Volume 43 Number 2 June 2019 ISSN 0350-5596

Informatica

An International Journal of Computing and Informatics

Editorial Boards

Informatica is a journal primarily covering intelligent systems in the European computer science, informatics and cognitive community; scientific and educational as well as technical, commercial and industrial. Its basic aim is to enhance communications between different European structures on the basis of equal rights and international refereeing. It publishes scientific papers accepted by at least two referees outside the author's country. In addition, it contains information about conferences, opinions, critical examinations of existing publications and news. Finally, major practical achievements and innovations in the computer and information industry are presented through commercial publications as well as through independent evaluations.

Editing and refereeing are distributed. Each editor from the Editorial Board can conduct the refereeing process by appointing two new referees or referees from the Board of Referees or Editorial Board. Referees should not be from the author's country. If new referees are appointed, their names will appear in the list of referees. Each paper bears the name of the editor who appointed the referees. Each editor can propose new members for the Editorial Board or referees. Editors and referees inactive for a longer period can be automatically replaced. Changes in the Editorial Board are confirmed by the Executive Editors.

The coordination necessary is made through the Executive Editors who examine the reviews, sort the accepted articles and maintain appropriate international distribution. The Executive Board is appointed by the Society Informatika. Informatica is partially supported by the Slovenian Ministry of Higher Education, Science and Technology.

Each author is guaranteed to receive the reviews of his article. When accepted, publication in Informatica is guaranteed in less than one year after the Executive Editors receive the corrected version of the article.

https://doi.org/10.31449/inf.v43i2.2179 Informatica **43** (2019) 151–159 **151**

A Review on CT and X-Ray Images Denoising Methods

Dang N. H. Thanh
Department of Information Technology, Hue College of Industry, Hue 530000 Vietnam
E-mail: dnhthanh@hueic.edu.vn

V. B. Surya Prasath
Division of Biomedical Informatics, Cincinnati Children's Hospital Medical Center, Cincinnati OH 45229 USA
Department of Biomedical Informatics, College of Medicine, University of Cincinnati, Cincinnati OH 45267 USA
Department of Electrical Engineering and Computer Science, University of Cincinnati, Cincinnati OH 45221 USA
E-mail: surya.prasath@cchmc.org, prasatsa@uc.edu

Le Minh Hieu
Department of Economics, University of Economics, The University of Danang, Danang 550000 Vietnam
E-mail: hieulm@due.udn.vn

Overview paper

Keywords: poisson noise, medical imaging, image processing, medical image processing, denoising

Received: February 6, 2018

In medical imaging systems, denoising is one of the important image processing tasks. Automatic noise removal will improve the quality of diagnosis and requires careful treatment of obtained imagery. Computed tomography (CT) and X-Ray imaging systems use the X radiation to capture images and they are usually corrupted by noise following a Poisson distribution. Due to the importance of Poisson noise removal in medical imaging, there are many state-of-the-art methods that have been studied in the image processing literature. These include methods that are based on total variation (TV) regularization, wavelets, principal component analysis, machine learning etc. In this work, we will provide a review of the following important Poisson removal methods: the method based on the modified TV model, the adaptive TV method, the adaptive non-local total variation method, the method based on the higher-order natural image prior model, the Poisson reducing bilateral filter, the PURE-LET method, and the variance stabilizing transform-based methods. Our task focuses on methodology overview, accuracy, execution time and their advantage/disadvantage assessments. The goal of this paper is to provide an apt choice of denoising method that suits to CT and X-ray images. The integration of several high-quality denoising methods in image processing software for medical imaging systems will be always excellent option and help further image analysis for computer-aided diagnosis.

Povzetek: Pregledni članek opisuje metode za čiščenje slike, narejene z rentgenom ali CT.

1 Introduction

Image denoising and noise removal with structure preservation is one of important tasks that are integrated in medical diagnostic imaging system, such as X-Ray, computed tomography (CT). X-ray and CT images are formed when an area under consideration of a patient is exposed under X-ray/CT and resulting attenuation is captured [1]. The noise density in these systems follows by the Poisson distribution and well known as the Poisson noise, shot noise, photon noise, Schott noise or quantum noise. Although Poisson noise does not depend on temperature and frequency, it depends on photon counters. Poisson noise strength is proportional with the pixel intensity growth: Poisson noise at higher intensity pixel is greater than one at less intensity pixel [2].

Nowadays, digitization is an important technique to improve image quality in medical imaging systems and the Poisson noise characteristics needs to be considered to remove it effectively [1]. Because the Poisson noise is a type of signal dependent noises, applying the usual denoising methods like for additive noises is ineffective, we need to design specific methods based on its characteristics.

There are many approaches were used to remove the Poisson noise, including total variation, mathematical transforms (wavelets, etc.), Markov random field, principal component analysis (PCA), machine learning etc. This paper mainly focuses on non-learning-based methods, learning technique is just a tiny part of this review that relates to the field of expert image prior model.

The approach that has been widely studied in the past few year and earn many achievements is regularization by total variation. This approach based on the regularization that was developed long time ago. Rudin et al. [3] used the total variation regularization to remove noise on digital images. Basically, they minimized an energy functional based on L^2 norm of image gradient with fixed constraint

for noise variance. The proposed model was also known as ROF (Rudin-Osher-Fatemi) model. This work is well-known and was cited by tens of thousands of times. However, the ROF model focuses on restoring images that are degraded by Gaussian noise. This model is ineffective to process Poisson noise: in the resulting image, the edge is not well preserved; if regularization strength is decreased, the noise in higher intensity-region still remains.

To pass over those limitations of the ROF model, Triet et al. [2] proposed an improved version that can process the Poisson noise well. This model is known as modified ROF model (MROF). However, both of original methods that based on ROF and MROF create an effect: artificial artifacts [1]. The artificial artifacts on digital images are misrepresentations of image processing. This effect makes some regions of images get unnatural [4]. The artifacts have many types, such as: staircasing, star, halo etc. In medical imaging, these artifacts can cause doctors to mistake for actual pathology. Usually, they need to learn to recognize these artifacts to avoid mistaking. So, during the processing, these artificial regions should not to be created. Prasath [1] proposed an adaptive version of MROF to remove this effect. This method is known as the adaptive total variation method (ATV).

A common problem of both MROF and ATV methods is ineffective to process on photon-limited image. To enhance quality of this type of image in denoising process, Salmon et al. [5] proposed the non-local PCA method. Thereafter, Liu et al. [6] proposed another adaptive non-local total variation method (ANLTV). This method increases the information structure of image and gives the better denoising result on photon-limited images.

Non-local approaches like ANLTV are state-of-the-art. However, if the local models are combined with training process, we can get the result that is not inferior to other state-of-the-art non-local models. Wensen et al. [7] proposed a local variational model that incorporates the fields of expert prior image that is widely used in image prior and regularization model. This model is known as the higher-order natural image prior model (HNIPM). The HNIPM can remove Poisson noise on both high and low peak images. Although this model is local, since the model is trained on the Anscombe transform domain (very effective for Poisson denoising), it is also a competitive model to compare to other state-of-the-art Poisson denoising models.

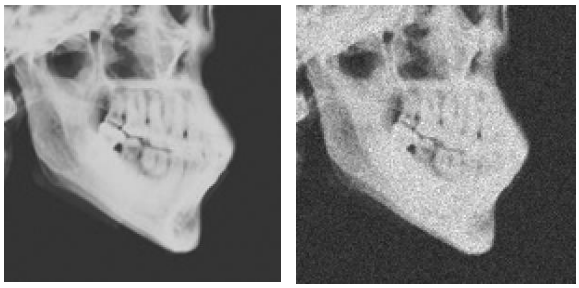
However, above methods are performed on iteration and this requires more execution time to remove noise. Kirti et al. [8] proposed a spatial domain filter by modifying bilateral filter framework to remove Poisson noise. The Poisson reducing bilateral filter (PRBF) is non-iterative nature. So, it can treat Poisson noise faster than iterative based approaches.

Another approach is highly expected – wavelet and its modifications. Thierry et al. proposed a denoising method based on image-domain minimization of Poisson unbiased risk estimation: PURE-LET (**P**oisson **U**nbiased **R**isk **E**stimation – **L**inear **E**xpansion of **T**hresholds) [9]. This method is performed in a transformed domain: undecimated discrete wavelet transform and can be extended with some other transforms. Zhang et al. [10] also proposed a multiscale variance stabilizing transform (MS-VST) that can be deemed as an extension of Anscombe transform. This transform also can be combined with wavelet, ridgelet, and curvelet [10]. Both PURE-LET and MS-VST are competitive relative to many existing denoising methods, in which, the VST based methods are new research trend for CT and X-Ray images denoising [11] [12] [13] [14], because of using VST, Poisson noise can be treated as the additive Gaussian noise. Hence, researchers can reuse the existing Gaussian denoising methods, that get many achievements and it is unnecessary to develop a partial denoising method to treat Poisson noise.

Our paper is organized as follows: in Section 2, a detail about image formation on CT/X-Ray imaging systems and characteristics of Poisson noise are provided; in Section 3, methodology of Poison denoising methods are covered shortly; Section 4 and Section 5 present the discussion about accuracy, performance, advantages/disadvantages of methods and the conclusion.

2 Image formation in medical imaging systems and Poisson noise

In CT and X-Ray imaging systems, to produce a radiographic image, X-Ray photons must pass through tissue and interact with an image receptor. The process of image formation is a result of differential absorption of the X-Ray beam as it interacts with the anatomic tissue [15]. Differential absorption is a process whereby some of the X-Ray beam is absorbed in the tissue and some passes through the anatomic part. Because varying anatomic parts do not absorb the primary beam to the same degree, anatomic parts composed of bone absorb more X-Ray photons than parts filled with air. Differential absorption of the primary X-Ray beam creates an image that structurally represents the anatomic area of interest.



a b

Figure 1: The Poisson noise generation: a) The expected noise-free image; b) The noisy image

Poisson noise is a fundamental form of uncertainty associated with the measurement of light, inherent to the quantized nature of light and the independence of photon detection [16]. Its expected magnitude is signal-dependent and causes the dominant source of image noise except in low-light conditions.

Image sensors measure scene irradiance by counting the number of discrete photons incident on the sensor over a given time interval. Because of the photoelectric effect

in digital sensors, photons are converted into electrons, whereas film-based sensors rely on photo-sensitive chemical reaction. Then, the random individual photon arrival leads to Poisson noise.

Individual photon detections can be considered as independent events that follow a random temporal distribution. The photon counting is a Poisson process, and the number of photons k measured by a given sensor element over a time interval t is described by the discrete probability distribution

$$P(k) = \frac{e^{-\lambda t}(\lambda t)^k}{k!},$$

where λ – expected number of photons per unit time interval, which is proportional to the incident scene irradiance.

Since the Poisson noise is derived from the nature of signal itself, it provides a lower bound on the uncertainty of measuring light. Any measurement would relate to Poisson noise, even under the ideal conditions of free-noise sources. When Poisson noise is the only significant source of uncertainty, as commonly occurs in bright photon-rich environments, imaging is called photon-limited [16]. By the Poisson distribution, to reduce the Poisson noise, need to capture more photons. This requires longer exposures times or increasing the X-Ray intensity beam. However, the number of photons captured in a single shot is limited by the full well capacity of the sensor. Moreover, increasing exposures times or photon intensity beam would be harmful for health of patients. Since this limitation of technology, it is necessary to reduce the Poisson noise by image processing algorithms.

Figure 1 simulates the Poisson noise generation on image. We use the built-in *imnoise* function of MATLAB to generate the Poisson noise on skull image [17]. The Poisson noise in the higher intensity regions is greater than one of the lower intensity regions.

3 Denoising methods on CT and X-Ray images

3.1 The modified ROF model

Suppose that f – a given grayscale image on Ω (a bounded open subset of $\mathbb{R}^2$, i.e. $\Omega \subset \mathbb{R}^2$), u – an expected denoising image that closely matches to observed image, $x = (x_1, x_2) \in \Omega$ – pixels.

By using total variation regularization, Triet et al. convert the Poisson denoising problem to the following minimization problem:

$$u = \underset{u}{\operatorname{argmin}} \left(\int_\Omega \big(u - f.ln(u)\big)dx + \beta \int_\Omega |\nabla u| dx \right) \quad (1)$$

where, $\beta > 0$ – regularization parameter.

To solve this problem, Triet et al. used the gradient descent method that replaces the regularization parameter by function that is suitable to process noise on image regions with both low and high intensity. This manner exactly suits the signal-dependent nature of Poisson noise.

3.2 The adaptive Total variation method

The adaptive total variation method is similar with above method. However, the second term in (1) is replaced by an adaptive total variation:

$$u = \underset{u}{\operatorname{argmin}} \left(\int_\Omega \big(u - f.ln(u)\big)dx + \int_\Omega \omega(x)|\nabla u| dx \right) \quad (2)$$

where,

$$\omega(x) = \frac{1}{1 + k|G_\sigma * \nabla u|},$$

G_σ – the Gaussian kernel for smoothing with σ variance, $k > 0$ – contrast parameter, operator $*$ is convolution.

In order avoid staircasing artifacts, Prasath [1] proposed the generalized inverse gradient term incorporating to the local statistics with patches extracted from image. The detail about this term is presented below.

Let $\mathcal{N}_{x,r}$ be the local region centered at x with radius r. Consider the local histogram of a pixel $x \in \Omega$ and its corresponding cumulative distribution function [18]:

$$H_x(y) = \frac{\big|\{z \in \mathcal{N}_{x,r} \cap \Omega | u(z) = y|\}\big|}{\big|\mathcal{N}_{x,r} \cap \Omega\big|},$$

$$C_x(y) = \frac{\big|\{z \in \mathcal{N}_{x,r} \cap \Omega | u(z) \le y|\}\big|}{\big|\mathcal{N}_{x,r} \cap \Omega\big|},$$

Where $0 \le y \le L$, L – maximum possible pixel value of the image, $|\cdot|$ – the number of elements of set (cardinality).

The local histogram quantity to quantify local regions of given image is:

$$\mathcal{Q}(x) = \int_0^L C_x(y)dy.$$

Finally, the adaptive weight in (2) is defined as:

$$\omega(x) = \frac{1}{1 + k(|G_\sigma * \nabla u(x)|/\mathcal{Q}(x))^2}$$

The alternating direction method of multipliers [1] is provided to solve the problem (2). This iterative manner also gives good performance.

3.3 The adaptive non-local Total Variation method

In the case of photon-limited image, a lot of useful structure information of original image has been lost. So, the corrupted image is close to the binary image. If we only apply the denoising methods, such as the modified ROF model or the adaptive total variation, the denoising result is not really effective.

For this type of images, firstly, we need to enhance image (improve light, contrast, etc.) and after that, perform the denoising process.

The method that Liu et al. [6] proposed is similar with above idea. For first step, they enhance the image detail by using Euler's elastica. In second step, they remove noise by using non-local total variation to aim to preserve the structure information.

The Euler's elastica-based noise image enhancement model is proposed hereafter:

$$u = \underset{u}{\operatorname{argmin}} \left(\int_{\Omega} u - f.ln(u)dx + \lambda \int_{\Omega} \left(a + b \left(\nabla . \frac{\nabla u}{|\nabla u|} \right)^2 \right) |\nabla u| dx \right) \quad (3)$$

where $\lambda > 0$ – regularization parameter, $a > 0, b > 0$ – weight parameters.

The Poisson denoising model based on non-local total variation is provided as follows:

$$U = \underset{u}{\operatorname{argmin}} \left(\int_{\Omega} (u - U)^2 dx + \alpha \int_{\Omega} |\nabla_{\mathrm{NL}} u| dx \right), \quad (4)$$

where

$$\int_{\Omega} |\nabla_{\mathrm{NL}} u| dx = \int_{\Omega} \sqrt{\int_{\Omega} \big(u(x) - u(y)\big)^2 \omega(x,y) dy} \, dx$$

– is non-local total variation, $\omega(x, y)$ – the non-local weight to measure the similarity of patches centered at the pixels x and y. The denoised version will be restored from (4) by using an inverse Anscombe transform as bellow:

$$u = \left(\frac{U}{2}\right)^2 - \frac{3}{8}.$$

The alternating direction method of multipliers is also recommended to solve the models (3) and (4).

3.4 The higher-order natural image prior model

The denoising method by the higher-order natural image prior model is based on the fields of expert image prior model that can be presented as follows:

$$\underset{u}{\operatorname{argmin}} \sum_{i=1}^{N_f} \alpha_i \sum_{p=1}^{N} \rho\big((k_i * u)_p\big) + D(u, f), \quad (5)$$

where N_f – number of filters, k_i – set of learned linear filters with corresponding weights $\alpha_i > 0$, N – number of image pixels, $\rho(z) = \ln(1 + z^2)$ – the potential function, $(k_i * u)_p$ – a convolution at pixel p. The first term is derived from the fields of expert image prior model, the second term $D(u, f)$ is data fidelity that has various forms.

By using model (5), Wensen et al. [7] proposed two models that were trained in various transform domains: the first model – is trained in the original image domain with the Poisson noise statistics derived data term; the second model – is trained in the Anscombe transform domain with a quadratic data term. The first model removes Poisson noise on high peak images effectively, but it fails for low peak image. The reason is for the low peak image, there are large regions of image, in which, there are many pixels with zero intensity. This leads to those pixels with zero intensity cannot be updated and fixed at 0 in the iterations. Hence, noise still remains. The second model is powerful to remove noise for low peak images, but the quadratic data term is only effective to treat Gaussian noise. So, Wensen et al. combined the advantages of two models to make a novel model by replacing the quadratic data term in the second model by the Poisson noise statistics of data term in the first model. The resulting model proved its own power to remove the Poisson noise for both cases of high and low peak images.

The iPiano algorithm [19] is recommended to solve the resulting model. It is an efficient algorithm for non-convex optimization problems.

3.5 The Poisson reducing bilateral filter

The bilateral filter was proposed by Tomasi et al. [20] to reduce additive Gaussian noise. This filter was developed based on the geometric and photometric distances in a local window. Kirti et al. [8] modified this filter by replacing the geometric distance by Poisson distribution. Therefore, the mean value is selected as mean of image intensity in a local window. For every mean value in the local window, the expected value is estimated by the maximum likelihood estimation method.

Since the Poisson reducing bilateral filter is non-iterative nature, its performance primarily depends on the maximum likelihood estimation method.

3.6 The PURE-LET method

The PURE-LET (**P**oisson **U**nbiased **R**isk **E**stimation – **L**inear **E**xpansion of **T**hresholds) method [9] is extended from SURE-LET method [21]. The PURE-LET method is used to reduce Poisson noise. Basically, this denoising method was proposed based on a minimization of Poisson unbiased risk estimation by using the linear expansion of thresholds (LET). Luisier et al. [9] proposed the PURE-LET to reduce Poisson noise without any priori hypotheses on noise-free image.

The main goal of this proposed denoising method is a minimization of the mean squared error of the noise-free image and the denoised image. However, since the noise-free image is unknown, unbiased risk estimation was used that known as the Poisson unbiased risk estimation. This estimation was given in the unnormalized-Haar-discrete-wavelet-transform domain. In this estimation, an unknown image function used to replace for the noise-free image.

To minimize this estimation, above unknow image function will be expressed in the linear expansion of thresholds. If elementary denoising functions are given, the minimization problem gets to be the problem of finding weight parameters in the linear expansion. Hence, the main task of this PURE-LET method focuses on solving a linear system of equations, in which the variables are weight parameters of the linear expansion.

The linear expansion of thresholds can be presented in transformed domain, such as unnormalized wavelet transform, Anscombe transform and Haar-Fisz transform.

Another important task in the PURE-LET methods is choosing a set of elementary denoising functions (or thresholding functions). These functions need to be satisfied the following minimal properties: differentiability, anti-symmetry, linear behavior for large coefficients.

The PURE-LET method is a competitive method to compare to other state-of-the-art Poisson denoising methods. This method is also easy to be extended to treat other noises, such as Gaussian noise [22] [23] [24] [25], the mixed noise [26] [27] [28] [29] [30]. The method performance much depends on the performance of methods of

solving linear system of equations, for example, the Gauss-Seidel method.

3.7 The multiscale variance stabilizing transform method

The multiscale variance stabilizing transform method is proposed by Zhang et al. [10] to reduce Poisson noise on photon-limited image. This method is based on the variance stabilizing transform (VST) that is incorporated within the multiscale framework offered by the undecimated wavelet transform (UWT). This transform is used because of its translation-invariant denoising. The denoising task comes to finding coefficients of the multiscale variance stabilizing transform. By using these coefficients, we can estimate the noise-free image.

The denoising method involves in the following steps: transformation – computation of UWT in conjunction with MS-VST; detection – detection of significant detail coefficients by hypotheses test; estimation – reconstruction of the final estimate by using the knowledge of the detected coefficients. Since the signal reconstruction requires inverting the MS-VST-combined UWT, this reconstruction process is formulated as a convex sparsity-promotion optimization problem. This optimization problem can be solved by many iterative methods, such as the iterative hybrid steepest descent method.

The MS-VST method can be combined with wavelet, as well as ridgelet (wavelet analysis in Radon domain) or curvelet. Further, this method can also to be extended to reduce other types of noise.

3.8 Adaptive variance stabilizing transform based methods

The Poisson denoising methods by VST-based approach is often performed by three steps: applying the variance stabilizing transform, such as Anscombe transform; applying the denoising methods to resulting image, in which the denoising methods are the one for additive Gaussian noise; using inverse transformation to denoised image to get the Poisson denoised image.

Hence, VST-based methods can use state-of-the-art Gaussian denoising methods. By this idea, there are some very effective methods, such as BM3D [31], SAFIR [32], BLS-GSM [33].

For VST-based methods, the choice of inverse transformation is very important. Makitalo and Foi [11] proposed the optimal inverse Anscombe transform. The adaptive variance stabilizing transform-based method of Makitalo et al. can be covered as follows:

Step 1: Apply the Anscombe transform to Poisson noisy image to get asymptotically additive Gaussian noisy image. For z – the observed pixel values obtained through an image acquisition device, the Anscombe transform is

$$f(z) = 2\sqrt{z + \frac{3}{8}}, z = (z_1, \dots, z_N), N - \text{pixel numbers}.$$

Step 2: Denoise the transformed images by additive Gaussian denoising method.

Step 3: The denoising of $f(z)$ produces a signal D that considered as an estimate of $E\{f(z)y\}, y = (y_1, \dots, y_N)$ – pixel values of denoising image, $E\{.\}$ – the mean. So, it is necessary to apply inverse transformation to D to obtain the desired estimate of y. The inverse transformations can be used include:

a) The exact Unbiased inverse

$$\mathcal{I}_C(D) = 2\sum_{z=0}^{+\infty}\left(\sqrt{z + \frac{3}{8}}.\frac{D^z e^{-D}}{z!}\right).$$

b) The ML inverse

$$\mathcal{I}_{ML}(D) = \begin{cases}\mathcal{I}_C(D), & if\ D \geq 2\sqrt{3/8} \\ 0, & if\ D < 2\sqrt{3/8}\end{cases}.$$

c) The MMSE inverse

$$\mathcal{I}_{MMSE}(D) = \int_{-\infty}^{+\infty} p(D|y)ydy \Big/ \int_{-\infty}^{+\infty} p(D|y)dy,$$

where, $p(D|y) = \frac{1}{\sqrt{2\pi\epsilon^2}}\mathrm{e}^{-\frac{1}{2\epsilon^2}(D - E\{f(z)|y\})^2}$ – the generalized probability density function of z conditioned on y.

Another adaptive VST-based method that has high accuracy and performance to treat Poisson noise was proposed by Azzari and Foi [12]. This method is known as the iterative VST-based method.

This method is also handled via three steps as above. However, in step 2, authors proposed another method to remove noise, but they did not use existing additive Gaussian denoising methods. The method is effective and has high performance because it exploited characteristics of Anscombe transformation.

The algorithm starts by setting $\hat{y}_0 = z$. At each iteration $i = 1, \dots, K$, a convex combination needs to be computed:

$$\bar{z}_i = \lambda_i z + (1 - \lambda_i)\hat{y}_{i-1},$$

where $0 < \lambda_i < 1$, $\hat{y}$ – estimate of y. So, $\hat{y}_{i-1}$ can be treated as a surrogate for y:

$$E\{\overline{z}_i|y\} = y = \lambda_i^{-2} var\{\overline{z}_i|y\},$$

Where, $E\{.\}, var\{.\}$ – the mean and variance respectively, and $\overline{z}_i$ has higher SNR (signal-to-noise ratio) than z for any $\lambda_i < 1$.

Apply a VST f_i to $\overline{z}_i$ and obtain an image $\overline{\overline{z}}_i = f_i(\overline{z}_i)$, which can be denoised by a filter Φ for additive white Gaussian noise to get a filtered image $D_i = \Phi(\overline{\overline{z}}_i)$. Assuming $D_i = E\{f_i(\overline{z}_i)|y\}$, the exact unbiased inverse of f_i,

$$\mathcal{I}: E\{f_i(\overline{z}_i)|y\} \to E\{\overline{z}_i|y\} = y,$$

Will restore the original image:

$$\hat{y}_i = \mathcal{I}_{f_i}^{\lambda_i}(D_i).$$

This process loops until $i = K$.

The accuracy and performance of this method are competitive to other state-of-the-art Poisson denoising methods.

3.9 Other Poisson denoising methods

Since the Poisson denoising problem has important role not only in medicine, but also in other fields, such material science, astronomy etc., beside above state-of-the-art denoising methods, there are also many other denoising methods are highly assessed, such as:

The adaptive BLS-GSM method of Li et al. [34]. They proposed this method based on Bayesian least squares method. Basically, this Poisson denoising method is a term of VST-based approach.

The optimized anisotropic Poisson denoising method of Radow et al. [35]. This method is proposed based on variational approach and anisotropic regulariser in the spirit of anisotropic diffusion. This method can be considered as a part of the total variation regularization.

The Poisson denoising based on greedy approach of Dupe and Anthoine [36]. The goal of this method is combination of a greedy method with Moreau-Yosida regularization of the Poisson likelihood.

The Poisson reduction based on region classification and response median filtering of Kirti et al [37]. Their contribution is usage of modified Harris corner point detector to predict noisy pixels and responsive median filtering in spatial domain.

The primal-dual hybrid gradient algorithm [38] is a Poisson denoising method that should be also noticed. This method is based on total variation regularization and primal-dual hybrid gradient. So, this method has very good performance.

4 Discussion

Firstly, we will discuss on the accuracy of Poisson denoising methods. The MROF, ATV, ANLTV and HNIPM methods based on regularization, their accuracy is good enough to perform in medical imaging systems. Since the HNIPM method is trained on Anscombe transform domain, regardless of its localization, its accuracy is competitive enough to other methods. If we combine the MROF, ATV, ANLTV methods with training process to select optimal parameters in iterative manners, their accuracy might be so far better than the HNIPM method, especially, for the ANLTV method, because it does not change the information structure of image in denoising process.

An effect that reduces the accuracy in denoising process is artificial artifacts. Almost of local methods usually create this effect. So, we need to perform some techniques to avoid adding artifacts to images, such in the case of the ATV method. For non-local methods, since the information structure of image is preserved, the artifacts will be seldom added. The PRBF method has the lowest accuracy to compare to other denoising methods, including the PURE-LET, MS-VST and adaptive VST-based methods. When filter noise by PRBF, the hallo artifacts will appear in resulting images and the artifacts strength depends on filter parameters. Although we can control these parameters to reduce the hallo artifacts, it is very hard to select optimal values. There are some methods were developed to reduce this type of artifacts [39] [40], but it is still unfinished, especially, on Poisson noise reduction process by bilateral filter. For the PURE-LET, MS-VST and adaptive VST-based methods, the accuracy might be better than local variational based methods without training process, particularly, for the photon-limited images. However, the PURE-LET method is usually unstable. In our test, the denoising result by the PURE-LET method is slightly different in every execution, regardless of unchangeable input setting of parameters and configuration. When we compare the MS-VST method to the PURE-LET method, the MS-VST method has better accuracy, especially, for photon-limited images [10]. The adaptive VST-based methods have better accuracy and performance to compare to MS-VST method [11] [12]. Both the PURE-LET and MS-VST cause the artifacts. For the adaptive VST-based methods, appearance of the artifacts depends on selection of Gaussian denoising methods.

Secondly, we focus on method performance by assessing the execution time. Poisson denoising methods, such as MROF, ATV, ANLTV, PURE-LET, MS-VST and adaptive VST-based methods are designed on iterative manner, so their execution time is longer than one of the PRBF method. The PRBF method is very fast and this is proven in processing large images. Execution time of both of PURE-LET, MS-VST and adaptive VST-based methods also depends on computation time of transforms. Otherwise, for the PURE-LET method, it also depends on execution time of solving system of linear equations, and for the MS-VST method – depends on performance of method to solve convex optimization problem, such as the hybrid steepest decent method, and for adaptive VST-based methods – depends on performance of selective Gaussian denoising methods. For other methods: MROF, ATV, ANLTV, HNIPM, execution time much depend on performance of method to solve optimization problem (convex optimization for the MROF, ATV, ANLTV methods and nonconvex optimization for the HNIPM method). There were some methods are recommended in their proposed works to solve these optimization problems: the gradient descent method, the alternating direction method of multipliers for convex optimization; iPiano for non-convex optimization. However, for the convex optimization, we can use other faster methods, such as: the primal-dual modified extragradient method, the primal-dual Arrow-Hurwitz method, the graph-cut method [41]. In work [41], Chambolle et al. showed comparison of execution time of above methods with the alternating direction method of multipliers. Among of these methods, the primal-dual Arrow-Hurwitz method is the fastest, but proof of its convergence is open problem. The primal-dual modified extragradient method is certainly convergent and it is easy to parallelize on GPU. The graph-cut method is very fast and give exact discrete solutions, but an efficient parallelization on GPU is still open problem. For the non-convex optimization, the iPiano method is state-of-the-art algorithm and fast enough to applied in this situation. Parallelization of the iPiano method is still open problem. Hence, the execution time problem of all above methods can be solved by combining with higher performance algorithms and/or parallel processing.

Finally, about methodology, the MROF, ATV, ANLTV and PRBF methods are simple and easy to understand and easy to write program. The HNIPM is slightly more complex and requires the training process. Both of PURE-LET, MS-VST and adaptive VST-based methods are the most complex. They are performed in various transform domains. Their accuracy and performance also depend on calculation of these transforms.

To choose suitable method for Poisson denoising in specific cases, we need to know their advantages and disadvantages. These advantages and disadvantages are listed in Table 1 in terms of denoising capabilities of the reviewed denoising methods here. After decades of denoising research there are no universal denoising method even in the case of additive Gaussian noise. However, by concentrating on the state of the art denoising methods with emphasize of domain specific techniques will pave the way for choosing an optimal denoising method. We believe the overview of Poisson denoising methods based on mathematically well-defined techniques studied here can be used by researchers in developing and utilizing these in various domains.

5 Conclusion

The denoising on CT/X-Ray images is still a challenge in medical image processing, especially, on the photon-limited images. The state-of-the-art methods cannot solve simultaneously the following tasks: high accuracy on both photon-limited and photon-unlimited images, avoid adding artificial artifacts and the performance. The goal to develop an effective universal method that reduces multiple types of noise is even more difficult challenge.

In this paper, we reviewed on the following methods: MROF, ATV, ANLTV, HNIPM, PRBF, PURE-LET and MS-VST. The PRBF is excellent choice if the execution time is the most important. However, if the accuracy is priority, non-local methods are recommended. If we need to process the photon-limited images, the ANLTV, MS-VST and adaptive VST-based methods are very good choices. If we want to exploit the existing Gaussian denoising methods, we can use adaptive VST-based methods, including MS-VST.

During denoising process is performed, it is necessary to avoid adding artificial structures, and one can choose ATV or ANLTV methods that provide good denoising performance without introducing discernible artifacts. In this case, the VST-based methods can be used if they are combined to the image structure preservation Gaussian denoising methods, such as BM3D [31], SAFIR [32] etc.

By the research trend, the VST-based approach is a novel option by the criteria to create an “universal” method to remove multiple type of noises. This approach has potential if it is possible to expand the VST-based approach to apply to other signal dependent noises.

6 References

[1] V. B. S. Prasath, "Quantum Noise Removal in X-Ray Images with Adaptive Total Variation Regularization," *Informatica*, vol. 28, no. 3, pp. 505-515, 2017. *http://dx.doi.org/10.15388/Informatica.2017.141*

[2] L. Triet, C. Rick and J. A. Thomas, "A Variational Approach to Reconstructing Images Corrupted by Poisson Noise," *Journal of Mathematical Imaging and Vision*, vol. 27, no. 3, pp. 257-263, 2007. *https://doi.org/10.1007/s10851-007-0652-y*

[3] I. R. Leonid, O. Stanley and F. Emad, "Nonlinear total variation based noise removal algorithms," *Physica D: Nonlinear phenomena*, vol. 60, no. 1-4, pp. 259-268, 1992. *https://doi.org/10.1016/0167-2789(92)90242-F*

[4] W. Zhifeng, L. Si, Z. Xueying, X. Yuesheng and K. Andrzej, "Reducing staircasing artifacts in spect reconstruction by an infimal convolution regularization," *Journal of Computational Mathematics*, vol. 34, no. 6, pp. 626-647, 2016. *https://doi.org/10.4208/jcm.1607-m2016-0537*

[5] S. Joseph, H. Zachary, D. Charles-Alban and W. Rebecca, "Poisson Noise Reduction with Non-local PCA," *Journal of Mathematical Imaging and Vision*, vol. 48, no. 2, pp. 279-294, 2014. *https://doi.org/10.1007/s10851-013-0435-6*

[6] H. Liu, Z. Zhang, L. Xiao and Z. Wei, "Poisson noise removal based on non-local total variation with Euler's elastica pre-processing," *Journal of Shanghai Jiao Tong University*, vol. 22, no. 5, pp. 609-614, 2017. *https://doi.org/10.1007/s12204-017-1878-5*

[7] F. Wensen, Q. Hong and C. Yunjin, "Poisson noise reduction with higher-order natural image prior model," *SIAM journal on imaging sciences*, vol. 9, no. 3, pp. 1502-1524, 2016. *https://doi.org/10.1137/16m1072930*

[8] V. T. Kirti, H. D. Omkar and M. S. Ashok, "Poisson noise reducing Bilateral filter," *Procedia Computer Science*, vol. 79, pp. 861-865, 2016. *https://doi.org/10.1016/j.procs.2016.03.087*

[9] F. Luisier, C. Vonesch, T. Blu and M. Unser, "Fast Interscale Wavelet Denoising of Poisson-corrupted Images," *Signal Processing*, vol. 90, pp. 415-427, 2010. *https://doi.org/10.1016/j.sigpro.2009.07.009*

[10] Z. Bo, M. F. Jalal and S. Jean-Luc, "Wavelets ridgelets and curvlets for Poisson noise removal," *IEEE Transaction on image processing*, vol. 17, no. 7, pp. 1093-1108, 2008. *https://doi.org/10.1109/tip.2008.924386*

[11] M. Markku and F. Alessandro, "Optimal Inversion of the Anscombe Transformation in Low-Count Poisson Image Denoising," *IEEE Transactions on Image Processing*, vol. 20, no. 1, pp. 99-109, 2011. *https://doi.org/10.1109/tip.2010.2056693*

[12] A. Lucio and F. Alessandro, "Variance Stabilization for Noisy+Estimate Combination in Iterative Poisson Denoising," *IEEE Signal Processing Letters*, vol. 23, no. 8, pp. 1086-1090, 2016. *https://doi.org/10.1109/lsp.2016.2580600*

[13] M. Niklas, B. Peter, D. Wolfgang, M. V. Paul and B. Y. Andrew, "Poisson noise removal from high-resolution STEM images based on periodic block matching," *Advanced Structural and Chemical Imaging*, vol. 1, no. 3, pp. 1-19, 2015. *https://doi.org/10.1186/s40679-015-0004-8*

[14] A. S. Sid, Z. Messali, F. Poyer, R. L. Lumbroso-Le, L. Desjardins, T. C. D. Cassoux N., S. Marco and S. Lemaitre, "Iterative Variance Stabilizing Transformation Denoising of Spectral Domain Optical Coherence Tomography Images Applied to Retinoblastoma," *Ophthalmic Research*, vol. 59, pp. 164-169, 2018. *https://doi.org/10.1159/000486283*

[15] T. L. Fauber, Radiographic Imaging and Exposure, Missouri: Elsevier, 2017.

[16] S. W. Hasinoff, "Photon, Poisson Noise," in *Computer Vision: A Reference Guide*, Boston, MA, Springer US, 2014, pp. 608-610. *https://doi.org/10.1007/978-0-387-31439-6_482*

[17] N. Veasey, "X-ray of skull showing brain and neurons," Getty Images.

[18] V. B. S. Prasath and R. Delhibabu, "Automatic contrast parameter estimation in anisotropic diffusion for image restoration," in *International Conference on Analysis of Images, Social Networks and Texts*, p. 198-206, Yekaterinburg, 2014. *https://doi.org/10.1007/978-3-319-12580-0_20*

[19] O. Peter, C. Yunjin, B. Thomas and P. Thomas, "iPiano: Inertial Proximal Algorithm for Nonconvex Optimization," *SIAM Journal on Imaging Sciences*, vol. 7, no. 2, p. 1388–1419, 2014. *https://doi.org/10.1137/130942954*

[20] C. Tomasi and R. Manuchi, "Bilateral filtering for gray and color images," in *Sixth International Conference on Computer Vision*, Bombay, 1998. *https://doi.org/10.1109/iccv.1998.710815*

[21] B. Thierry and L. Florian, "The SURE-LET approach to image denoising," *IEEE transaction on image processing*, vol. 16, no. 11, pp. 2778-2786, 2007. *https://doi.org/10.1109/tip.2007.906002*

[22] V. B. S. Prasath, D. N. H. Thanh, N. H. Hai and N. X. Cuong, "Image Restoration With Total Variation and Iterative Regularization Parameter Estimation," in *Proceedings of the Eighth International Symposium on Information and Communication Technology*, pp. 378-384, Nha Trang, 2017. *https://doi.org/10.1145/3155133.3155191*

[23] V. B. S. Prasath and D. Vorotnikov, "On a System of Adaptive Coupled PDEs for Image Restoration," *Journal of Mathematical Imaging and Vision*, vol. 48, no. 1, pp. 35-52, 2014. *https://doi.org/10.1007/s10851-012-0386-3*

[24] V. B. S. Prasath and A. Singh, "A hybrid convex variational model for image restoration," *Applied Mathematics and Computation*, vol. 215, no. 10, pp. 3655-3664, 2010. *https://doi.org/10.1016/j.amc.2009.11.003*

[25] V. B. S. Prasath, D. Vorotnikov, P. Rengarajan, S. Jose, G. Seetharaman and K. Palaniappan, "Multiscale Tikhonov-Total Variation Image Restoration Using Spatially Varying Edge Coherence Exponent," *IEEE Transactions on Image Processing*, vol. 24, no. 12, pp. 5220 - 5235, 2015. *https://doi.org/10.1109/tip.2015.2479471*

[26] D. N. H. Thanh and S. D. Dvoenko, "A method of total variation to remove the mixed Poisson-Gaussian noise," *Pattern Recognition and Image Analysis*, vol. 26, no. 2, pp. 285-293, 2016. *https://doi.org/10.1134/s1054661816020231*

[27] D. N. H. Thanh and S. D. Dvoenko, "A Mixed Noise Removal Method Based on Total Variation," *Informatica*, vol. 26, no. 2, pp. 159-167, 2016.

[28] D. N. H. Thanh and S. D. Dvoenko, "A Variational Method to Remove the Combination of Poisson and Gaussian Noises," in *Proceedings of the 5th International Workshop on Image Mining. Theory and Applications (IMTA-5-2015) in conjunction with VISIGRAPP 2015*, pp. 38-45, Berlin, 2015. *https://doi.org/10.5220/0005460900380045*

[29] D. N. H. Thanh and S. D. Dvoenko, "Image noise removal based on total variation," *Computer Optics*, vol. 39, no. 4, pp. 564-571, 2015. *https://doi.org/10.18287/0134-2452-2015-39-4-564-571*

[30] D. N. H. Thanh, S. D. Dvoenko and D. V. Sang, "A Denoising Method Based on Total Variation," in *Proceedings of the Sixth International Symposium on Information and Communication Technology*, pp. 223-230, Hue, 2015. *https://doi.org/10.1145/2833258.2833281*

[31] K. Makitalo and A. Foi, "On the inversion of the Anscombe transformation in low-count Poisson image denoising," in *Workshop Local and Non-Local approximation Image Processing*, pp. 26-32, Tuusula, 2009. *https://doi.org/10.1109/lnla.2009.5278406*

[32] J. Boulanger, J. B. Sibarita, C. Kervrann and P. Bouthemy, "Non-parametric regression for patch-based fluorescence microscopy image sequence denoising," in *Fifth IEEE International symposium on Biomedical Imaging*, Paris, 2008. *https://doi.org/10.1109/isbi.2008.4541104*

[33] J. Portilla, V. Strela, M. J. Wainwright and E. P. Simoncelli, "Image denoising using scale mixtures of Gaussian in the wavelet domain," *IEEE Trans. Image Process.*, vol. 12, no. 11, pp. 1338-1351, 2003. *https://doi.org/10.1109/tip.2003.818640*

[34] L. Li, N. Kasabov, J. Yang, L. Yao and Z. Jia, "Poisson Image Denoising Based on BLS-GSM Method," in *International conference on Neural Information Processing*, pp. 513-522, Istanbul, 2015. *https://doi.org/10.1007/978-3-319-26561-2_61*

[35] G. Radow, M. Breub, L. Hoeltgen and T. Fischer, "Optimised Anisotropic Poisson Denoising," in *Scandinavian conference on Image Analysis*, pp. 502-514, Tromso, 2017. *https://doi.org/10.1007/978-3-319-59126-1_42*

[36] F. X. Dupe and S. Anthoine, "A greedy approach to sparse Poisson denoising," in *IEEE International workshop on Machine Learning for Signal Processing (MLSP)*, pp. 1-6, Southampton, 2013. *https://doi.org/10.1109/mlsp.2013.6661993*

[37] T. Kirti, K. Jitendra and S. Ashok, "Poisson noise reduction from X-Ray images by region classification and response median filtering," *Sadhana*, vol. 42, no. 6, pp. 855-863, 2017. *https://doi.org/10.1007/s12046-017-0654-4*

[38] S. Bonettini and V. Ruggiero, "On the convergence of primal-dual hybrid gradient algorithms for total variation image restoration," *Journal of Mathematical Imaging and Vision*, vol. 44, no. 3, pp. 236-253, 2012. *https://doi.org/10.1007/s10851-011-0324-9*

[39] X. Chuanmin and S. Zelin, "Adaptive Bilateral Filtering and Its Application in Retinex Image Enhancement," in *Seventh International Conference on Image and Graphics (ICIG)*, pp. 45-49, Qingdao, 2013. *https://doi.org/10.1109/icig.2013.15*

[40] V. K. Nath, D. Hazarika and A. Mahanta, "Blocking artifacts reduction using adaptive bilateral filtering," in *International Conference on Signal Processing and Communications (SPCOM)*, pp. 1-5, Bangalore, 2010. *https://doi.org/10.1109/spcom.2010.5560517*

[41] A. Chambolle, M. Novaga, D. Cremers and T. Pock, "An introduction to total variation for image analysis," in *Theoretical Foundations and Numerical Methods for Sparse Recovery*, De Gruyter, 2010, pp. 1-87.

Property / Method	Ability of de-noising on photon-limited image	Add artifacts	Level of methodology	Level of parallelization	Level of execution time*
MROF	No	Staircasing	Easy	Easy	Immediate
ATV	No	No	Easy	Easy	Fast
ANLTV	Yes	No	Easy	Easy	Fast
HNIPM	No	Staircasing	Immediate	Immediate	Fast
PRBF	No	Hallo	Easy	Unnecessary	Very Fast
PURE-LET	No	Star	Hard	Easy	Fast
MS-VST	Yes	Blocky	Hard	Easy	Fast
Adaptive VST-based methods	Yes	Yes/No†	Hard	Easy	Fast

Table 1: Advantages and disadvantages of Poisson denoising methods

* Execution time by method that was used in their proposed works.
† This depends on selection of Gaussian denoising methods.

https://doi.org/10.31449/inf.v43i2.1617 Informatica **43** (2019) 161–175 **161**

On the Properties of Epistemic and Temporal Epistemic Logics of Authentication

Sharar Ahmadi and Mehran S. Fallah
Department of Computer Engineering and Information Technology
Amirkabir University of Technology (Tehran Polytechnic), Hafez Ave., Tehran, Iran
E-mail: sharar.ahmadi@aut.ac.ir, msfallah@aut.ac.ir

Massoud Pourmahdian
Department of Mathematics and Computer Science
AmirKabir University of Technology (Tehran Polytechnic), Hafez Ave., Tehran, Iran
E-mail: pourmahd@ipm.ir

Overview paper

Keywords: epistemic logic, temporal epistemic logic, formal verification, authentication protocol

Received: May 2, 2017

The authentication properties of a security protocol are specified based on the knowledge gained by the principals that exchange messages with respect to the steps of that protocol. As there are many successful attacks on authentication protocols, different formal systems, in particular epistemic and temporal epistemic logics, have been developed for analyzing such protocols. However, such logics may fail to detect some attacks. To promote the specification and verification power of these logics, researchers may try to construct them in such a way that they preserve some properties such as soundness, completeness, being omniscience-free, or expressiveness. The aim of this paper is to provide an overview of the epistemic and temporal epistemic logics which are applied in the analysis of authentication protocols to find out how far these logical properties may affect analyzing such protocols.

Povzetek: V preglednem prispevku je prestavljena epistemska in časovna epistemska logika overitvenega postopka z namenom izboljšave delovanja.

1 Introduction

The principals communicating in a network need to be assured that they are sending/receiving messages to/from the intended principals as otherwise an attacker may impersonate an authorized principal and gain access to confidential information. To prevent this, the principals use authentication protocols, which are built on cryptography, for exchanging messages [13]. Since there are many successful attacks on authentication protocols [47, 60, 49, 35, 37, 33], different formal sytems have been developed for analyzing such protocols. Many of these systems are logical and are known as logics of authentication [14, 8, 7, 36, 38].

The first formal system designated for the specification and verification of authentication protocols is an epistemic logic - called BAN [14]. Although BAN can safely verify some protocols, it does not verify some other ones successfully, e.g., it proved that the Needham-Schroeder Public Key protocol (NSPK for short) was secure but later it was shown that NSPK was vulnerable to man-in-the-middle attack [46]. To promote the verification power of BAN, some extensions of it have been developed [27, 3, 60, 62, 61, 17, 4]. Moreover, researchers have developed some other logics of authentication that are not BAN-like, but are inherited from standard logics. Many of these logics are epistemic and temporal epistemic ones that can model different runs of a protocol or can be applied to investigate the knowledge acquired by principals at different instants in protocol runs [16, 45, 50, 52, 8, 53]. For example, a principal may find out who originated a received message at specific step of a protocol run and may agree with the sender on the received information.

There are also dynamic epistemic logics that are useful for modeling knowledge protocols, which model higher-order information and uncertainties in terms of agents' knowledge about each other. However, since these logics are inconvenient in a cryptographic setting for generating equivalence relations among messages, we do not consider them in this paper [21].

Although the proposed epistemic and temporal epistemic logics have significantly improved the analysis of authentication protocols, every now and then a problem is found and we need to improve the logics to solve that problem. For example, an attack may be detected by an omniscience-free logic while it is ignored by another logic that is not omniscience-free. Similarly, an authentication protocol can be specified by a temporal epistemic logic while it cannot be specified by a logic whose modalities are only epis-

temic ones. Such issues encourage researchers to find out if logics of authentication should preserve specific logical properties. The properties that are usually discussed in this context are soundness, completeness, expressiveness, and being omniscience-free. Moreover, since a powerful attacker is traditionally modeled as the well-known Dolev-Yao message deduction system [24], it is valuable to see if these logics can model such a system. In this way, if a logic of authentication proves a security goal about an authentication protocol, one can trust that the result is indeed valid in the presence of a powerful attacker who can eavesdrop all communications, drop, manipulate and replay messages, and perform cryptographic operations using his known keys and messages.

The aim of this paper is not to compare epistemic logics of authentication to alternative security protocol analysis, such as applied pi calculus and other process calculi, strands, multiset and other forms of rewriting. The aim of this paper is to provide an overview of the epistemic and temporal epistemic logics of authentication to find out how far some of their logical properties such as soundness, completeness, being omniscience-free, and expressiveness may affect analyzing authentication protocols. To do so, we discuss not only the conditions under which these logics support the Dolev-Yao message deduction, but also the logical properties that encourage us to trust the derived judgements about the authentication protocols.

The rest of the paper is as follows: In Section 2, we provide an overview of the notions of cryptography, Kripke semantics, and epistemic logics of authentication. In Section 3, we compare epistemic and temporal epistemic logics of authentication and show how far some of their logical properties may affect them in analyzing authentication protocols. Section 4 concludes the paper.

2 Basic notions

Authentication protocols are rules built on cryptographic primitives that help principals authenticate each other while communicating in a hostile environment [13]. An authentication goal can be expressed in terms of a knowledge notion, e.g., the sender authentication can be read as "the receiver knows the sender of a received message". Consider the NSPK protocol shown in Figure 1. Every principal in this protocol has a public key and a private key such that the public key of any principal A is known to everyone but only A has the corresponding private key.

In this protocol, principal A generates a nonce n_a, pairs n_a with its name A, encrypts $n_a.A$ with principal B's public-key $pk(B)$ so that only B can decrypt it by his private key, and sends $\{n_a.A\}_{pk(B)}$ to B. By receiving this message, B decrypts it and sends n_a back along with his nonce n_b in an encrypted message so that only A can decrypt it. Then, A sends n_b back to B. The goal of the NSPK protocol is that both A and B can be assured that they are talking to each other and not to an attacker. BAN logic proved that the NSPK protocol was safe [14], whereas Lowe showed that it was vulnerable to the man-in-the-middle attack [46]. Although such a result seems confusing, it is suggested by the well-known fact that the NSPK protocol is safe assuming that no compliant initiator will ever select a non-compliant responder for a session. Needham and Schroeder assumed this fact about the principals. However, it was certainly no longer a reasonable assumption when cryptographic protocols were beginning to be used on the open internet and Lowe outlined the man-in-the-middle attack.

The man-in-the-middle attack, shown in Figure. 1, consists of two interleaved sessions of the NSPK protocol. After A initiates a protocol run with I, the intruder I extracts the message, impersonates A, and sends n_a to B. When B replies, I forwards this message to A and misuses A to obtain n_b. Then, I sends n_b back to B. Thus, B is deceived to believe that he is talking to A while he is in fact communicating with I. This attack shows that the result of analyzing the NSPK protocol using BAN logic is questionable. Since the original BAN did not have formal semantics, finding such a semantics that could model the above attack became an important topic of research.

As said earlier, the formal analysis of an authentication protocol using epistemic logics depends on the knowledge gained by the principals executing that protocol. There are two main ways to formalize such knowledge. Assume the statement: "B has sent m", where the underlying semantics of a logic of authentication interprets this statement as follows: "B is engaging in an event of a protocol sending message m". If we formalize this statement with a logical formula ϕ, A knows ϕ means: "A knows that B has sent m". This is called propositional knowledge which is implicit and does not care about the details of computation [58]. There is also algorithmic knowledge formalizing the exact models of principals' knowledge such that if a principal has some bit strings, he can apply cryptographic operators to compute more strings using some predefined algorithms [30]. In this paper, we consider both of these knowledge formalizations, but first we need to explain some primitive notions.

Assume that θ is a set of principals exchanging messages by executing an authentication protocol. We may use a logical language $\mathcal{L}$ to specify not only the steps of such a protocol, but also the intended authentication properties that we want to prove about that protocol. To do so, we need to formalize exchanged messages as message terms in $\mathcal{L}$ because protocols are a type of messages passing multi-agent systems [26]. A message may be a plain term c or a compound one constructed by encryption or pairing such that $\{m\}_k$ is the encryption of message m with the key k and $m.m'$ is the pairing of messages m and m'. There is a need for a derivation system to derive new messages from known ones using cryptographic functions. In this paper, we use the well-known Dolev-Yao message deduction system [24] as follows: $m.m'$ is a message if and only if both m and m' are messages. If $\{m\}_k$ and k are messages, then so is m.

NSPK protocol	**man-in-the-middle attack**
$A \to B : \{n_a.A\}_{pk(B)}$	1. $A \to I : \{n_a, A\}_{pk(I)}$
$B \to A : \{n_a.n_b\}_{pk(A)}$	1′. $I(A) \to B : \{n_a, A\}_{pk(B)}$
$A \to B : \{n_b\}_{pk(B)}$	2′. $B \to I(A) : \{n_a, n_b\}_{pk(A)}$
	2. $I \to A : \{n_a, n_b\}_{pk(A)}$
	3. $A \to I : \{n_b\}_{pk(I)}$
	3′. $I(A) \to B : \{n_b\}_{pk(B)}$

Figure 1: NSPK protocol and the man-in-the-middle attack

Finally, if m and k are messages, then so is $\{m\}_k$. Given a set of message terms τ and a finite set of Dolev-Yao message deduction rules σ, we say that m is derivable from τ if either $m \in \tau$ or m is derivable from τ by applying the rules in σ. Assuming a set of message terms τ, there are two interpretations for knowledge.

The first interpretation says that a principal i knows a formula ϕ if he is aware of ϕ and ϕ is true in all the worlds he considers possible. In this case, a set of formulas, denoted by $\mathcal{A}_i(w)$, is associated to every possible world w such that i is aware of every formula in $\mathcal{A}_i(w)$ [31]. The intuition behind such an interpretation is that a principal needs to be aware of a formula before he can know it. For instance, a principal i may be aware of an encrypted message $\{m\}_k$ that he receives without being aware of message m. In this way, i may know that he receives $\{m\}_k$ if this message holds in all the worlds that are possible to him while he may not know that he receives m. In the context of verifying security protocols, $\mathcal{A}_i(w)$ is implemented as an algorithm that says "YES" for the formulas that agent i is aware of in his local state in w. In this way, we say that i knows ϕ explicitly using algorithmic knowledge [45].

The second interpretation says that a principal i knows a formula ϕ implicitly, shown by an epistemic formula $K_i\phi$, if i knows that ϕ is true. The set $\mathcal{F}$ of $\mathcal{L}$-formulas then, comprises not only atomic formulas about sending or receiving messages, but also compound formulas built inductively as follows: For every $\phi, \psi \in \mathcal{F}$, $i \in \theta$, and $m \in \tau$, we have $\phi \wedge \psi$, $\neg\phi$, $K_i\ \phi$, and $\mathcal{A}_i\phi$ are in $\mathcal{F}$.

The authentication protocols and goals formalized by formulas in $\mathcal{F}$ need to be interpreted in a proper formal semantics. Since an authentication protocol can be seen as a multi-agent system and it is known that an interpreted system [1] (IS for short) is a standard semantics for a multi-agent system, authentication protocols can be modeled by interpreted systems too. This can build a foundation for constructing Kripke semantics for epistemic logics of authentication as follows [26].

A Kripke model of an epistemic logic of authentication can reflect an authentication protocol. Such a model has a set of possible worlds that can be defined as $W = R \times \mathbb{N}$, where R is the set of all runs of that protocol and $\mathbb{N}$ is the set of natural numbers. Thus, a pair $\langle r, n\rangle$ - called a point - represents a run r at a time instant t. Such a point can be associated to a set of formulas that hold (are true) in that point. A Kripke model is then a tuple of the form $\mathcal{M} = \big(W, \{\sim_i\}_{i\in\mathcal{A}}, \pi\big)$ where W is the set of all possible points of the protocol. Moreover, the accessibility relation $\sim_i$ can be interpreted in different ways.

In one interpretation, for every $w_1, w_2 \in W$, $w_1 \sim_i w_2$ holds if and only if the local states of a principal i are the same in w_1 and w_2. For example, the local states of B at the end of both runs of the NSPK protocol shown in Figure. 1 are the same because B sends and receives the same messages by completing the execution of these two runs. In another interpretation, for every $w_1, w_2 \in W$, $w_1 \sim_i w_2$ holds if and only if the local states of a principal i are indistinguishable in w_1 and w_2. For example, assume that there is a protocol P such that $w_1 = \langle r_1, t_1\rangle$ and $w_2 = \langle r_2, t_2\rangle$ are two possible worlds of a Kripke model that reflects P. Principals A and B participate in two runs of P, denoted by r_1 and r_2, and formulas $A\ sends\ \{m\}_K$ and $A\ sends\ \{m'\}_{k'}$ hold in w_1 and w_2, respectively. Assume that B does not know the proper decryption keys to decrypt these messages, so he cannot distinguish formulas $A\ sends\ \{m\}_K$ and $A\ sends\ \{m'\}_{k'}$ because he sees $\{m\}_k$ and $\{m'\}_{k'}$ as two random messages. In this way, he considers both of the formulas the same. If all of the other formulas that hold in w_1 and w_2 are equal, B cannot distinguish between w_1 and w_2 even if $\{m\}_k \neq \{m'\}_{k'}$, i.e., we have: $w_1 \sim_B w_2$ [2]. In this model, $K_i\ \phi$ is true at $w \in W$ if ϕ is true at every $w' \in W$ that is accessible from w in A's view. Moreover, $\mathcal{A}_i\phi$ is true in $w \in W$ if ϕ can be computed by an awareness algorithm $\mathcal{A}_i$ in w. Such

[1] Assume that $\theta = \{i_1, \ldots, i_n, e\}$ is a set of principals such that "e" denotes a specific principal called the environment. For each $i \in \theta$, there is a finite set L_i of local states, a finite set a_i of local actions, and a local protocol $p_i : L_i \to 2^{a_i}$. The transition relation $t_i : L_i \times a_1 \times \ldots \times a_n \to L_i$ is then defined to return the next local state of i after all the principals perform their actions at the local state. Consider a set of global states $G \subseteq L_1 \times \ldots L_n \times L_e$, a set of joint actions $a = a_1 \times \ldots \times a_n \times a_e$, a joint protocol $p : (p_1, \ldots, p_n, p_e)$, and a global transition relation $t = (t_1, \ldots, t_n, t_e)$, which operates on global states by composing all local and environmental transition relations. An IS is then a tuple $\big(G, I_0, t, \{\sim_i\}_{i\in\mathcal{A}}, \pi\big)$, where G is the set of all global states accessible from any initial global state in I_0 via the transition relation t. For each $i \in \theta$, there is an accessibility relation $\sim_i \subseteq G \times G$ such that $g \sim_i g'$ if and only if $l_i(g) = l_i(g')$, where $l_i : G \to L_i(g)$ returns i's local state in the global state g, and $\pi : G \times Atom \to \{true, false\}$ is an interpretation function [26].

[2] There are also some other interpretations for the accessibility relation. We refer the interested reader to Ref. [17, 8].

an algorithm is defined specifically for every protocol and for computing intended formulas [30]. The truth of other non-atomic formulas is defined in a standard way and the atomic formulas are interpreted by the interpretation function π [15].

Assume that $\mathcal{M} = \big(W, \{\sim_i\}_{i\in\mathcal{A}}, \pi\big)$ is a Kripke model that models a protocol P, and $AuthR$ is an authentication requirement formalized by a logical formula ϕ. We say that ϕ is satisfiable with respect to $\mathcal{M}$ when there is a $w \in W$ such that ϕ is true in w, i.e., $AuthR$ holds in a run of P that is associated to w. We say that ϕ is valid with respect to $\mathcal{M}$ if ϕ is true in every $w \in W$ i.e. $AuthR$ holds in all runs of P. We discuss authentication and formalizing authentication in more detail below.

2.1 Formalizing authentication

Most of the security protocols have been designated for attaining authentication i.e. one principal should be assured of the identity of another principal. A protocol designer may assign different roles such as initiator, responder, or server to principals. Authentication protocols can be classified into two categories with respect to these roles: the protocols that try to authenticate a responder B to an initiator A, and the protocols that try to authenticate an initiator A to a responder B.

The notion of authentication does not have a clear consensus definition in the academic literature. However, the most clear and hierarchical definition for authentication has been devised by Lowe. In this definition, authentication requirements depend on the use to which the security protocol is put. These requirements can then be classified as aliveness, weak agreement, non-injective agreement, and agreement [46]. A protocol guarantees to a principal A "aliveness" of another principal B if the following condition holds: whenever the initiator A completes a run of the protocol, apparently with the responder B, then B has previously been running the protocol. Aliveness can be extended to "weak agreement" if B has previously been running the protocol with A. "Weak agreement" can be extended to non-injective agreement on a set of data items (where V is a set of free variables of the protocol) if B has previously been running the protocol with A, B was acting as responder in his run, and the two principals agreed on the values of all the variables in V. Weak agreement can be extended to "agreement" if each such a run of A corresponds to a unique run of B [46].

There are many attacks that occur due to parallel runs of a protocol [47]. The definition of weak agreement for authentication guarantees a one to one relationship between the runs of two principals as follows: a protocol authenticates a responder to an initiator, whenever a principal A starts j runs of the protocol as an initiator and l runs as a responder all in parallel; and completes $k \leq j$ runs of the protocol acting as initiator apparently with a responder B, then B has recently been running k runs acting as responder in parallel, apparently with A. Moreover, A protocol authenticates an initiator to a responder, whenever a principal B starts j runs of the protocol as a responder and l runs as an initiator, all in parallel; and completes $k \leq j$ runs of the protocol acting as responder, apparently with initiator A, then A has recently been running k runs acting as initiator in parallel, apparently with B [59]. In the following example, we explain the definition of agreement in more detail.

Example 2.1. *Consider the following challenge-response protocol that aims to authenticate an initiator A to a responder B, and to authenticate a responder B to an initiator A. In this protocol, k_{ab} is a shared key between A and B. Moreover, n_a and n_b are two nonces generated by A and B, respectively.*

$$A \to B : n_a$$
$$B \to A : \{n_a\}_{k_{ab}}.n_b$$
$$A \to B : \{n_b\}_{k_{ab}}$$

There is the following reflection attack on the protocol that consists of two sessions of the protocol executed in parallel. In this attack, B has the responder role and $I(A)$ denotes an intruder who impersonates A:

$$1.\ I(A) \to B : n_a$$
$$2.\ B \to I(A) : \{n_a\}_{k_{ab}}.n_b$$
$$1'.\ I(A) \to B : n_b$$
$$2'.\ B \to I(A) : \{n_b\}_{k_{ab}}.n'_b$$
$$3.\ I(A) \to B : \{n_b\}_{k_{ab}}$$

B starts two runs of the protocol as a responder to A, but it completes only one run (lines: 1, 2, and 3) with $I(A)$ while A does not participate in these runs. So, the protocol fails to aim the agreement requirement.

In the next example, we show how we can formalize an authentication requirement.

Example 2.2. *Consider the NSPK protocol, shown in Figure 1. We want to formalize the non-injective agreement authentication requirement. To do so, we use epistemic modalities as follows:*

$$K_B\ K_A\ msg\ n_a.n_b$$

This formula can be read as follows: "B knows that A knows the message $n_a.n_b$". If this formula can be proven for the NSPK protocol, then we say that the protocol guarantees "non-injective agreement" to B, where $\{n_a.n_b\}$ appears as the set of data items that the two principals agree on their value. Since B encrypts n_b with A's public-key and sends $\{n_a.n_b\}_{pk(A)}$ to A, whenever B receives a message containing n_b, he concludes that A has previously been running the protocol with B because A is the only principal who has A's private key to decrypt $\{n_a.n_b\}_{pk(A)}$ in order to extract n_b. The man-in-the-middle attack deceives B to believe that he is talking to A while he is in fact talking to I, who is an intruder. All BAN-like logics

proved the above formula for the NSPK protocol, whereas an omniscience-free epistemic BAN-like logic, that is referred to WS5 throughout this paper, could identify this insider attack [17]. In fact, being omniscience-free enabled WS5 to model the Dolev-Yao message deduction properly. We will explain this logic in detail at next sections.

3 Logical properties

In this section, we investigate how far some properties of epistemic and temporal epistemic logics of authentication may affect the analysis of authentication protocols. The properties that we investigate are soundness, completeness, being omniscience-free, and expressiveness.

3.1 Soundness and completeness

Beside syntax and semantics, every logic may have a proof system $\mathcal{X}$ consisting of some axioms and rules where the axioms are valid with respect to the logic's semantics and the rules preserve validity i.e. if the premise of a rule is valid, the result of it is also valid. Let $\mathcal{X}$ be a proof system of a logic of authentication that is based on the Dolev-Yao deduction system. Moreover, let the following statement be an authentication property: "principals i and j know that they are talking to each other", where i and j are engaging only in one session and both peers has received certain messages as common knowledge to authenticate each other. In this case, proving ϕ in $\mathcal{X}$ means that i and j know that they are indeed talking to each other even in an environment where there are attackers who can derive messages due to the Dolev-Yao deduction system.

The proof system $\mathcal{X}$ may have some interesting properties, two of which are soundness and completeness: $\mathcal{X}$ is sound if every derivable formula ϕ in $\mathcal{X}$ is also valid. $\mathcal{X}$ is complete if every valid formula ϕ is provable in $\mathcal{X}$. This is also called "weak completeness" by some researchers [15]. Logical analysis of security protocols relies on formal models of cryptography where cryptographic operations and security properties are defined as formal expressions. Such models ignore the details of encryption and focus on an abstract high-level specification and analysis of a system [1, 14, 24, 28].

Proving the soundness and completeness of a logic of authentication gives a strong intuition that the formal semantics of that logic is defined properly and it is working as expected. So, the logic can be applied safely in analyzing authentication protocols. For formal verification of a security protocol, there is a need for a formal model to reflect that protocol appropriately i.e. there is a need for a sound formal model for that protocol. Using a logical model, the verification is then dependent on the following parameters: first, the protocol must be described in the language of the logic. This description will be a part of a trust theory which consists of correct and acceptable propositions used in deducing security requirements. Even with a bad description of a protocol and its initial assumptions, the logic should consider all possible runs of that protocol.

To discuss the second parameter for defining a sound formal model, we first provide an overview of the Dolev-Yao indistinguishability relation. The intuitive idea for defining this relation is the fact that two messages are indistinguishable if any test - based on a limited set of operations on messages - gives the same result about the configuration of those messages. The Dolev-Yao indistinguishability relation can be related to cryptographic computing models. In this case, a formal model is said to be computationally sound. This is expressed for static equivalence, which is a general form of indistinguishability, explicitly: two local states are static equivalents if they satisfy the same equivalence tests. For a given theory of equation, static equivalence is based on a computable efficient set of operations such as symmetric and asymmetric encryption and decryption. For example, consider the simplest equivalence theory satisfying an equation of the form $dec(enc(m, pk), pr) = m$, where pk, pr, and m are a public key, a private key, and a message, respectively. Moreover, enc is an asymmetric encryption operator that encrypts m with pk and dec is an asymmetric decryption operator that decrypts an encrypted message with pr [2].

There is also another parameter for defining a sound formal model. Assume that there is a specification of an authentication protocol P and some initial assumptions using a logic of authentication L. We need to show whatever is deduced in L about P should be consistent with what the principals involved in executions of P actually infer. Assume that Γ is a finite set of logical formulas including the specification of P and its initial assumptions. Moreover, assume that the desired security goal is formalized by a formula ϕ that can be proven by applying the formuals in Γ and the axioms and rules of L's proof system. If L is logically sound and $\mathcal{M}$ is its model that satisfies the formulas in Γ, $\mathcal{M}$ also satisfies ϕ. If we show that $\mathcal{M}$ considers all possible runs of P, including those that attackers may participate in, the model reflects P properly.

As said earlier, the other theorem that is usually investigated for every logic is completeness. All valid formulas of a complete logic are also provable in its proof system. This motivates researchers to provide provers for the analysis of security protocols [22, 52, 51, 29]. If such a logic is also sound, the derived statements are more trusted since completeness shows that the formal semantics work as expected. Completeness may be a result of another property. As an example, the completeness of BAN-like logics has been an open problem for many years because some of them do not have any formal semantics and some other ones have inaccurate formal semantics. So, the logics could not model some possible runs executed by a Dolev-Yao attacker. However, it has been shown that the completeness of BAN-like logics can be proven by presenting a formal semantics that avoids logical omniscience [18]. We will discuss logical omniscience in more details later.

There is also another line of research that proves com-

pleteness for monadic fragments of a first-order temporal epistemic logic with respect to their corresponding classes of quantified interpreted systems. Such systems may have the following typical properties: synchronicity, perfect recall, no learning, and unique initial state. In contrast to most of the logics of authentication, such a logic has some axioms and rules that explore the relationship between its time and knowledge modalities [5, 6].

3.2 Logical omniscience

The semantics of a logic of authentication can be defined based on the standard Kripke structure. Such a semantics may lead to the logical omniscience problem where principals know all logical truths i.e. they know all consequences of what they know [31]. In fact, the problem bypasses the limitations placed on the knowledge of a principal who receives an encrypted message but does not have the right key to decrypt that message. Assume that L is an epistemic logic of authentication, which has a formal semantics built on the standard Kripke structure, and Γ is a set of logical formulas in L. Moreover, assume that a formula ϕ can be derived from Γ in L's proof system and an agent i knows all of the formulas in Γ. Then, the formal semantics of L leads to the logical omniscience where i knows ϕ. This fact is an immediate result of the interpretation of the knowledge modality of L with respect to the underlying standard Kripke semantics. In this way, a formula $K_i\phi$ is true in a state (possible world) w if and only if ϕ is true in every state that is indistinguishable (accessible) from w in i's view.

For example, assume that the following formula is true in all possible runs of a protocol,

$$i\ sent\ msg\ \{m\}_k \rightarrow submsg(m)$$

where $submsg(m)$ is read as m is a sub-message i.e. m is a sub-message of $\{m\}_k$. Using the standard Kripke semantics and for every principal j, the following formula is also true in all runs of that protocol:

$$K_j\ i\ sent\ msg\ \{m\}_k \rightarrow K_j\ submsg(m)$$

Now, assume that the anonymity of i fails and the formula $K_j\ i\ sent\ msg\ \{m\}_k$ is valid. Therefore, $K_j\ submsg(m)$ can be deduced by applying modus ponens. But in fact, this judgment is true only if j knows the symmetric key k to decrypt $\{m\}_k$. Thus, logical omniscience should be avoided in order to restrict principals' knowledge to what they can compute from their known facts, messages, and keys.

There are different approaches for solving the logical omniscience problem in the analysis of security protocols. In Ref. [19], the problem is solved by presenting a generalized Kripke semantics based on a permutation-based IS. Such a semantics results in a weakened necessitation rule for a logic faithful to BAN. Hence the logic becomes an omniscience-free weakened $S5$. Such a logic formalizes an implicit form of knowledge that results in abstract high-level reasoning of security protocols [17]. The logical omniscience problem can also be avoided by exact models of knowledge that a principal acquires during protocol runs. For example, such models are applied by a logic - called TDL [45]. So, a part of the logic that links epistemic modalities to awareness algorithms becomes omniscience-free. In this way, a principal knows a fact if he is aware of that fact. The idea of using awareness algorithms in formal security was originated in Ref. [30]. We will talk about the logics that use such algorithms at the end of this section.

3.3 Expressiveness

Epistemic logics of authentication usually have three different operators besides the standard ones of propositional logics. These operators are temporal, epistemic, and awareness. Hybrid systems may also have some other operators such as type operators or algebraic operators, but we do not consider hybrid systems in this paper and refer the interested reader to Ref. [39]. Temporal modalities formalize precedence of actions, time intervals, etc., such as "next" and "in a time interval $[t_1, t_2]$". Epistemic modalities formalize the knowledge of principals. Awareness operators show the algorithms that principals use to become aware of facts. The expressiveness of epistemic logics of authentication relies on their logical order and modalities. Moreover, if a logic has temporal operators, its expressiveness also relies on the method that the epistemic core is augmented by temporal modalities.

There are three approaches for adding temporal modalities to an epistemic core of a logic of authentication. The first approach, which makes the resulting temporal epistemic logic very expressive, is the fusion approach where epistemic and temporal modalities may appear in each other's scope without any restrictions. Moreover, the resulting logic may have axioms and rules that explore interactions between time and knowledge [5, 8, 23, 4]. This approach has been used in developing logics applied in analyzing a wide range of security protocols. Two examples of these protocols are classical authentication protocols such as NSPK and stream authentication protocols such as TESLA [57], which is used for sending streams of messages: videos, audios, etc.

The second approach is to use fibring technique where temporal and epistemic modalities may appear in each other's scope without any restrictions. But, the time and knowledge dimensions of a fibred logic are orthogonal. So, such a logic has no axioms and rules to explore the relationship between time and knowledge. Fibring technique does not model the knowledge which is obtained as a consequence of a particular event. Thus, a fibred logic is less intuitive for modeling security protocols and less expressive in comparison with other logics built on the fusion approach. However, theorems such as soundness and completeness may be easily proven for a fibred logic if its constituent logics are sound and complete. Moreover, a prover can be easily constructed for a fibred logic if its constituent logics have provers. Such a logic has been developed for

the analysis of the TESLA protocol [56].

Finally, the last approach for adding temporal modalities to an epistemic core of a logic of authentication is using the temporalization technique. This technique operates in a hierarchical way such that the temporal modalities can never appear in the scope of epistemic modalities i.e. the resulting logic does not have any formula of the form $K_i \bigcirc \phi$, which can be read as follows: agent i knows that at the next step ϕ holds. This approach has been used for verifying different protocols such as TESLA and WMF [53, 52].

In Figure 2, we compare some important epistemic and temporal epistemic logics of authentication against the above properties where every row of the figure is dedicated to a specific logic. The 1^{st} column of each row shows the logic name. The 2^{nd} column shows if the logic order is "propositional" or "first-order", denoted by "PR" and "FO", respectively. The 3^{rd} column shows the type of operators used in the logic where "E", "A", and "T" denote "epistemic", "awareness", and "temporal", in order. The 4^{th} column shows if the logic has a proof system, model checker, tableau .... The 5–7^{th} columns show if the logic is sound, complete, or omniscience-free, respectively. If a logic is sound, complete, or omniscience-free, we show this by a "✓" symbol. If any of these properties does not hold, we show this by a "×" symbol. If a logic does not have a proof system, it has no soundness and completeness theorems. In this case, we use a "−" symbol. Finally in the last column, "EXP" denotes that the explicit part of the logic is omniscience-free. Some of the security protocols analyzed by these logics are shown in Figure 3. The attacker models of these logics are also summarized in Figure 4. In what follows, we explain the above logical properties in more detail.

3.4 Inside the logics

In 1989 BAN logic was proposed as the first formal system for the specification and verification of authentication protocols [14]. This is a simple intuitive propositional epistemic logic named after its developers Burrows, Abadi, and Needham. The syntax of BAN consists of inference rules about principals' beliefs and their actions. This syntax enables BAN not only to specify the steps of an authentication protocol and its security goals, but also to derive the intended goals about that protocol. The first step of applying BAN is to idealize a protocol into an abstraction. In the second step, one should translate the initial assumptions and the security goals into BAN language, relate each idealized step of the protocol to a BAN formula, and then use BAN inference rules to derive intended goals.

The soundness and completeness theorems cannot be proven for BAN because it has no formal semantics. This logic has only epistemic modalities and it is not expressive enough to specify or verify highly time-dependent protocols such as stream authentication protocols [29]. This, along with BAN's propositional order, results in its low specification power. BAN has no formal attacker model but the capabilities of attackers are somehow embedded in its proof system.

For example, BAN has a rule - called the message-meaning rule - which has two premises. Assume that P and Q are two agents, m is a message, k is a key, and $\{m\}_k$ denotes that m is encrypted with k. The first premise of this rule says that P believes that k is a shared key between P and Q. This is formalized as follows: $P\ Believes\ P \leftrightarrow^k Q$. The second premise says that someone has sent a message which contains $\{m\}_k$ to P. This is shown as follows: $P\ sees\ \{m\}_k$. The conclusion of this rule is that P believes that at some time Q sent a message which contains m. This is formalized by the following formula: $P\ Believes\ Q\ said\ \{m\}_k$. As k is a shared key between P and Q, only these two agents can use k to encrypt m and no other agent can create $\{m\}_k$. Thus, when P receives $\{m\}_k$ it concludes that at some time Q has sent a message which contains $\{m\}_k$. In this way, even if an attacker has sent a message containing $\{m\}_k$ to P, P believes that this message has not been originated by Q [14]. As an example of formalizing authentication in BAN, we say that authentication is complete for the NSPK protocol if there are nonces n_a and n_b, generated by A and B, respectively, such that the following statements hold:

$$A\ Believes\ A \leftrightarrow^{n_a.n_b} B$$
$$B\ Believes\ A \leftrightarrow^{n_a.n_b} B.$$

The above formalizations can be classified as non-injective agreement. However, other weaker formalizations can be presented too.

As said earlier, BAN has some problems while verifying authentication protocols. Many extensions of this logic have been developed to resolve its problems. One of these extensions is GNY developed in 1990 for verifying a wider range of authentication protocols. GNY emphasizes separating the content and meaning of messages while it follows the same method as BAN for formalizing authentication. This logic is named after its developers Gong, Needham, and Yahalom. Although GNY can be applied in verifying a sample voting protocol successfully, the logic still suffers the same problems as its predecessor. Neither BAN nor GNY preserves the properties discussed in Subsections 3.1, 3.2, and 3.3. Thus, their derivations are not trustworthy. Moreover, these logics cannot analyze highly time-dependent protocols such as the TESLA protocol.

The first attempt for developing a formal semantics for BAN was in 1991 when Abadi and Tuttle improved the syntax and inference rules of BAN and also presented a formal semantics - called AT - for BAN [3]. This semantics is based on the standard Kripke structure constructed from interpreted systems. In this model, a principal i knows a formula ϕ if we have: "i knows ϕ" is true at a point $\langle r, k\rangle$ if it is true in every point $\langle r', k'\rangle$ that is indistinguishable from $\langle r, k\rangle$ in i's view. This extension of BAN - called AT logic - is sound with respect to the AT semantics. Thus, the proofs in this logic are more trustworthy. However, the completeness of AT remained an open problem for years

Logic	Order	Operators	Prover, Model Checker, ...	Sound	Ccomplete	O-free
BAN [14]	PR	E	proof system	–	–	×
GNY [27]	PR	E	proof system	–	–	×
AT [3]	PR	E	proof system	✓	×	×
TBAN [60]	PR	E/T	proof system	✓	×	×
VO [62]	PR	E	proof system	–	–	×
SVO [61]	PR	E	proof system	✓	–	×
WS5 [17]	PR	E	proof system	✓	✓	✓
FWS5 [17]	FO	E	proof system	✓	✓	✓
TWS5 [4]	PR	E/T	proof system	✓	✓	✓
L_n^{KX} [30]	PR	E/A	knowledge algorithm	–	–	EXP
TDL [45]	FO	E/A/T	proof system	✓	✓	EXP
KL_n [23]	FO	E/T	resolution prover [22]	–	–	×
TBL [53]	FO	E/T	proof system	✓	✓	×
TML+ [41]	FO	E/T	tableau prover [51,52]	✓	✓	×
FL [56]	FO	E/T	KEM prover [29]	✓	✓	×
$ECKL_n$ [50]	FO	E/T	MCTK model checker	–	–	×
CTLK [8]	FO	E/T	MCMAS model checker [43,44]	–	–	×
CTLS5 [9]	FO	E/T	MCMAS model checker [43,44]	–	–	×
CTLKR [10]	FO	E/T	MCMAS-E model checker	–	–	×
ICTLK [12]	FO	E/T	MCMAS-S model checker [11]	–	–	×

Figure 2: Some Logics Applied in Analyzing Authentication Protocols

Protocol	Logics	Protocol	Logics
NSPK	WS5,KL_n,$ECKL_n$,CTLK,CTLS5,ICTLK	Mix	FWS5
WMF	BAN,TWS5,TML+,CTLK,CTLS5	Duck-Duck-Goose	WS5, L_n^{KX}
TESLA	TWS5,TDL,TBL,FL	M-TESLA	TWS5
KSL2	CTLK	ISO2PUCCF	CTLK
ISOSK2PU	CTLK	S-RPC	CTLS5
KSL	CTLS5	FOO e-voting	CTLKR

Figure 3: Some Protocols Analyzed by The Logics in Figure 2

due to the logical omniscience problem, which was an immediate consequence of its standard Kripke semantics [19]. Because of the logical omniscience problem, the logic bypasses the principals' restricted knowledge. Thus, AT is not enough for modeling different runs of a protocol. This logic follows BAN's method for formalizing security properties. However, it can also formalize such properties at specific points of protocol runs. For example, we may want to verify whether a formula such as $P \leftrightarrow^k Q$ is true at a specific point $\langle r, t\rangle$ of a protocol run or not.

The correctness of a security protocol highly depends on the evolving knowledge of principals communicating through the protocol steps while time is passing. In 1993, Syverson added temporal operators to BAN for the first time. We call this logic TBAN throughout this paper [60]. TBAN could verify a key distribution protocol that the previous BAN-like logics could not because they lacked temporal modalities and ignored a casual consistency attack on the protocol. TBAN is sound with respect to the AT semantics. Moreover, it is able to formalize temporal modalities and statements. Thus, the specification power of TBAN is more than its predecessors. This logic was a starting point in using both temporal and epistemic modalities for analyzing authentication protocols and later many other logics followed this approach [53, 4, 23, 22, 52, 56, 8]. For example, we can formalize authentication for NSPK as follows where the symbol $\Box$ is read as "always":

$$\Box\ A\ Believes\ A \leftrightarrow^{n_a.n_b} B$$
$$\Box\ B\ Believes\ A \leftrightarrow^{n_a.n_b} B.$$

Although TBAN is more expressive than its predecessors, it cannot analyze such protocols as M-TESLA, Mix, and Dual Signature protocols [4, 17] since it is not omniscience-free.

Contemporary to TBAN, van Oorschot followed another line of research and extended BAN to facilitate the verification of key agreement protocols [62]. The extended logic was named VO. Although BAN, GNY, and VO have proof

Logic	Attacker	Comments
BAN [14]	Implicit	It has no formal attacker model, but the capabilities of the attacker are somehow embedded in the proof system.
GNY [27]	Implicit	It considers the attacker similar to that one of BAN.
AT [3]	Implicit	It considers the attacker similar to that one of BAN.
TBAN [60]	Implicit	It can verify a protocol against an attack that needs temporal modalities to be formalized.
VO [62]	Implicit	It considers the attacker similar to that one of BAN.
SVO [61]	Implicit	It considers the attacker similar to that one of BAN.
WS5 [17]	Implicit	The underlying generalized Kripke semantics restricts the knowledge gained by the attacker.
FWS5 [17]	Implicit	It considers the attacker similar to that one of WS5.
TWS5 [4]	Implicit	It considers the attacker similar to that one of WS5, but it can also verify highly time-dependent protocols.
L_n^{KX} [30]	Explicit	It models the Dolev-Yao attacker by awareness algorithms.
TDL [45]	Explicit	It models the Dolev-Yao attacker by awareness algorithms.
KL_n [22]	Explicit	It specifies the attacker's capabilities by logical formulas.
TBL [53]	Implicit	It does not prove if it can model the defined attacker.
TML+ [52]	Implicit	The attacker capabilities are embedded in the proof system.
FL [56]	Implicit	It does not prove if it can model the defined attacker.
$ECKL_n$ [50]	Explicit	The attacker is defined similar to that one of Dolev-Yao.
CTLK [8]	Explicit	It models the Dolev-Yao attacker as an environment.
CTLS5 [9]	Explicit	The attacker model is similar to that one of CTLK.
CTLKR [10]	Explicit	The passive attacker links receipt provider and its vote.
ICTLK [12]	Explicit	The attacker model is similar to that one of CTLK.

Figure 4: The Attacker Model of The Logics in Figure 2

systems, we cannot analyze their soundness or completeness because they lack formal semantics. All of these BAN extensions were unified into a sound logic - called SVO [61] - whose axioms and rules were simplified. The completeness of BAN was finally proven in 2007 when a proper proof system and formal semantics were provided for this logic. GNY, AT, VO, and SVO have no formal attacker model, but the capabilities of the attacker are somehow embedded in the proof system. So, the attacker model of these logics is similar to that of BAN.

In 2005, it was shown that the AT semantics could not identify some possible attacks because of the logical omniscience problem. To solve this problem, Cohen and Dam provided a generalized Kripke semantics for BAN such that BAN's soundness, completeness, and decidability were proven [19, 18]. Using this semantics, BAN can be embedded into an S5 logic where some specific message permutations are defined over messages. In this way, a formula $K_i\phi$ is true in a possible world w if for every possible world w' of the model which is indistinguishable from w in i's view and with respect to a message permutation ρ, $\rho(\phi)$ is true in w'. The formula $\rho(\phi)$ is the one in which every message m is replaced by $\rho(m)$ [19].

Such a generalized Kripke semantics results in a weak necessitation rule for BAN. In this way, the application of a weak necessitation rule along with axiom K does not lead to logical omniscience in the derivations. So, the underlying semantics restricts the knowledge gained by an attacker. We call this logic Weakened S5 - WS5 for short - throughout the paper. WS5 can be extended by first-order quantifiers. The resulting logic is a sound and complete first-order logic [20], denoted by FOWS5 in this paper. It is shown that these logics can safely specify and verify the Mix protocol since they are omniscience-free. However, they do not have temporal modalities to analyze such protocols as stream authentication.

WS5 can also be extended by temporal modalities. The resulting logic is called TWS5. This logic successfully verifies a modified TESLA protocol - called M-TESLA - which cannot be analyzed by the previous temporal epistemic logics that are not omniscience-free [4]. However, these logics make use of message permutation functions,

which cause exponential run time [39]. Although WS5 and its extensions are sound, complete, and omniscience-free, the expressiveness of WS5 is less than its extensions since it is propositional and does not have temporal modalities.

WS5, FWS5, and TWS5 use different symbols to formalize security properties. While WS5 can use epistemic modalities, FWS5 and TWS5 can make use of quantifiers and temporal modalities along with epistemic modalities, respectively. As an example, assume that we want to formalize a message sending axiom in TESLA which says that if the sender S sends a message M to the receiver R, then R may receive this message in time interval [u, v] [4]. This can be done as follows where $next^u$ denotes u clock ticks later:

$S\ sends\ M\ \rightarrow\ (\bigcirc^u\ R\ receives\ M) \vee \ldots \vee (\bigcirc^v R\, receives\, M)$.

It is proven that the underlying generalized Kripke semantics of WS5 restricts knowledge gained by an attacker because the message permutation functions make the logic omniscience-free. In fact, it is shown that the Dolev-Yao deduction system reflects the semantics of WS5 implicitly because such a semantics considers all of the possible runs of a protocol in the formal model by applying the permutation functions on messages even those runs executed by a Dolev-Yao attacker [17, 4].

There are also many standard logics that were not originally developed for analyzing authentication protocols, but they are well adapted to this purpose. One of these logics is L_n^{KX} proposed by Halpern and Pucella in 2003 [30]. This logic uses two kinds of knowledge: implicit knowledge which is similar to that of BAN-like logics and explicit knowledge which links to some knowledge algorithms. The attacker model is defined based on the explicit knowledge in L_n^{KX}. In fact, L_n^{KX} defines attackers as the Dolev-Yao deduction system explicitly using a Dolev-Yao knowledge algorithm. We refer the interested reader to Ref. [30] to review the full algorithm.

Explicit knowledge prevents logical omniscience. Thus, attackers infer the statements that they can compute. It has been proven that algorithmic knowledge can model the Dolev-Yao deduction system. This model is a useful abstraction because it does not consider the cryptosystem used in the protocol and it can easily capture probability for guessing appropriate keys. L_n^{KX} does not have a proof system. Thus, it does not have soundness and completeness theorems. However, a proof system may also be developed for it. This logic is flexible and modular. Moreover, it can be extended with probabilities to guess keys [25, 32] according to Lowe's model for guessing attacks [49]. L_n^{KX} cannot verify highly time-dependent protocols because it has no temporal modalities. This logic also uses implicit knowledge to model principals' beliefs about what is happening during a protocol run.

Lomuscio and Wozan followed the same approach to develop a temporal epistemic logic called TDL for the specification and verification of TESLA [45]. The logic not only has traditional knowledge modalities but also has awareness operators to represent explicit knowledge. Thus, a part of TDL that links to explicit knowledge is omniscience-free. The logic defines attackers as the Dolev-Yao deduction system explicitly. This is formalized through a derivation relation that shows how an attacker can extract a message from a set of received messages and keys using admissible operations. As an example, there is a derivation rule as follows: if an agent i derives $m.m'$ using the Dolev-Yao knowledge algorithm, denoted by an awareness operator $\mathcal{A}_i^{DY}$, i can also derive m using this algorithm. TDL has a computationally-grounded semantics. Moreover, it is intuitive, sound, complete, and decidable. TDL has a high specification power because it is a first-order modal logic that has three types of modalities: epistemic, awareness, and temporal.

In 2004, Dixon, Fernandez Gago, Fisher, and van der Hoek introduced a first-order temporal logic of knowledge - called KL_n - for the specification and verification of authentication protocols and verified NSPK as a case study [23]. This logic is useful for reasoning about the evolving knowledge of a principal over time. Especially, this is important if we want to be sure that a principal obtains certain knowledge at a time instant. KL_n is a fusion of a linear time temporal logic and a multi modal epistemic logic S5. Thus, the logic is powerful enough to specify agents' capabilities explicitly by logical formulas. As an example of using quantifiers and modalities of KL_n for formalizing security requirements, consider the NSPK protocol as shown in Figure 1. Assuming that the predicate $value - nonce(N, V)$ of the logic indicates that the value of nonce N is V, we want to show that every other principal, other than A and B, who are running the protocol can never know the value of A's nonce. This is formalized as follows:

$$\forall\, V\ \ \Box\neg K_C\ value - nonce(N_a, V)$$

where $\Box$ and K_C are two modalities of KL_n that are interpreted as "always" and "agent C knows", respectively.

To prove a security goal ϕ from protocol specification ψ, where both ϕ and ψ are formulas in KL_n, we must prove $\psi \rightarrow \phi$. To do so, a resolution method is applied which is in fact a refutation system showing that $\psi \wedge \neg\phi$ is not satisfiable. A prototype theorem prover has been developed for a single modal version of temporal logics of knowledge but the developers insisted on a need to develop a more powerful prover to deal with the multi-modal case in order to prove theorems automatically [23].

Contemporary to KL_n, Liu, Ozols, and Orgun developed the TML+ logic for analyzing authentication protocols [41]. The logic uses the temporalization technique to combine an epistemic logic - called TML - with a simple linear-time temporal logic - called SLTL. This technique allows adding SLTL to TML in a hierarchical way such that the temporal operators can never appear in the scope of epistemic ones. TML+ can be applied for the verification of time-dependent properties of security protocols. To do so, a trust theory for a protocol that is to be verified is provided. The theory consists of TML+ axioms and rules along with

a specification of the protocol and initial assumptions in the form of TML+ formulas. Then, the theory is applied to drive security goals. TML+ is sound and complete, yet it is not omniscience-free. However, the logic assumes that an attacker cannot decrypt an encrypted message if he does not have the right key for decryption.

In 2007, a tableau system was developed for TML+ [51]. This was used for verifying both static and dynamic aspects of some protocols such as WMF, Needham-Schroeder symmetric key, and Kerbros. The developers proved the soundness and completeness of the labelled tableau calculus based on the soundness and completeness results of the constituent logics of TML+. Then, they sketched a resolution-type proof procedure and proposed a model checking algorithm for TML+ [51].

In 2006, Orgun, Ji, Chuchang, and Governatori constructed the FL logic for analyzing stream authentication protocols from TML and SLTL using the fibring technique [56]. In this way, FL is sound and complete with respect to its fibered model because its constituent logics are proven to be sound and complete [40, 34]. The original idea of the fibring technique is based on the assumption that both constituent logics are endowed with point-based semantics so that a model of the fibred logic is a point-based semantics and every point of the fibred model is closely related to a point in a model of each constituent logic. Thus, a model of a fibred logic can be related to several models in each of the constituent logics. As a consequence, a fibred logic preserves the theorems proven for its original logics [42]. The FL fibred model is such that the formulas of FL may contain any number of temporal and belief operators without any restrictions. Any formula whose main operator is a temporal modality, is interpreted by referring to the SLTL semantics and any formula whose main operator is a belief modality, is interpreted by referring to the Kripke semantics of TML.

The fibring technique also allows developing proof procedures from the constituent logics since the time and knowledge dimensions of FL are orthogonal[29]. In 2008, a modal tableau was developed for FL[29] by adapting KEM, a modal tableau system, showing how combinations of multi-modal logics can provide an effective tool for verifying the TESLA protocol. It has been proven that the adapted KEM is sound and complete for SLTL and TML. As a result of the fibring technique, these theorems have also been proven for FL. KEM can be used to automatically check for formal properties of security protocols [29].

The logic FL is more expressive than a temporalised belief logic such as TML+ [52] because temporal and belief modalities may appear in each other's domains. This logic is flexible because of its modular nature and it has a high specification power since it is a first-order logic making use of temporal and knowledge modalities. However, it is not omniscience-free because of the standard Kripke semantics of its belief aspect. In 2012, Ma *et al.* constructed a temporalized belief logic - called TBL [53] - that was less expressive than FL because of using the temporalization technique. However, it was appllied in verifying the TESLA protocol successfully. TBL is sound and complete, yet it is not omniscience-free. This logic follows FL to define the attacker's capabilities where they are defined similar to the Dolev-Yao model. However, neither of these two logics prove if they can model such attackers.

In 2009, Boureanu, Cohen, and Lomuscio presented an effective fully automatic approach for analyzing security protocols in the presence of the Dolev-Yao attackers. This approach makes use of a temporal epistemic logic, called CTLK [8]. The first step is to specify a security protocol in CAPSL [54], which is a high-level specification language formally describing protocols. Then, the specifications of the security protocol and security goals in CAPSL are translated by an automatic compiler into an ISPL (Interpreted Systems Programming Language), which is a set of CTLK specifications to be checked. The result of this translation is a file that a MCMAS model checker[3] takes as input and checks whether or not the security goals are satisfied by the protocol. In most cases, this gives a countermodel if they are not satisfied by the protocol. The main contribution is a compiler from protocol descriptions given in CAPSL into ISPL, the input language for MCMAS. The translation is optimised to limit the state explosion and benefit from MCMAS's various optimisations. To do so, the authors developed PD2IS (Protocol Descriptions to Interpreted Systems), an automatic compiler from CAPSL protocol descriptions to ISPL programs. This verification method assumes a bounded number of concurrent protocol sessions instantiated to run concurrently [8].

This CTLK approach makes use of the consistent message permutations introduced in Ref. [19] through a smart translation of security goals into CTLK formulas in order to prevent bad effects of the logical omniscience. However, CTLK is not omniscience-free. The approach models the Dolev-Yao attacker explicitly as an environment which has the capabilities of honest principals, can eavesdrop all communications, compose and replay messages into any protocol session, and perform cryptographic operators using his known keys. The attacker also has an identity and public and private keys that may be used to communicate with other principals and record their send actions. To avoid the state-explosion problem while model checking, MCMAS employs a fixed limited number of interleaved sessions. Moreover, it is assumed that the verification process does not support all types of nesting while encrypting messages and the messages are of finite-length. These assumptions are applied because an attack to a protocol may be usually found in a small run of the protocol consisting of only a few sessions [48].

No proof system has been presented for the CTLK logic yet. So, this logic does not have any proven soundness and completeness theorems [8]. However, one can develop a proof system with respect to the underlying semantics of CTLK and check if the logic is sound and complete accord-

[3]This is a BDD-based model checker for multi-agent systems supporting temporal-epistemic specifications [43]

ing to that proof system. CTLK is powerful and flexible enough to be extended. As a variant of this logic, CTLS5$_n$ has been developed to model-check detectability of attacks in multi-agent systems that are automatically generated for security protocols [9]. As another extension of CTLK, CTLKR has been developed for the analysis of an e-voting protocol - called Foo - with a passive attacker model who can link the receipt-providing principal and its vote [10]. Finally, ICTLK [12] has been developed for analyzing an unbounded number of concurrent protocol sessions. These extensions of CTLK use the same tools for verifying security protocols automatically. Contemporary to CTLK, Luo *et al.* provided a temporal epistemic logic - called ECKLn - that was applied in verifying the NSPK protocol using model checking techniques similarly. To do so, the authors implemented and automatically verified security protocols in a model checker for multiagent systems [9].

4 Conclusions

One of the main approaches for formal analysis of authentication protocols is using epistemic or temporal epistemic logics. The semantics of such logics should be able to model the protocol runs, including those executed by attackers. Then, theorem provers or model checkers are built on these logics to analyze the protocols against intended properties. Comparing the logics in Figure 2, it seems that WS5 and its extensions are the best choices for protocol analysis since they are sound, complete, and omniscience-free. However, Figures 3 and 4 show that CTLK and its extensions are very powerful in security protocol analysis, make use of automatic compilers and model checkers, and prevent bad effects of logical omniscience while modeling the Dolev-Yao message deduction.

It has been shown that it is an undecidable problem to find whether a security protocol is indeed secure or not [55]. Thus, it is practical to use trusted security protocols analyzed by different formal methods to provide fast solutions for existing problems of real life systems. At the same time, it is practical to move toward the best formal system for verifying authentication protocols. A good measure for finding such a formal system for security protocol analysis is to prove that it can model the Dolev-Yao message deduction. In this line, we investigated the epistemic and temporal epistemic logics and showed how far their properties such as soundness, completeness, being omniscience-free, and expressiveness may affect the analysis of authentication protocols. Some of these logics have no proof system, thus they have no soundness or completeness theorems. However, they may apply model checkers for analysis and use awareness algorithms for preventing the logical omniscience problem while modeling the Dolev-Yao message deduction explicitly [30, 45, 8]. Preventing the logical omniscience or avoiding its bad effects restricts a principal's knowledge to what he can compute or derive with his known keys and messages. Some other epistemic logics have proof systems, are omniscience-free, and model the attacker capabilities implicitly [19, 20, 4]. If such logics are logically sound, their derived judgments are more trustworthy. If they are complete, they may make use of automatic provers.

Comparing Figures 2 and 4, the epistemic logics modeling the Dolev-Yao message deduction are either omniscience-free or prevent the bad effects of omniscience. In fact, provers are usually built on logics that preserve properties such as soundness, completeness, and being omniscience-free so that one can trust their output. At the same time, model checkers deal with the state explosion problem by imposing some assumptions so that they can verify security protocols within an acceptable time. However, such assumptions may not stop us from using model checkers since they cover a wide range of security protocols and use existing tools. Finally, the expressiveness of such logics makes them powerful enough to both formalize attacker's capabilities by logical formulas and analyze different classes of authentication protocols. Specifically, if the logic has both temporal and epistemic modalities, it can analyze highly time-dependent protocols such as stream authentication protocols.

Although it has been shown that logics using algorithmic knowledge can model the Dolev-Yao message deductive explicitly [30], to the best of our knowledge it has not been proven if a temporal epistemic logic of authentication can model such an attacker implicitly using permutation-based generalized Kripke semantics. This can be a topic for further research. As seen in this paper, all of the logics that model the attacker capabilities implicitly are omniscience-free. Thus, this can be a starting point for this topic of research. Finally, it would be beneficial if such an overview extends to the use of epistemic and temporal epistemic logics in analyzing other security/privacy properties.

References

[1] M. Abadi and A.D. Gordon (1999) A calculus for cryptographic protocols: The spi calculus, *Information and computation*, Elsevier, 148(1): 1–70. https://doi.org/10.1006/inco.1998.2740

[2] M. Abadi and P. Rogaway (2002) Reconciling two views of cryptography (the computational soundness of formal encryption), *Journal of cryptology*, Springer, 15(2): 103–127. http://dx.doi.org/10.1007/s00145-007-0203-0

[3] M. Abadi and M.R. Tuttle (1991), A semantics for a logic of authentication, *Proceedings of the tenth annual ACM symposium on Principles of distributed computing*, ACM, 201–216. https://doi.org/10.1145/112600.112618

[4] S. Ahmadi and M.S. Fallah (2018) An Omniscience-Free Temporal Logic of Knowledge for Verify-

ing Authentication Protocols, *Bulletin of the Iranian Mathematical Society*, Springer, 44(5): 1–23. https://doi.org/10.1007/s41980-018-0087-9

[5] F. Belardinelli and A. Lomuscio (2010) Interactions between time and knowledge in a first-order logic for multi-agent systems, *Proceedings of the Twelfth International Conference on the Principles of Knowledge Representation and Reasoning*, AAAI Press, 38–48.

[6] F. Belardinelli and A. Lomuscio (2012) Interactions between time and knowledge in a first-order logic for multi-agent systems: completeness results, *Journal of Artificial Intelligence Research*, AI Access Foundation, 1–45. https://doi.org/10.1613/jair.3547

[7] B. Blanchet, B. Smyth, and V. Cheval (2015) ProVerif 1.90: Automatic Cryptographic Protocol Verifier, User Manual and Tutorial.

[8] , I. Boureanu, M. Cohen, and A. Lomuscio (2009) Automatic verification of temporal-epistemic properties of cryptographic protocols, *Journal of Applied Non-Classical Logics*, Taylor & Francis, 19(4): 463–487. https://doi.org/10.3166/jancl.19.463-487

[9] I. Boureanu, M. Cohen, and A. Lomuscio (2010) Model checking detectability of attacks in multiagent systems, *Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems*, International Foundation for Autonomous Agents and Multiagent Systems, 1: 691–698.

[10] I. Boureanu, A.V. Jones, and A. Lomuscio (2012) Automatic verification of epistemic specifications under convergent equational theories, *Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems-Volume 2*, International Foundation for Autonomous Agents and Multiagent Systems, 1141–1148.

[11] I. Boureanu, P. Kouvaros, and A. Lomuscio (2016) MCMAS-S- An experimental model checker or the verification of security properties in unbounded multi-agent systems. https://vas.doc.ic.ac.uk/software/mcmas-extensions/

[12] I. Boureanu, P. Kouvaros, and A. Lomuscio (2016) Verifying Security Properties in Unbounded Multiagent Systems, *Proceedings of the 2016 International Conference on Autonomous Agents & Multiagent Systems*, International Foundation for Autonomous Agents and Multiagent Systems,1209–1217.

[13] C. Boyd and A. Mathuria (2013)*Protocols for authentication and key establishment*, Springer Science & Business Media, 2013. https://doi.org/10.1007/978-3-662-09527-0

[14] M. Burrows, M. Abadi, and R.M. Needham (1989) A logic of authentication, *Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, The Royal Society, 426(1871): 233–271.

[15] C.C. Chang and H.J. Keisler (1990) *Model theory*, North Holland, 73.

[16] L. Chao, L. Hui, and M. Jianfeng (2009) Analysis the Properties of TLS Based on Temporal Logic of Knowledge, *Proceedings of the 5th International Conference on Information Assurance and Security*, IEEE, 2: 19–22. https://doi.org/10.1109/ias.2009.49

[17] M. Cohen (2007) Logics of Knowledge and Cryptography: Completeness and Expressiveness, PhD Thesis, KTH, Stockholm, Sweden.

[18] M. Cohen and M. Dam (2005) A completeness result for BAN logic, *Prococeedings of Methods for Modalities*, 4.

[19] M. Cohen and M. Dam (2005) Logical omniscience in the semantics of BAN logic, *Proceedings of the Foundations of Computer Security*, 121–132.

[20] M. Cohen and M. Dam (2007) A complete axiomatization of knowledge and cryptography, *Proceedings of the 22nd Annual IEEE Symposium on Logic in Computer Science*, IEEE, 77–88.

[21] F. Dechesne and Y. Wang (2010) To know or not to know: epistemic approaches to security protocol verification, *Synthese*, Springer, 177(1): 51–76. https://doi.org/10.1007/s11229-010-9765-8

[22] C. Dixon, M.C. Fernández Gago, M. Fisher, and W. van der Hoek (2007) Temporal logics of knowledge and their applications in security, *Electronic Notes in Theoretical Computer Science*, Elsevier, 186: 27–42. https://doi.org/10.1016/j.entcs.2006.11.043

[23] C. Dixon, M.C.F. Gago, M. Fisher, and W. Van Der Hoek (2004) Using temporal logics of knowledge in the formal verification of security protocols, *Proceedings of the 11th International Symposium on Temporal Representation and Reasoning*, IEEE,148–151.

[24] D. Dolev and A. Yao (1983) On the security of public key protocols, *Proceedings of the IEEE Transactions on Information Theory*, IEEE, 29(2):198–208.

[25] R. Fagin and J.Y. Halpern (1994) Reasoning about knowledge and probability, *Journal of the ACM*, ACM, 41(2):340–367. https://doi.org/10.1145/273865.274429

[26] R. Fagin, Y. Moses, J.Y. Halpern, and M.Y. Vardi (2003) *Reasoning about knowledge*, The MIT Press. https://doi.org/10.7551/mitpress/5803.001.0001

[27] L. Gong, R. Needham, and R. Yahalom (1990) Reasoning about belief in cryptographic protocols, *Proceedings of the IEEE Computer Society Symposium on Research in Security and Privacy*, IEEE, 234–248. https://doi.org/10.1109/risp.1990.63854

[28] A.D. Gordon and A. Jeffrey (2003) Authenticity by typing for security protocols, *Journal of computer security*, IOS Press, 11(4): 451–519. https://doi.org/10.3233/jcs-2003-11402

[29] G. Governatori, A.M. Orgun, and C. Liu (2008) Modal tableaux for verifying stream authentication protocols, *Journal of Autonomous Agents and Multi Agent Systems*. https://doi.org/10.1007/s10458-007-9027-4

[30] J.Y. Halpern and R. Pucella (2003) Modeling adversaries in a logic for security protocol analysis, *Formal Aspects of Security*, Springer, 115–132. https://doi.org/10.1007/978-3-540-40981-6_11

[31] J.Y. Halpern and R. Pucella (2011) Dealing with logical omniscience: Expressiveness and pragmatics, *Artificial intelligence*, Elsevier, 175(1): 220–235. https://doi.org/10.1016/j.artint.2010.04.009

[32] J.Y. Halpern and M.R. Tuttle (1993) Knowledge, probability, and adversaries, *Journal of the ACM*, ACM,40(4):917–960. https://doi.org/10.1145/153724.153770

[33] J. Heather, G. Lowe, and S. Schneider (2003) How to prevent type flaw attacks on security protocols, *Journal of Computer Security*, IOS Press, 11(2): 217–244. https://doi.org/10.3233/jcs-2003-11204

[34] G.E. Hughes and M.J. Cresswell (2012) *A new introduction to modal logic*, Routledge. https://doi.org/10.4324/9780203028100

[35] G. Jakubowska, P. Dembinski, W. Penczek, and M. Szreter (2009) Simulation of Security Protocols based on Scenarios of Attacks, *Fundamenta Informaticae*, 93(1): 185-203.

[36] A. Jurcut, T. Coffey, and R. Dojen (2013) Establishing and fixing security protocols weaknesses using a logic-based verification tool, *Journal of Communication*, 8(11): 795–806. https://doi.org/10.12720/jcm.8.11.795-805

[37] A.D. Jurcut, T. Coffey, and R. Dojen (2014) Design guidelines for security protocols to prevent replay & parallel session attacks, *computers & security*, Elsevier, 45: 255–273. https://doi.org/10.1016/j.cose.2014.05.010

[38] F. Kammuller and C.W. Probst (2015) Modeling and verification of insider threats using logical analysis, *IEEE Systems Journal*, IEEE, 11(2): 534–545. https://doi.org/10.1109/jsyst.2015.2453215

[39] S. Kramer (2007) Logical concepts in cryptography, PhD Thesis, École Polytechnique Fédérale de Lausanne.

[40] C. Liu and M. Orgun (1996) Dealing with multiple granularity of time in temporal logic programming, *Journal of Symbolic Computation*, Elsevier, 22(5): 699–720. https://doi.org/10.1006/jsco.1996.0072

[41] C. Liu, M.A. Ozols, and M. Orgun (2004) A temporalised belief logic for specifying the dynamics of trust for multi-agent systems *Advances in Computer Science-ASIAN 2004. Higher-Level Decision Making*, Springer, 142–156. https://doi.org/10.1007/978-3-540-30502-6_10

[42] C. Liu, M. Ozols, and M. Orgun (2005) A fibred belief logic for multi-agent systems, *AI 2015: Advances in Artificial Intelligence*, Springer, 29–38. https://doi.org/10.1007/11589990_6

[43] A. Lomuscio, H. Qu, and F. Raimondi (2009) MCMAS: A model checker for the verification of multi-agent systems, *Computer Aided Verification*, Springer, 682–688. https://doi.org/10.1007/978-3-642-02658-4_55

[44] A. Lomuscio, H. Qu, and F. Raimondi (2015) MCMAS: an open-source model checker for the verification of multi-agent systems *International Journal on Software Tools for Technology Transfer*, Springer, 19(1): 9–30. https://doi.org/10.1007/s10009-015-0378-x

[45] A. Lomuscio and B. Woźna (2006) A complete and decidable security-specialised logic and its application to the TESLA protocol, *Proceedings of the 5th international joint conference on Autonomous agents and multiagent systems*, ACM, 145–152. https://doi.org/10.1145/1160633.1160658

[46] G. Lowe (1995) An attack on the Needham-Schroeder public-key authentication protocol, *Information processing letters*, Elsevier, 56(3): 131–133. https://doi.org/10.1016/0020-0190(95)00144-2

[47] G. Lowe (1997) A family of attacks upon authentication protocols, epartment of Mathematics and Computer Science, University of Leicester.

[48] G. Lowe (1998) Casper: A Compiler for the Analysis of Security Protocols, *Journal of Computer Security*, IOS Press, 6(1-2): 53–84. https://doi.org/10.3233/jcs-1998-61-204

[49] G. Lowe (2004) Analysing protocols subject to guessing attacks, *Journal of Computer Security*, IOS Press, 12(1): 83–97. https://doi.org/10.3233/jcs-2004-12104

[50] X. Luo, Y. Chen, M. Gu, and L. Wu (2009) Model Checking Needham-Schroeder Security Protocol Based on Temporal Logic of Knowledge, *Proceedings of the NSWCTC'09. International Conference on Networks Security, Wireless Communications and Trusted Computing*, IEEE, 2: 551–554. https://doi.org/10.1109/nswctc.2009.384

[51] J. Ma, M.A. Orgun, and K. Adi (2011) An analytic tableau calculus for a temporalised belief logic, *Journal of Applied Logic*, Elsevier, 9(4): 289–304. https://doi.org/10.1016/j.jal.2011.08.003

[52] J. Ma, M.A. Orgun, and A. Sattar (2009) Analysis of authentication protocols in agent-based systems using labeled tableaux *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, IEEE, 39(4): 889–900. https://doi.org/10.1109/tsmcb.2009.2019263

[53] J. Ma and K. Schewe (2012) A Temporalised Belief Logic for Reasoning about Authentication Protocols, *Proceedings of the IEEE 11th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom)*, IEEE, 1721–1728. https://doi.org/10.1109/trustcom.2012.59

[54] J.K. Millen (1996) CAPSL: Common Authentication Protocol Specification Language, *Proceedings of the 1996 workshop on New security Paradigms (NSPW)*, ACM, 96: 132. https://doi.org/10.1145/304851.304879

[55] J. Mitchell, A. Scedrov, N. Durgin, and P. Lincoln (1999) Undecidability of bounded security protocols, *Proceedings of the Workshop on Formal Methods and Security Protocols*.

[56] M.A. Orgun, J. Ma, C. Liu, and G. Governatori (2006) Analysing stream authentication protocols in autonomous agent-based systems2nd IEEE International Symposium on, *Proceedings of the 2nd IEEE International Symposium on Dependable, Autonomic and Secure Computing*, IEEE, 325–332. https://doi.org/10.1109/dasc.2006.19

[57] A. Perrig, R. Canetti, J.D. Tygar, and D. Song (2000) Efficient authentication and signing of multicast streams over lossy channels, *Proceedings of the IEEE Symposium on Security and Privacy*, IEEE, 56–73. https://doi.org/10.1109/secpri.2000.848446

[58] R. Ramanujam and S.P. Suresh (2005) Deciding knowledge properties of security protocols, *Proceedings of the 10th conference on theoretical aspects of rationality and knowledge*, National University of Singapore, 219–235.

[59] R. Ramezanian (2015) Process Algebraic Modeling of Authentication Protocols for Analysis of Parallel Multi-Session Executions, *The ISC International Journal of Information Security*, Iranian Society of Cryptology, 1(1): 55-67. https://doi.org/10.22042/isecure.2015.1.1.6

[60] P.F. Syverson (1993) Adding time to a logic of authentication, *Proceedings of the 1st ACM conference on Computer and communications security*, ACM, 97–101. https://doi.org/10.1145/168588.168600

[61] P.F. Syverson and P.C. Van Oorschot (1994) On unifying some cryptographic protocol logics, *Proceedings of the IEEE Computer Society Symposium on Research in Security and Privacy*, IEEE, 14–28. https://doi.org/10.21236/ada465512

[62] P. van Oorschot (1993) Extending cryptographic logics of belief to key agreement protocols, *Proceedings of the 1st ACM Conference on Computer and Communications Security*, ACM, 232–243. https://doi.org/10.1145/168588.168617

Benchmark Problems for Exhaustive Exact Maximum Clique Search Algorithms

Sándor Szabó
Institute of Mathematics and Informatics, University of Pécs
Ifjuság utja 6, H-7624 Pécs, Hungary
E-mail: sszabo7@hotmail.com

Bogdán Zaválnij
Alfréd Rényi Institute of Mathematics, Hungarian Academy of Sciences
Reáltanoda u. 13–15, H-1053 Budapest, Hungary
E-mail: bogdan@renyi.hu

Keywords: clique, maximal clique, maximum clique, graph coloring, random graph, Mycielski graph

Received: February 12, 2019

There are well established widely used benchmark tests to assess the performance of practical exact clique search algorithms. In this paper a family of further benchmark problems is proposed mainly to test exhaustive clique search procedures.

Povzetek: Podanih je nekaj novih standardnih problemov za testiranje algoritmov za iskanje klik.

1 Preliminaries

Let $G = (V, E)$ be a finite simple graph. Here V is the set of vertices of G and E is the set of edges of G. The finiteness of G means that G has finitely many nodes and finitely many vertices, that is, $|V| < \infty$, $|E| < \infty$. The simplicity of G means that G does not have any loop and it does not have double edges.

Let C be a subset of V. If two distinct nodes in C are always adjacent in G, then C is called a clique in G. When C has k elements, then we talk about a k-clique. We include the cases $k = 0$ and $k = 1$ as well when $|C| = 0$ and $|C| = 1$, respectively. Though in these cases C does not have two distinct elements.

A clique is maximal in G if it is not part of any larger clique in G. In other words a clique is maximal clique of the graph G if it cannot be extended to a larger clique by adding a new node of the graph G. A k-clique is a maximum clique in G if G does not contain any $(k + 1)$-clique. All the maximum cliques of a graph G have the same number of nodes. We call this well defined number the clique number of G and we denote it by $\omega(G)$.

A number of problems is referred as clique search problems [1].

Problem 1. *Given a finite simple graph G. List all maximal cliques of G without repetition.*

Problem 2. *Given a finite simple graph G and given a positive integer k. Decide if G has a k-clique.*

Problem 3. *Given a finite simple graph G. Determine $\omega(G)$.*

Problem 4. *Given a finite simple graph G. List all maximum cliques of G without repetition.*

An algorithm to solve Problem 1 can be found in [2]. The algorithm is commonly known as Bron-Kerbosch algorithm. Obviously, the Bron-Kerbosch algorithm can be used to solve Problems 2, 3 and 4. A more efficient algorithm to solve these problems was first given in [3]. The algorithm is known under the name Caraghan-Pardalos algorithm. The Bron-Kerbosch and Carraghan-Pardalos algorithms are the classical algorithms that form the base of many further clique search procedures. These algorithms are presented in [14], [26], [24], [10], [9]. But this list is not intended to be complete.

The complexity theory of the algorithm tells us that Problem 2 is in the NP-complete complexity class. (See for instance [6].) Consequently, Problem 3 must be NP-hard.

We color the vertices of G such that the following conditions are satisfied.

(1) Each vertex of G is colored with exactly one color.

(2) Vertices of G connected by an edge receive distinct colors.

A coloring of the nodes of G that satisfies conditions (1), (2) is called a legal coloring or well coloring of the nodes of G.

Suppose that the nodes of G can be colored legally using k colors. We may use a map $f : V \to \{1, \ldots, k\}$ to describe a coloring of the nodes of G. The numbers $1, \ldots, k$ play the roles of the colors and $f(v)$ is the color of the vertex v for each $v \in V$. If for adjacent nodes u and v of G

the equation $f(u) = f(v)$ implies $u = v$, then the coloring defined by the map f is a legal coloring.

There is a number of the colors k such that the nodes of G can be colored legally using k colors and the nodes of G cannot be colored legally using $k-1$ colors. This well defined number k is called the chromatic number of the graph G and it is denoted by $\chi(G)$.

Graph coloring is a vast subject and we cannot make justice to this venerable field. In this paper we take a very narrow view. We are interested in only one particular application. Note that $\omega(G) \leq \chi(G)$ holds for each finite simple graph G and so coloring of the nodes can be used to estimate the size of a maximum clique. However, the gap between $\omega(G)$ and $\chi(G)$ can be arbitrarily large. J. Mycielski [13] exhibited a graph $M^{(k)}$ for which $\omega(M^{(k)}) = 2$ and $\chi(M^{(k)}) = k$ for each integer $k \geq 2$.

In order to find bounds for $\omega(G)$ the following node coloring was proposed in [21]. Let us choose an integer $s \geq 2$ and consider a coloring of the nodes of G that satisfies the following conditions.

(1′) Each vertex of G is colored with exactly one color.

(2′) If the vertices $v_1, \ldots, v_s$ are vertices of a clique in G, then all the vertices $v_1, \ldots, v_s$ cannot receive the same color.

A coloring of the nodes of G satisfying the conditions (1′), (2′), is called a monochrome s-clique free coloring. In short we will talk about s-clique free coloring. For $s = 2$ the monochrome s-clique free coloring of the nodes gives back the legal coloring of the nodes. There is a well defined minimum number k such that the nodes of G have an s-clique free coloring using k colors. This k is referred to as the the s-clique free chromatic number of G and it is denoted by $\chi^{(s)}(G)$. The inequality $\omega(G) \leq (s-1)\chi^{(s)}(G)$ shows that s-clique free coloring of the nodes can be used to establish upper bound for the clique number.

A number of problems is considered in connection with coloring the nodes of a graph customarily.

Problem 5. *Given a finite simple graph G and given a positive integer k. Decide whether the nodes of G admit a legal coloring using k colors.*

Problem 6. *Given a finite simple graph G. Determine $\chi(G)$.*

It is a well known result of the complexity theory of algorithms that Problem 5 belongs to the P complexity class for $k = 2$ and it belongs to the NP-complete complexity class for $k \geq 3$. (See for example [15].) It follows that for $k \geq 3$ Problem 6 is NP-hard.

Problem 7. *Given a finite simple graph G and two positive integers s, k. Decide if the nodes of G have a legal s-clique free coloring with k colors.*

It was established in [23] that for $k = 3$, $s \geq 3$ Problem 7 is NP-complete.

2 A Mycielski type result

As we have already mentioned the chromatic number can be a poor upper estimate of the clique number. By Mycielski's construction there are 3-clique free graphs with arbitrarily large chromatic number. P. Erdős [5] generalized this result. Let us call the length of a shortest cordless circle in a graph the girth of the graph. Clearly, the girth of a 3-clique free graph must be at least 4. Erdős has proved that for given positive integers k and l, there is a finite simple graph G with $\mathrm{girth}(G) \geq l$, $\chi(G) \geq k$. Erdős's proof is not constructive and so it is not at all straight forward how the resulting graphs could be used in constructing test instances.

In this section we present another extension of Mycielski's result. We replace the legal coloring of the nodes of a graph by a legal s-clique free coloring of the nodes of the graph. Consequently, the s-clique free chromatic number $\chi^{(s)}(G)$ will play the role of the chromatic number $\chi(G)$.

The result is motivated by the fact that one might try to construct clique search test instances by replacing the Mycielski graph by the graph emerging from the proof the generalized version.

Theorem 1. *Let us choose two positive integers s and k with $s \geq 3$ and $k \geq 2^{(s-1)}/(s-1)$. There is a finite simple graph $L^{(s,k)}$ such that $\omega(L^{(s,k)}) \leq 2^{(s-1)}$ and $\chi^{(s)}(L^{(s,k)}) \geq k$.*

The reader may notice that the graph $L^{(2,k)}$ is isomorphic to the Mycielski graph $M^{(k)}$.

Proof. The proof will be constructive. We start with the special case $s = 3$. We choose an integer k for which $k \geq 2^{(3-1)}/(3-1) = 2$. Let $M^{(k)}$ be the Mycielski graph with parameter k. Let $u_1, \ldots, u_n$ be the nodes of $M^{(k)}$. We substitute the node u_i of $M^{(k)}$ by an isomorphic copy $M_i^{(k)}$ of the Mycielski graph for each i, $1 \leq i \leq n$. Let $v_{i,1}, \ldots, v_{i,n}$ be the nodes of $M_i^{(k)}$. We assume that the nodes

$$v_{1,1}, \ldots, v_{1,n}, \ldots, v_{n,1}, \ldots, v_{n,n}$$

are pair-wise distinct. These nodes will be the nodes of the graph $L^{(3,k)}$.

The edges of $M_i^{(k)}$ are going to be edges of $L^{(3,k)}$ for each i, $1 \leq i \leq n$. Further, whenever the unordered pair $\{u_i, u_j\}$ is an edge of $M^{(k)}$, then we add the edge $\{v_{i,\alpha}, v_{j,\beta}\}$ to $L^{(3,k)}$ for each α, β, $1 \leq \alpha, \beta \leq n$.

The dedicated reader will not fail to notice that the construction we just presented is the so called lexicographic product of the graphs $M^{(k)}$ and $M^{(k)}$.

We claim that $\omega(L^{(3,k)}) \leq 4$.

In order to verify the claim we assume on the contrary that $\omega(L^{(3,k)}) \geq 5$. Let C be a 5-clique in $L^{(3,k)}$. Set $V_i = \{v_{i,1,}, \ldots, v_{i,n}\}$. Note that the set V_i induces $M_i^{(k)}$ in $L^{(3,k)}$ as a subgraph of $L^{(3,k)}$. From the fact that $\omega(M_i^{(k)}) \leq 2$ it follows that C may have at most 2 nodes in V_i for each i, $1 \leq i \leq n$. Therefore C has nodes in some $M_i^{(k)}$ for at least 3 distinct values of i.

Suppose that i and j are distinct numbers in the set $\{1,\ldots,n\}$. A node in $M_i^{(k)}$ and a node in $M_j^{(k)}$ can be adjacent only if the unordered pair $\{u_i,u_j\}$ is an edge of $M^{(k)}$. This means that $M^{(k)}$ contains a 3-clique. But $M^{(k)}$ does not contain any 3-clique. This contradiction completes the verification of the claim.

We claim that $\chi^{(3)}(L^{(3,k)}) \geq k$.

In order to prove the claim let us assume on the contrary that $\chi^{(3)}(L^{(3,k)}) \leq k-1$. Set

$$\begin{aligned} V &= V_1 \cup \cdots \cup V_n \\ &= \{v_{1,1},\ldots,v_{1,n},\ldots,v_{n,1},\ldots,v_{n,n}\}. \end{aligned}$$

Suppose that the map $f : V \to \{1,\ldots,k-1\}$ defines a 3-clique free coloring of the nodes of $L^{(3,k)}$.

The restriction of f to the set V_i is a map $g_i : V_i \to \{1,\ldots,k-1\}$. Clearly, the map g_i defines a coloring of the nodes of the graph $M_i^{(k)}$. From the fact that $\chi(M_i^{(k)}) \geq k$ it follows that there are two distinct adjacent nodes of $M_i^{(k)}$ such that the two nodes receive the same color c_i. Set $U = \{u_1,\ldots,u_n\}$. Using the color c_i we define a map $h : U \to \{1,\ldots,k-1\}$. We set $h(u_i) = c_i$ for each i, $1 \leq i \leq n$.

Note that the map h defines a legal coloring of the nodes of the graph $M^{(k)}$. The only thing which needs verification is that if u_i and u_j are distinct adjacent nodes of $M^{(k)}$, then $c_i \neq c_j$.

Let us assume on the contrary that u_i and u_j are distinct adjacent nodes of $M^{(k)}$ and $c_i = c_j$. The graph $M_i^{(k)}$ has two distinct adjacent nodes $v_{i,i(1)}$ and $v_{i,i(2)}$ such that

$$f(v_{i,i(1)}) = f(v_{i,i(2)}) = c_i.$$

Similarly, the graph $M_j^{(k)}$ has two distinct adjacent nodes $v_{j,j(1)}$ and $v_{j,j(2)}$ such that

$$f(v_{j,j(1)}) = f(v_{j,j(2)}) = c_j.$$

Note that the nodes $v_{i,i(1)}$, $v_{i,i(2)}$, $v_{j,j(1)}$, $v_{j,j(2)}$ are the nodes of a 4-clique in $L^{(3,k)}$. This means that the coloring defined by the map f is not a 3-clique free coloring of the nodes of $L^{(3,k)}$. This shows that the coloring defined by the map h is a legal coloring of the nodes of $M^{(k)}$.

The coloring defined by h is using at most $k-1$ colors. This contradicts the fact that $\chi(M^{(k)}) \geq k$. Thus we may conclude that $\chi^{(3)}(L^{(3,k)}) \geq k$ as we claimed.

Let us turn to the special case $s = 4$. We choose a integer k for which $k \geq 2^{(4-1)}/(4-1)$, that is, $k \geq 3$. Let $M^{(k)}$ be the Mycielski graph with parameter k. Let $u_1,\ldots,u_n$ be the nodes of $M^{(k)}$. We substitute the node u_i of $M^{(k)}$ by an isomorphic copy $L_i^{(3,k)}$ of the graph $L^{(3,k)}$ for each i, $1 \leq i \leq n$. Let $v_{i,1},\ldots,v_{i,m}$ be the nodes of $L_i^{(3,k)}$. We assume that the nodes

$$v_{1,1},\ldots,v_{1,m},\ldots,v_{n,1},\ldots,v_{n,m}$$

are pair-wise distinct. These nodes will be the nodes of the graph $L^{(4,k)}$.

The edges of $L_i^{(3,k)}$ are going to be edges of $L^{(4,k)}$ for each i, $1 \leq i \leq n$. Further, whenever the unordered pair $\{u_i,u_j\}$ is an edge of $M^{(k)}$, then we add the edge $\{v_{i,\alpha},v_{j,\beta}\}$ to $L^{(4,k)}$ for each α, β, $1 \leq \alpha,\beta \leq m$.

We claim that $\omega(L^{(4,k)}) \leq 8$.

In order to verify the claim we assume on the contrary that $\omega(L^{(4,k)}) \geq 9$. Let C be a 9-clique in $L^{(4,k)}$. Note that the set $V_i = \{v_{i,1,},\ldots,v_{i,m}\}$ induces $L_i^{(3,k)}$ in $L^{(4,k)}$ as a subgraph of $L^{(4,k)}$. From the fact that $\omega(L_i^{(3,k)}) \leq 4$ it follows that C may have at most 4 nodes in V_i for each i, $1 \leq i \leq n$. Therefore C has nodes in some $L_i^{(3,k)}$ for at least 3 distinct values of i.

Suppose that i and j are distinct numbers in the set $\{1,\ldots,n\}$. A node in $L_i^{(3,k)}$ and a node in $L_j^{(3,k)}$ can be adjacent only if the unordered pair $\{u_i,u_j\}$ is an edge of $M^{(k)}$. This means that $M^{(k)}$ contains a 3-clique. But $M^{(k)}$ does not contain any 3-clique. This contradiction completes the proof of the claim.

We claim that $\chi^{(4)}(L^{(4,k)}) \geq k$.

In order to prove the claim let us assume on the contrary that $\chi^{(4)}(L^{(4,k)}) \leq k-1$. Set

$$\begin{aligned} V &= V_1 \cup \cdots \cup V_n \\ &= \{v_{1,1},\ldots,v_{1,n},\ldots,v_{n,1},\ldots,v_{n,n}\}. \end{aligned}$$

Suppose that the map $f : V \to \{1,\ldots,k-1\}$ defines a 4-clique free coloring of the nodes of $L^{(4,k)}$.

The restriction of f to the set V_i is a map $g_i : V_i \to \{1,\ldots,k-1\}$. Clearly, the map g_i defines a coloring of the nodes of the graph $L_i^{(3,k)}$. From the fact that $\chi^{(3)}(L_i^{(3,k)}) \geq k$ it follows that there is a 3-clique in $L_i^{(3,k)}$ such that the 3 nodes of the clique receive the same color c_i. Set $U = \{u_1,\ldots,u_n\}$. Using the color c_i we define a map $h : U \to \{1,\ldots,k-1\}$. We set $h(u_i) = c_i$ for each i, $1 \leq i \leq n$.

Note that the map h defines a legal coloring of the nodes of the graph $M^{(k)}$. The only thing which needs verification is that if u_i and u_j are distinct adjacent nodes of $M^{(k)}$, then $c_i \neq c_j$.

Let us assume on the contrary that $c_i = c_j$. The graph $L_i^{(3,k)}$ has 3 distinct pair-wise adjacent nodes $v_{i,i(1)}$, $v_{i,i(2)}$, $v_{i,i(3)}$ such that

$$f(v_{i,i(1)}) = f(v_{i,i(2)}) = f(v_{i,i(2)}) = c_i.$$

Similarly, the graph $L_i^{(3,k)}$ has 3 distinct pair-wise adjacent nodes $v_{j,j(1)}$, $v_{j,j(2)}$, $v_{j,j(2)}$ such that

$$f(v_{j,j(1)}) = f(v_{j,j(2)}) = f(v_{j,j(3)}) = c_j.$$

Note that the nodes

$$v_{i,i(1)}, v_{i,i(2)}, v_{i,i(3)}, v_{j,j(1)}, v_{j,j(2)}, v_{j,j(3)}$$

are the nodes of a 6-clique in $L^{(4,k)}$. This means that the coloring defined by the map f is not a 4-clique free coloring of the nodes of $L^{(4,k)}$. This shows that the coloring defined by the map h is a legal coloring of the nodes of $M^{(k)}$.

In the coloring defined by h most $k-1$ colors occur. This contradicts the fact that $\chi(M^{(k)}) \geq k$. Thus we may draw the conclusion that $\chi^{(4)}(L^{(4,k)}) \geq k$ as we claimed.

Continuing in this way we can complete the proof of the theorem. □

3 Test problems

Problems 2 and 3 are commonly referred as k-clique and maximum clique problems, respectively. As we have pointed out it is a well known result of the complexity theory of algorithms that the maximum clique problem is NP-hard. Loosely speaking it can be interpreted such that the maximum clique problem is computationally demanding.

As at this moment there are no readily available mathematical tools to evaluate the performance of practical clique search algorithms, the standard procedure is to carry out numerical experiments on a battery of well selected benchmark tests.

The most widely used test instances are the Erdős–Rényi random graphs, graphs from the the second DIMACS challenge[1] [8], combinatorial problems of monotonic matrices [25, 22], and hard coding problems of Deletion-Correcting Codes[2] [20].

Evaluating the performances of various clique search algorithms is a delicate matter. On one hand one would like to reach some practically relevant conclusion about the competing algorithms. On the other hand this conclusion is based on a finite list of instances.

One has to be ever cautious not to draw overly sweeping conclusions from these inherently limited nature experiments. (We intended to contrast this approach to the asymptotic techniques which are intimately tied to infinity.) The situation is of course not completely pessimistic. After all, these benchmarks were successful at shedding light on the practicality of many of the latest clique search procedures. However, we should strive for enhancing the test procedures. The main purpose of this paper is to propose new benchmark instances.

There are occasions when we are trying to locate a large clique in a given graph such that the clique is not necessarily optimal. This approach is referred as non-exact method to contrast it to the exhaustive search. For instance constructing a large time table in this way can be practically important and useful even without a certificate of optimality.

The benchmark tests are of course relevant in connection with non-exact procedures too. In order to avoid any unnecessary confusion we would like emphasize that in this paper we are focusing solely on the exact clique search methods.

Let n be a positive integer and let p be a real number such that $0 \leq p \leq 1$. An Erdős-Rényi random graph with parameters n, p is a graph G with vertices $1, 2, \ldots, n$. The probability that the unordered pair $\{x, y\}$ is an edge of G is equal to p for each x, y, $1 \leq x < y \leq n$. The events that the distinct pairs

$$\{x_{i(1)}, y_{i(1)}\}, \ldots, \{x_{i(s)}, y_{i(s)}\}$$

are edges of G are independent of each other for each subset $\{i(1), \ldots, i(s)\}$ of $\{1, 2, \ldots, n\}$, where $s \geq 2$.

In a more formal way the Erdős-Rényi random graph of parameters n, p is a random variable whose values are all the simple graphs with n vertices. The probability distribution over these graphs is specified in the manner we have described above. In this paper we can work safely in a more intuitive level. We start with a complete graph on n vertices and we decide the fate of each edge by flipping a biased coin.

In the case $p = 0$ we end up with a graph consisting of n isolated nodes. In the case $p = 1$ we end up with a complete graph on n nodes. (Paper [4] is the basic reference on Erdős-Rényi random graph.)

Let l, m be positive integers. Let $H_i = (V_i, E_i)$ be a graph consisting of l isolated nodes. This means that $|V_i| = l$ and $E_i = \emptyset$ for each i, $1 \leq i \leq m$. Let $V_i = \{v_{i,1}, \ldots, v_{i,l}\}$. We construct a new graph $G = (V, E)$. We set $V = V_1 \cup \cdots \cup V_m$. The nodes $v_{i,r}$, $v_{j,s}$ are connected by an edge in G whenever $i \neq j$. We may say that the graph G is isomorphic to the lexicographic product of the graphs H and K_m, where H consists of l isolated nodes and K_m is the complete graph on m nodes. (For further details of graph products see [7].)

Clearly, V_i is an independent set in G for each i, $1 \leq i \leq m$. The subgraph induced by $V_i \cup V_j$ in G is a complete bipartite graph for each i, j, $1 \leq i < j \leq m$. Obviously, $\chi(G) = m$ and $\omega(G) = m$ hold. In fact G contains l^m distinct m-cliques.

At this stage we choose a real number p such that $0 \leq p \leq 1$. At each edge of G we flip a biased coin. The edge stays with probability p. We call this step randomizing G. The resulting random graph G' belongs to the parameters l, m, p. The $l = 1$ particular case corresponds to the Erdős-Rényi random graph of parameters m, p.

It is clear that $\chi(G') \leq m$ and $\omega(G') \leq m$. In order to guarantee that $\omega(G') = n$ holds we will plant an m-clique into G'. One can achieve this by picking $x_i \in V_i$ for each i, $1 \leq i \leq m$ and connect each distinct pairs among $x_1 \ldots, x_m$ by an edge in G'.

Benchmark tests based on these random graphs are collected in the BHOSLIB library.[3] (The acronym BHOSLIB stands for Benchmarks with Hidden Optimum Solutions Library.)

After all these preparations we are ready to describe the graphs we would like to propose for testing clique search algorithms. Let k, m be positive integers. Let $M_i^{(k)} = (V_i, E_i)$ be the Mycielski graph of parameter k. Let $V_i =$

[1] ftp://dimacs.rutgers.edu/pub/challenge/
[2] http://neilsloane.com/doc/graphs.html
[3] http://www.nlsde.buaa.edu.cn/~kexu/benchmarks/graph-benchmarks.htm

$\{v_{i,1}, \ldots, v_{i,n}\}$ for each i, $1 \leq i \leq m$. We construct a new graph $G = (V, E)$. We set $V = V_1 \cup \cdots \cup V_m$. Let $v_{i,r}, v_{i,s} \in V_i$. If the unordered pair $\{v_{i,r}, v_{i,s}\}$ is an edge of $M_i^{(k)}$, then we add this pair as an edge to G. These edges will be the blue edges of G. In other words the subgraph induced by V_i in G is isomorphic to $M_i^{(k)}$ for each i, $1 \leq i \leq m$.

Pick $v_{i,r} \in V_i$, $v_{j,s} \in V_j$. We connect the nodes $v_{i,r}$, $v_{j,s}$ by an edge in G whenever $i \neq j$. These edges will be the red edges of G.

Note that the graph G is isomorphic to the lexicographic product of the graphs $M^{(k)}$ and K_m, where $M^{(k)}$ is the Mycielski graph of parameter k and K_m is the complete graph on m nodes. One can verify that $\chi(G) = (k)(m)$ and $\omega(G) = (2)(m)$.

We choose a real number p such that $0 \leq p \leq 1$. We randomize the red edges of G. We flip a biased coin and keep each red edge with probability p. The resulted random graph is denoted by G'. It is obvious that $\chi(G') \leq (k)(m)$ and $\omega(G') \leq (2)(m)$. By planting a $(2m)$-clique into G' we can guarantee that $\omega(G') = (2)(m)$. We pick $x_i, y_i \in V_i$ such that the unordered pair $\{x_i, y_i\}$ is an edge in G' for each i, $1 \leq i \leq m$. Finally, we construct a $(2m)$-clique whose nodes are $x_1, y_1, \ldots, x_m, y_m$.

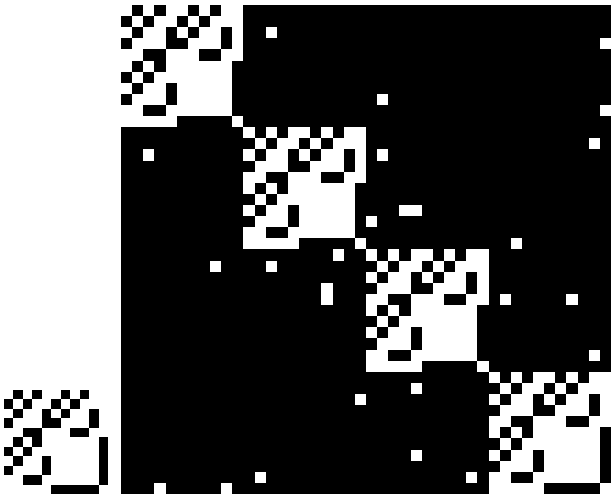


Figure 1: The adjacency matrices of the Mycielski graph $M^{(4)}$ and the random graph G'.

4 Numerical experiments

The proposed new collection of test graphs can be found on the site `clique.ttk.pte.hu/evil`. The source code of the program that generates the adjacency matrices of these graphs are also available on this site.

As an illustration Figure 1 exhibits the adjacency matrix of the Mycielski graph of parameter 4. Further it depicts the adjacency matrix of the randomized version of the lexicographic product of $M^{(4)}$ and K_4, where $M^{(4)}$ is the Mycielsky graph of parameter 4 and K_4 is the complete graph on 4 vertices. As the number of vertices of $M^{(4)}$ and K_4 are equal to 11 and 4, respectively, the product graph has 44 nodes. The probability we used for randomizing is $p = 0.98$.

In the constructions we systematically replaced the Mycielski graph $M^{(k)}$ by the graph $L^{(s,k)}$ that appeared in the proof of Theorem 1 for small values of the parameters s and k. The graph $M^{(4)}$ has 11 nodes. Its chromatic and clique numbers are 4 and 2, respectively. There is a graph with the same chromatic and clique numbers the so-called Chvatal graph. The Chvatal graph has 12 vertices but more symmetric than $M^{(4)}$. We replaced $M^{(4)}$ by the Chvatal graph systematically when we constructed test instances. Note that other graphs also can be used instead of these two examples. Presumably the kind of graphs where the clique number is far from the chromatic number. As the last step we randomly permuted the nodes of the graphs.

We carried out a large scale numerical experiment to check the proposed EVIL benchmark problems. We used 55 test graphs. We took 35 BHOSLIB graphs and 20 EVIL graphs. The experiment involved 7 programs implementing 12 different algorithms and so we are able to compare the running times of 660 clique searches. The processor of the computer we used was a 2.3 GHz, Xeon E5-2670v3.

The 12 clique search algorithms we used are the following.

(1) Östergård[4] [14] (cliquer),

(2) Li[5] [11], [12] (M-cql 10, M-cql 13-1 and M-cql 13-2).

(3) Konc[6] [9] (mcqd and mcqd-dyn)

(4) Prosser[7] (who implemented Tomita's algorithm [24]) (MCR)

(5) San Segundo[8] [16], [17], [18], [19] (BBMC, BBMC-R, BBMC-L and BBMC-X).

There are three ways to use the 2013 version of C.-M. Li program. A switch can be set to either "1" or "2" to select between two built in orderings of the nodes of the graph. In case no value of the switch is specified the program chooses between the "1" and "2" possibilities. During our test we explicitly used the switch "1" and "2" (M-cql 13-1 and M-cql 13-2).

The above programs are high quality state of art programs. It seemed reasonable to enter an unsophisticated program to the competion. The program can be found on the same site as the EVIL instances and goes under the name antiB. The brief description of the program is the following.

[4] `http://users.aalto.fi/~pat/cliquer.html`

[5] `http://home.mis.u-picardie.fr/~cli/EnglishPage.html`

[6] `http://www.sicmm.org/konc/maxclique/`

[7] `http://www.dcs.gla.ac.uk/~pat/maxClique/distribution/`

[8] `https://www.biicode.com/pablodev/examples_clique`

1) Using the simplest sequential greedy algorithm color legally the nodes of the graph and save the colors of the nodes.
2) Set k to be the number of colors of the legal coloring we have located.
3) Carry out a k-clique search.
4) If a k-clique is found, then it is a maximum clique of the graph. Otherwise reduce the value of k and go to step 3.

The k-clique search is based on the Carraghan-Pardalos algorithm, where we utilized original coloring of the nodes. The ordering of the nodes was done by the size of the color classes and the node degrees.

The results of the experiment are summarized in Table 1.

The running times on the BHOSLIB instances are shown in the first part of the table. The evaluation of the results shows that the 2013 version of Li's program with the "2" switch is on is performing unexpectedly well. The running times of he cliquer and the antiB programs are outliers as well.

Our possible explanation is that although the BHOSLIB tests are excellent for heuristic big clique search programs however they are not so good for evaluating exact maximum clique search algorithms. One problem with these instances is that the nodes of these graphs are not randomly permuted. This means that a simple sequential greedy coloring will find the chromatic number at once, as the color classes are laid out consecutively. This can be remedied easily. An other possible problem is that most maximum clique search programs use coloring as an auxiliary algorithm. For these graphs the chromatic number is equal to the clique size leaving a zero optimality or duality gap. So specially designed programs, as the presented antiB, are able to take advantage of this.

The running times for the EVIL benchmark problems are shown in the second part of the table. Here the running times are more evenly distributed.

One particular result was that there is a test graph with 220 nodes – 20 copies of the $M^{(4)}$ graph, $p = 98\%$ edge probability – whose clique number could be determined by only one program in slightly less than 12 hours. We suppose that this problem is the hardest one of such small size.

For certain graph the running times of the cliquer program are again outliers. These short running times could be explained by the fact that the cliquer does not rely on legal coloring of the nodes as do the other programs. After all, we constructed the tests to widen the gap between the chromatic number and the clique number. On the same graphs but with non-permuted nodes the cliquer runs even faster. This again points out the importance of randomly permuting the nodes of the test graphs. The antiB program finishes at the last place as one would expect.

We would like to close this section with a few remarks on why the reader should appreciate the proposed benchmark problems. Although it seems that there is a large number of benchmark problems for maximum clique search algorithms the plain fact is that there are not enough of them. Many of these test problems are too easy for the modern solvers as the sizes of these problems are small. On the other hand there are test instances that are overly hard for the contemporary clique solvers. The proposed EVIL test graphs are forming parameterized families. The parameters can be tuned to produce benchmark problems in various degrees of difficulty.

5 A historical thread

The graphs that are used for benchmarking clique search algorithms are coming from various walks of discrete mathematics and its applications. In this section we will follow a particular thread in order to shed some light on the forces and demands that shaped the evolution of certain benchmark problems.

By the lack of other possibilities the performance of clique search algorithms are commonly evaluated by carrying out large scale numerical experiments. Historically the first clique search procedures were tested almost exclusively on random graphs. The Erdős-Rényi random graphs are readily available with any specified number of nodes and with any specified edge densities. These random graph are popular and useful test instances.

Since the clique size and the chromatic number of a random graph is unknown at the moment of generating them, these test graphs are not the ideal choices to test k-clique search algorithms or for algorithms to list all maximum cliques.

Let $G = (V, E)$ be a complete k-partite graph with n nodes. For the sake of definiteness we assume that V is partitioned into the sets $V_1, \ldots, V_k$ such that $|V_1| = \cdots = |V_k| = s$. Consequently $n = ks$. Suppose $V_i = \{v_{i,1}, \ldots, v_{i,s}\}$. If $i \neq j$, then we connect each node $v_{i,\alpha}$ in V_i with each node $v_{j,\beta}$ in V_j. Thus G has $(1/2)k(k-1)s^2$ edges, that is, $|E| = (1/2)k(k-1)s^2$.

The nodes of G can be legally colored using k colors. The sets $V_1, \ldots, V_k$ can play the roles of the color classes. It is clear that $\omega(G) = k$. Picking exactly one node from each V_i we get mutually adjacent nodes that form the nodes of a k-clique in G. The number of the maximum cliques in G is equal to s^k. These benchmark problems are good candidates to test clique search algorithms that list all maximum cliques. These benchmark problems painfully lack the unpredictability of random graphs.

Therefore when we connect the nodes of the sets V_i and V_j we should do it in a randomized manner. Each of the s^2 edges of the bipartite graph induced by the set $V_i \cup V_j$ in G is chosen with a fixed probability $p_{i,j}$. The clique size of the resulting random graph may decrease and the chromatic number of the resulting random graph may increase. By planting a randomly chosen k-clique into the graph we guarantee that the clique and the chromatic number are equal to k.

These graphs are the BHOSLIB instances. The development of these graphs greatly enhanced the utility of random graphs in testing clique search algorithms.

A graph whose chromatic number is equal to its clique number is by no means is a typical graph. This is why we propose a family of test graphs that have all the desired properties of the BHOSLIB graph and in the same time the gap between the clique and chromatic numbers can be set in a more flexible manner.

Let $G = (V, E)$ be the graph we intend to construct. Let us start with a graph $M = (W, F)$ such that $\omega(M) = r$, $\chi(M) = s$ are known. Suppose $W = \{w_1, \ldots, w_m\}$. We consider k isomorphic copies $M_1, \ldots, M_k$ of M. Let $M_i = (W_i, F_i)$, where $W_i = \{w_{i,1}, \ldots, w_{i,m}\}$.

In order to construct the graph G we set $V = W_1 \cup \cdots \cup W_k$. Here we assume that the sets $W_1, \ldots, W_k$ are pair-wise disjoint. Consequently G has km nodes. Let us pick two distinct W_i and W_j. We connect $w_{i\alpha}$ and $w_{j,\beta}$ for each α, β, $1 \leq \alpha, \beta \leq m$. The resulting graph G has km nodes and $k|F| + (1/2)k(k-1)m^2$ edges. Clearly, $\omega(G) = k\omega(M) = kr$ and $\chi(G) = k\chi(M) = ks$.

As a next step we randomize G. Namely, we pick each of the m^2 edges running between W_i and W_j with a fixed probability $p_{i,j}$. (In many cases we choose all $p_{i,j}$ to be equal.) Because we drop edges from the graph G, the quantities $\omega(G)$, $\chi(G)$ may change. By planting a random (kr)-clique we may restore the original $\omega(G)$. The chromatic number of the new random graph may decrease. If the probability $p_{i,j}$ is close to one then most likely the decrease in the chromatic number is small. When we choose a graph M for which $\omega(M)$ is much smaller than $\chi(M)$, then for the resulted random graph the gap between the clique and chromatic numbers most likely does not disappear.

The resulting random graphs are the test instances proposed in this paper.

The Szemerédi regularity lemma teaches us that a large dense graph can be transformed into a form which is very much similar to our proposed graph. So the proposed graph in a sense is not overly artificial.

6 Reducing the number of nodes

We use the graphs $L^{(s,k)}$ defined in the proof of Theorem 1 to construct benchmark instances. It is imperative to construct hard benchmark instances with as few nodes as possible. In this section we will show that with a little extra care we can reduce the number of the nodes and still get a graph which has the required properties of the graphs $L^{(s,k)}$. For the sake of simplicity we will restrict our attention to the special case $s = 3$.

Let $G = (V, E)$ be a finite simple graph. A subset C of V is called an edge covering set of G if each edge of G has at least one end point in C.

Let $\mu(k)$ be the number of the nodes of the Mycielski graph $M^{(k)}$ of parameter k. The Mycielski graph $M^{(3)}$ is a circle consisting of 5 nodes and 5 edges. Thus $\mu(3) = 5$. The Mycielski graph $M^{(k+1)}$ is constructed from the Mycielski graph $M^{(k)}$ in the following way.

Suppose $U = \{u_1, \ldots, u_n\}$ is the set of nodes of $M^{(k)}$. Here of course $n = \mu(k)$. In order to construct $M^{(k+1)}$ from $M^{(k)}$ we add new nodes $v_1, \ldots, v_n$ to the existing nodes $u_1, \ldots, u_n$. We set $V = \{v_1, \ldots, v_n\}$. We assume that the nodes $u_1, \ldots, u_n$ and $v_1, \ldots, v_n$ are pair-wise distinct. In other words we assume that $U \cap V = \emptyset$. We draw an edge between the node v_i and each neighbor of the node u_i for each i, $1 \leq i \leq n$. Finally we add a new node w to the set of nodes $U \cup V$ and draw an edge between the node w and v_i for each i, $1 \leq i \leq n$.

Note that the equation $\mu(k+1) = 2\mu(k) + 1$ holds and using $\mu(3) = 5$ one can compute the number of nodes of the Mycielski graph $M^{(k)}$.

Lemma 1. *The Mycielski graph $M^{(k)}$ has an edge covering set of size $\mu(k-1)+1$ for each $k \geq 3$.*

Proof. For $k = 3$ the Mycielski graph is a circle consisting of 5 vetices and 5 edges. Two adjacent nodes x, y and a node z that is not adjacent to any of x, y form an edge covering set in $M^{(3)}$. This proves the lemma in the special case $k = 3$.

For the remaining part of the proof we may assume that $k \geq 4$. We proceed by an induction on k. The Mycielski graph $M^{(k)}$ has some nodes $u_1, \ldots, u_n$ such that $n = \mu(k-1)$ and the subset $U = \{u_1, \ldots, u_n\}$ induces a subgraph that is isomorphic to $M^{(k-1)}$.

The Mycielski graph $M^{(k)}$ has some nodes $v_1, \ldots, v_n$ such that v_i is adjacent to each neighbor of u_i for each i, $1 \leq i \leq n$. Finally, $M^{(k)}$ has a node w which is adjacent to v_i for each i, $1 \leq i \leq n$.

The set $C = U \cup \{w\}$ is an edge covering set in $M^{(k)}$. This completes the proof of the lemma. □

We do not claim that this edge covering set has the smallest possible number of nodes.

Lemma 2. *For each integer $k \geq 3$ there is a graph $N^{(k)}$ such that it has $[\mu(k-1)+1]\mu(k) + \mu(k-1)$ nodes $\omega(N^{(k)}) \leq 4$ and $\chi^{(3)}(N^{(k)}) \geq k$.*

Proof. We start the construction of $N^{(k)}$ with the Mycielski graph $M^{(k)}$. We assume that $U = \{u_1, \ldots, u_n\}$ is the set of nodes of $M^{(k)}$. By Lemma 1, the graph $M^{(k)}$ has an edge covering set C, where $n = \mu(k)$ and $|C| = \mu(k-1)+1$.

We will assign a subgraph L_i to the node u_i of the graph $M^{(k)}$ for each i, $1 \leq i \leq n$. The graph L_i is either a graph consisting of one single node $v_{i,1}$ or L_i is an isomorphic copy of the Mycielski graph $M^{(k)}$. In this second case the set of nodes of L_i is equal to $V_i = \{v_{i,1}, \ldots, v_{i,r}\}$, where $r = \mu(k)$.

If $u_i \notin C$, then to the node u_i we assign the graph L_i which consists of a single node. If $u_i \in C$, then to the node u_i we assign the graph L_i which has $\mu(k)$ nodes.

Let us suppose that the unordered pair $\{u_i, u_j\}$ is an edge of the graph $M^{(k)}$. We draw edges between each node

of L_i and each node of L_j. The result of repeating this procedure for each edges of the graph $M^{(k)}$ will be the graph $N^{(k)}$. Clearly, $N^{(k)}$ has

$$[\mu(k-1)+1]\mu(k)+\mu(k)-\mu(k-1)-1=$$
$$[\mu(k-1)+1]\mu(k)+2\mu(k-1)+1-\mu(k-1)-1=$$
$$[\mu(k-1)+1]\mu(k)+\mu(k-1)$$

nodes. It remains to show that $\omega(N^{(k)}) \leq 4$ and $\chi^{(3)}(N^{(k)}) \geq k$.

In order to prove that $\omega(N^{(k)}) \leq 4$ we assume on the contrary that $\omega(N^{(k)}) \geq 5$. Let Δ be a 5-clique in $N^{(k)}$. Note that

$$\omega(L_i) = \begin{cases} 1, & \text{if } |V_i| = 1, \\ 2, & \text{if } |V_i| = \mu(k). \end{cases}$$

The clique Δ may contain at most one node from L_i for which $|V_i| = 1$. The clique Δ may contain at most two nodes from L_i when $|V_i| = \mu(k)$. It follows that there must be at least three distinct values of i for which the graph L_i contains a node from the 5-clique Δ. This gives at least three distinct values of i for which the nodes u_i pair-wise adjacent. But $M^{(k)}$ does not have any 3-clique as $\omega(M^{(k)}) \leq 2$.

In order to prove that $\chi^{(3)}(N^{(k)}) \geq k$ we assume on the contrary that $\chi^{(3)}(N^{(k)}) \leq k-1$. Let $V = V_1 \cup \cdots \cup V_n$ be the set of nodes of $N^{(k)}$ and let $f : V \to \{1, \ldots, k-1\}$ be a map that describes the monochrome 3-clique free coloring of the nodes of $N^{(k)}$.

Consider a subgraph L_i of $N^{(k)}$. Let g_i be the restriction of f to $V_i = \{v_{i,1}, \ldots, v_{i,r}\}$. Plainly, the map $g_i : V_i \to \{1, \ldots, k-1\}$ describes a coloring of the nodes of the graph L_i. Using the facts that L_i is isomorphic to $M^{(k)}$ and $\chi(M^{(k)}) \geq k$ we can draw the conclusion that there must be two distinct adjacent nodes of $M^{(k)}$ that receive the same color.

Remember that $U = \{u_1, \ldots, u_n\}$ is the set of nodes of the Mycielski graph $M^{(k)}$ we used in the construction of $N^{(k)}$. We define a map $h : U \to \{1, \ldots, k-1\}$.

If the subgraph L_i of $N^{(k)}$ has only one node, then we set $h(u_i)$ to be $f(v_{i,1})$. In plain English we assign the color of the only node of the graph L_i to the node u_i of the graph $M^{(k)}$.

If the subgraph L_i of $N^{(k)}$ has $\mu(k)$ nodes, then there are two distinct adjacent nodes of L_i, say $v_{i,i(1)}$, $v_{i,i(2)}$ such that $f(v_{i,i(1)}) = f(v_{i,i(2)})$. In this case we set $h(u_i)$ to be $f(v_{i,i(1)})$. In simple English assign the common color of the adjacent nodes $v_{i,i(1)}$, $v_{i,i(2)}$ of $N^{(k)}$ to the node u_i of $M^{(k)}$.

We claim that the map $h : U \to \{1, \ldots, k-1\}$ describes a legal coloring of the nodes of $M^{(k)}$. The only thing we should check is that if u_i, u_j are distinct adjacent nodes of $M^{(k)}$, then $h(u_i) \neq h(u_j)$ must hold.

Let us start with the case when $u_i \notin C$, $u_j \notin C$. The assumption that u_i, u_j are distinct adjacent nodes of $M^{(k)}$ contradicts the fact that C is an edge covering set in $M^{(k)}$. Therefore this case cannot occur.

If $u_i \in C$, $u_j \in C$, then the subgraphs L_i, L_j of $N^{(k)}$ both have $\mu(k)$ nodes. There are distinct adjacent nodes $v_{i,i(1)}$, $v_{i,i(2)}$ in L_i such that $f(v_{i,i(1)}) = f(v_{i,i(2)}) = h(u_i)$. Similarly, there are distinct adjacent nodes $v_{j,j(1)}$, $v_{j,j(2)}$ in L_j such that $f(v_{j,j(1)}) = f(v_{j,j(2)}) = h(u_j)$. Here $h(u_i) \neq h(u_j)$ must hold since otherwise we have a 4-clique in $N^{(k)}$ whose nodes receive the same color. This cannot happen as the map $f : V \to \{1, \ldots, k-1\}$ describes a monochrome 3-clique free coloring of the nodes of $N^{(k)}$.

If $u_i \notin C$, $u_j \in C$, then the subgraph L_i of $N^{(k)}$ has only one node and the subgraph L_j of $N^{(k)}$ has $\mu(k)$ nodes. Now $f(v_{i,1}) = h(u_i)$. There are distinct adjacent nodes $v_{j,j(1)}$, $v_{j,j(2)}$ in L_j such that $f(v_{j,j(1)}) = f(v_{j,j(2)}) = h(u_j)$. This time $h(u_i) \neq h(u_j)$ must hold since otherwise we have a 3-clique in $N^{(k)}$ whose nodes receive the same color. This cannot happen as the map $f : V \to \{1, \ldots, k-1\}$ describes a monochrome 3-clique free coloring of the nodes of $N^{(k)}$. □

Acknowledgments

This research was supported by National Research, Development and Innovation Office – NKFIH Fund No. SNN-117879.

References

[1] I. M. Bomze, M. Budinich, P. M. Pardalos, M. Pelillo (1999) The Maximum Clique Problem, *Handbook of Combinatorial Optimization* Vol. 4, Kluwer Academic Publisher. https://doi.org/10.1007/978-1-4757-3023-4_1

[2] C. Bron and J. Kerbosch (1973) Finding all cliques of an undirected graph, *Communications of ACM* **16**, 575–577. https://doi.org/10.1145/362342.362367

[3] R. Carraghan, P. M. Pardalos (1990) An exact algorithm for the maximum clique problem, *Operation Research Letters* **9**, 375–382. https://doi.org/10.1016/0167-6377(90)90057-C

[4] P. Erdős, A. Rényi (1960) On the evolution of random graphs, *Magyar Tud. Akad. Mat. Kutató Int. Közl.* **5**, 17–61.

[5] P. Erdős (1959), Graph theory and probability, *Canad. J. Math.* **11**, 34–38.

[6] M. R. Garey and D. S. Johnson (2003), *Computers and Intractability: A Guide to the Theory of NP-completeness*, Freeman, New York.

[7] R. Hammack, W. Imrich, S. Klavžar (2011) *Handbook of Product Graphs*, CRC Press, Boca Raton, FL.

[8] J. Hasselberg, P. M. Pardalos, and G. Vairaktarakis (1993) Test case generators and computational results for the maximum clique problem, *Journal of Global Optimization* **3** , 463–482. http://www.springerlink.com/content/p2m65n57u657605n https://doi.org/10.1007/bf01096415

[9] J. Konc and D. Janežič (2007) An improved branch and bound algorithm for the maximum clique problem, *MATCH Communications in Mathematical and Computer Chemistry* **58**, 569–590.

[10] D. Kumlander (2005) *Some Practical Algorithms to Solve the Maximum Clique problem* PhD. Thesis, Tallin University of Technology.

[11] C.-M. Li, Z. Quan (2010) An efficient branch-and-bound algorithm based on MaxSAT for the maximum clique problem, *Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence.* (AAAI-10), pp. 128–133.

[12] C.-M. Li, Z. Fang, K. Xu (2013) Combining MaxSAT reasoning and incremental upper bound for the maximum clique problem, *Proceedings of the 2013 IEEE 25th International Conference on Tools with Artificial Intelligence.* (ICTAI2013), pp. 939–946. https://doi.org/10.1109/ictai.2013.143

[13] J. Mycielski (1955) Sur le coloriage des graphes, *Colloq. Math.* **3**, 161–162.

[14] P. R. J. Östergård (2002) A fast algorithm for the maximum clique problem, *Discrete Applied Mathematics* **120**, 197–207. https://doi.org/10.1016/S0166-218X(01)00290-6

[15] C. H. Papadimitriou (1994) *Computational Complexity*, Addison-Wesley Publishing Company, Inc.

[16] P. San Segundo, D. Rodriguez-Losada, A. Jimenez (2011) An exact bit-parallel algorithm for the maximum clique problem, *Computers & Operations Research.* **38**, 571–581. https://doi.org/10.1016/j.cor.2010.07.019

[17] P. San Segundo, F. Matia, D. Rodriguez-Losada, M. Hernando (2013) An improved bit parallel exact maximum clique algorithm, *Optimization Letters.* **7**, 467–479. https://doi.org/10.1007/s11590-011-0431-y

[18] P. San Segundo, C. Tapia (2014) Relaxed approximate coloring in exact maximum clique search, *Computers & Operations Research.* **44**, 185–192. https://doi.org/10.1016/j.cor.2013.10.018

[19] P. San Segundo, A. Nikolaev, M. Batsyn (2015) Infra-chromatic bound for exact maximum clique search, *Computers & Operations Research.* **64**, 293–303. https://doi.org/10.1016/j.cor.2015.06.009

[20] N. J. A. Sloane, *Challenge Problems: Independent Sets in Graphs.* http://neilsloane.com/doc/graphs.html

[21] S. Szabó (2011) Parallel algorithms for finding cliques in a graph, *Journal of Physics: Conference Series* **268** 012030 https://doi.org/10.1088/1742-6596/268/1/012030

[22] S. Szabó (2013) Monotonic matrices and clique search in graphs, *Annales Univ. Sci. Budapest., Sect. Computatorica* **41**, 307–322.

[23] S. Szabó and B. Zaválnij (2012) Greedy algorithms for triangle free coloring, *AKCE International Journal of Graphs and Combinatorics* **9** No. 2, 169–186.

[24] E. Tomita and T. Seki (2003) An efficient branch-and-bound algorithm for finding a maximum clique, *Lecture Notes in Computer Science* **2731**, 278–289.

[25] E. W. Weisstein, Monotonic Matrix, In: *MathWorld–A Wolfram Web Resource.* http://mathworld.wolfram.com/MonotonicMatrix.html

[26] D. R. Wood (1997) An algorithm for finding a maximum clique in a graph, *Oper. Res. Lett.* **21**, 211–217. https://doi.org/10.1016/S0167-6377(97)00054-0

name	$\|V\|$	%	$\omega(G)$	antiB	BBMC	BBMC -R	BBMC -L	BBMC -X	M-clq 10	M-clq 13-1	M-clq 13-2	mcqd	mcqd -dyn	MCR	cliquer
frb30-15-1.clq	450	82	30	0	1611	1645	1694	1613	575	629	0	2735	2541	3673	21
frb30-15-2.clq	450	82	30	0	1010	1094	1191	1047	921	990	0	3329	4155	905	43
frb30-15-3.clq	450	82	30	0	602	581	623	556	432	429	0	1305	2767	300	194
frb30-15-4.clq	450	82	30	0	1855	1768	1901	1676	1154	617	0	5763	4996	2698	2
frb30-15-5.clq	450	82	30	0	1273	1213	1437	1154	726	1110	0	1997	4536	1355	0
frb35-17-1.clq	595	84	35	1	–	–	–	–	–	–	1	–	–	34983	18
frb35-17-2.clq	595	84	35	1	–	–	–	–	–	–	1	–	–	–	104
frb35-17-3.clq	595	84	35	2	–	–	–	–	–	–	0	–	–	22607	14493
frb35-17-4.clq	595	84	35	1	–	–	–	–	27231	42219	0	–	–	5249	7923
frb35-17-5.clq	595	84	35	5	–	–	–	–	–	–	0	–	–	–	181
frb40-19-1.clq	760	86	40	0	–	–	–	–	–	–	1	–	–	11589	–
frb40-19-2.clq	760	86	40	1	–	–	–	–	–	–	0	–	–	–	–
frb40-19-3.clq	760	86	40	8	–	–	–	–	–	–	0	–	–	–	353
frb40-19-4.clq	760	86	40	38	–	–	–	–	–	–	7	–	–	–	2296
frb40-19-5.clq	760	86	40	10	–	–	–	–	–	–	5	–	–	–	78
frb45-21-1.clq	945	87	45	0	–	–	–	–	–	–	119	–	–	–	–
frb45-21-2.clq	945	87	45	118	–	–	–	–	–	–	72	–	–	–	–
frb45-21-3.clq	945	87	45	122	–	–	–	–	–	–	44	–	–	–	–
frb45-21-4.clq	945	87	45	218	–	–	–	–	–	–	36	–	–	–	–
frb45-21-5.clq	945	87	45	475	–	–	–	–	–	–	203	–	–	–	–
frb50-23-1.clq	1150	88	50	16385	–	–	–	–	–	–	764	–	–	–	–
frb50-23-2.clq	1150	88	50	10145	–	–	–	–	–	–	363	–	–	–	–
frb50-23-3.clq	1150	88	50	12585	–	–	–	–	–	–	7938	–	–	–	–
frb50-23-4.clq	1150	88	50	501	–	–	–	–	–	–	17	–	–	–	2754
frb50-23-5.clq	1150	88	50	18256	–	–	–	–	–	–	221	–	–	–	–
frb53-24-1.clq	1272	88	53	73	–	–	–	–	–	–	4771	–	–	–	–
frb53-24-2.clq	1272	88	53	–	–	–	–	–	–	–	190	–	–	–	–
frb53-24-3.clq	1272	88	53	2910	–	–	–	–	–	–	2091	–	–	–	–
frb53-24-4.clq	1272	88	53	29815	–	–	–	–	–	–	4022	–	–	–	–
frb53-24-5.clq	1272	88	53	27671	–	–	–	–	–	–	1071	–	–	–	–
frb59-26-1.clq	1534	89	59	–	–	–	–	–	–	–	–	–	–	–	–
frb59-26-2.clq	1534	89	59	42408	–	–	–	–	–	–	–	–	–	–	–
frb59-26-3.clq	1534	89	59	–	–	–	–	–	–	–	–	–	–	–	–
frb59-26-4.clq	1534	89	59	–	–	–	–	–	–	–	–	–	–	–	–
frb59-26-5.clq	1534	89	59	–	–	–	–	–	–	–	11661	–	–	–	–
chv12x10.clq	120	92	20	–	4	5	4	1	1	8	0	4759	48	700	0
myc5x24.clq	120	97	48	14536	0	0	0	0	0	0	0	2	1	4	112
myc11x11.clq	121	93	22	–	8	12	7	2	1	7	0	1097	85	1081	239
s3m25x5.clq	125	89	20	5440	6	8	6	5	1	1	0	5	6	9	0
myc23x6.clq	138	87	12	1681	2	3	2	2	2	57	0	4	7	56	699
myc5x30.clq	150	97	60	–	1	1	1	0	0	0	0	10	2	47	42042
s3m25x6.clq	150	90	24	–	192	278	195	186	4	12	8	64	92	128	0
myc11x14.clq	154	94	28	–	486	566	422	66	33	235	23	–	11563	–	–
chv12x15.clq	180	94	30	–	26019	34161	26045	7796	1235	26798	184	–	–	–	0
myc5x36.clq	180	97	72	–	3	2	3	2	0	0	0	17	6	118	–
myc23x8.clq	184	90	16	–	115	165	112	88	215	23434	90	1138	1390	–	–
myc11x17.clq	187	95	34	–	40109	–	33957	5056	2378	43935	2375	–	–	–	3159
s3m25x8.clq	200	92	32	–	46253	–	44843	38987	181	1206	478	22778	18148	40089	0
myc5x42.clq	210	98	84	–	26	15	25	4	0	0	0	443	36	1414	–
myc11x20.clq	220	95	40	–	–	–	–	–	–	–	38519	–	–	–	–
myc23x10.clq	230	91	20	–	–	–	–	38104	26210	–	7545	–	–	–	–
chv12x20.clq	240	95	40	–	–	–	–	–	–	–	–	–	–	–	–
myc5x48.clq	240	97	96	–	23	22	25	13	0	0	0	319	39	3316	–
s3m25x10.clq	250	93	40	–	–	–	–	–	6980	44122	–	–	–	–	18
myc11x23.clq	253	95	46	–	–	–	–	–	–	–	–	–	–	–	–

Table 1: Running time results in seconds for the BHOSLIB and EVIL instances. The "–" sign indicates that the running times are exceeding the 12 hour limit.

https://doi.org/10.31449/inf.v43i2.1621 Informatica **43** (2019) 187–198 **187**

Mutual Information Based Feature Selection for Fingerprint Identification

Ahlem Adjimi and Abdenour Hacine-Gharbi
LMSE laboratory, University of Bordj Bou Arreridj, Elanasser, 34030 Bordj Bou Arreridj, Algeria
E-mail: adjimia@yahoo.fr, hacgharbi@yahoo.fr

Philippe Ravier
PRISME laboratory, University of Orleans, 12 rue de Blois, 45067 Orléans, France
E-mail: philippe.ravier@univ-orleans.fr, +0033238494863

Messaoud Mostefai
LMSE laboratory, University of Bordj Bou Arreridj, Elanasser, 34030 Bordj Bou Arreridj, Algeria
E-mail: mostefaimess@gmail.com

Keywords: fingerprint identification, feature selection, dimensionality reduction, mutual information, local binary patterns, local phase quantization, histogram of gradients, binarized statistical image features

Received: May 3, 2017

In the field of fingerprint identification, local histograms coding is one of the most popular techniques used for fingerprint representation, due to its simplicity. This technique is based on the concatenation of the local histograms resulting in a high dimension histogram, which causes two problems. First, long computing time and big memory capacities are required with databases growing. Second, the recognition rate may be degraded due to the curse of dimensionality phenomenon. In order to resolve these problems, we propose to reduce the dimensionality of histograms by choosing only the pertinent bins from them using a feature selection approach based on the mutual information computation. For fingerprint features extraction we use four descriptors: Local Binary Patterns (LBP), Histogram of Gradients (HoG), Local Phase Quantization (LPQ) and Binarized Statistical Image Features (BSIF). As mutual information based selection methods, we use four strategies: Maximization of Mutual Information (MIFS), minimum Redundancy and Maximal Relevance (mRMR), Conditional Info max Feature Extraction (CIFE) and Joint Mutual Information (JMI). We compare results in terms of recognition rates and number of selected features for the investigated descriptors and selection strategies. Our results are conducted on the four FVC 2002 datasets which present different image qualities. We show that the combination of mRMR or CIFE feature selection methods with HoG features gives the best results. We also show that the selection of useful fingerprint features can surely improve the recognition rate and reduce the complexity of the system in terms of computation cost. The feature selection algorithms may reach 98% of time reduction by considering only 20% of the total number of features while also improving the recognition rate of about 2% by avoiding the curse of dimensionality phenomena.

Povzetek: Analizirani so različni načini opisa in preiskovanja pri histogramskem kodiranju identifikacije prstnih odtisov.

1 Introduction

Biometric recognition has gained a considerable interest in the recent years because of the various applications in the large field of security. Security can be categorized in data access security (computer and mobile access, USB key, bank cards) or in person access security (forensic identification, ID access). Many technological solutions exist relying on distinctive biometric identifiers (e.g. fingerprints, face, iris or speech) each one having its own qualities. However, the most used biometric identifiers are the fingerprints due to their uniqueness, persistence, simplicity of acquisition and the availability of the electronic acquisition devices [1]. Indeed, the fingerprints are single to each person and they remain unchanged during all the life of the person.

Fingerprint recognition systems can be categorized into three main approaches: minutiae-based systems, image-based correlation systems and image-based distance systems [2]. For the first category, the fingerprint image must pass through several preprocessing steps to detect and extract some points of interest called minutiae: smoothing, local ridge orientation estimation, binarization, thinning, and minutia detection. The second category directly estimates the similarity between a test and a reference fingerprint pattern by the autocorrelation method. For the third category, global or local features are extracted from the fingerprint image such that the features also called descriptors retain most of the pertinent information representing the fingerprint. This kind of

fingerprint recognition systems is preferred in the case of low quality images, because it is difficult to extract reliable minutiae sets in this case [3]. A distance measure between a test and a reference fingerprint pattern or any other classifier are finally used for making a matching decision [3].

Within this last category, many descriptors have been proposed. These descriptors can be principally grouped into histogram-based features or linear transformed features. The descriptors of the first group exploit some statistical characteristics of the fingerprint by transforming the image into a histogram of fixed length like Local Binary Patterns (LBP), Gabor filter with Local Binary Patterns (GLBP) hybrid method [4], Local Phase Quantization (LPQ) [5], Histogram of Gradients [6] or Binarized Statistical Image Features (BSIF) [7] or Scale Invariant Feature Transform (SIFT) [8][9]. In the second group, the fingerprint image is transformed into a vector of different features extracted from the fingerprint image such as Discrete Cosine Transform (DCT) features [10], Gabor filters based descriptors [11][12] and Discrete Wavelet Transform (DWT) features [13][14][15][16].

In this work, we focus on the histogram-based fingerprint representation techniques such as LBP, LPQ, HoG and BSIF. Indeed, these techniques are very used for fingerprint recognition due to their simplicity. These techniques are based on the concatenation of the local histograms leading to a histogram of great dimension (*e.g.*1024 features for each fingerprint in the case of LBP), which requires long computing time, big memory capacity and requires a huge training dataset to model the classes. Practically, it has been observed that features addition can cause a performance degradation of the classifier if the number of data used for the classifier designing is too low relatively to the number of features [17][18]. This phenomenon called the curse of dimensionality leads to the phenomenon of "peaking" [19]. So it is desirable to keep the number of features as small as possible which is also of benefit for reducing computational cost in the fingerprint identification task and for avoiding memory obstruction too. Keeping a small number of features is a dimensionality reduction operation, which can be done with two approaches: the first approach is a features transformation in which the initial features set is replaced by a new reduced set using transformation algorithm like PCA (Principal Component Analysis), LDA (Linear Discriminant Analysis)…. The second approach is a features selection which selects the relevant features from the initial features set [20]. However, using a reduced set of features by transformation needs greater memory capacity and more computing time in the testing phase compared to using a reduced set of features obtained by selection algorithms [20] because the former requires computation of all the features before reduction. So, in the present work, we have considered the features selection algorithms to select the relevant bins of histograms for the histogram-based fingerprint representation techniques.

The feature selection methods are also divided into two categories, which are "*wrapper*" or "*filter*". In "*wrapper*" methods, the relevance measure for a features subset is the training/testing recognition rate of the used classifier. Consequently, the wrapper selection procedure makes the computational cost rapidly increase, because a new classifier has to be built with training and testing phases each time a features subset is tested. Moreover, the features selected by wrapper methods are adapted to the used classifier, so their performance results are dependent on the type of classifier. In contrast, "*filter*" methods evaluate the features subset relevance independently of the classifier, so the selected features can be used for any classifier modelling [20][21]. For all these reasons, we have chosen the "*filter*" methods, which are the preferable methods in the case of high dimensionality and large datasets for computational reasons.

The "*filter*" methods use a selection criterion typically based on information theory tools like Mutual Information (MI) useful for measuring the quantity of information that features may have for describing the data. To our knowledge, only few works have investigated the MI based criteria in the field of biometric identification.

In [22], an efficient code selection method for face recognition is presented and compact LBP codes are obtained. The code selection is based on the maximization of mutual information (MMI) between features (LBP codes) and class labels. Applying this principle for selection is achieved by using the max-relevance and min-redundancy (mRMR) criterion. The method proposed consists of transforming the face images into LBP histograms, then selecting the relevant codes from these histograms using the maximization of the mutual information. In this work the authors have used the chi-square formula for measuring the distance between the histograms of the reference and the test templates.

In [23], the BSIF features have been investigated in the frame of a fingerprint recognition system, with preliminary results of feature selection using the FVC2002 fingerprint dataset [24]. The experiments have shown that an increasing number of extracted sub-images leads to an increasing recognition rate, but also leads to higher dimension histograms which decreased accordingly performance of the system regarding computing time and memory capacity. This motivated the use of MI feature selection strategy, namely interaction capping (ICAP).

In this work, we extend the fingerprint recognition system proposed in [23] by considering more datasets within the FVC2002 fingerprint database, more descriptor types and by investigating several other feature selection strategies, all based on mutual information computation to select the relevant bins of histograms that are extracted from the fingerprint images. The present study will focus on robustness of the fingerprint system regarding various descriptors and noisy datasets. The main aim of this work is to find a combination of feature selection method with a pertinent descriptor type in a larger context than in study [23]. To that aim, next section introduces the former developments of [23] and explains the novelty of the present paper comparatively. Section 3 proposes a brief review of all the descriptors used in this paper. Section 4 describes the feature selection methods based on mutual information. In section 5 we present the experimental

procedure and we discuss the obtained results using a public fingerprint dataset in section 6. Finally, we draw a conclusion in section 7.

2 Related work

In our previous works [23] and [25], a fingerprint recognition system was created following the flowchart of Fig. 1. A sequence of many preprocessing steps were applied on the training and testing image datasets before extracting the LBP, LPQ or BSIF features, namely enhancement, alignment, extraction of the region of interest (ROI) around the core point and division of the ROI into sub-regions. This procedure is detailed in [23]. So the set of sub-regions are inputs for the features computation. In [25], we used the novel BSIF descriptor [7] compared with LBP and LPQ descriptors, for fingerprint images. From each sub-region, a histogram of BSIF is extracted and the final feature vector is obtained by concatenating all BSIF histograms extracted from the sub-regions. In [23] an extended work of this previous work was presented, in which the relevant bins of the BSIF descriptor extracted histograms were selected using ICAP features selection method. The last step of Fig. 1 is the decision making. It is based on the distance between the histograms of the reference fingerprints and the tested one. The distance is computed as a chi-square measure which formula is given below [22]

$$\chi^2(R,T) = \sum_{i=1}^{n} \frac{(R_i - T_i)^2}{R_i + T_i} \quad (1)$$

where R_i and T_i are the reference and the tested fingerprint histogram magnitudes respectively and n is the number of bins.

The recognition system uses the following rule to make a decision: if a test fingerprint gives the best match for the fingerprint of the same person it is declared to be a correct match; else it is declared to be a false match.

The recognition rate is computed as

$$Recognition\ rate\ (\%) = \frac{\text{number of correctly recognized images}}{\text{number of test images}} \times 100, \quad (2)$$

In the current paper, many extensions are proposed with respect to our former work [23]. The purpose is to evaluate the robustness of the system regarding changes in the datasets, depending on the descriptors type. We thus consider the new descriptor histogram of gradients (HoG). Then all the descriptors LBP, LPQ, HoG and BSIF are evaluated on all the datasets DB1, DB2, DB3, DB4 of the FVC2002 fingerprint dataset [24]. Indeed, the DB2 and DB3 datasets were discarded for the preliminary study in work [23] while interesting for a robustness study because these are noisy datasets. Moreover, four MI strategies instead of only one in work [23] are investigated for achieving a comparison between them, also by considering the four descriptors instead of BSIF only as proposed in [23]. These novelties are described in the flowchart of Fig. 2. Furthermore, the impact of feature selection on computing time is analyzed. A deep performance analysis of the dimensionality reduction procedure is also proposed.

The parameter values of the fingerprint recognition system depicted in Fig. 2 will be given in section 5.2 of the experimental part.

3 A brief review of descriptors LBP, LPQ, HoG and BSIF

In this section we give a brief review of the descriptors LBP, LPQ, HoG and BSIF used in this work for features extraction.

3.1 LBP (Local Binary Patterns)

This operator was proposed by Ojala et al [26] for texture analysis. It is characterized by its tolerance to illumination changes, its computational simplicity and its invariance against changes in gray levels. The LBP descriptor works on eight neighbors of a pixel and uses the gray value of this pixel as a threshold; thus, if a neighbor pixel has a higher or a same gray value than the center pixel then a binary one is assigned to that pixel, else it gets a binary zero. The LBP code for the center pixel is then produced by concatenating the eight ones or zeros to obtain a binary number that is transformed after that to a decimal number. The LBP code has a certain value from 0 to 255. Therefore, a histogram of 256 bins is composed from these values and used for matching.

3.2 LPQ (Local Phase Quantization)

This texture descriptor was originally proposed by Ojansivu and Heikkila [27]. It is based on the blur invariance property of the Fourier phase spectrum. It has shown good performance in recognition of textures even when there is no blur and outperforms the Local Binary Pattern operator in texture classification. It uses the local phase information extracted using the 2-D local Fourier transform computed over a window of size (2R+1) by (2R+1) neighborhood at each pixel position in image of size n by n. For LPQ, only four complex coefficients corresponding to 2-D spatial frequencies $v_1 = [a, 0], v_2 = [0, a], v_3 = [a, a]$ and $v_4 = [-a, a]$ where $a = \frac{1}{2R+1}$ are retained. The real and the imaginary parts of the complex values are stacked in a vector of 8 components for each pixel which gives a matrix of size 8 by n x n. Then, the coefficients are decorrelated by a whitening operation assuming a correlation coefficient of 0.95 between adjacent pixel values and a Gaussian distribution of the pixel values. Finally, this matrix is binarized by looking the sign of each element, so that if it has a positive value, a binary 1 is assigned to that element otherwise a binary 0 is assigned. The last step is the histogram construction by transforming each column of 8 elements to a decimal value between 0 and 255. Finally a 256-dimensional histogram is composed from these values and used in classification.

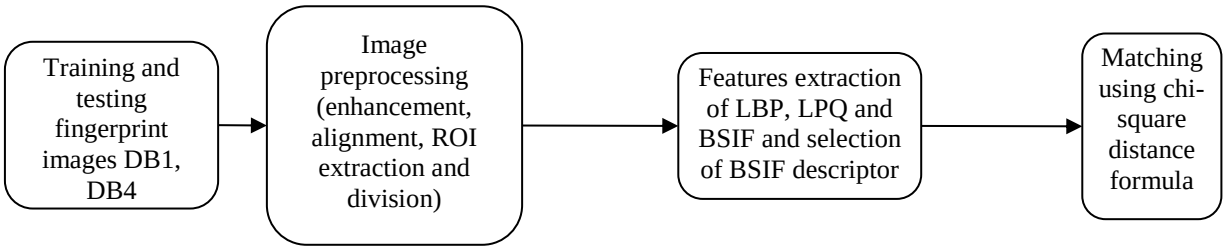


Figure 1: Flowchart of the related work system of fingerprint recognition.

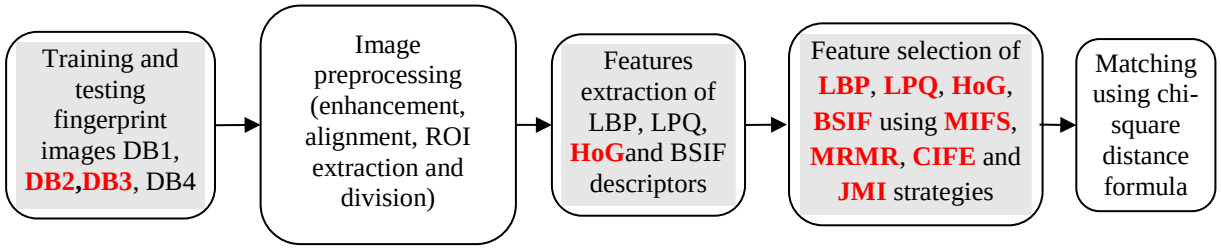


Figure 2: Flowchart of the proposed system. The red characters indicate the added elements for a deep study of the system (details of image preprocessing and matching steps can be found in reference [23]).

3.3 HoG (Histogram of Gradients)

The HoG descriptor has been first proposed by Dalal and Triggs [28] as an image descriptor used in computer vision and image processing for object detection. The basic idea of this descriptor is that local object appearance and shape can be characterized rather well by the distribution of local intensity gradients. The gradient filter is applied in both directions x and y of the image. The two obtained images are then transformed in magnitude and orientation gradients. After, they are divided into small spatial regions (cells). For each cell, each pixel has a gradient magnitude which accumulates the distribution at the bin corresponding to its orientation value. The concatenation of these histograms gives the HoG histogram. For example, if the number of orientation bins spaced over 0° - 180° is 9 (180°/20°) and the image is split into 3x4 cells (12 is the total number of cells), we then obtain a histogram of G with 3x4x9=108 bins. Actually, the obtained histogram is not a genuine one since the bins cumulative does not reach the total number of pixels. A histogram-like is finally obtained with sqrt L2-normalization [28].

3.4 BSIF (Binarized Statistical Image Features)

BSIF is a new descriptor recently proposed by Kannla&Rahtu [7] for texture classification and face recognition. Its main idea is that it automatically learns a set of filters from a small set of natural images instead of using manual filters such as in LBP and LPQ descriptors. BSIF is a binary code string which length is the number of filters. Each bit of the code string is computed by binarizing the response of the image to a linear filter from the set with a fixed threshold. Given an image patch X of size $l \times l$ pixels and the #i linear filter W_i of the same size from the set of learned filters, the response s_i is obtained by

$$s_i = \sum_{u,v} W_i(u,v)X(u,v) = w_i^T x, \quad (3)$$

where vectors w_i and x contain the pixels of W_i and X. The binarized feature b_i is obtained by setting $b_i = 1$ if $s_i > 0$ and $b_i = 0$ otherwise [7]. The BSIF descriptor depends on two parameters which are the filter window size and the number of bits representing the binary code string. So, the number of bits determines the number of extracted features. If the binary code string is represented with 8 bits, we get 256 features vector, which means a histogram of BSIF features of 256 bins.

4 Feature selection using Mutual Information

Feature selection is used to identify the useful features and remove the features that are redundant and irrelevant for the task of classification. For this reason, it is necessary to reach a measurement of features relevance which makes it possible to quantify their importance in this task. In this section we briefly give some basic concepts and notions from information theory that are useful for understanding the four feature selection methods used in this work. In information theory, MI measures the statistical dependence between two random variables. So, MI can be used to evaluate the relative utility of each feature to classification, in which entropy and mutual information are two principal concepts.

Entropy H can be interpreted as a measure of the uncertainty of random variables. Let X be (or represent) a discrete random variable with probabilistic distribution p(x). The entropy of X is defined as [29]:

$$H(X) = -\sum_{x \in X} p(x) \log\big(p(x)\big) \quad (4)$$

The mutual information MI between two discrete variables X and Y is defined using their joint probabilistic distribution $p(x,y)$ and their respective marginal probabilities $p(x)$ and $p(y)$ as:

$$MI(X;Y) = \sum_{x\in X}\sum_{y\in Y} p(x,y)\log\frac{p(x,y)}{p(x)p(y)} \quad (5)$$

The objective of using MI is to select a subset S of relevant features from a set F of features, which share the most information with the class variable. The treatment of each feature needs a very big number of possible subsets (combination C_k^n), this leads to the iterative "greedy" algorithms which select the relevant features one by one (sequential forward selection) or deletes the unneeded features (sequential backward selection). The use of the greedy forward selection procedure with the MI based relevance criterion is generally a good choice of feature selection procedure [30].

The Forward ‘‘greedy’’ algorithm based on MI is presented as follows [31][32]:

1) (Initialization) set F ←“initial set of n features”; S ← “empty subset”
2) (Calculation of MI), $\forall f_i \in F$, calculate$MI(C; f_i)$.
3) (Choose the first feature f_{s_1}), find the feature that maximizes $MI(C; f_i)$, affect $F \leftarrow F - \{f_{s_1}\}$, $S \leftarrow \{f_{s_1}\}$.
4) (Greedy selection), repeat until the desired number of features:
   a. (Compute MI between features), $\forall f_i \in F$, compute $MI(C; S, f_i)$.
   b. (Select the next feature f_{s_j}), choose the feature $f_i \in F$ that maximizes $MI(C; S, f_i)$ at the step j, affect $F \leftarrow F - \{f_{s_j}\}$, $S \leftarrow S \cup \{f_{s_j}\}$.
5) Take out the subset S of the selected features.

Practically, it is difficult to compute $MI(C; S, f_i)$ when the cardinal of the subset S increases because it requires an estimation of high dimension probability density functions, which cannot be correctly estimated with a limited number of samples [20]. So the majority of the algorithms use measurements which are maximally based on three variables: two features plus the class index. For this reason, many proposed criteria based on MI are heuristic [32][33].

As previously stated, “*filter*”methods are preferred to wrapper ones. These methods are defined by a criterion J, also called relevance index or scoring criterion, which is planned to measure the relevance of a feature or a feature subset for the task of classification. The simplest feature-scoring criterion is referred as MIM (Mutual Information Maximization) [21]:

$$J_{mim}(f_i) = MI(C; f_i) \quad (6)$$

The J_{mim} criterion does not include the features already selected which leads to selecting redundant features (sharing the same information with the class index C) that must be eliminated. Numerous “*filter*”criteria have been proposed taking into account the redundancy [33][32]. We use four criteria in this work: MIFS, mRMR, CIFE and JMI [21].

4.1 Mutual Information Feature Selection strategy (MIFS)

Proposed by Battiti [31], it is very useful in feature selection problems and classifying systems due to its simplicity. MIFS selects the feature that maximizes the information about the class label C, and subtract the MI between features f_i and the already selected variable f_j to achieve the minimum redundancy:

$$J_{mifs}(f_i) = MI(C; f_i) - \beta\sum_{f_j\in S} MI(f_i; f_j) \quad (7)$$

In this latter expression, S stands for the set of already selected features.

The parameter β is a configurable parameter that determines the degree of redundancy checking within MIFS. It must be set experimentally [21][34]. The performance of MIFS degrades if there are many irrelevant and redundant features because it penalizes redundancy too much.

4.2 Minimum Redundancy and Maximal Relevance strategy (mRMR)

Proposed by Peng et al [35], it is equivalent to MIFS with $\beta = \frac{1}{|S|}$ where $|S| = \text{card}(S)$ is the number of already selected features. It finds a balance between the relevance, which is the dependence between the features and the class, and the redundancy of features with respect to the subset of previously selected features. The criterion can be written as:

$$J_{mrmr}(f_i) = MI(C; f_i) - \frac{1}{|S|}\sum_{f_j\in S} MI(f_i; f_j). \quad (8)$$

With the minimum redundancy criterion of mRMR method, we can get more representative features of the class variable, which are maximally dissimilar to already selected ones, so it gives a small number of features which effectively covers the same space as a larger number of features.

4.3 Conditional Infomax Feature Extraction strategy (CIFE)

Lin and Tang [36] proposed a criterion, called *Conditional Infomax Feature Extraction*, in which the joint class-relevant information is maximized by explicitly reducing the class-relevant redundancies among features [33]. Note that this criterion has been proposed by several authors in different ways [20][32][33][37]:

$$J_{cife}(f_i) = MI(C; f_i) - \sum_{f_j\in S} MI(f_i; f_j) + \sum_{f_j\in S} MI(f_i; f_j|C). \quad (9)$$

	Technology	Scanner	Size of image (pixel × pixel)	Set A	Set B	Resolution
DB1	Optical	IdentixTouchView II	388×374	100 persons with 8 impressions per person (800)	10 persons with 8 impressions per person (80)	500 dpi
DB2	Optical	Biometrika FX2000	296×560			569 dpi
DB3	Capacitive	Precise Biometrics 100 SC	300×300			500 dpi
DB4	Synthetic	SFinGEv2.51	288×384			About 500 dpi

Table 1: The technologies and scanners used to collect the FVC2002 datasets and the size of images in each dataset.

The CIFE criterion is same as MIFS plus the conditional redundancy term.

4.4 Joint Mutual Information strategy (JMI)

Proposed by Yang and Moody [38], the Joint Mutual Information score is

$$J_{jmi}(f_i) = MI(C; f_i) - \frac{1}{|S|}\sum_{f_j \in S}[MI(f_i; f_j) - MI(f_i; f_j|C)] \quad (10)$$

JMI method studies relevancy and redundancy by taking the mean value, and takes into consideration the class label when calculating MI. JMI and mRMR are very similar but the difference is the conditional redundancy term.

5 Experimental procedure

First, we give a brief description of the public fingerprint dataset FVC2002 [24]. Second, we present the experimental parameters chosen for our fingerprint recognition system. Third, we describe the way we select the relevant bins from LBP, LPQ, HoG and BSIF histograms using the Brown's toolbox for feature selection [21].

5.1 Datasets

The experimental results have been conducted on the FVC2002 fingerprint dataset [24], which has been divided into two sets A and B. Each set is divided in 4 datasets DB1, DB2, DB3 and DB4. Three different scanners and the SFinGe synthetic generator were used to collect the fingerprints [24]. A total of 120 fingers and 12 impressions per finger (1440 impressions) using 30 volunteers have been collected. The top-ten quality fingers were removed from each dataset since they do not constitute an interesting case study [24]. The size of each dataset in the FVC2002 test, however, was established as 110 fingers, 8 impressions per finger (880 impressions) and split into set A (100 fingers - evaluation set) and set B (10 fingers - training set). To make set B representative of the whole dataset, the 110 collected fingers were ordered by quality, and then the 8 images from every tenth finger were included in set B. The remaining fingers constituted set A. In this work, we have used set A to conduct our experimental results [6].

Table 1 presents the technologies and the scanners used to collect the FVC2002 datasets and the size of images in each dataset for each set.

5.2 Fingerprint recognition system

This section describes the experimental parameters chosen for our fingerprint recognition system.

The related work in section 2 mentioned the region around the core point of the fingerprint image. The region of size (100x100 pixels) is extracted and divided into 4 sub-regions of size (50x50 pixels) for each one. For features extraction we use the four descriptors LBP, LPQ, HoG and BSIF applied for each sub-region.

- For LBP features extraction, we convert the gray value of each pixel to one of the 256 LBP codes. Next we construct the histogram of LBP codes.
- For LPQ we use a radius equal to 3, so a histogram of 256 bins is extracted.
- For HoG, each sub-region is divided into sub windows of 3 rows and 3 columns (9 cells total). The orientation and magnitude of each pixel is calculated. The absolute orientation is divided into 9 equally sized bins, which results in a 9-bin histogram per each of the 9 cells, so a histogram of 81 bins is produced.
- For BSIF we use a filter of 11x11 size and number of bits equal to 8 to extract a histogram of 256 bins. The learnt filters are provided by [7].

For each region, the histograms of LBP, LPQ, HoG and BSIF are extracted independently and concatenated to construct the final normalized histogram for each descriptor. The LBP, LPQ, HoG and BSIF histograms are extracted using SIfingToolbox[1]. For LBP, BSIF and LPQ features, the normalization is carried out by dividing the value of each bin of the histogram by the sum of the values of the bins of this histogram. For HoG features, the normalization is done with sqrt L2-normalization as stated in [28].

Table 2 presents the number of bins in each extracted histogram for the different descriptors.

In this work, the first results are obtained by training the system over 7 images of each person for each dataset. That is, we use 700 dataset images for training and use remaining 100 dataset images for testing for each dataset. In the experiments, the 8 fold-cross validation was applied, so the test step was repeated 8 times.

[1]https://www.dropbox.com/s/wregrs3ah0qcfdd/SIfing.rar

Feature extraction method	Number of regions around the core point	Number of histogram bins
LBP	4 regions of size 50x50	256*4=1024
LPQ		256*4=1024
HoG		81*4=324
BSIF		256*4=1024

Table 2: Number of histogram bins for each descriptor.

5.3 Bins selection

Table 2 shows that the number of extracted features is high (histogram of 1024 in the case of BSIF, LBP and LPQ and 324 in the case of HoG) which makes the response time in the matching stage very long. The dimensionality reduction is achieved by a feature selection stage. To that aim, we have used the Brown's Toolbox (FEAST toolbox)[2], which contains the implementation of 13 different features selection methods based on mutual information. In our case we have only used 4 feature selection methods. Two of them are based on the redundancy (MIFS and mRMR). The two other ones are based on the conditional redundancy (CIFE and JMI).

Practically, the LBP, LPQ, BSIF and HoG histogram bins are extracted from all the training images that are also used for feature selection. At this point, each bin is considered as a feature in the feature selection process. This means that each feature is a random variable which probability density function can be estimated with a histogram construction using many realizations of the variable, each image being associated to a realization. Building the histogram of features necessitates the magnitude variation ranges to be properly discretized. This step is required for a low biased estimation of mutual information and entropies used in the Brown's Toolbox. Now, we assume that the number of images is Nwhich is the number of samples or realizations used for histogram estimation of the features. The number m of bins representing the histogram for each feature can be obtained by Sturges' formula [39]:

$$m = log_2(N) + 1 \quad (11)$$

6 Results and discussion

6.1 Impact of the descriptor type on classification performance

In this section, we analyze performance results of the proposed descriptors for the fingerprint recognition task. Performance is measured in terms of recognition rates and computing time for the identification stage. Table 3 shows the recognition rates and the computing time with all extracted features obtained for each descriptor applied on the different datasets. It is clearly shown from Table 3 (a) that the LBP features provide the poorest recognition rates compared to the other descriptors in all datasets with an about 10% drop in the recognition rate by comparison with the other rates. The BSIF descriptor gives the best recognition rates except in the DB2 dataset. For all the datasets, the HoG and LPQ descriptors give approximately the same results. It is also observed that DB3 dataset gives the poorest recognition rates. This is due to the fact that DB3 is the most difficult dataset among the four datasets in FVC2002 in terms of image quality [40]. Mainly it can be concluded that the HoG and LPQ descriptors are robust with respect to the dataset diversity because of general high recognition rates compared to the other descriptors. This is confirmed by an average rate over the four datasets reaching near 86.8% for both descriptors. Conversely, BSIF also reaches an average rate of 86% but with extreme values with the highest rates for three datasets and the poorest rate for one dataset. From Table 3 (b), it is clearly shown that the HoG descriptor requires less computing time than the other descriptors for the identification stage. This is due to the smaller number of histogram bins required for this method. Moreover, the computing time is rather independent of the tested dataset. So generally, we can conclude that HoG features outperform the other used features in terms of calculation complexity (only 324 features) and in recognition rate.

A natural perspective is to deal with higher dimension datasets and/or real-time recognition systems. This requires keeping the number of the extracted features as small as possible, which implies computational and memory cost reductions for the training and testing stages. For this reason, many feature selection algorithms have been investigated to solve the problem of computational and memory cost reduction.

6.2 Impact of the feature selection algorithm on classification performance

Fig. 3 shows the results obtained by the four feature selection methods (MIFS, mRMR, CIFE and JMI) on the four datasets DB1, DB2, DB3 and DB4 and with all the descriptors.

The results obtained with LPQ features are very close to those of HoG and BSIF, like observed in the previous study [23] with LBP also giving the poorest results. It can be noted that all the curves reach approximately a plateau as soon as 20% of the total number of features are selected by any of the selection algorithm except MIFS. A first conclusion is that dimensionality feature reduction can be achieved for all the datasets. In many cases, the MIFS algorithm shows an abrupt change at the beginning of the curve. Among the feature selection algorithms, the mRMR is slightly better than the other ones in average over all the datasets.

The curse of dimensionality phenomenon can clearly be observed with DB3 and DB4 datasets in Fig.3, where higher recognition rates can be reached with a smaller number of features than the maximal one. However, the

[2]http://www.cs.man.ac.uk/~gbrown/fstoolbox/

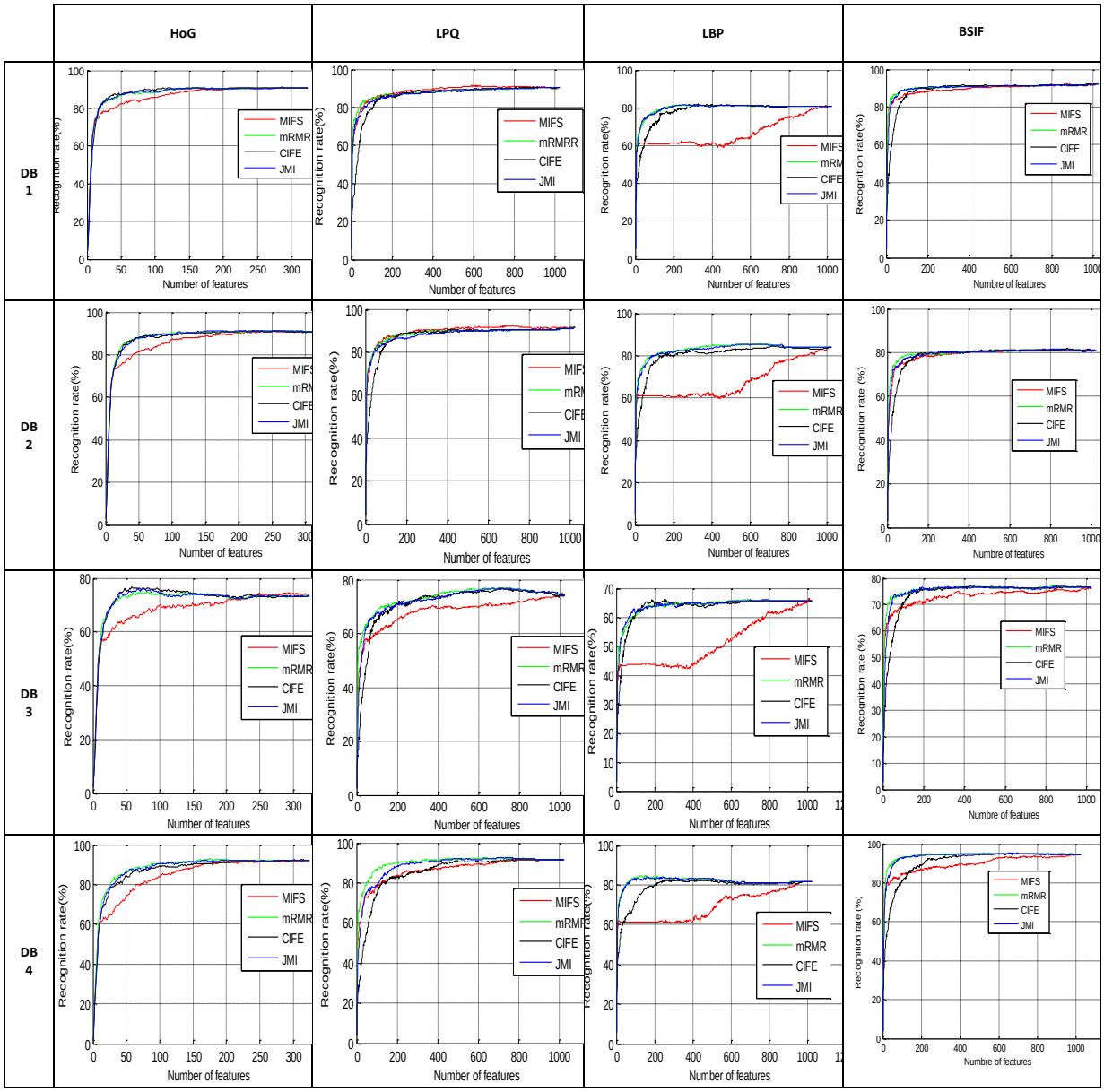


Figure 3: Recognition rates on all the four datasets using HoG, LPQ, LBP and BSIF selected features and using MIFS, mRMR, CIFE and JMI feature selection strategies.

(a)	DB1	DB2	DB3	DB4
HoG	90.75	90.86	73.25	**92.13**
LPQ	90.25	91.25	74.13	**91.50**
LBP	80.75	**84.00**	65.75	81.38
BSIF	92.25	80.75	76.37	**94.50**

(b)	DB1	DB2	DB3	DB4
HoG	563	569	554	**564**
LPQ	10350	10493	10304	**10378**
LBP	10161	**10219**	10253	10256
BSIF	11381	10905	10609	11257

Table 3: **(a)** Recognition rate results (%) **(b)** Computing time results (s) with HoG, LBP, LPQ and BSIF features on the four FVC 2002 datasets.

(a)	DB1	DB2	DB3	DB4
HoG	96.44	96.48	96.39	96.45
LPQ	98.08	98.15	98.11	98.09
LBP	98.08	98.09	98.09	98.11
BSIF	98.01	97.79	97.84	97.94

(b)	DB1	DB2	DB3	DB4
HoG	2.62	2.19	**-2.73**	4.88
LPQ	4.55	5.2	4.4	3.55
LBP	0	2.98	3.04	**-1.99**
BSIF	1.76	1.55	0.65	0.66

Table 4: **(a)** Reduction Rate (%) of computing Time **(b)** Loss of Recognition Rate (%) caused by dimensionality reduction.

phenomenon of peaking can be far more significant in some curves without cross-validation. Indeed, the curves of Fig.3 are the result of cross-validation which makes an average of 8 recognition rate curves. This operation may mask outlier curves. As an example, we consider a case without cross-validation with HoG features on DB3 by taking the 7th image as a test image and the remainder images as references. From Fig.4, the CIFE algorithm allows 74% of recognition rate to be attained by selecting 28 HoG features which is far better than the recognition rate of 66% obtained with all the features (324).
Note in addition that such a case corresponds to the practical use of a feature selection algorithm because of averaging effect of the cross-validation process, which prevents delivering a common sequence of selected features.

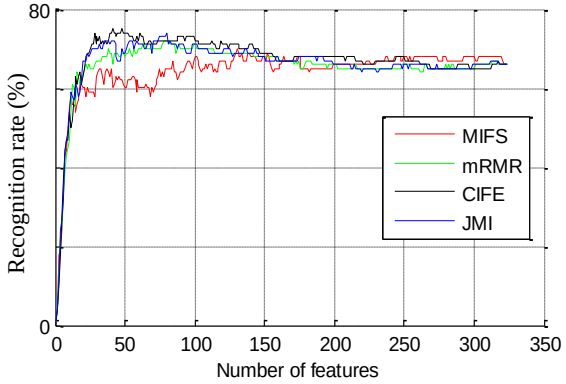


Figure 4: The curse of dimensionality phenomenon (peaking) for DB3 dataset with HoG selected features.

6.3 Impact of feature selection on computing time

In this section, we evaluate the benefit of the selection procedure on the complexity of the system in terms of computing time and its effect on the recognition rate of the system. For this experiment, we use the JMI features selection method.

Table 4(a) presents the Reduction Rate of the computing Time (RRT) given as follow:

$$RRT = (TF - TS)/TF \tag{12}$$

where TF is the computing Time corresponding to number of Full features and TS is the computing Time corresponding to the number of Selected features.

Table 4(b) presents the Loss of Recognition Rate (LRR) caused by the dimensionality reduction. This is given by:

$$LRR = (RRF - RRS)/RRF \tag{13}$$

where RRF is the Recognition Rate corresponding to the number of Full features and RRS is the Recognition Rate corresponding to the number of Selected features.
In this experiment, we consider the first 20% of the selected features w.r.t. to the full number of features.

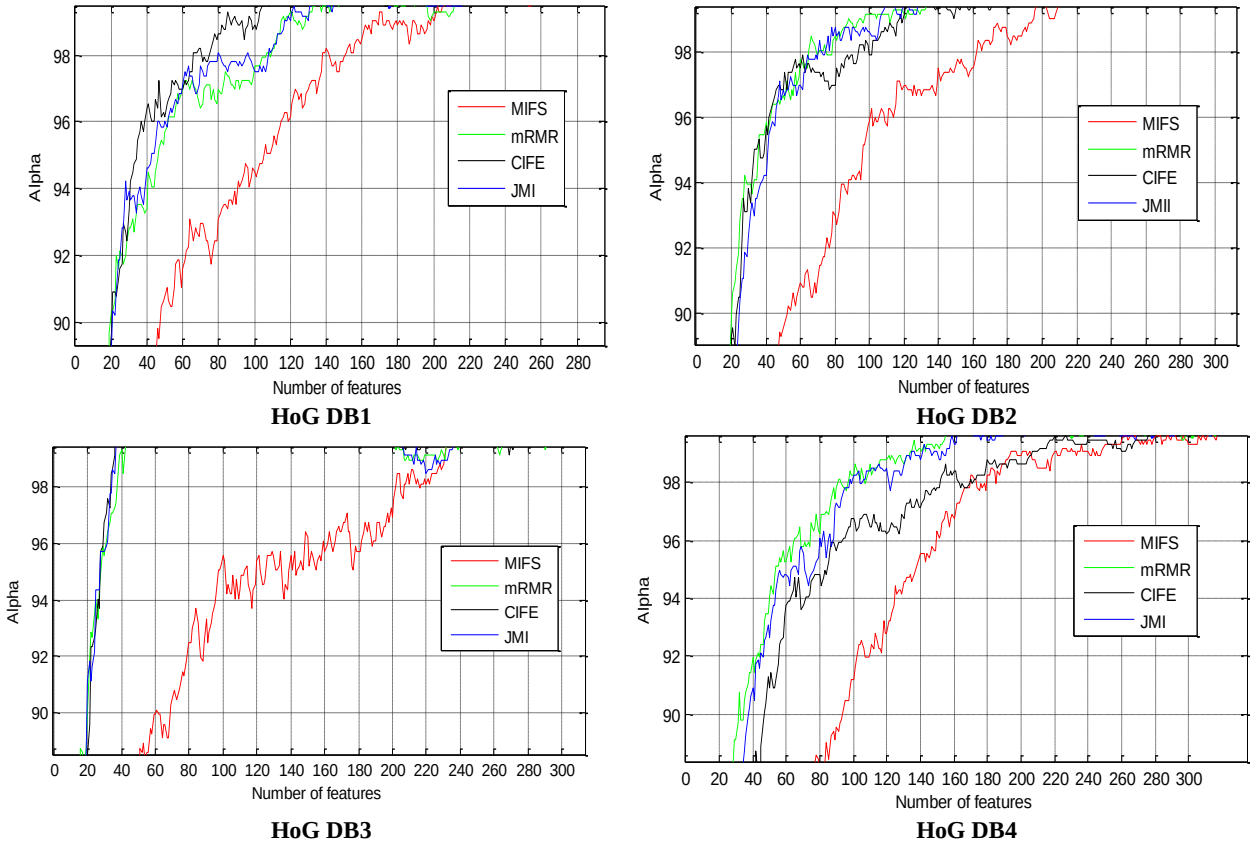


Figure 5: Number of HoG selected features with ***alpha*** = {90%….99%} on all datasets, using MIFS, mRMR, CIFE and JMI features selection strategies.

From table 4(a), it can be concluded that considering 20% of BSIF, LBP or LPQ selected features improves the computation time of about 98% compared to the computation time needed with the full number of features. Table 4(b) indicates that the loss of recognition rate may grow up to about 5% while some cases may improve the recognition rate (1.99% when selecting 20% of LBP features with DB3 or 2.73% when selecting 20% of HoG features with DB4 respectively).

6.4 Performance analysis of the dimensionality reduction procedure

It is interesting to know to what extent the number of features could be decreased by considering a small degradation of the recognition rate. For this experiment, we thus determine the number of selected HoG features that allows a recognition rate greater than an $alpha$ percent value of the rate obtained with the minimum number of features using the formula

$$alpha = \frac{RRS}{RRF} * 100 \quad (14)$$

where RRS is the recognition rate corresponding to the selected features. RRF is the recognition rate obtained with all the features. The *alpha* parameter can take values from 0% to 100%. Fig.5 reports the number of HOG selected features corresponding to $alpha$ values located in {90%...99%}. From these results, it can be observed that the three feature-selection methods mRMR, CIFE and JMI give very close results, unlike MIFS that always shows poorer performance except in the case of DB3. It can also be observed that CIFE seems to show better results in the case of real bases (DB1, DB2 and DB3) with respect to the synthetic base (DB4). The number of features can be strongly reduced for DB3 with very little concession on the recognition rate (for example 34 features with CIFE are sufficient with $alpha$=98%), the profit being very weak for smaller $alpha$ values. On the other hand, willing to keep the same number of features (34) with the other bases, it is necessary to go down to $alpha$= 94% for DB1, 95% for DB2 and less than $alpha$= 90% for DB4 (with mRMR).

Table.5 presents the optimal number of BSIF, HoG, LPQ and LBP selected features by the used feature selection methods with $alpha$=98%. Table.6 presents their corresponding recognition rates.

From Tables 5 and 6, the following points can be highlighted:

- For DB1 and DB3, the combination of HoG features with the feature selection method CIFE gives the best performance results with a reduced number of 66 features in the case of DB1 and 34 features in the case of DB3.
- For DB2 and DB4, the combination of HoG features with the feature selection method mRMR gives the best performance results with a reduced number of 66 features in the case of DB2 and 91 in the case of DB4.
- For DB4, using LBP features with feature selection method mRMR gives a reduced number of features equal to 48 but with a poor recognition rate compared to HoG and LPQ. The best performance result is obtained with 87 BSIF features.

As a conclusion, the two feature-selection methods mRMR and CIFE allow obtaining the reduced number of the features in the majority of cases.

7 Conclusion

Histogram based techniques are very used for fingerprint image representation. Generally, concatenation of the histograms leads to the problem of high dimension, which degrades performance results of the identification system in terms of complexity (computing time and memory cost) and recognition rate. In this paper, we have deeply studied the problem of dimensionality reduction in a fingerprint identification system in order to reduce the complexity with possible improvement of the recognition rate avoiding the curse of dimensionality phenomenon. We have presented a fingerprint recognition system based on 4 descriptors: local binary pattern (LBP), local phase quantization (LPQ), Histogram of gradients (HoG) and Binarized Statistical Image Features (BSIF). For the dimensionality reduction we used 4 feature selection methods based on mutual information: MIFS, mRMR, CIFE and JMI. The experiments were conducted on the public FVC 2002 fingerprint dataset.

The use of several types of features and several datasets allows efficiently to validate the feature selection

	BSIF				HoG				LPQ				LBP			
	MIFS	**mRMR**	**CIFE**	**JMI**	**MIFS**	**mRMR**	**CIFE**	**JMI**	**MIFS**	**mRMR**	**CIFE**	**JMI**	**MIFS**	**mRMR**	**CIFE**	**JMI**
DB1	425	202	**176**	201	138	107	**66**	80	**261**	472	313	448	918	144	220	**137**
DB2	274	**113**	152	194	162	**66**	94	75	**234**	303	255	411	953	**207**	472	222
DB3	363	**121**	152	124	202	38	**34**	35	845	303	**290**	348	950	260	**150**	216
DB4	589	90	297	**87**	170	**91**	152	98	653	**184**	425	248	932	**48**	197	52

Table 5: Number of BSIF, HoG, LPQ and LBP selected features with $alpha$ =98%. The green values correspond to the minimum number of selected features with a 98% degradation acceptance with respect to the rate obtained with all the features.

	BSIF				HoG				LPQ				LBP			
	MIFS	**mRMR**	**CIFE**	**JMI**	**MIFS**	**mRMR**	**CIFE**	**JMI**	**MIFS**	**mRMR**	**CIFE**	**JMI**	**MIFS**	**mRMR**	**CIFE**	**JMI**
DB1	90	90.37	**90.10**	90.37	89	89	**89**	89	**88.5**	88.5	88.5	88.62	79.38	79.38	79.25	**79.38**
DB2	79	**79.12**	79	79.12	89.10	**89.5**	89.25	89.25	**89.5**	89.5	89.5	89.5	82.83	**82.83**	82.5	82.38
DB3	74.74	**74.75**	74.75	74.75	71.8	72.25	**72.10**	72.25	72.75	72.75	**72.75**	72.87	64.5	64.63	**64.75**	64.5
DB4	92.5	92.5	92.6	**92.5**	90.5	**90.37**	90.37	90.30	89.75	**89.75**	89.87	89.75	79.75	**79.75**	80.25	79.75

Table 6: Recognition rates obtained by BSIF, HoG, LPQ and LBP selected features with $alpha$ =98%. The green numbers are those giving the smallest numbers of selected features.

techniques and to choose the best combination (type of features/feature selection method) for the task of fingerprint identification. From all the results we can conclude that the use of feature selection methods can reduce the number of features whatever the type of features and whatever the dataset, except in the case of using MIFS with LBP features that present bad performance result. We can conclude also that the feature selection techniques can reduce the curse of dimensionality phenomenon and probably improve the recognition rate of the identification system. The combination of HoG features with CIFE or mRMR gives the best performance in terms of recognition rate, robustness and complexity of the system. In terms of complexity, a huge computation time reduction (98%) is obtained by considering only 20% of the total number of features without much affecting the recognition rate.

In definitive, employing feature selection algorithms will always provide a benefit when compared to no selection since higher or equal identification performance can be obtained and at the same time the computation complexity for the identification stage can be reduced. As perspective, we plan to investigate other descriptors and biometric modalities.

References

[1] D. Maio, D. Maltoni, A. K. Jain and S. Prabhakar, " *Handbook of fingerprint recognition*," Springer, New York, NY, 2003. https://doi.org/10.1007/b97303

[2] K. S. Sunil, "A Review of Image Based Fingerprint Authentication Algorithms," *International Journal of Advanced Research in Computer Science and Software Engineering*, vol. 3, no. 6, pp. 553-556, 2013.

[3] Y. Jucheng, "Non-minutiae based fingerprint descriptor," in *Biometrics*, Nanchang, In Tech, 2012, pp. 80-98. https://doi.org/10.5772/21642

[4] N. Nanni and A. Lumini, "Local Binary Patterns for a hybrid fingerprint matcher," *Pattern Recognition*, vol. 41, no. 11, pp. 3461-3466, 2008. https://doi.org/10.1016/j.patcog.2008.05.013

[5] S. Brahnam, C. Casanova, L. Nanni and A. Lumini, "A Hybrid Fingerprint Multimatcher," in *16th International Conference on Image Processing, Computer Vision, and Pattern Recognition*, Las Vegas, Nevada, USA, pp. 877-882, 2012.

[6] L. Nanni and A. Lumini, "Descriptors for image-based fingerprint matchers," *Expert Systems With Applications*, vol. 36, no. 10, pp. 12414-12422, 2009. https://doi.org/10.1016/j.eswa.2009.04.041

[7] J. Kanala and E. Rahtu, "BSIF: binarized statistical image features", *21st International Conference on Pattern Recognition (ICPR 2012)*," IEEE, Tsukuba, Japan, pp. 1363-1366, 2012.

[8] A I Awad and K Baba, "Evaluation of a fingerprint identification algorithm with SIFT features," in *IIAI International Conference on Advanced Applied Informatics*, Fukuoka, 2012, pp. 129-132. https://doi.org/10.1109/iiai-aai.2012.34

[9] S Egawa, A I Awad, and K Baba, "Evaluation of acceleration algorithm for biometric identification," *Network Digital Technologies NDT 2012. Communications in Computer and Information Science. Springer, Berlin, Heidelberg*, vol. 294, pp. 231-242, 2012. https://doi.org/10.1007/978-3-642-30567-2_19

[10] T. Amornraksa and S. Tachaphetpiboon, "Fingerprint recognition using DCT features," *Electronic Letters*, vol. 42, no. 9, pp. 522–523, 2006. https://doi.org/10.1049/el:20064330

[11] A. K. Jain, S. Prabhakar, L. Hong and S. Pankanti, "Filterbank-based fingerprint matching," *Image Processing, IEEE Transactions*, vol. 9, no. 5, pp. 846-859, 2000. https://doi.org/10.1109/83.841531

[12] S. Lifeng, Z. Feng and T. Xiaoou, "Improved fingercode for filterbank-based fingerprint matching," *In* International Conference on Image *Processing* , vol. 2, no. 2, pp. 895-898, 2003. https://doi.org/10.1109/icip.2003.1246825

[13] R. Kumar, P. Chandra and M. Hanmandlu, "Fingerprint Matching Based on Texture Feature," *In Mobile Communication and Power Engineering*, Springer-Verlag Berlin, vol. 296, pp. 86-91, 2013. https://doi.org/10.1007/978-3-642-35864-7_12

[14] M. Saha, J. Chaki and R. Parekh, "Fingerprint Recognition using Texture Features," *International Journal of Science and Research*, vol. 2, no. 12, pp. 2319-7064, 2013.

[15] K. Tewari and R. L. Kalakoti, "Fingerprint Recognition Using Transform Domain Techniques," in *International Technological Conference*, pp.136-140, 2014.

[16] M. W. Zin and M. M. Sein, "Texture feature based fingerprint recognition for low quality imagesTexture Feature based Fingerprint Recognition for Low Quality Images," in *Micro-NanoMechatronics and Human Science (MHS), International Symposium*, IEEE, Nagoya, Japan, pp. 333–338, 2011. https://doi.org/10.1109/mhs.2011.6102204

[17] C. M. Bishop, Neural Networks for Pattern Recognition, Oxford University Press, 1995.

[18] A. K. Jain and B. Chandrasekaran, "Dimensionality and Sample Size Considerations in Pattern Recognition Practice," in *Handbook of Statistics*, Amsterdam, 1982, pp. 835-855. https://doi.org/10.1016/s0169-7161(82)02042-2

[19] A. K. Jain, R. P. Duin and J. Mao, "Statistical Pattern Recognition: A Review," *IEEE Transaction on Pattern Analysis and Machine Intelligence*, vol. 22, no. 1, pp. 4-37, 2000. https://doi.org/10.1109/34.824819

[20] A. Hacine-Gharbi, M. Deriche, P. Ravier and T. Mohamadi, "A new histogram-based estimation

technique of entropy and mutual information using mean squared error minimization," *Computers & Electrical Engineering*, vol. 39, no. 3, pp. 918-933, 2013.
https://doi.org/10.1016/j.compeleceng.2013.02.010

[21] G. Brown, A. Pocock, M. Lujan and M. J.Zhao, "Conditional Likelihood Maximisation: A Unifying Framework for Information Theoretic Feature Selection," *Journal of Machine Learning Research*, vol. 13, pp. 27-66, 2012.

[22] B. Jun, T. Kim and D. Kim, "A compact local binary pattern using maximization of mutual information for face analysis Pattern Recognition," *Pattern Recognition*, vol. 44, pp. 532–543, 2011.
https://doi.org/10.1016/j.patcog.2010.10.008

[23] A. Adjimi, A. Hacine-Gharbi, P. Ravier and M. Mostefai, "Extraction and selection of binarised statistical image features for fingerprint recognition," *Int. J. Biometrics*, vol. 9, no. 1, p. 67–80., 2017.
https://doi.org/10.1504/ijbm.2017.10005054

[24] D. Maio, D. Maltoni, R. Cappelli, J. L. Wayman and A. K. Jain, "FVC2002: Second Fingerprint Verification Competition," *in 16 th international conference in Pattern Recognition*, 2002.

[25] A. Adjimi, A. Hacine-Gharbi and M. Mostefai, "Application of Binarized Statistical Image Features for Fingerprint Recognition," *in SIVA 2015, 3rd international conference signal image vision and their applications*, Guelma, Algeria, 2015.

[26] T. Ojala, M. Pietikainen and T. Maenpaa, "Multiresolution gray-scale and rotation invariant texture classification with local binary patterns," Pattern Analysis and Machine Intelligence, *IEEE Transactions*, vol. 24, no. 7, pp. 971-987, 2002.
https://doi.org/10.1109/tpami.2002.1017623

[27] T. Ojala, M. Pietikainen and D. Harwood, "A comparative study of texture measures with classification based on feature distributions," *Pattern Recognition*, vol. 29, pp. 51–59, 1996.
https://doi.org/10.1016/0031-3203(95)00067-4

[28] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," *in IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, San Diego, USA, p. 886-893, 2005.
https://doi.org/10.1109/CVPR.2005.177

[29] T. Cover and J. Thomas, Elements of information theory, 2e édition ed., Canada: John Wiley & Sons, 2006.
https://doi.org/10.1002/047174882x

[30] D. François, F. Rossi, V. Wertz and M. Verleysen, "Resampling methods for parameter-free and robust feature selection with mutual information," *Neurocomputing*, vol. 70, pp. 1276–1288, 2007.
https://doi.org/10.1016/j.neucom.2006.11.019

[31] R. Battiti, "Using mutual information for selecting features in supervised neural net learning," *IEEE Trans. Neural Networks*, vol. 5, no. 4, pp. 537–550, 1994.
https://doi.org/10.1109/72.298224

[32] A. Hacine-Gharbi, P. Ravier and T. Mohamadi, "Une nouvelle méthode de sélection des paramètres pertinents : application en reconnaissance de la parole," in *conférence TAIMA*, Hammamet, Tunisie, pp. 399-407, 2009.

[33] G. Brown, "A new perspective for information theoretic feature selection," in *International Conference on Artificial Intelligence and Statistics*, Florida, USA, pp, 49-56, 2009.

[34] N. Kwak and C. H. Choi, "Input feature selection for classification problems," *IEEE Transactions on Neural Networks*, vol. 13, no. 1, pp. 143–159, 2002.
https://doi.org/10.1109/72.977291

[35] H. Peng, F. Long and C. Ding, "Feature selection based on mutual information: Criteria of max dependency, max-relevance, and min-redundancy," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, no. 8, pp. 1226–1238, 2005.
https://doi.org/10.1109/tpami.2005.159

[36] D. Lin and X. Tang, "Conditional infomax learning: An integrated framework for feature extraction and fusion," in *European Conference on Computer Vision*, Springer-Verlag Berlin, Graz, Austria, pp. 68-82 , 2006.
https://doi.org/10.1007/11744023_6

[37] I. Kojadinovic, "Relevance measures for subset variable selection in regression problems based on k-additive mutual information," *Comput. Statist. Data Anal.*, vol. 49, pp. 1205–1227, 2005.
https://doi.org/10.1016/j.csda.2004.07.026

[38] H. Yang and J. Moody, "Data Visualization and Feature Selection: New Algorithms for Non Gaussian Data," *Advances in Neural Information Processing Systems*, MIT Press, pp. 688-695, 1999.

[39] H. Sturges, "The choice of a class-interval," *J. Amer. Statist. Assoc*, vol. 21, pp. 65–66, 1926.
https://doi.org/10.1080/01621459.1926.10502161

[40] Y. Chen, S. C. Dass and A. K. Jain, "Fingerprint Quality Indices for Predicting Authentication Performance," in *Audio- and Video-Based Biometric Person Authentication*, Springer-Verlag Berlin Heidelberg, Hilton Rye Town, USA, pp. 160-170, 2005.
https://doi.org/10.1007/11527923_17

Some Remarks and Tests on the DH1 Cryptosystem Based on Automata Compositions

Pál Dömösi
Institute of Mathematics and Informatics, University of Nyíregyháza
H-4400 Nyíregyháza, Sóstói út 31/B, Hungary

Faculty of Informatics, University of Debrecen
H-4028 Debrecen, Kassai út 26, Hungary
E-mail: domosi@unideb.hu

József Gáll, Géza Horváth
Faculty of Informatics, University of Debrecen
H-4028 Debrecen, Kassai út 26, Hungary
E-mail: gall.jozsef@inf.unideb.hu, horvath.geza@inf.unideb.hu

Norbert Tihanyi
Faculty of Informatics, Eötvös Loránd University
H-1117 Budapest, Pázmány Péter sétány 1/C, Hungary
E-mail: tihanyi.pgp@gmail.com

Keywords: automata network, NIST test, block cipher, statistics

Received: February 17, 2019

In this paper we discuss NIST test results of a previously introduced cryptosystem based on automata compositions. We conclude that the requirements of NIST test are all fulfilled by the cryptosystem.

Povzetek: Analiziran je kriptirni sistem DH1 na osnovi končnih avtomatov s testom NIST.

1 Introduction and problem statement

Dömösi and Horváth in their previous works (see [Dömösi and Horváth, 2015a] and [Dömösi and Horváth, 2015b]) introduced new block ciphers based on Gluškov-type product of automata. In what follows we will refer to the cipher in [Dömösi and Horváth, 2015a] as the first Dömösi-Horváth cryptosystem, or in short, DH1-cipher, whereas to the cipher in [Dömösi and Horváth, 2015b] as the second Dömösi-Horváth cryptosystem, or in short, DH2-cipher. In this paper we investigate some properties of the DH1-cipher. However, we do not discuss all details of definition and motivation regarding DH1-chipers in this paper.

Both systems use the following simple idea: consider a giant-size permutation automaton such that the set of states and the set of inputs consisting of all given length of strings over a non-trivial alphabet as all possible plaintext/ciphertext blocks. Moreover consider a cryptographically secure pseudo random number generator with large periodicity having the property that, getting its really random kernel, it serves a sequence of pseudo random strings as inputs for the automaton. For each plaintext block the system calculates the new state into which the actual pseudorandom string takes the automaton from the state which is identified as the actual plaintext block. The string – identified as the new state– will be the ciphertext block ordered to the considered plaintext block. Of course, the ciphertext will be the concatenation of the generated ciphertext blocks. The giant size of the automaton makes it infeasible to break the system by brute-force method.

For all notions and notations not defined in this paper we refer to the monographs [Dömösi and Nehaniv, 2005, Mezenes and Vanstone, 1996]. The cryptosystem discussed here is a block cipher. Since the key automaton is a permutation automaton, for every ciphertext there exists exactly one plaintext making the encryption and decryption unambiguous. Moreover, there is a huge number of corresponding encoded messages to each plaintext so that several encryptions of the same plaintext yield several distinct ciphertexts.

Given the cryptosystem DH1-cipher described above a natural question is the investigation of the statistical properties of the system from many perspectives. For instance, the avalanche effect of the system –as a natural property required in the profession– may be tested by several classical hypothesis tests. Some early results are given in [Dömösi et al., 2017] where they confirm that the avalanche effect is fulfilled. However, further tests can and should also be used, in particular the ones used for testing whether the output of it can be distinguished from 'true' random sources. That is why we turned to the well known

NIST package of statistical tests in this paper, which can be considered as a 'standard' in the profession for such purposes. Our main aim is to give the results of the NIST test regarding the cryptosystem at issue (Section 5). For this we describe the system (Section 3) together with some theoretical background (Section 2), as well as the necessary details, of course, of our experimental analysis done for the tests (Section 4). We show in this paper that the system we discuss has passed all statistical tests in the NIST package.

2 Theoretical background

The automata are systems that can be used for the transmission of information of certain type. In wider sense, every system that accepts signals from its environment and, as a result, changes its internal state, can be considered as an automaton. By an *automaton* we mean a deterministic finite automaton without outputs. The automaton $\mathcal{A} = (A, \Sigma, \delta)$ consists of the finite set of *states* A, the finite set of *input signals* Σ, and the *transition function* δ, which is often written in a matrix form. The *transition matrix* of the automaton $\mathcal{A} = (A, \Sigma, \delta)$ consists of its states such that it has as many rows as input signals, and there are as many columns as states of the automaton. For the sake of simplicity we assume that A and Σ are ordered sets. The j-th element of the i-th row of the transition matrix will be the state which is assigned by the transition function to the pair consisting of j-th state and i-th input signal. We say about this element a of the i-th row and j-th column of the transition matrix that the i-th input signal takes the automaton from its j-th state to state a. (In fact, in this case it is also usual to say that the automaton goes from its j-th state to state a by the effect of the i-th input signal.) The rows of the transition matrix can be identified with the input signals of the automaton, and its columns with its states, while the transition matrix itself with the transition.

If all the rows of the transition matrix are permutations of the state set then we have a *permutation automaton*.

***Lemma* 1.** *An automaton $\mathcal{A} = (A, \Sigma, \delta)$ is a permutation automaton if and only if for any $a, b \in A, x \in \Sigma$, $\delta(a, x) = \delta(b, x)$ implies $a = b$.*

Proof. Suppose that $\mathcal{A}$ is a permutation automaton. Then all rows in its transition matrix are permutations of the state set. But then none of the rows of the transition matrix has a repetition. Therefore, for any states $a, b \in A$ and input $x \in \Sigma$, $\delta(a, x) = \delta(b, x)$ implies $a = b$. Conversely, assume that for any states $a, b \in A$ and input $x \in \Sigma$, $\delta(a, x) = \delta(b, x)$ implies $a = b$. Then none of the rows of the transition matrix has a repetition. Therefore all of its rows are permutations of the state set. This completes the proof.

The *Gluškov-type product* of the automata $\mathcal{A}_i$ with respect to the feedback functions φ_i $(i \in \{1, \ldots, n\})$ is defined to be the automaton $\mathcal{A} = \mathcal{A}_1 \times \cdots \times \mathcal{A}_n(\Sigma, (\varphi_1, \ldots, \varphi_n))$ with state set

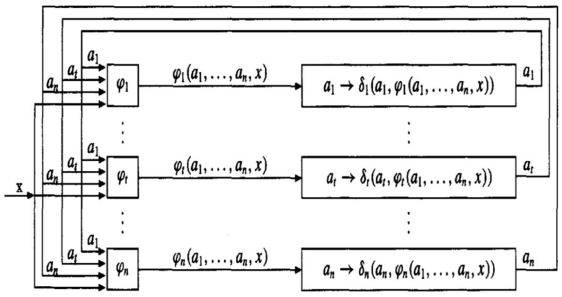


Figure 1: Gluškov-type product.

$A = A_1 \times \cdots \times A_n$, input set Σ, transition function δ given by $\delta((a_1, \ldots, a_n), x) = (\delta_1(a_1, \varphi_1(a_1, \ldots, a_n, x)), \ldots, \delta_n(a_n, \varphi_n(a_1, \ldots, a_n, x)))$ for all $(a_1, \ldots, a_n) \in A$ and $x \in \Sigma$ (see also Figure 1). In particular, if $\mathcal{A}_1 = \ldots = \mathcal{A}_n$ then we say that $\mathcal{A}$ is a *Gluškov-type power*.

We shall use the feedback functions $\varphi_i, i = 1, \ldots, n$ in an extended sense as mappings $\varphi_i^* : A_1 \times \cdots \times A_n \times \Sigma^*$, where $\varphi_i^*(a_1, \ldots, a_n, \lambda) = \lambda$ and $\varphi_i^*(a_1, \ldots, a_n, px) = \varphi_i^*(a_1, \ldots, a_n, p)\varphi_i(\delta_1(a_1, \varphi_1^*(a_1, \ldots, a_n, p)), \ldots, \delta_n(a_n, \varphi_n^*(a_1, \ldots, a_n, p)), x)$, $a_i \in A_i, i = 1, \ldots, n, p \in \Sigma^*$, $x \in \Sigma$. In the sequel, $\varphi_i^*, i \in \{1, \ldots, n\}$ will also be denoted by φ_i.

Next we define the concept of *temporal product* of automata. It is a model for multichannel automata networks where the network may cyclically change its internal structure during its work on each channel.

Let $\mathcal{A}_t = (A, \Sigma_t, \delta_t), t = 1, 2$ be automata having a common state set A. Take a finite nonvoid set Σ and a mapping φ of Σ into $\Sigma_1 \times \Sigma_2$. Then the automaton $\mathcal{A} = (A, \Sigma, \delta)$ is a *temporal product* (t-product) of $\mathcal{A}_1$ by $\mathcal{A}_2$ with respect to Σ and φ if for any $a \in A$ and $x \in \Sigma$, $\delta(a, x) = \delta_2(\delta_1(a, x_1), x_2)$, where $(x_1, x_2) = \varphi(x)$ (see also Figure 2). The concept of temporal product is generalized in the natural way to an arbitrary finite family of $n > 0$ automata $\mathcal{A}_t$ $(t = 1, \ldots, n)$, all with the same state set A, for any mapping $\varphi : \Sigma \to \prod_{t=1}^{n} \Sigma_t$, by defining $\delta(a, x) = \delta_n(\cdots \delta_2(\delta_1(a, x_1), x_2), \cdots, x_n)$ when $\varphi(x) = (x_1, \ldots, x_n)$. In particular, a temporal product of automata with a single factor is just a (one-to-many) relabeling of the input letters of some input-subautomaton of its factor.

***Lemma* 2.** *Every temporal product of permutation automata is a permutation automaton.*

Proof. It is clear from the above mentioned remark that every temporal product of permutation automata with a single factor is a permutation automaton. Now let $\mathcal{A}_t = (A, \Sigma_t, \delta_t), t = 1, 2$ be permutation automata with the same state set A. Consider a temporal product of $\mathcal{A}_1$ and $\mathcal{A}_2$ with respect to an arbitrary input set Σ and mapping $\varphi : \Sigma \to \Sigma_1 \times \Sigma_2$. Prove that for any $a, b \in A, z \in \Sigma$ with $\varphi(z) = (x, y)$, $\delta_2(\delta_1(a, x), y) = \delta_2(\delta_1(b, x), y)$ implies $a = b$.

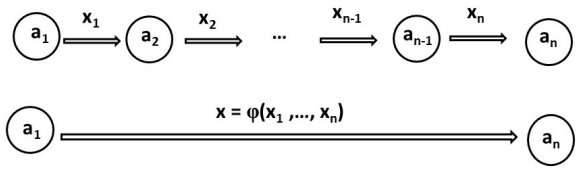


Figure 2: Temporal product.

Indeed, let $\delta_1(a,x) = c$ and $\delta_1(b,x) = d$. Recall that $\mathcal{A}_2$ is a permutation automaton. Therefore, by Lemma 1, $\delta_2(c,y) = \delta_2(d,y)$ implies $c = d$. On the other hand, $\mathcal{A}_1$ is also a permutation automaton. Thus, by Lemma 1, $c = d$ with $\delta_1(a,x) = c$ and $\delta_1(b,x) = d$ imply $a = b$. Applying Lemma 1 again, we receive that the temporal product of $\mathcal{A}_1$ and $\mathcal{A}_2$ with respect to Σ and φ is a permutation automaton. Therefore our statement holds for all temporal products having two factors. Now we consider a temporal product of permutation automata $\mathcal{A}_1, \ldots, \mathcal{A}_n, n > 2$ with respect to a given set Σ and mapping φ.

Define the mappings $\varphi_1 : \Sigma \to \Sigma_1 \times \Sigma_2$, $\varphi_2 : \Sigma \to (\Sigma_1 \times \Sigma_2) \times \Sigma_3, \ldots, \varphi_{n-1} : \Sigma \to (...(\Sigma_1 \times \Sigma_2) \times \ldots \times \Sigma_{n-1}) \times \Sigma_n$ with $\varphi_1(x) = (x_1, x_2)$, $\varphi_2(x) = ((x_1,x_2),x_3), \ldots, \varphi_{n-1}(x) = ((...((x_1,x_2),x_3)...),x_n)$ whenever $\varphi(x) = (x_1, \ldots, x_n)$. Let $\mathcal{B}_1$ denote the temporal product of $\mathcal{A}_1$ and $\mathcal{A}_2$ with respect to Σ and φ_1, $\mathcal{B}_2$ denote the temporal product of $\mathcal{B}_1$ and $\mathcal{A}_3$ with respect to Σ and $\varphi_2, \ldots,$ $\mathcal{B}_{n-1}$ denote the temporal product of $\mathcal{B}_{n-2}$ and $\mathcal{A}_n$ with respect to Σ and φ_n, respectively.

Then using the fact that our statement holds for all temporal products with two factors we obtain that all of $\mathcal{B}_1, \ldots, \mathcal{B}_{n-1}$ are permutation automata. On the other hand, it is clear that $\mathcal{B}_{n-1}$ is equal to the temporal product of permutation automata $\mathcal{A}_1, \ldots, \mathcal{A}_n$ with respect to Σ and φ. Thus the proof is complete.

Given a function $f : X_1 \times \cdots \times X_n \to Y$, we say that f is *really independent of its i-th variable* if for every pair $(x_1, \ldots, x_n), (x_1, \ldots, x_{i-1}, x_i', x_{i+1}, \ldots, x_n) \in X_1 \times \cdots \times X_n$, $f(x_1, \ldots, x_n) = f(x_1, \ldots, x_{i-1}, x_i', x_{i+1}, \ldots, x_n)$. Otherwise we say that f *really depends on its i-th variable*.

A (finite) *directed graph* (or, in short, a *digraph*) $\mathcal{D} = (V, E)$ (of order $n > 0$) is a pair consisting of sets of *vertices* $V = \{v_1, \ldots, v_n\}$ and *edges* $E \subseteq V \times V$. Elements of V are sometimes called *nodes*. An edge $(v, v') \in E$ is said to have a *source* v and a *target* v'. Moreover, we say that $v \in V$ is a *source* if there exists a $v' \in V$ having $(v, v') \in E$, and $v' \in V$ is a *target* if there exists a $v \in V$ with $(v, v') \in E$. The pair $(v, v'), (v'', v''')$ is called a *branch* if $v = v''$ and $v' \neq v'''$. In addition,the pair $(v, v'), (v'', v''')$ is called a *collapse* if $v \neq v''$ and $v' = v'''$. If $|V| = n$ then we also say that $\mathcal{D}$ is a digraph of order n. If V can be decomposed into two disjoint (nonempty) subsets V_1, V_2 such that V_1 is the set of all targets and V_2 is the set of all sources then we say that $\mathcal{D}$ is a *bipartite digraph*. If the bipartite graph $\mathcal{D}$ has neither branches nor collapses then we say that $\mathcal{D}$ is a *simple bipartite digraph*.

Let Σ be the set of all binary strings with a given length $\ell > 0$ and let n be a positive integer power of 2, let $\mathcal{A}_1 = (\Sigma, \Sigma \times \Sigma, \delta_{\mathcal{A}_1})$ be a permutation automaton such that for every $a, x, x', y, y' \in \Sigma, \delta_{\mathcal{A}_1}(a,(x,y)) \neq \delta_{\mathcal{A}_1}(a(x',y)), \delta_{\mathcal{A}_1}(a,(x,y)) \neq \delta_{\mathcal{A}_1}(a(x,y'))$, and let $\mathcal{A}_i = (\Sigma, \Sigma \times \Sigma, \delta_{\mathcal{A}_i}), i = 2, \ldots, n$ be state-isomorphic copies of $\mathcal{A}_1$ such that $\mathcal{A}_1, \ldots, \mathcal{A}_n$ are not necessarily pairwise distinct, and let n be a power of 2. Consider the following simple bipartite digraphs:

$\mathcal{D}_1 = (\{1, \ldots, n\}, \{(n/2 + 1, 1), (n/2 + 2, 2), \ldots, (n, n/2)\})$,

$\mathcal{D}_2 = (\{1, \ldots, n\}, \{(n/4 + 1, 1), (n/4 + 2, 2), \ldots, (n/2, n/4)$,
$(3n/4 + 1, n/2 + 1), (3n/4 + 2, n/2 + 2), \ldots, (n, 3n/4)\})$,

...,

$\mathcal{D}_{log_2 n-1} = (\{1, \ldots, n\}, \{(3,1), (4,2), (7,5), \ldots,$
$(8,6), (n-1, n-3), (n, n-2)\})$,

$\mathcal{D}_{log_2 n} = (\{1, \ldots, n\}, \{(2,1), (4,3), \ldots, (n, n-1)\})$,

$\mathcal{D}_{log_2 n+1} = \mathcal{D}_1$,

...,

$\mathcal{D}_{2log_2 n} = \mathcal{D}_{log_2 n}$.

For every digraph $\mathcal{D} = (V, E)$ with $\mathcal{D} \in \{\mathcal{D}_1, \ldots, \mathcal{D}_{2log_2 n}\}$ let us define the Gluškov-type product, called *two-phase $\mathcal{D}$-product*, $\mathcal{A}_\mathcal{D} = \mathcal{A}_1 \times \cdots \times \mathcal{A}_n(\Sigma^n, (\varphi_1, \ldots, \varphi_n))$ of $\mathcal{A}_1, \ldots, \mathcal{A}_n$ so that for every $(a_1, \ldots, a_n), (x_1, \ldots, x_n) \in \Sigma^n, i \in \{1, \ldots, n\}$, $\varphi_i(a_1, \ldots, a_n, (x_1, \ldots, x_n)) = (a_j \oplus x_j, x_i)$, if $(j,i) \in E$, and $a_j \oplus x_j$ is the bitwise addition modulo 2 of a_j and x_j, $\varphi_j(a_1, \ldots, a_n, (x_1, \ldots, x_n)) = (a_i' \oplus x_i, x_j)$, if $(j,i) \in E$, a_i' denotes the state into which $\varphi_i(a_1, \ldots, a_n, (x_1, \ldots, x_n))$ takes the automaton $\mathcal{A}_i$ from its state a_j, and $a_i' \oplus x_i$ is the bitwise addition modulo 2 of a_i' and x_i. [1]

Let $\mathcal{B} = (\Sigma^n, (\Sigma^n)^{2log_2 n}, \delta_\mathcal{B})$ be the temporal product of $\mathcal{A}_{\mathcal{D}_1}, \ldots, \mathcal{A}_{\mathcal{D}_{2log_2 n}}$ with respect to $(\Sigma^n)^{2log_2 n}$ and the identity map $\varphi : (\Sigma^n)^{2log_2 n} \to (\Sigma^n)^{2log_2 n}$. We say that $\mathcal{B}$ is a *key-automaton* with respect to $\mathcal{A}_1, \ldots, \mathcal{A}_n$.[2] Obviously, $\mathcal{B}$ is unambigously defined by the transition matrix of $\mathcal{A}_1$ and the bijective mappings $\tau_1 : \Sigma \to \Sigma, \ldots, \tau_n : \Sigma \to \Sigma$ which represent the state isomorphisms of $\mathcal{A}_1, \ldots, \mathcal{A}_n$ to $\mathcal{A}$.

An important property of key-automata is explained in the following result.

***Theorem* 1.** *Every key-automaton is a permutation automaton.*

Proof. Let $\mathcal{B} = (\Sigma^n, (\Sigma^n)^{2log_2 n}, \delta_\mathcal{B})$ be a key-automaton. By definition, it is a temporal product of automata $\mathcal{A}_{\mathcal{D}_1}, \ldots, \mathcal{A}_{\mathcal{D}_{2log_2 n}}$ with respect to $(\Sigma^n)^{2log_2 n}$ and

[1] We remark, that for every $j \in V_2$ there exists exactly one $i \in V_1$ with $(j,i) \in E$, and conversely, for every $i \in V_1$ there exists exactly one $j \in V_2$ with $(j,i) \in E$. Therefore, all of $\varphi_1, \ldots, \varphi_n$ are well-defined.

[2] Recall that n should be a positive integer power of 2.

the identity map $\varphi : (\Sigma^n)^{2log_2 n} \to (\Sigma^n)^{2log_2 n}$ as defined above. By Lemma 2, it is enough to prove that each of $\mathcal{A}_{\mathcal{D}_1}, \ldots, \mathcal{A}_{\mathcal{D}_{2log_2 n}}$ is a permutation automaton.

Consider an automaton $\mathcal{A}_{\mathcal{D}} = (\Sigma^n, \Sigma^n, \delta_{\mathcal{D}})$ with $\mathcal{A}_{\mathcal{D}} \in \{\mathcal{A}_{\mathcal{D}_1}, \ldots, \mathcal{A}_{\mathcal{D}_{2log_2 n}}\}$ and the simple bipartite digraph $\mathcal{D} = (V, E)$ assigned to $\mathcal{A}_{\mathcal{D}}$. Let V_1 denote the set of targets and V_2 denote the set of sources of $\mathcal{D}$ as before.

By Lemma 1 it is enough to prove that for any states $(a_1, \ldots, a_n), (a'_1, \ldots, a'_n) \in \Sigma^n$ and input $(x_1, \ldots, x_n) \in \Sigma^n$, $\delta_{\mathcal{D}}((a_1, \ldots, a_n), (x_1, \ldots, x_n)) = \delta_{\mathcal{D}}((a'_1, \ldots, a'_n), (x_1, \ldots, x_n))$ implies $(a_1, \ldots, a_n) = (a'_1, \ldots, a'_n)$.

Suppose $\delta_{\mathcal{D}}((a_1, \ldots, a_n), (x_1, \ldots, x_n)) = \delta_{\mathcal{D}}((a'_1, \ldots, a'_n), (x_1, \ldots, x_n)) = (b_1, \ldots, b_n)$ for some state $(b_1, \ldots, b_n)$ of $\mathcal{A}_{\mathcal{D}}$ and let $(i, j) \in E$. Observe that for every $i \in V_1$ there exists exactly one $j \in V_2$ with $(j, i) \in E$, and vice versa, for every $j \in V_2$ there exists exactly one $i \in V_1$ with $(j, i) \in E$. This means that the transitions in the i-th and j-th component automata depend only on the i-th and j-th state and input components.

Then, by the effect of its input $(a_j \oplus x_j, x_i)$ the i-th component of $\mathcal{A}_{\mathcal{D}}$ goes from its state a_i into state b_i, and similarly, by the effect of its input $(b_i \oplus x_i, x_j)$ the j-th component of $\mathcal{A}_{\mathcal{D}}$ goes from its state a_j into state b_j.

But then by the effect of its input $(a'_j \oplus x_j, x_i)$, the i-th component of $\mathcal{A}_{\mathcal{D}}$ goes from its state a'_i into state b_i, and similarly, by the effect of its input $(b_i \oplus x_i, x_j)$, the j-th component of $\mathcal{A}_{\mathcal{D}}$ goes from its state a'_j into state b_j.

Recall that $\mathcal{A}_j$ is a permutation automaton. Therefore, applying Lemma 1, $a_j = a'_j$. Therefore, using our previous assumptions we can derive that by the effect of its input $(a_j \oplus x_j, x_i)$ the i-th component of $\mathcal{A}_{\mathcal{D}}$ goes from its state a'_i into state b_i. On the other hand, we assumed that by the effect of its input $(a_j \oplus x_j, x_i)$, the i-th component of $\mathcal{A}_{\mathcal{D}}$ goes from its state a_i into state b_i. Applying Lemma 1 again we obtain that $a_i = a'_i$.

Applying the above treatment to every $(i, j) \in E$, we receive $(a_1, \ldots, a_n) = (a'_1, \ldots, a'_n)$. This completes the proof.

The basic idea of DH1 cryptosystem is to use a finite automaton and a pseudo random generator. The set of states of the automaton consists of all possible plaintext/cyphertext blocks and the input set of the automaton contains all possible pseudo random blocks. The size of the pseudo random blocks are the same as the size of the plaintext/cyphertext blocks. For each plaintext block the pseudo random generator generates the next pseudo random block and the automaton transforms the plaintext block into a cyphertext block by the effect of the pseudo random block. The key is the transformation matrix of the automaton.

It is easy to see that the key must be a permutation automaton, since this property grants an unambiguous decryption. This condition is satisfied by Theorem 1.

On the other hand we can have more than one corresponding ciphertext for each plaintext even if we use the same key-automaton. The reason for this is that we can change the pseudo random numbers generated by the pseudo random generator. We can save a secret number n –as a part of the key– and before encryption we can choose a (public) random number m. This number m will be the first block of the ciphertext, and before encryption and decryption, the seed of the pseudo random number generator can be calculated with an XOR operation from n and m $(n \oplus m)$. This way each encryption process uses different pseudo random numbers and results different ciphertext for the same plaintext.

The problem with this idea is the following. Modern block ciphers operate on fixed-length groups of bits called blocks. The size of the blocks is at least 128 bits (16 bytes), so the size of the transition matrix of the automaton is huge, namely $2^{128} \times 2^{128} \times 16$ bytes, which is impossible to be stored in the memory or on a hard disk. The solution is to use an automata network. Gluškov-type product of automata consists of smaller component automata and it is able to simulate the operation of a huge automaton. In this case we should store only the transition matrix of the isomorphic component-automata, the structure of the composition and the secret number n to calculate the seed of the pseudo random number generator.

3 Encryption and decryption

A symmetric cryptosystem consists of the following:

- a set of plaintexts $\mathcal{P}$,
- a set of ciphertexts $\mathcal{C}$,
- a key space $\mathcal{K}$,
- an encryption function $e : \mathcal{P} \times \mathcal{K} \to \mathcal{C}$, and
- a decryption function $d : \mathcal{C} \times \mathcal{K} \to \mathcal{P}$.

Furthermore, the following property must hold for each $x \in \mathcal{P}$ and $k \in K$: $d((e(x, k), k) = x$. Moreover, the cryptosystem is called approved block cipher if and only if the elements of the set of plaintexts and the set of ciphertexts are at least 128 bit long ($|\mathcal{P}| \geq 2^{128}$ and $|\mathcal{C}| \geq 2^{128}$).

Our cryptosystem is a block cipher one. Both of the encryption and decryption apparatus have a pseudo random generator and a key-automaton.

The encryption procedure is the following. Before the encryption procedure, the pseudo random generator gets its initialization vector as a true random string $r_1 \ldots r_n \in \Sigma^n$, where the pseudo random alphabet Σ is also the plaintext and the ciphertext alphabet simultaneously. This initialization vector will also be the first block of the ciphertext.

Then the apparatus reads the plaintext block-by-block and, after reading the next plaintext block $a_1 \cdots a_n \in \Sigma^n$ (the first block first), it generates the second, third, and the further blocks of the ciphertext in the following way.

The apparatus takes the key-automaton $\mathcal{B} = (\Sigma^n, (\Sigma^n)^{2log_2 n}, \delta_{\mathcal{B}})$ into the state $a_1 \cdots a_n \in \Sigma^n$

which coincides with the actual one, i.e. the last received plaintext block.

Next the pseudo random number generator generates a $2log_2n$ long number of pseudo random sequences $w_1, \ldots, w_{2log_2n} \in \Sigma^n$ such that each of them takes the next temporal component (the first one first) $\mathcal{A}_\mathcal{D} = (\Sigma^n, \Sigma^n, \delta_\mathcal{D})$ ($\mathcal{A}_\mathcal{D} \in \{\mathcal{A}_{\mathcal{D}_1}, \ldots, \mathcal{A}_{\mathcal{D}_{2log_2^n}}\}$) of the key automaton into the state $a_{k,1} \cdots a_{k,n} = \delta_\mathcal{D}(a_{k-1,1} \cdots a_{k-1,n}, w_k), k = 1, \ldots, 2log_2n$, where $a_{0,1} \cdots a_{0,n}$ denotes the actual plaintext block.

The last state $a_{2log_2n,1} \cdots a_{2log_2n,n}$ will be the generated ciphertext block of the plaintext block $a_1 \cdots a_n$.

The i-th transition $a_{i,1} \cdots a_{i,n} = \delta_\mathcal{D}(a_{i-1,1} \cdots a_{i-1,n}, w_i)$ will be performed in the following way.

Recall that $\mathcal{D}$ is a Gluškov product $\mathcal{A}_\mathcal{D} = \mathcal{A}_1 \times \cdots \times \mathcal{A}_n(\Sigma^n, (\varphi_1, \ldots, \varphi_n))$ of appropriate permutation automata $\mathcal{A}_m = (\Sigma, \Sigma^2, \delta_m), m = 1, \ldots, n$ that are state isomorphic to each other so that for an appropriate bipartite digraph $\mathcal{D} = (V, E)$ with the set V_1 of targets and V_2 of sources we have as follows:

$\delta_i(a_{k-1,i}, \varphi_i(a_{k-1,1}, \cdots, a_{k-1,n}, (x_1, \ldots, x_n))) = a_{k,i}$, where $a_{k,i} = \delta_i(a_{k-1,i}, (a_{k-1,j} \oplus x_j, x_i))$, if $(j, i) \in E$, and $a_{k,i} = \delta_i(a_{k-1,i}, (a_{k,j} \oplus x_j, x_i))$, if $(i, j) \in E$, and $a_{k,j} = \delta_i(a_{k-1,i}, (a_{k-1,j} \oplus x_j, x_i))$,

(1)

where $w_m = x_1 \cdots x_n \in \Sigma^n$ is the actual pseudo random string. Obviously, using the transition matrix of $\mathcal{A}_i$, from $a_{k-1,i}, a_{k-1,j}, x_i, x_j$ we can determine $a_{k,i}$ for every $i \in V_1, (j, i) \in E$. Moreover, after calculating the values $a_i (i \in V_1)$, using the transition table of $\mathcal{A}_i$, from $a_{k-1,j}, a_{k,i}, x_i, x_j$ we can determine $a_{k,j}$ for every $i \in V_2, (i, j) \in E$.

Then, concatenating the calculated blocks, we will get the ciphertext.

The decryption procedure is the following. Similarly as before, before the decryption procedure the pseudo random generator gets the first ciphertext block as its initialization vector $r_1 \ldots r_n \in \Sigma^n$.

Then the apparatus reads the ciphertext block-by-block and, after reading the next ciphertext block $c_1 \cdots c_n \in \Sigma^n$ (the first block first), it generates the second, third and the further blocks of the plaintext in the following way.

The apparatus determines the state $a_1 \cdots a_n \in \Sigma^n$ of key-automaton $\mathcal{B} = (\Sigma^n, (\Sigma^n)^{2log_2n}, \delta_\mathcal{B})$ into which the automaton $\mathcal{B}$ is taken from the state $a_1 \cdots a_n \in \Sigma^n$ by the effect of $2log_2n$ consecutive strings in Σ^n generated by the pseudo random generator.

Thus the pseudo random generator should generate a $2log_2n$ -long number of pseudo random sequences $w_1, \ldots, w_{2log_2n} \in \Sigma^n$ and going back from the last member w_{2log_2n} to the first one w_1 the following procedure is performed.

Each of them takes the next temporal component (in opposite direction, i.e., the last one first and the first one last) $\mathcal{A}_\mathcal{D} = (\Sigma^n, \Sigma^n, \delta_\mathcal{D})$ ($\mathcal{A}_\mathcal{D} \in \{\mathcal{A}_{\mathcal{D}_1}, \ldots, \mathcal{A}_{\mathcal{D}_{2log_2^n}}\}$) of the key automaton into the state $a_{k-1,1} \cdots a_{k-1,n}$ back from the state $a_{k,1} \cdots a_{k,n} = \delta_\mathcal{D}(a_{k-1,1} \cdots a_{k-1,n}, w_k), k = 1, \ldots, 2log_2n$, where $a_{2log_2n,1} \cdots a_{2log_2n,n}$ denotes the actual ciphertext block $c_1 \cdots c_n$.

The last state $a_{0,1} \cdots a_{0,n}$ will be the generated plaintext block of the ciphertext block $c_1 \cdots c_n$.

The state $a_{i-1,1} \cdots a_{i-1,n}$ obtained from the i-th state transition $a_{i,1} \cdots a_{i,n} = \delta_\mathcal{D}(a_{i-1,1} \cdots a_{i-1,n}, w_i)$ will be performed in the following way.

Recall again that $\mathcal{D}$ is a Gluškov product $\mathcal{A}_\mathcal{D} = \mathcal{A}_1 \times \cdots \times \mathcal{A}_n(\Sigma^n, (\varphi_1, \ldots, \varphi_n))$ of appropriate permutation automata $\mathcal{A}_m = (\Sigma^2, \Sigma, \delta_m), m = 1, \ldots, n$ that are state isomorphic to each other so that for an appropriate bipartite digraph $\mathcal{D} = (V, E)$ with the set V_1 of targets and V_2 of sources, we conclude as in (1).

Recall also that all of $\mathcal{A}_1, \ldots, \mathcal{A}_n$ are permutation automata. Therefore, for every $a_{k,i}, a_{k,j}, x_i, x_j, j \in V_2, (j, i) \in E$, there exists only one $a_{k-1,j}$ with $a_{k,j} = \delta_i(a_{k-1,j}, (a_{k,i} \oplus x_i, x_j))$. Thus, using the transition table we can unambiguously determine $a_{k-1,j}$ for every $j \in V_2$. Moreover, for every $a_{k,i}, a_{k-1,j}, x_i, x_j$, $i \in V_1, (j, i) \in E$, there exists exactly one $a_{k-1,i}$ with $a_{k,i} = \delta_i(a_{k-1,i}, (a_{k-1,j} \oplus x_j, x_i))$. Therefore, using the transition table again we can unambiguously determine $a_{k-1,i}$ as well for every $i \in V_1$.

Then by concatenating the determined plaintext blocks we will get the plaintext back.

To sum up, the discussed cryptosystem is a block cipher. Because of Theorem 1, for every ciphertext there exists exactly one plaintext making the encryption and decryption unambiguous. Moreover, there is a huge number of corresponding encoded messages to each plaintext so that several encryptions of the same plaintext yield several distinct ciphertexts.

4 Experimental results

The practical test was done using 16 byte (128 bit) long input blocks, output blocks and pseudo random blocks. First we present the size of the keyspace, then we continue our investigation with the test results of the the speed of the algorithm, and finally the effectiveness of the avalanche effect.

Using the above mentioned parameters with 256 possible states (1 byte long states) we need 16 automata having a transition matrix of $2^{16} = 65536$ lines and $2^8 = 256$ columns. Each cell of the automaton contains 1 byte long data (One state). The size of the matrix is 16 megabytes and the number of possible matrices is $256!^{65536}$, where the exclamation mark means the factorial operation. This protection is much more than good enough against brute-force

attacks. When we use isomorphic automata this huge number should be further increased to have $256!^{65536} * 256!^{15} = 256!^{65551}$ possible keys. Using the above mentioned parameters with half byte (4bits) long states, we need 32 automata having a transition matrix of $2^8 = 256$ lines and $2^4 = 16$ columns and each cell of the automaton contains half byte long data. In this case the size of the matrix is only 2 kilobytes and the number of possible matrices is $16!^{256}$. Using permutation automata this can be increased to $16!^{287}$ possible keys, which is still more than enough against brute-force attacks. However, we recommend the 8 bit version, because the number of calculations during the encoding and decoding process is less and the effectiveness of the avalanche effect is better.

The practical test of the encoding and decoding algorithm was done on an average desktop PC, (3,1 GHz Intel Core I3-2100 processor, 4 Gigabyte RAM). The program we used was a well written C# implementation. The results of the speed tests of the 8 bit version can be seen in Table 1.

The results of the speed tests show that using an average PC the encoding time is more than 4 megabytes per second, and decoding time is about the same.

The avalanche effect is a very important property of block ciphers. The block cipher is said to have avalanche effect when a small change in the plaintext block results in a significant change in the corresponding ciphertext block, further, a small change in the ciphertext block results in a significant change in the corresponding plaintext block. We tested the avalanche effect in the following way. We chose 1000000 random plaintext blocks, encoded them and then we changed 1 bit in each plaintext block, encoded again, then we calculated the number of different bytes in the ciphertext blocks pair-wise. We also tested the opposite case, namely, we chose 1000000 random ciphertext blocks, decoded them and then we changed 1 bit in each ciphertext block, decoded again and calculated the number of different bytes in each plaintext block pair-wise. During the first test we used just the first two rounds of encoding and decoding. The results can be seen in Table 2. When we change only one bit in the plaintext block the difference between the corresponding ciphertext blocks will be really huge in the majority of cases. The same effect can be seen in the opposite case: changing one bit in the ciphertext block results in a huge difference in the plaintext block as well. Although it was a good result, we also made a further test with the full 4-round algorithm. The results can be seen in Table 3.

Furthermore, we calculated the optimal avalanche effect. For this, we chose 2×1000000 completely random blocks and then calculated the difference between them pair-wise. The results are in Table 4

We can assume that using the 8-bit version of the algorithm with 128 bit long blocks and 4 rounds the algorithm has the maximal avalanche effect and an appropriate speed (4 megabyte/s). Of course the speed of the algorithm depends on the hardware, the programming language and the actual program code as well.

5 The NIST test

The National Institute of Standards and Technology (NIST) published a statistical package consisting of 15 statistical tests that were developed to test the randomness of arbitrarily long binary sequences produced by either hardware or software based cryptographic random or pseudo random number generators. In case of each statistical test a set of P-values was produced. Given a significance level α, if the P-value is less than or equal to α then test suggests that the observed data is inconsistent with our null hypothesis, i.e. the 'hypothesis of randomness', so we reject it. We used $\alpha = 0.01$ as it is common in such problems in cryptography. An α of 0.01 indicates that one would expect 1 sequence in 100 sequences to be rejected under the null hypothesis. Hence a P-value exceeding 0.01 would mean that the sequence would be considered to be random, and P-value less than or equal to 0.01 would lead to the conclusion that the sequence is non-random.

One of the criteria used to evaluate the AES candidate algorithms was their demonstrated suitability as random number generators. That is, the evaluation of their output utilizing statistical tests should not provide any means by which to distinguish them computationally from a truly random source. Randomness testing was performed using the same parameters as for the AES candidates in order to achieve the most reliable and comparable results. First the input parameters –such as the sequence length, sample size, and significance level– were fixed. Namely, these parameters were set at 2^{20} bits, 300 binary sequences, and $\alpha = 0.01$, respectively. Furthermore, Table 5 shows the length parameters we used.

In order to analyze the output of the algorithm we encrypted the Rockyou database, which contains more than 32 millions of cleartext passwords (see e.g: [Tihanyi et al., 2015]). Applying the NIST test for the encrypted file it has turned out that the output of the algorithm can not be distinguished in polynomial time from true random sources by statistical tests. The exact p-values of the evaluation of the ciphertext are shown in Table (6). We also tested the uniformity of the distribution of the p-values obtained by the statistical tests included in NIST, which is a usual requirement in the literature (see e.g. [Rukhin et al., 2010]). The uniformity of p-values provide no additional information about the type of the cryptosystem. We have also shown that the proportions of binary sequences which passed the 0.01 level lie in the required confidence interval (see e.g. [Rukhin et al., 2010]).

6 Conclusions

The output of our crypto algorithm has passed all statistical tests of the NIST suite we performed and **we were not**

Table 1: Encoding and decoding spped test.

Type of the plaintext	Size	Encoding time	Decoding time	Encoded bytes per second
Image:JPG	205216	00:00.0470960	00:00.0456500	4357397.6558519
Document:PDF	204768	00:00.0459240	00:00.0454752	4458845.0483407
Text:TXT	204848	00:00.0467470	00:00.0461294	4382056.6025627
Compressed:RAR	204848	00:00.0471470	00:00.0454830	4344878.7833796
Compressed:RAR	104883392	00:25.9539778	00:27.2784568	4041129.7569962
Compressed:RAR	524613552	02:10.6843636	02:08.6140492	4014355.9454882
Compressed:RAR	1102971104	04:28.121944	04:08.2624464	4442762.5683785

Table 2: Character differences after 2 rounds of encoding.

different characters in one block	encoding	decoding
0	0	0
1	0	0
2	1	1
3	0	0
4	36	40
5	3	1
6	72	89
7	125	136
8	5574	5594
9	11	4
10	179	225
11	410	396
12	11050	11064
13	880	921
14	22670	22397
15	43064	42710
16	915924	916422

Table 3: Character differences after 4 rounds of encoding.

different characters in one block	encoding	decoding
0-12	0	0
13	37	28
14	1717	1746
15	59403	59145
16	938842	939081

Table 4: Optimal avalanche effect.

different characters in one block	
0-12	0
13	32
14	1693
15	58681
16	939594

Table 5: Parameters used for NIST Test Suite.

Test Name	Block length
Block Frequency	128
Non-overlapping Template	9
Overlapping Template	9
Approximate Entropy	10
Serial	16
Linear Complexity	500

able to distinguish it from true random sources by statistical methods. Statistical analyses of a cryptosystem is a must-have requirement, and these test results are good indicators that further analyses can and should be done in order to check further properties. Cryptanalysis methods like chosen-plaintext, known-plaintext and related-key attack techniques will be used to prove or disprove the strength of the cryptosystem. These problems are the subject of our future research.

Many information systems such as computers and computer networks may be simulated by means of a queueing system. In general, queueing systems model is developed assuming the arrival rate and service intensity to be in the equilibrium state. The well-known methods of the queueing system investigation are based on the stationary behaviour of the input flow and service duration. Taking into account these characteristics as well as technical-economical criteria, the optimal system performance parameters are determined.

In real conditions the input flow arrival rate is affected by the step-by-step influence and the system state can essentially differ from the desired one. Here we come across the problem of compensating these differences with the purpose of equalizing the real value of output of customers' flow to the desirable one.

The main idea of this work lies in the identification of the queueing system as the control object with further construction of discrete control closed scheme.

7 Acknowledgments

This work was supported by the National Research, Development and Innovation Office of Hungary under Grant No. TÉT 16-1-2016-0193.

References

[Dömösi and Horváth, 2015a] Dömösi, P. and Horváth, G. (2015). A novel cryptosystem based on abstract automata and Latin cubes. *Studia Scientiarum Mathematicarum Hungarica*, 52(2):221–232. https://doi.org/10.1556/012.2015.52.2.1309

[Dömösi and Horváth, 2015b] Dömösi, P. and Horváth, G. (2015). A novel cryptosystem based on Gluškov product of automata. *Acta Cybernetica*, 22:359–371. https://doi.org/10.14232/actacyb.22.2.2015.8

[Dömösi et al., 2017] Dömösi, P., Gáll, J., Horváth, G and Tihanyi, N. (2017). Statistical Analysis of DH1 Cryptosystem. *Acta Cybrnetica*, 23:371–378. https://doi.org/10.14232/actacyb.23.1.2017.20

[Dömösi and Nehaniv, 2005] Dömösi, P. and Nehaniv,C.L. (2005). *Algebraic theory of automata networks: An introduction. ser. SIAM monographs on Discrete Mathematics and Applications*, vol. **11**, Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA. https://doi.org/10.1137/1.9780898718492

[Mezenes and Vanstone, 1996] Menezes, P. C. O. A. J. and Vanstone, S. A. (1996). *Handbook of Applied Cryptography ser. Discrete Mathematics and Its Applications. CRC Press*. https://doi.org/10.1201/9781439821916

[Rukhin et al., 2010] Rukhin, A., Soto, J., Nechvatal, J., Smid, M., Barker, E., Leigh, S., Levenson, M., Vangel, M., Banks, D., Heckert, A., Dray, J., Vo, S. (2010). NIST Special Publication 800-22: A Statistical Test Suite for Random and Pseudo Random Number Generators for Cryptographic Applications. *National Institute of Standards and Technology*, https://nvlpubs.nist.gov/nistpubs/legacy/sp/nistspecialpublication800-22r1a.pdf, downloaded in August 2016. https://doi.org/10.6028/nist.sp.800-22

[Tihanyi et al., 2015] Tihanyi, N., Kovács, A., Vargha, G., Lénárt, Á. *Unrevealed Patterns in Password Databases Part One: Analyses of Cleartext Passwords*. Technology and Practice of Passwords. PASSWORDS 2014. Lecture Notes in Computer Science, vol 9393. https://doi.org/10.1007/978-3-319-24192-0_6

Table 6: Results for the uniformity of p-values and the proportion of passing sequences.

C1	*C2*	*C3*	*C4*	*C5*	*C6*	*C7*	*C8*	*C9*	*C10*	*P-value*	PROPORTION	STATISTICAL TEST
28	35	23	33	43	34	32	23	26	23	0.162606	296/300	$Frequency$
25	29	35	38	27	23	26	27	31	39	0.407091	298/300	$BlockFrequency$
28	37	26	37	32	28	25	36	25	26	0.574903	297/300	$CumulativeSums$
26	30	31	30	33	27	24	38	28	33	0.840081	295/300	$CumulativeSums$
33	20	33	26	32	28	44	25	30	29	0.205897	297/300	$Runs$
23	33	40	24	31	22	31	29	38	29	0.284959	297/300	$LongestRun$
24	28	40	32	24	30	30	27	37	28	0.527442	297/300	$Rank$
34	35	23	33	30	35	27	34	23	26	0.623240	298/300	FFT
35	31	30	29	30	29	32	28	23	33	0.958773	295/300	$NonOverlapping-Template$
.	.	.	.	.	.	.	.	.	.	.	.	.
...	...	...	...	...	...	...	...	...	...	...	...	...
25	27	25	29	40	39	29	33	26	27	0.419021	299/300	$OverlappingTemplate$
32	29	21	20	29	37	34	28	30	40	0.220931	298/300	$Universal$
35	33	28	34	26	26	27	30	33	28	0.935716	299/300	$ApproximateEntropy$
21	17	24	23	15	15	18	12	15	17	0.516465	171/177	$RandomExcursions$
.	.	.	.	.	.	.	.	.	.	.	.	.
...	...	...	...	...	...	...	...	...	...	...	...	...
23	16	15	16	14	26	12	18	18	19	0.384836	172/177	$RandomExcursions-Variant$
.	.	.	.	.	.	.	.	.	.	.	.	.
...	...	...	...	...	...	...	...	...	...	...	...	...
23	27	38	25	27	43	41	24	24	28	0.042808	298/300	$Serial$
28	28	25	24	45	32	32	33	28	25	0.253551	296/300	$Serial$
32	25	33	34	40	20	31	35	15	35	0.039244	295/300	$LinearComplexity$

Agent-Based Modeling of Society Resistance against Unpopular Norms

Kashif Zia, Dinesh Kumar Saini and Arshad Muhammad
Faculty of Computing and Information Technology, Sohar University, Sohar, Oman
E-mail: kzia@su.edu.om, dinesh@su.edu.om, amuhammad@su.edu.om

Umar Farooq
University of Science and Technology Bannu, Khyber Pakhtunkhwa, Pakistan
E-mail: umar@ustb.edu.pk

Keywords: agent-based modeling and simulation, unpopular norms, emperor's dilemma, norm aversion

Received: June 24, 2017

People lives in a society adhering to different types of norms, and some of these norms are unpopular. This paper proposes an agent-based model for unpopular norm aversion. The proposed model is simulated asking important "what-if" questions to elaborate on the conditions and reasons behind the emergence, spreading and aversion of unpopular norms. Such conditions can thus be analyzed and mapped onto the behavioral progression of real people and patterns of their interactions to achieve improved societal traits particularly using the new social landscape dominated by digital content and social networking. Hence, it can be argued that careful amalgamation of social media content can not only educate the people but also help them in an aversion of undesirable behaviors such as retention and spreading of unpopular norms. Simulation results revealed that to achieve a dominant norm aversion, an agent population must incorporate a rational model, besides, active participation of agents in averting unpopular norms.

Povzetek: Razvit je agentni sistem obnašanja množic, ki omogoča analizo odpora proti nezaželenim normam.

1 Introduction

Social norms are concepts and practices prevalent in a society [7]. Formally, "Norms are practical prescriptions, permissions, or prohibitions, accepted by members of particular groups, organizations, or societies, and capable of guiding the actions of those individuals" [21].

Norms are accepted which means that their existence is evident from empirical inquiry. However, there is a contradiction in the viewpoint of the notion of existence. One view of norms existence (acceptance) is internalized that incorporates it into individuals' identity [21, 1]. The other view uses the intentions of individuals' as the criterion for norm acceptance instead of the identity of individuals [21]. Therefore, conforming/accepting a norm corresponds to the first view while following a norm relates to the second view [3]. This distinction allows thinking of a norm even without accepting it [21]. Norms describe a collective behavior of groups, organizations, or societies but they are the collective outcome of individuals' cognition.

Norms have the power to transform into actions. This can lead to norm transformation as well. Brennan et al [3] have distinguished between conforming and complying (following) with norms. Similarly, they differentiated between avoiding and acting opposite to norms. These actions are norm guided and in the absence of a norm, an action would not be performed or it would be performed but not in a similar fashion [21].

Norms play an important role in the development of social order [30]. They can change, create and affect behaviors. On the other hand, behaviors are capable of changing, creating and affecting norms [15]. Individual behavior affects the behavior of other individuals in its range of influence [8]. The process is often defined as norm being "externalized". These externalities are able to reproduce a regulatory impact on individuals' behavior [12]. According to Christine Horne, a higher degree of norm enforcement have large sanctioning benefits [9]. She designed a norm enforcement theory with the following features. In the case of group welfare, sanctioning benefits have a positive effect on norm and metanorm enforcement. However, the sanctioning cost has a negative effect on norm enforcement. In the case of social relations, interdependence has a positive effect on norm enforcement. Similarly, sanctioning cost and interdependence have a positive effect on metanorm enforcement. Meta-norms are a particular type of norms that regulate enforcement. Interdependence means the extent to which individuals value their relations. The experimental analysis of the theory of enforcement has revealed that theories that do not consider social relational contact may produce faulty predictions.

Generally, an individual in a society is expected to behave according to societal norms. However, the equation is not that simple. Following a societal norm does not mean that an individual is accepting it. There may be a number of conditions and incentives that force an individual to follow a social norm [21]. This clearly distinguishes

the distinction between following and conforming to the norm. If a norm is not confirmed or accepted from the inside of an individual, just following it as a visible trait is of weaker intensity. Hence, in the case of an individual, a norm can be followed as a result of social pressure, but not accepted, if the individual's personal belief does not correspond to it. Contrarily, an individual may accept/conform to a norm, if personal belief corresponds to following it. Christine Horne [10] has emphasized on relationships, which are more important than individual perception about norms. She argues that these relationships can even persuade an individual to enforce a norm, even if there is no apparent benefit of doing it. This situation becomes interesting when a particular norm is unpopular in nature.

Unpopular social norms are those norms with which the majority of people do not agree or believe in it internally. In fact, people personally do not agree with unpopular norms but still stick to them. Individuals' may even unintentionally enforce others to follow them. Such cases in sociology are dealt through a dilemma called, Emperor's Dilemma, as illustrated by Nkomo in [24]. Emperor's dilemma relates to a tale in which everyone shows fake admiration for a new gown worn by an emperor even though the emperor was naked. The cunning gown designers announced that the (non-existent) gown would not be visible to those who are not loyal to the emperor or who are really dumb. The fear of being punished and of being identified as having inferior societal traits, no one spoke the truth. The truth that the emperor was in fact naked.

It is evident that the Emperor's Dilemma is demonstrated in many places around the world in one way or the other. Whether it is foot-binding in neo-Confucian China or inter-cousin marriages and dowry in Asia (indicated by Blake in [2] and Hughes in [11], respectively), the nature of the thought process is the same. People do not reveal what they really believe from the fear of being identified as ignorant or anti-social.

It is not that harmful if unpopular norms are followed at an individual level. However, when a large population adopts unpopular norms, following it becomes a kind of default behavior that might influence the neutral part of the population. As a consequence, it has been observed that people even start enforcing unpopular norm which they disapprove in private. This behavior is generally termed as false enforcement. Willer et al. in [29] focused on finding out the reasons for false enforcement. According to them, people falsely enforce unpopular norms to create an illusion of sincerity rather than conviction. They performed experiments using two scenarios, namely, wine tasting and text evaluation. Experimental results revealed that people who enforced a norm even against their actual belief, in fact, criticized different alternate variations of an unpopular norm. In short, their outcomes indicate that how social pressure can lead to false enforcement of an unpopular norm.

Un-popular Norms (UNs) could have an adverse impact on society and it is, therefore, sometimes necessary to oppose and possibly avert them. To achieve this goal, it is important to know the conditions which enable the persistence of unpopular norms and models that support possible aversion of these norms. This study attempts to elaborate on the conditions and reasons behind the emergence, spreading and aversion of unpopular norms in a society, using a theory-driven agent-based simulation.

The rest of the paper is structured as follows. Section 1 introduced the research work presented in this work. Related work is provided in section 2. The current models and then the proposed model is presented in section 3. Detailed analysis and comparison are provided in 4. Section 5 ends this paper with conclusions and future direction of this work.

2 Related work

The propagation and transformation of norms co-evolve with each other. Norms propagate through diffused influence. Since the subjects being influenced may have their own perspective, they may decide to adhere or reject it. As a result, the reciprocating influence of the subjects may transform the norm itself. According to Macy and Flache, exploration of scenarios of such a nature has been a subject of complex adaptive systems and they are investigated by developing agent-based models [18]. Understanding the emergence of norms in a society of agents is a challenge and an area of ongoing research [27].

Studying norms in society have been one of the research focus of agent-based modeling community. Theoretical studies on norms such as those conducted by Conte and Castelfranchi [6] and Meneguzzi et al. [20] explored that agent are supposed to comply with social norms. The sense of punishment from the society is evident as the predominant factor behind compliance of norms [4]. Studies conducted by Sanchez-Anguix et al. [25] and Sato and Hashimoto [26] focused on the emergence of norms and they described strategies showing how norms prevail in a society. This is basically governed by societal influence. Agents set their goals and frequently change their behavior based on societal influence until a global equilibrium in achieved [27]. Though lots of work has been done on the emergence and prevalence of norms, very limited is carried out for the aversion of unpopular norms. To the best, our knowledge, our previous work [31, 22, 23] is the only agent-based study on this exciting research area. Willer et al. have pointed out many "empirical cases in which individuals are persuaded to publicly support behaviors or beliefs that they privately question" [29]. The term, Preference falsification, coined by Kuran [17] is defined as "the act of misrepresenting one's genius wants under perceived social pressures". According to him, an equilibrium is the sum of three utilities namely, intrinsic, expressive, and reputation. The intrinsic utility is about an individual's personal satisfaction being part of the society. The expressive

utility is about an individual gain in the response of presenting himself/herself to be what is expected. The utility that is acquired through the reaction of others is termed as reputation utility. The concept of an unpopular norm is very close to the concept of preference falsification, in which individuals publicly lie about their privately held preferences [16]. According to Makowsky and Rubin [19], such societies are "prone to cascades of preference revelation if preferences are interconnected - where individuals derive utility from conforming to the actions of others". Further, "ICTs and preference falsification complement each other in the production of revolutionary activity. The former facilitates the transmission of shock while the latter increases the magnitude of change that arises after a shock." The utility acts in two different ways in the propagation of unpopular norms. At one end, it can force an individual to follow an unpopular norm, or even falsely enforce it. On the other end, it can propagate an opposite sentiment as a result of private preference revelation. There is a number of evidence that a minority of activists (capable of revealing their private preferences on will) can make a big difference but in a conducive environment [13]. So, the relevant question in this context becomes *"Can a minority of activists change an unpopular norm adopted by the majority?"*.

3 The proposed extended model

To avert UNs, it is important to understand conditions that might help to stop the propagation of these norms. Particularly, it is imperative to find the conditions necessary to establish an alternative norm - a reciprocal norm of prevailing UN, and the conditions that enforce people other than activists to follow the alternate norm. This section first introduces the social interaction model for following UNs proposed by Centola et al. [5]. It, then, provides briefly our previous extension to this model followed by the proposed extension in this paper.

3.1 Centola's model of norm aversion

Centola's model [5] is capable of elaborating the conditions and reasons behind the emergence, spreading, and aversion of UNs in society but using theory-driven approach. Consider an Un-popular Norm (UN) prevailing in a society. Assume that a minority of the population truly believe in it due to some vested interest. Agents representing this population are termed as True Believerss (TBs). Contrarily, a majority of the population do not believe in the UN. Agents representing this population are termed as Dis-Believerss (DBs). Figure 1(a) illustrates a sample distribution scenario.

Centola's model is based on four variables explained below:

1) Belief: an agent's belief in UN which is 1 in case of TBs and -1 in case of DBs.
2) Compliance: means that an agent is complying with a UN or not? Initially, all TBs are complying ($compliance = 1$) and DBs are not complying ($compliance = -1$).
3) Enforcement: is an agent influence on the neighborhood. Starting with a default value of 0, it can either be -1 or 1.
4) Strength: is an agent's resistance against compliance of a UN.

An agent *i*'s belief is a static value. The value of compliance may change using Equation 1.

$$compliance_i = \begin{cases} -belief_i \;\; if \; (\frac{-belief_i}{N_i} \times NE_i) > strength_i \\ belief_i \qquad\qquad\qquad\qquad otherwise \end{cases} \tag{1}$$

$$enforcement_i = \begin{cases} -belief_i, \; if \; (\frac{-belief_i}{N_i} \times NE_i) > (strength_i + k) \bigwedge (belief_i \neq compliance_i) \\ belief_i, \; if \; (strength_i \times enforcement\_need_i > k) \bigwedge (belief_i = compliance_i) \\ 0, \;\; otherwise \end{cases} \tag{2}$$

Where, NE_i = count of (Moore's) neighbors enforcing opposite belief and N_i = count of (Moore's) neighbors. This means that an agent's decision to comply with UN or not is dependent on the enforcement of opposite belief by the neighborhood. If NE_i is greater than the strength of a DB, the agent would comply against its belief. Since, TBs compliance (which equals their belief about a UN) and strength are already equal to 1, Equation 1 would not change the compliance value of TBs.

When compliance is decided, an enforcement decision is made next. Enforcement value may change using Equation 2.

Equation 3 is used to compute $enforcement\_need_i$ - that is the need of enforcement reflecting influence of neighborhood compliance.

$$enforcement\_need_i = \frac{(1 - \frac{belief_i}{N_i}) \times NC_i}{2} \tag{3}$$

Where, NC_i = number of (Moore's) neighbors whose

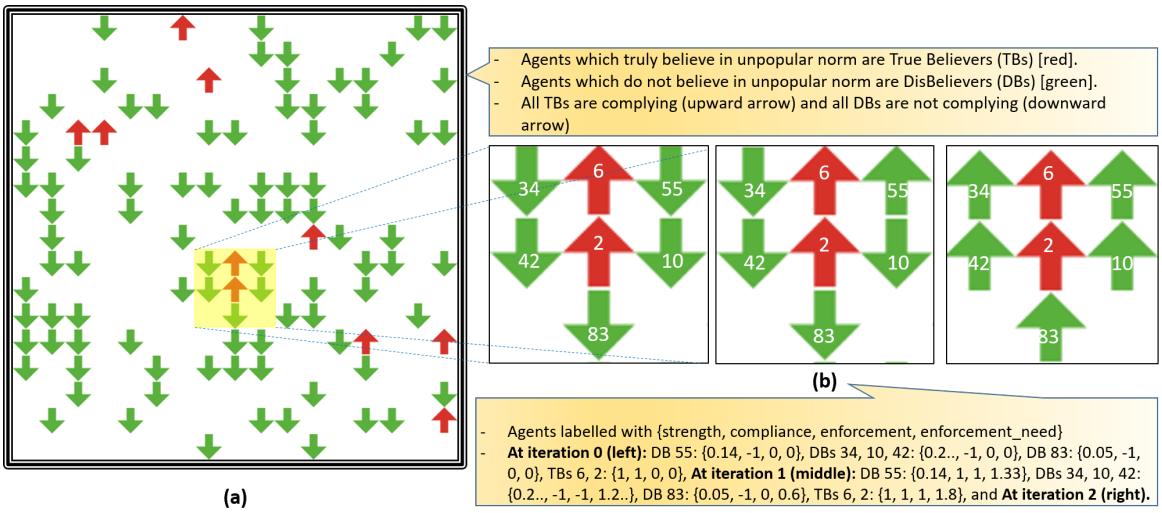


Figure 1: (a) Simulation set-up for 100 agents including 10% TBs. Initial values: TBs (belief = 1, strength = 1.0, compliance = 1, enforcement = 0), DBs (belief = -1, strength = [0.01-0.29], compliance = -1, enforcement = 0). (b) Changes in arrow directions in response of the application of Centola's model.

compliance is different than the agent's belief.

When an agent's belief is equal to compliance (true is the case of TBs and starting values of DBs), then the enforcement will be equal to belief but only when strength $\times$ enforcement_need of an agent is greater than a threshold variable k. Otherwise, it would remain 0. Since, the strength of DBs is kept very low, thus the condition would not result in enforcing -1 value by DBs. This condition will always enforce a value of 1 by TBs. On the other hand, when an agent's belief is not equal to compliance (true is the case of DBs with belief -1 and compliance 1 when Equation 1 is applied), the enforcement will be equal to the negation of belief but only when the enforcement of opposite belief in the neighborhood is greater than strength plus k value of the agent. This means that such a DB itself start enforcing a UN.

For example, the TB with ID 6 (see Figure 1 (b) - middle) uses Equation 3 to calculate the enforcement_need value equal to 1.6 for $belief_i = 1$, $N_i = 5$ and $NC_i = 4$. Thus, the value of compliance for the agent (from Equation 1) remains equal to its belief, because, neighborhood enforcement $(-(1)/5 \times 0$, where $0 = NE_i)$ is not greater than its strength 1. However, the second condition of Equation 2 changes enforcement from 0 equal to 1, because, the strength value of 1 multiplied with enforcement_need value of 1.6 gives a much greater than the enforcement threshold k which is considered 0.2 in this case. The same explanation applies to TB named 2.

DB (BD check it please), numbered 55 (see Figure 1(b)) applies Equation 3 to get the enforcement_need value equal to 1.33 for $belief_i = -1$, $N_i = 3$ and $NC_i = 2$. In this case, the value of compliance for the agent (from Equation 1) changes to the opposite of its belief, because, neighborhood enforcement $(-(-1)/3 \times 2$, where $2 = NE_i)$ is greater than its strength value of 0.14. The first condition of expression 2 changes the enforcement value from 0 equal to 1 as neighborhood enforcement $(-(-1)/3 \times 2$, where $2 = NE_i)$ is much greater than the enforcement threshold k (0.2 in this case) plus strength (0.14).

There are some DBs that do not comply at this point. For example, DB 34 by using Equation 3 calculates the enforcement_need value equal to 1.2 for $belief_i = -1$, $N_i = 5$ and $NC_i = 2$. The value of compliance for the agent (from Equation 1) remains unchanged, because, the neighborhood enforcement $(-(-1)/5 \times 1$, where $1 = NE_i)$ is not greater than its strength value of 0.216. The second condition of Equation 2 would make enforcement from 0 equal to -1, because strength (0.216) multiplied with enforcement_need (1.2) is slightly greater than the enforcement threshold k considered 0.2 in this case. The same applies to DB 10 and 42. Contrarily, the enforcement of DB 83 remains 0. These DBs, however, start complying at next iteration (see Figure 1 (b) - right) due to combined enforcement of their neighbors.

3.2 Our previous extension

Since, acpdb compliance in basic centolla's model is undesirable, in our previous work [31], we extended it and introduced a special kind of DBs (called Activists (ACTs)) with more desire to avert (act against) a UN. These ACTs are triggered by the presence of TBs in the surrounding, particularly who are enforcing. Their strength is progressively incremented proportionally to the intensity of enforcement from TBs. The strength of an ACT is calculated using Equation 4.

$$strength_i = strength_i + (\frac{E_{jb}}{N_i}) \tag{4}$$

Where, $E_{jb} =$ is the number of enforcing TBs.

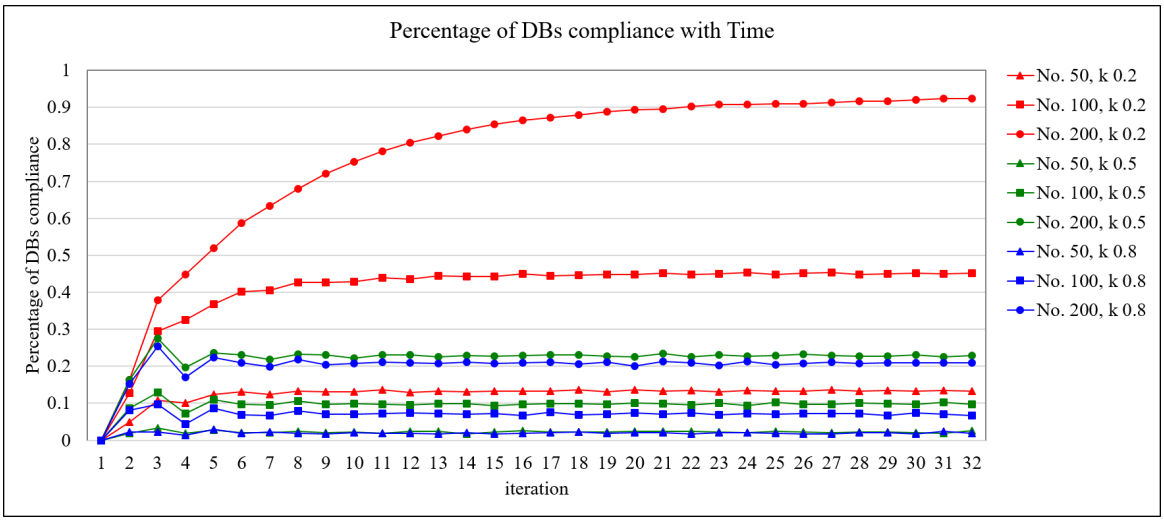


Figure 2: Simulation results of the basic Centolla's model for various scenarios based on number of agents (considered 50, 100, and 200) and threshold value k - showing an agent's desire to comply (considered 0.2, 0.5, and 0.8).

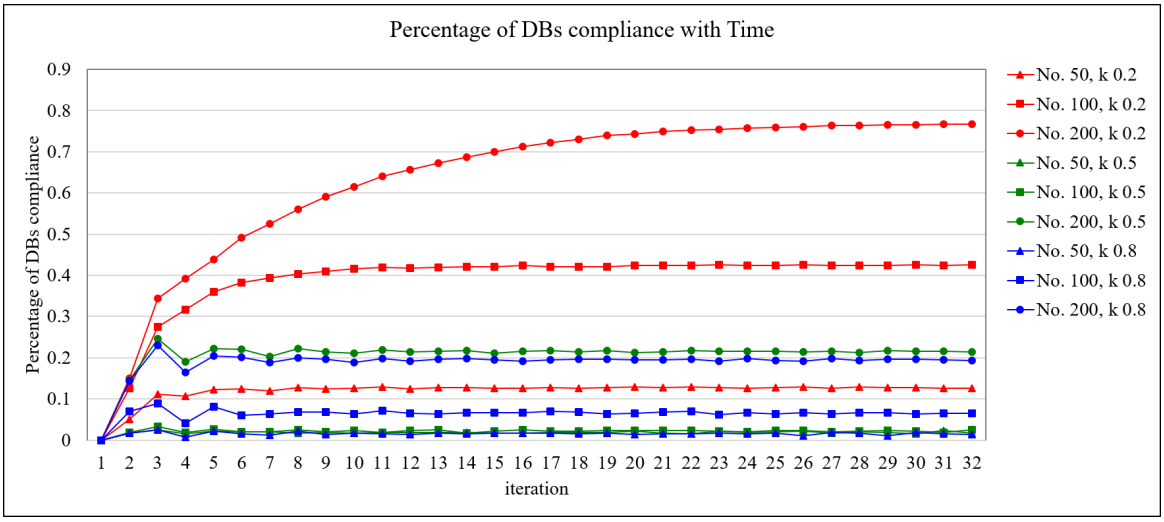


Figure 3: Simulation results of our previous extension to Centolla's model for various scenarios based on number of agents (considered 50, 100, and 200) and threshold value k - showing an agent's desire to comply (considered 0.2, 0.5, and 0.8).

3.3 The proposed extension

In this paper, the model is further extended to incorporate the decision-making of a DB as a result of neighborhood condition. It is proposed that DBs (who are not ACTs) should not be considered as entirely a numb entity. We propose a decision-making model represented in Equation 5. In this model, the strength of DBs (who are not ACTs) is changed (increased or decreased) based on its type being either "optimistic" or "pessimistic". The difference between percentage of enforcing TBs (termed as, P_{jb}) and percentage of complying DBs (termed as, P_{jd}) is divided by neighborhood density (N_i) times the fraction of DBs of that type (consider opt for an optimistic and "$1-opt''$ for a pessimistic DB). If an agent belongs to the optimistic category, its strength would be increased/decreased based on the difference of "true enforcement" (represented as P_{jb}) and "false compliance" (represented as P_{jd}). When fast compliance is more then the strength will decrease. On the other hand, when true enforcement is more then the strength will increase.

4 Evaluation and results

4.1 Simulation environment

NetLogo [28] - a popular agent-based simulation tool with support for grid-based spaces, is used to simulate the work presented in this work. The agents reside on cells of a spatial grid. We have used the concept of Moore's neighborhood to represent the surrounding of an agent - a very

$$strength_i = \begin{cases} strength_i + (P_{jb} - P_{jd})/(N_i \times opt), & if\ i\ is\ optimistic \\ strength_i + (P_{jb} - P_{jd})/(N_i \times (1 - opt)), & otherwise \end{cases} \quad (5)$$

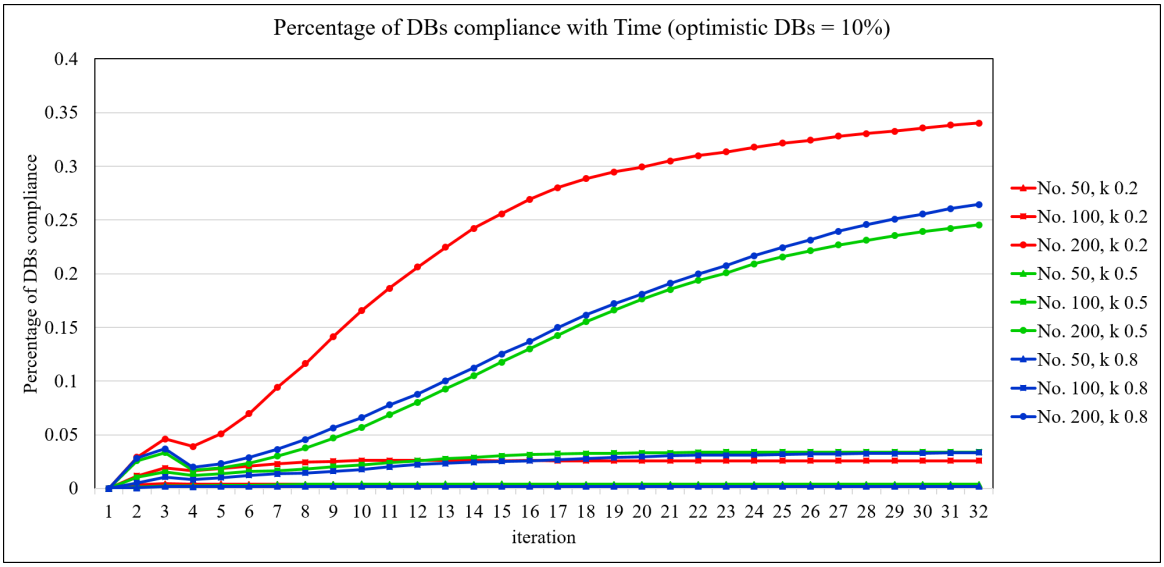


Figure 4: Simulation results of the proposed extension (with 10% agents of total population being optimistic) to Centolla's model for various scenarios based on number of agents (considered 50, 100, and 200) and threshold value k - showing an agent's desire to comply (considered 0.2, 0.5, and 0.8).

popular strategy in many cell-based spatial configurations [14]. For a coarse-grained evaluation, we used a simulation space consisting of a torus of 17 × 17 cells. Figure. 1(a) provides an illustration of this space filled with 100 agents.

4.2 Results and discussion

4.2.1 Previous findings

Due to the spatial nature of the neighborhood, it was expected that a more dense population is susceptible to more DBs compliance. This fact is evident from the results shown in Figure. 2. Further, DBs's compliance is inversely proportional to the value of k - an agent's desire to comply. Ironically, in all cases depicted in Figure. 2, the population achieves stability always being attracted towards various fixed points.

In our previous work [31], it was observed that in highly dense conditions with a large number of norm aversion ACTs, the aversion of unpopular norms can be achieved. This fact is highlighted in Figure. 3. There is a striking similarity between the basic model and our previously extended model whose results are presented in Figure. 2 and 3 in corresponding order. It is learned that the cases comprise of smaller values of k and a large number of agents are worst than the rest of the cases. A marginal improvement was achieved by introducing the ACTs where comparatively less number of DBs were witnessed complying with a UN.

4.2.2 Current findings: a brief analysis

This model uses optimistic DBs that are intrinsically believing in averting the UN. Simulation work conducted in this paper uses three different numbers of these optimistic DBs, which are counted as 10, 20 and 30% of the total population. It was learned that the proposed model significantly reduces the number of DBs complying with a UN. Even the scenario considered as a worst one (the one comprises of 200 agents and a threshold value k = 0.2) achieved a 100% improvement by drooping compliance rate from 70% to 35%. This is illustrated in Figure. 3 and 4.

When the proposed model is compared with the previous model, it was noted that DBs's compliance comparatively gets worse as the number of agents' increases irrespective of the value of k.

The cases where the number of agents is 200 always perform worse than other cases (comprising of 50 or 100 agents). This can be noticed while comparing the results presented in Figure. 3 with 4). Overall, with an increase in the number of optimistic DBs, the results get improved as witnessed by comparing the results given in Figure. 4, 5, and 6).

4.2.3 Current findings: a detailed analysis and comparison

In this section, we present a detailed analysis of the simulation results. The simulation space for these experiments comprises a torus of 33 × 33 cells. 1000 agents are placed on cells without overlapping. Figure 7(a) provides this

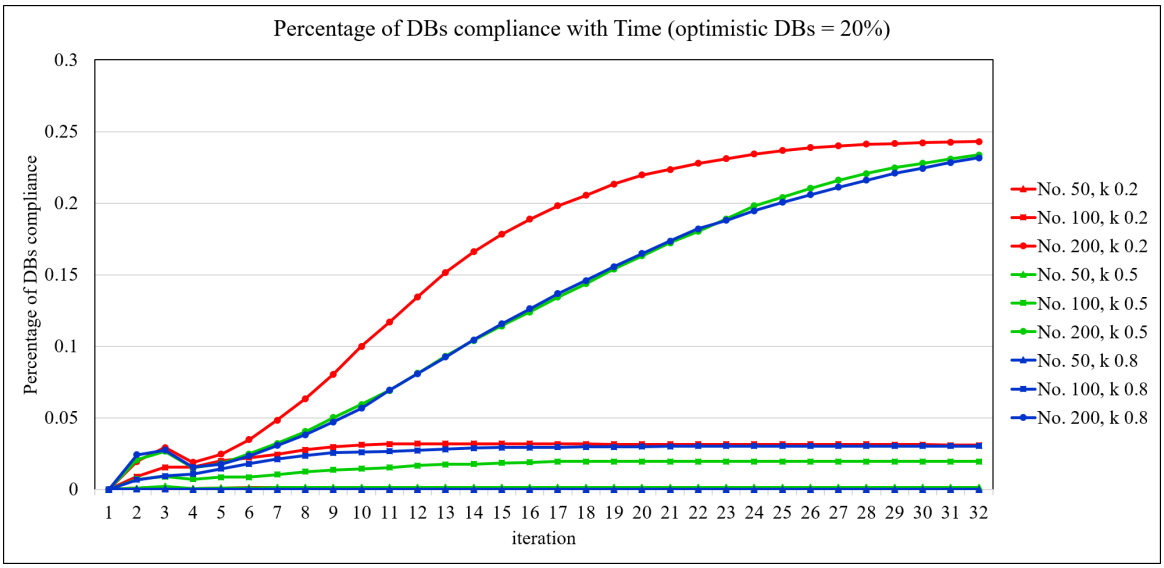


Figure 5: Simulation results of the proposed extension (with 20% agents of total population being optimistic) to Centolla's model for various scenarios based on number of agents (considered 50, 100, and 200) and threshold value k - showing an agent's desire to comply (considered 0.2, 0.5, and 0.8).

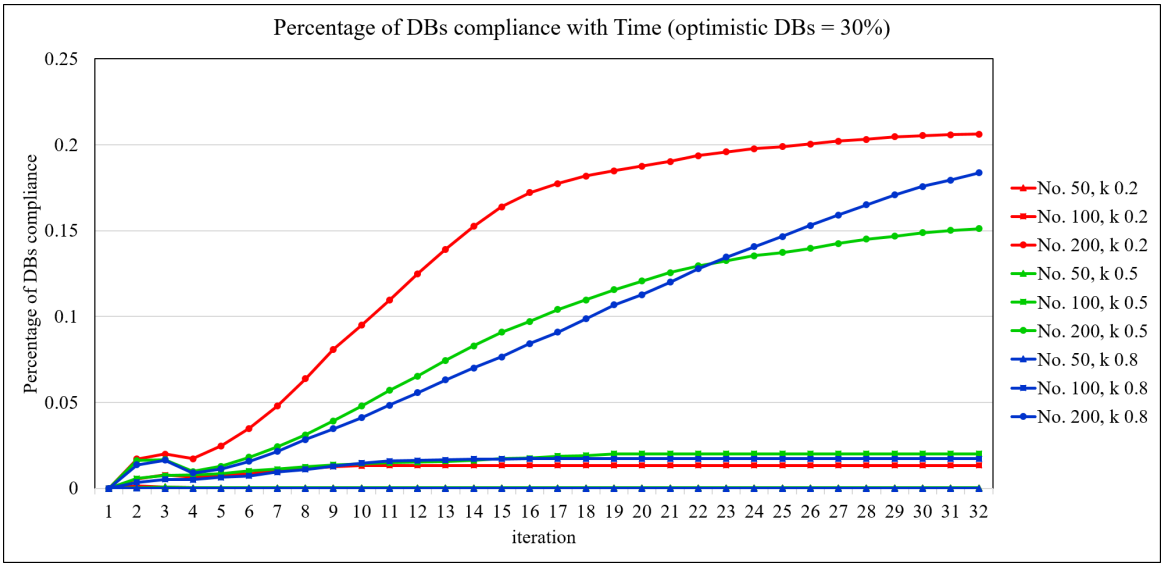


Figure 6: Simulation results of the proposed extension (with 30% agents of total population being optimistic) to Centolla's model for various scenarios based on number of agents (considered 50, 100, and 200) and threshold value k - showing an agent's desire to comply (considered 0.2, 0.5, and 0.8).

setup.

Simulation results are analyzed based on the following four quantities:

1) **DBComplBCount**: is the number of Dis-Believerss which comply with the Un-popular Norm, B, against their belief.

2) **DBFollBCount**: is the number of DBs which do not comply with the UN, B, but follow it against their belief.

3) **DBComplACount**: is the number of DBs which comply with the alternate norm, A, but still do not believe in it.

4) **DBBelACount**: is the number of DBs which comply with the alternate norm, A, and believe in it.

The purpose and intention of the proposed model are to reduce the value of **DBFollBCount** because these agents are unsure and their belief can potentially be averted. The possible aversion may transform agents status from "following" to those which are "complying" with the alternate norm (**DBComplACount**).

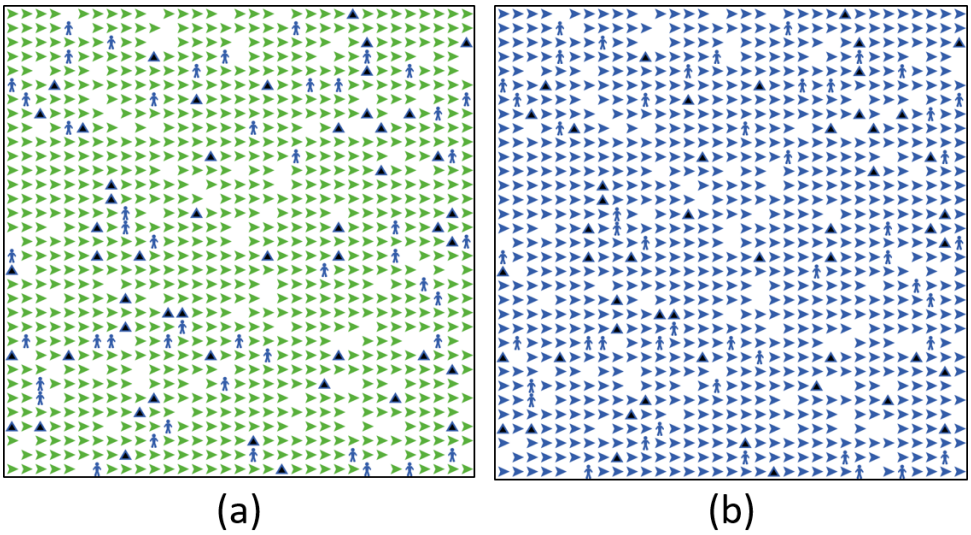


Figure 7: NetLogo Simulation. (a) Setup of 1000 agents with 5% TBs and 5% ACTs. TBs, ACTs, and DBs are represented as blue coloured triangles, blue coloured persons, and green coloured triangles correspondingly. (b) Results of basic Centola model [5]. Equilibrium state, where all DBs now comply with the unpopular norm, B, against their belief.

The basic model proposed by Centola [5] formulates the spread of a UN only. The results of the application of the model settle in an equilibrium state after the 5^{th} iteration. Figure 9(a) visualises the concept presented in Figure 7(b). It is evident from the results presented in Figure 9(a) that all DBs started with following the UN, quickly, started complying with it.

After all, DBs started complying with the norm, B, a change in strategy was tested. The ACTs was activated to play their role as proposed in [31]. The extended model proposed by Zareen et al. [31] reached at an equilibrium between 10^{th} to 12^{th} iteration. Figure 9(b) visualises the concept presented in Figure 8(a).

It is evident from the results shown in Figure 9(b) that DBs started complying with the alternate norm, A, under the influence of ACTs.

The number of DBs which transformed to compliance state merely changed to the following state again. Starting with an increase in the following agents, a decline was observed, however, it did not drop to 0. DBs following and complying to the norm, B, stabilizes with followers more than agents which are complying. As shown in Figure 8(a), DBs in the neighborhood of ACTs started following and complying norm, A, against their belief.

The proposed extended model achieved equilibrium with promising results. Figure 10(a) visualises the scenario presented in Figure 8(b). It is evident from the results given in Figure 10(a) that DBs started complying with alternate norm, A, under the influence of ACTs. Further, the majority of them started complying norm, A, with a belief in it. In response, the number of DBs following the norm, B, reduced to almost nothing.

Finally, we further increased the number of TBs and ACTs to a comparison with the scenarios just discussed. It was learned from Figure 10(a) and 10(b) that the pattern and state changes are similar. However, the aggregate number of DBs in state DBBelACount and DBFollBCount has decreased substantially when compared with the aggregate count of DBComplACount and DBComplBCount. It means that the DBs who believed in the norm, A, was decreased by almost 50% when the number of TBs and ACTs were doubled.

4.2.4 Discussion

The main objective of the proposed model was to reduce the number of disbelievers complying with an unpopular norm, B. It is clear from the simulation results given in Figure 9 that our previous model presented in [31] reduced the number of complying agents to 25% of the whole population as compared with 95% obtained by the standard model. However, 50% agents still follow the norm, B, and only 20% start complying with the alternate norm, A. The credit goes to the introduction of ACTs in the population of agents.

The introduction of optimistic agents in our current extension proposed in this paper significantly improved these results. Though the number of disbelievers complying with an unpopular norm does not change much, however, the majority of disbelievers started believing in alternate norm instead of following an unpopular norm. This is evident from comparing Figure 9(b) and Figure 10(a) with each other.

5 Conclusion

It is argued that for societal good, it is necessary to oppose and possibly avert unpopular norms. This work is an at-

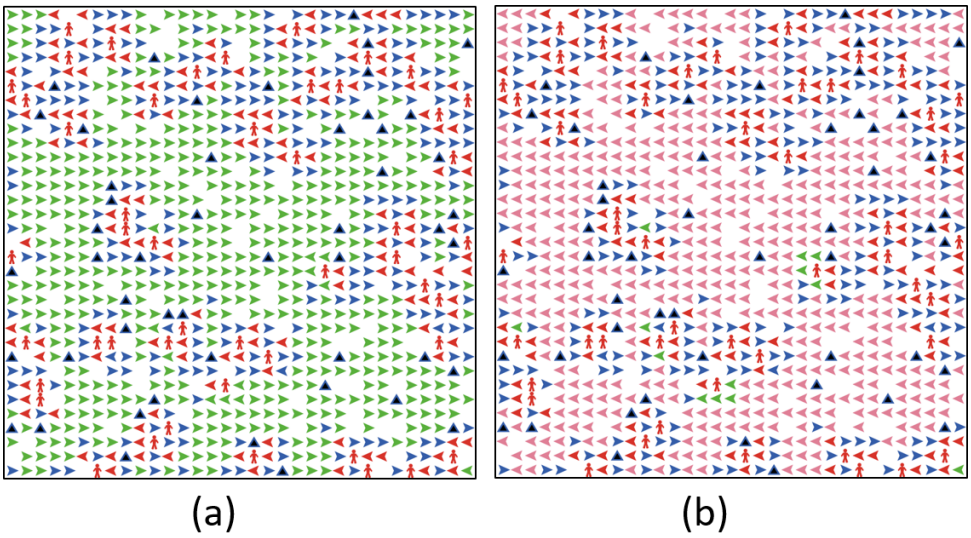


Figure 8: NetLogo Simulation. (a) Setup of 1000 agents with 5% TBs and 5% ACTs. TBs, and ACTs are represented as blue coloured triangles and blue coloured persons correspondingly.The rest of the agents are DBs.Simulation Result of the extended model proposed by Zareen et al. [31]. Equilibrium state, where DBs in the neighborhood of ACTs started following (blue) and complying (red) norm, A, against their belief. (b) Simulation results of the current proposed extended model.

tempt to realise the conditions that result in the emergence of unpopular norms and define situations under which these norms can be changed and averted. It presented an agent-based simulation for unpopular norm aversion. It utilised the reciprocal nature of persistence and aversion of norms to define situations under which these norms can be changed and averted. The simulation results revealed that in addition to agents actively participating in averting the unpopular norm, incorporating a rational decision-making model for normal agents is necessary to achieve a dominant norm aversion. Further, it was learned that the inclusion of true believers and activists play a significant role in norm aversion dynamics.

In short, this study revealed that more educated and socially active individuals are key to reduce undesirable norms in society. The significance of this fact is also applicable to digital societies primarily created by social networking applications nowadays.

References

[1] Cristina Bicchieri, Erte Xiao, and Ryan Muldoon. Trustworthiness is a social norm, but trusting is not. *Politics, Philosophy & Economics*, 10(2):170–187, 2011. URL: https://doi.org/10.1177/1470594X10387260.

[2] C Fred Blake. Foot-binding in neo-confucian china and the appropriation of female labor. *Signs: Journal of Women in Culture and Society*, 19(3):676–712, 1994. URL: https://doi.org/10.1086/494917.

[3] Geoffrey Brennan, Lina Eriksson, Robert E Goodin, and Nicholas Southwood. *Explaining norms*. Oxford University Press, 2013. URL: https://doi.org/10.1017/s0266267114000467.

[4] Will Briggs and Diane Cook. Flexible social laws. In *International Joint Conference on Artificial Intelligence*, volume 14, pages 688–693. Citeseer, 1995.

[5] Damon Centola, Robb Willer, and Michael Macy. The emperor's dilemma: A computational model of self-enforcing norms 1. *American Journal of Sociology*, 110(4):1009–1040, 2005. URL: https://doi.org/10.1086/427321.

[6] Rosaria Conte and Cristiano Castelfranchi. Are incentives good enough to achieve (info) social order? In *Social Order in Multiagent Systems*, pages 45–61. Springer, 2001. URL: https://doi.org/10.1007/978-1-4615-1555-5_3.

[7] Robert EL Faris. Handbook of modern sociology. 1964.

[8] Michael Hechter and Karl-Dieter Opp. *Social norms*. Russell Sage Foundation, 2001.

[9] Christine Horne. Explaining norm enforcement. *Rationality and Society*, 19(2):139–170, 2007. URL: https://doi.org/10.1177/1043463107077386.

[10] Christine Horne. *The rewards of punishment: A relational theory of norm enforcement*. Stanford University Press, 2009. URL: https://doi.org/10.1017/s0003975609990208.

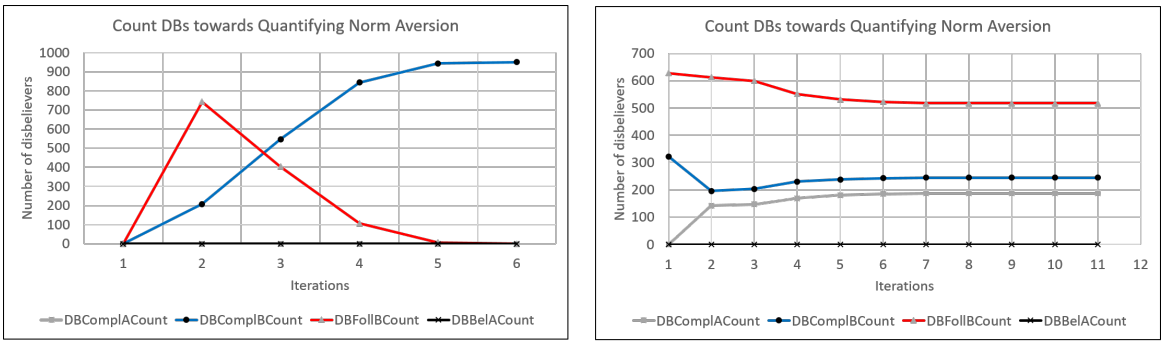


Figure 9: Simulation outcome in terms of number of DBs in various states against time of: (a) the basic Centola's model [5]. (b) the extended model proposed by Zareen et al. [31].

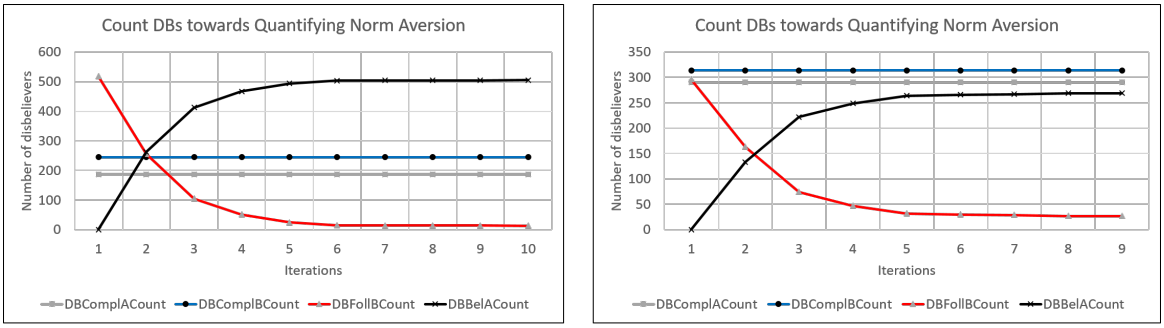


Figure 10: For optimistic agents, the simulation outcome of the proposed extended model in terms of number of DBs in various states against time for the scenarios comprises: (a) 5% TBs and 5% ACTs. (b) 10% TBs and 10% ACTs.

[11] Diane Owen Hughes. From brideprice to dowry in mediterranean europe. *Journal of family history*, 3(3):262–296, 1978. URL: https://doi.org/10.1177/036319907800300304.

[12] Coleman James. Foundations of social theory. *Cambridge, MA: Belknap*, 1990.

[13] Habibul Haque Khondker. Role of the new media in the arab spring. *Globalizations*, 8(5):675–679, 2011. URL: https://doi.org/10.1080/14747731.2011.621287.

[14] Tobias Kretz and Michael Schreckenberg. Moore and more and symmetry. In *Pedestrian and evacuation dynamics 2005*, pages 297–308. Springer, 2007. URL: https://doi.org/10.1007/978-3-540-47064-9_26.

[15] Dorothea Kübler. On the regulation of social norms. *Journal of Law, Economics, and Organization*, 17(2):449–476, 2001.

[16] Timur Kuran. The inevitability of future revolutionary surprises. *American Journal of Sociology*, 100(6):1528–1551, 1995. URL: https://doi.org/10.1086/230671.

[17] Timur Kuran. *Private truths, public lies: The social consequences of preference falsification*. Harvard University Press, 1997. URL: https://doi.org/10.1177/000169939603900109.

[18] Michael Macy and Andreas Flache. Social dynamics from the bottom up: Agent-based models of social interaction. *The oxford handbook of analytical sociology*, pages 245–268, 2009. URL: https://doi.org/10.1093/oxfordhb/9780199215362.013.11.

[19] Michael D Makowsky and Jared Rubin. An agent-based model of centralized institutions, social network technology, and revolution. *PloS one*, 8(11):e80380, 2013. URL: https://doi.org/10.1371/journal.pone.0080380.

[20] Felipe Meneguzzi, Odinaldo Rodrigues, Nir Oren, Wamberto W Vasconcelos, and Michael Luck. Bdi reasoning with normative considerations. *Engineering Applications of Artificial Intelligence*, 43:127–146, 2015. URL: https://doi.org/10.1016/j.engappai.2015.04.011.

[21] Paul Morrow. The thesis of norm transformation in the theory of mass atrocity. *Genocide Studies and Prevention: An International Journal*, 9(1):8, 2015.

[22] Arshad Muhammad, Kashif Zia, and Dinesh Kumar Saini. Agent-based simulation of socially-inspired model of resistance against unpopular norms. In *10th International Conference on Agents and Artificial Intelligence*, pages 133–139, 2018. URL: https://doi.org/10.5220/0006735501330139.

[23] Arshad Muhammad, Kashif Zia, and Dinesh Kumar Saini. Population dynamics necessary to avert unpopular norms. In *Agents and Artificial Intelligence*, pages 64–75. Springer International Publishing, 2019. URL: https://doi.org/10.1007/978-3-030-05453-3_4.

[24] Stella M Nkomo. The emperor has no clothes: Rewriting "race in organizations". *Academy of Management Review*, 17(3):487–513, 1992. URL: https://doi.org/10.5465/amr.1992.4281987.

[25] Victor Sanchez-Anguix, Vicente Julian, Vicente Botti, and Ana García-Fornes. Tasks for agent-based negotiation teams: Analysis, review, and challenges. *Engineering Applications of Artificial Intelligence*, 26(10):2480–2494, 2013. URL: https://doi.org/10.1016/j.engappai.2013.07.006.

[26] Takashi Sato and Takashi Hashimoto. Dynamic social simulation with multi-agents having internal dynamics. In *New Frontiers in Artificial Intelligence*, pages 237–251. Springer, 2007. URL: https://doi.org/10.1007/978-3-540-71009-7_21.

[27] George A Vouros. The emergence of norms via contextual agreements in open societies. In *Advances in Social Computing and Multiagent Systems*, pages 185–201. Springer, 2015.

[28] Uri Wilensky and William Rand. *An introduction to agent-based modeling: modeling natural, social, and engineered complex systems with NetLogo*. MIT Press, 2015. URL: https://doi.org/10.1063/pt.3.2884.

[29] Robb Willer, Ko Kuwabara, and Michael W Macy. The false enforcement of unpopular norms. *American Journal of Sociology*, 115(2):451–490, 2009. URL: https://doi.org/10.1086/599250.

[30] H Peyton Young. Social norms. 2007.

[31] Zoofishan Zareen, Muzna Zafar, and Kashif Zia. Conditions facilitating the aversion of unpopular norms: An agent-based simulation study. *International Journal of Advanced Computer Science and Applications*, 7(7), 2016. URL: https://doi.org/10.14569/ijacsa.2016.070769.

https://doi.org/10.31449/inf.v43i2.2217 Informatica **43** (2019) 221–234 **221**

A New Ensemble Semi-supervised Self-labeled Algorithm

Ioannis Livieris
Department of Computer & Informatics Engineering
Technological Educational Institute of Western Greece, Greece, GR 263-34
E-mail: livieris@teiwest.gr

Keywords: semi-supervised methods, self-labeled, ensemble methods, classification, voting

Received: March 13, 2018

As an alternative to traditional classification methods, semi-supervised learning algorithms have become a hot topic of significant research, exploiting the knowledge hidden in the unlabeled data for building powerful and effective classifiers. In this work, a new ensemble-based semi-supervised algorithm is proposed which is based on a maximum-probability voting scheme. The reported numerical results illustrate the efficacy of the proposed algorithm outperforming classical semi-supervised algorithms in term of classification accuracy, leading to more efficient and robust predictive models.

Povzetek: Razvit je nov delno nadzorovani učni algoritem s pomočjo ansamblov in glasovalno shemo na osnovi največje verjetnosti.

1 Introduction

The development of a powerful and accurate classifier is considered as one of the most significant and challenging tasks in machine learning and data mining [3]. Nevertheless, it is generally recognized that the key to recognition problems does not lie wholly in any particular solution since no single model exists for all pattern recognition problems [28, 15].

During the last decades, in the area of machine learning the development of an ensemble of classifiers has been proposed as a new direction for improving the classification accuracy. The basic idea of ensemble learning is the combination of a set of diverse prediction models, each of which solves the same original task, in order to obtain a better composite global model with more accurate and reliable estimates or decisions than can be obtained from using a single model [9, 28]. Therefore, several prediction models have been proposed based on ensembles techniques which have been successfully utilized to tackle difficult real-world problems [31, 14, 32, 30, 23, 27, 11]. Traditional ensemble methods usually combine the individual predictions of supervised algorithms which utilize only labeled data as training set. However, in most real-world classification problems, the acquisition of sufficient labeled samples is cumbersome and expensive and frequently requires the efforts of domain experts. On the other hand, unlabeled data are fairly easy to obtain and require less effort of experienced human annotators.

Semi-supervised learning algorithms constitute the appropriate and effective machine learning methodology for extracting useful knowledge from both labeled and unlabeled data. In contrast to traditional classification approaches, semi-supervised algorithms utilize a large amount of unlabeled samples to either modify or reprioritize the hypothesis obtained from labeled samples in order to build an efficient and accurate classifier. The general assumption of these algorithms is to leverage the large amount of unlabeled data in order to reduce data sparsity in the labeled training data and boost the classifier performance, particularly focusing on the setting where the amount of available labeled data is limited. Hence, these methods have received considerable attention due to their potential for reducing the effort of labeling data while still preserving competitive and sometimes better classification performance (see [18, 6, 7, 38, 17, 16, 21, 20, 22, 44, 45, 46, 43] and the references therein). The main issue in semi-supervised learning is how to exploit the information hidden in the unlabeled data. In the literature, several approaches have been proposed each with different philosophy related to the link between the distribution of labeled and unlabeled data [46, 4, 36].

Self-labeled methods constitute semi-supervised methods which address the shortage of labeled data via a self-learning process based on supervised prediction models. The main advantages of this class of methods are their simplicity and their wrapper-based philosophy. The former is related to the facility/comodity of application and implementation while the latter refers to the fact that any supervised classifier can be utilized, independent of its complexity [35]. In the literature, self-labeled methods are divided into self-training [41] and co-training [4]. Self-training constitutes an efficient semi-supervised method which iteratively enlarges the labeled training set by adding the most confident predictions of the utilized supervised classifier. The standard co-training method splits the feature space into two different conditionally independent views. Subsequently, it trains one classifier in each specific view and the classifiers teach each other the most confidently predicted examples. More sophisticated and advanced variants

of this method do not require explicit feature splits or the iterative mutual-teaching procedure imposed by co-training, as they are commonly based on disagreement-based classifiers [44, 12, 36, 46, 45]

By taking these into consideration, ensemble methods and semi-supervised methods constitute two significant classes of methods. The former attempt to achieve strong classification performance by combining individual classifiers while the later attempt to enhance the performance of a classifier by exploiting the information in the unlabeled data. Although both methodologies have been efficiently applied to a variety of real-world problems during the last decade, they were almost developed separately. In this context, Zhou [43] advocated that ensemble learning and semi-supervised learning are indeed beneficial to each other and stronger learning machines can be generated by leveraging unlabeled data with the combination of diverse classifiers. More specifically, ensemble learning could be useful to semi-supervised learning since an ensemble of classifiers could be more accurate than an individual classifier. Additionally, semi-supervised learning could assist ensemble learning since unlabeled data can enhance the diversity of the base learner which constitute the ensemble and increase the ensemble's classification accuracy.

In this work, a new ensemble semi-supervised self-labeled learning algorithm is proposed. The proposed algorithm combines the individual predictions of three of the most representative SSL algorithms: Self-training, Co-training and Tri-training via a maximum-probability voting scheme. The efficiency of the proposed algorithm is evaluated on various standard benchmark datasets and the reported experimental results illustrate its efficacy in terms of classification accuracy, leading to more efficient and robust prediction models.

The remainder of this paper is organized as follows: Section 3 presents some elementary semi-supervised learning definitions and Section 4 presents a detailed description of the proposed algorithm. Section 5 presents the experimental results of the comparison of the proposed algorithm with the most popular semi-supervised classification methods on standard benchmark datasets. Finally, Section 6 discusses the conclusions and some research topics for future work.

2 Related work

Semi-Supervised Learning (SSL) and Ensemble Learning (EL) constitute machine learning techniques which were independently developed to improve the performance of existing learning methods, though from different perspectives and methodologies. SSL provides approaches to improve model generalization performance by exploiting unlabeled data; while EL explores the possibility of achieving the same objective by aggregating a group of learners. Zhou [43] presented an extensive analysis of how semi-supervised learning and ensemble learning can be efficiently fuse for the development of efficient prediction models. A number of rewarding studies which fuse and exploit their advantages have been carried out in recent years; some useful outcomes of them are briefly presented below.

Zhou and Goldman [42] have adopted the idea of ensemble learning and majority voting and proposed a new SSL algorithm which is based on the multi-learning approach. More specifically, this algorithm utilizes multiple algorithms for producing the necessary information and endorses a voted majority process for the final decision, instead of asking for more than one views of the corresponding data.

Along this line, Li and Zhou [17] proposed another algorithm, in which a number of Random trees are trained on bootstrap data from the dataset, named Co-Forest. The main idea of this algorithm is the assignment of a few unlabeled examples to each Random tree during the training process. Eventually, the final decision is composed by a simple majority voting. Notice that the utilization of Random Tree classifier for random samples of the collected labeled data is the main reason why the behavior Co-Forest is efficient and robust although the number of the available labeled examples is reduced. Xu et al. [40] applied this method for the predictions of protein subcellular localization providing some promising results.

Sun and Zhang [34] attempted to combine the advantages of multiple-view learning and ensemble learning for semi-supervised learning. They proposed a novel multiple-view multiple-learner framework for semi-supervised learning which adopted a co-training based learning paradigm in enlarging labeled data from a much larger set of unlabeled data. Their motivation is based on the fact that the use of multiple views is promising to promote performance compared with single-view learning because information is more effectively exploited; while at the same time, as an ensemble of classifiers is learned from each view, predictions with higher accuracies can be obtained than solely adopting one classifier from the same view. The experiments conduced on several datasets presented some encouraging results, illustrating the efficacy of the proposed method.

Roy et al. [29] presented a novel approach by utilizing a multiple classifier system in the SSL framework instead of using a single weak classifier for change detection in remotely sensed images. The proposed algorithm during the iterative learning process uses the agreement between all the classifiers which constitute the ensemble for collecting the most confident labeled patterns. The effectiveness of the proposed technique was presented by a variety of experiments carried out on multi-temporal and multi-spectral

datasets.

In more recent works, Livieris et al. [21] proposed a new ensemble-based semi-supervised method for the prognosis of students' performance in the final examinations. They incorporated a ensemble of classifiers as base learner in the semi-supervised framework. Based on their numerical experiments, the authors concluded that ensemble methods and semi-supervised methodologies could efficiently combined to develop efficient prediction models. Motivated by the previous work, Livieris et al. [22] presented a new ensemble-based semi-supervised learning algorithm for the classification of chest X-rays of tuberculosis, presenting some encouraging results.

3 A review on semi-supervised self-labeled classification

In this section, we present a formal definition of the semi-supervised classification problem and briefly describe the most relevant self-labeled approaches proposed in the literature. Let $x_p = (x_{p1}, x_{p2}, \ldots, x_{pD}, y)$ be an example, where x_p belongs to a class y and a D-dimensional space in which x_{pi} is the i-th attribute of the p-th sample. Suppose L is a labeled set of N_L instances x_p with y known and U is an unlabeled set of N_U instance x_q with y unknown. Notice that the set $L \cup U$ consists the training set. Moreover, there exists a test set T of N_T unseen instances where y is unknown, which has not been utilized in the training stage. Notice that the aim of the semi-supervised classification is to obtain an accurate and robust learning hypothesis with the use of the training set.

Self-labeled techniques constitute a significant family of classification methods which progressively classify unlabeled data based on the most confident predictions and utilize them to modify the hypothesis learned from labeled samples. Therefore, the methods of this class accept that their own predictions tend to be correct, without making any specific assumptions about the input data. In the literature, a variety of self-labeled methods has been proposed each with different philosophy and methodology on exploiting the information hidden in the unlabeled data. In this work, we focus our attention to Self-training, Co-training and Tri-training which constitute the most efficient and commonly used self-labeled methods [21, 20, 22, 35, 37, 36].

3.1 Self-Training

Self-training [41] is generally considered as the simplest and one of the most efficient SSL algorithms. This algorithm is a wrapper based SSL approach which constitutes an iterative procedure of self-labeling unlabeled data. According to Ng and Cardie [25] "*self-training is a single-view weakly supervised algorithm*" which is based on its own predictions on unlabeled data to teach itself. Firstly, an arbitrary classifier is initially trained with a small amount of labeled data, constituting its training set which is iteratively augmented using its own most confident predictions of the unlabeled data. More analytically, each unlabeled instance which has achieved a probability over a specific threshold *ConLev* is considered sufficiently reliable to be added to the labeled training set and subsequently the classifier is retrained.

Clearly, the success of Self-training is heavily depended on the newly-labeled data based on its own predictions, hence its weakness is that erroneous initial predictions will probably lead the classifier to generate incorrectly labeled data [46]. A high-level description of Self-training algorithm is presented in Algorithm 1.

Algorithm 1: **Self-training**

Input: L − Set of labeled instances.
U − Set of unlabeled instances.
ConLev − Confidence level.
C − Base learner.

Output: Trained classifier.

1 : **repeat**
2 : Train C on L.
3 : Apply C on U.
4 : Select instances with a predicted probability more than *ConLev* per iteration (x_{MCP}).
5 : Remove x_{MCP} from U and add to L.
6 : **until** some stopping criterion is met or U is empty.

3.2 Co-training

Co-training [4] is a SSL algorithm which utilizes two classifiers, each trained on a different view of the labeled training set. The underlying assumptions of the Co-training approach is that feature space can be split into two different conditionally independent views and that each view is able to predict the classes perfectly [33]. Under these assumptions, two classifiers are trained separately for each view using the initial labeled set and then iteratively the classifiers augment the training set of the other with the most confident predictions on unlabeled examples.

Essentially, Co-training is a "*two-view weakly supervised algorithm*" since it uses the self-training approach on each view [25]. Blum and Mitchell [4] have extensively studied the efficacy of Co-training and they concluded that if the two views are conditionally independent, then the use of unlabeled data can significantly improve the predictive accuracy of a weak classifier. Nevertheless, the assumption about the existence of sufficient and redundant views is a luxury hardly met in most real world scenarios. Algorithm 2 presents a high-level description of Co-training algorithm.

Algorithm 2: **Co-training**

Input: L − Set of labeled instances.
U − Set of unlabeled instances.
C_i − Base learner ($i = 1, 2$).

Output: Trained classifier.

Create a pool U' of u examples by randomly choosing from U.
repeat
  Train C_1 on $L(V_1)$.
  Train C_2 on $L(V_2)$.
  for each classifier C_i **do** ($i = 1, 2$)
    C_i chooses p samples (P) that it most confidently labels as positive and n instances (N) that it most confidently labels as negative from U.
    Remove P and N from U'.
    Add P and N to L.
  end for
  Refill U' with examples from U to keep U' at constant size of u examples.
until some stopping criterion is met or U is empty.

Remark: V_1 and V_2 are two feature conditionally independent views of instances.

3.3 Tri-Training

Tri-Training [44] consists of an improved version of Co-Training which overcomes the requirements for multiple sufficient an redundant feature sets. This algorithm constitutes a bagging ensemble of three classifiers, trained on the data subsets generated through bootstrap sampling from the original labeled training set. In case two of the three classifiers agree on the categorization of an unlabeled instance, then this is considered to be labeled and augment the third classifier with the newly labeled example. The efficiency of the training process is based on the strategy the "*majority teach minority*" which avoids the use of a complicated time consuming approach to explicit measure the predictive confidence, serving as an implicit confidence measurement,

In contrast to several SSL algorithms, Tri-training does not require different supervised algorithms as base learners which leads to greater applicability in many real world classification problems [12, 46, 19]. A high-level description of Tri-training is presented in Algorithm 3.

Algorithm 3: **Tri-training algorithm**

Input: L − Set of labeled instances.
U − Set of unlabeled instances.
C_i − Base learner ($i = 1, 2, 3$).

Output: Trained classifier.

for $i = 1, 2, 3$ **do**
  $S_i = \text{BootstrapSample}(L)$.
  Train C_i on S_i.
end for

repeat
  for $i = 1, 2, 3$ **do**
    $L_i = \emptyset$.
    for $u \in U$ **do**
      if $C_j(u) = C_k(u)$ **then** ($j, k \neq i$)
        $L_i = L_i \cup (u, C_j(u))$.
      end if
    end for
  end for
  for $i = 1, 2, 3$ **do**
    Train C_i on S_i.
  end for
until some stopping criterion is met or U is empty.

4 An ensemble semi-supervised self-labeled algorithm

In this section, the proposed ensemble SSL algorithm is presented which is based on the hybridization of ensemble learning with semi-supervised learning. Generally, the development of an ensemble of classifiers consists of two main steps: *selection* and *combination*.

The selection of the appropriate component classifiers which constitute the ensemble is considered essential for its efficiency and the key points for its efficacy is based on the diversity and the accuracy the component classifiers. A commonly and widely utilized approach is to apply diverse classification algorithms (with heterogeneous model representations) to a single dataset [24]. Moreover, the combination of the individual predictions of the classification algorithms takes place through several methodologies and techniques with different philosophy and performance [28, 9].

By taking these into consideration, the development of an ensemble of classifiers is considered to be constituted by the SSL algorithms: Self-training, Co-training and Tri-training. These algorithms are self-labeled algorithms which exploit the hidden information in unlabeled data with complete different methodologies since Self-training and Tri-training are single-view methods while Co-training is a multi-view method.

A high-level description of the proposed Ensemble Semi-supervised Self-labeled Learning (EnSSL) algorithm is presented in Algorithm 4 which consists of two phases: *Training* phase and *Testing* phase.

In the Training phase, the SSL algorithms which constitute the ensemble are trained independently, using the same labeled L and unlabeled U datasets (steps 1-3). Clearly, the total computation time of this phase is the sum of computation times associated with each component SSL algorithm. In the Testing phase, initially the trained SSL algorithms are applied on each instance in the testing set (step 6). Subsequently, the individual predictions of the three SSL algorithms are combined via a maximum probability-based voting scheme. More specifically, the SSL algorithm which exhibits the most confident prediction over an unlabeled example of the test set is selected (step 8). In case the confidence of the prediction of the selected classifier meets a predefined threshold (*ThresLev*) then the classifier labels the example otherwise the prediction is not considered reliable enough (step 9). In this case, the output of the ensemble is defined as the combined predictions of three SSL learning algorithms via a simple majority voting, namely the ensemble output is the one made by more than half of them (step 11). This strategy has the advantage of exploiting the diversity of the errors of the learned models by using different classifiers and it does not require training on large quantities of representative recognition results from the individual learning algorithms.

Algorithm 4: **EnSSL**

Input: L − Set of labeled training instances.
U − Set of unlabeled training instances.
T − Set of test instances.
ThresLev − Threshold level.

Output: The labels of instances in the testing set.

/* Phase I: Training phase */

1: Train Self-train(L, U).

2: Train Co-train(L, U).

3: Train Tri-train(L, U).

/* Phase II: Testing phase */

5: **for** each x from T **do**

6: Apply Self-train, Co-train, Tri-train classifiers on x.

7: Find the classifier C^* with the highest confidence prediction on x.

8: **if** (Confidence of $C^* \geq$ *ThresLev*) **then**

9: C^* predicts the label y of x.

10: **else**

11: Use majority vote to predict the label y of x.

12: **end if**

13: end for

5 Experimental results

In this section, the classification performance of the proposed algorithm is compared with that of Self-training, Co-training and Tri-training on 40 benchmark datasets from KEEL repository [2] in terms of classification accuracy.

Each self-labeled algorithm was evaluated deploying as base learners:

- C4.5 decision tree algorithm [26].
- RIPPER (JRip) [5] as the representative of the classification rules.
- kNN algorithm [1] as instance-based learner.

These algorithms probably constitute three of the most effective and most popular data mining algorithms for classification problems [39]. In order to study the influence of the amount of labeled data, four different ratios of the training data were used: 10%, 20%, 30% and 40%. Moreover, we compared the classification performance of the proposed algorithm for each utilized base learner against the corresponding supervised learner.

The implementation code was written in JAVA, using WEKA Machine Learning Toolkit [13]. The configuration parameters of all the SSL methods and base learners used in the experiments are presented in Tables 1 and 2, respectively. It is worth noticing that the base learners were utilized with their the default parameter settings included in the WEKA software in order to minimize the effect of any expert bias by not attempting to tune any of the algorithms to the specific datasets.

Table 3 presents a brief description of the datasets structure i.e. number of instances (#Instances), number of attributes (#Features) and number of output classes (#Classes). The datasets considered contain between 101 and 7400 instances, the number of attributes ranges from 3 to 90 and the number of classes varies between 2 and 15.

SSL Algorithm	Parameters
Self-training	Maximum number of iterations $= 40$. $c = 95\%$.
Co-training	Maximum number of iterations $= 40$. Initial unlabeled pool $= 75$.
Tri-training	No parameters specified.
EnSSL	*ThresLev* $= 95\%$.

Table 1: Parameter specification for all SSL algorithms employed in the experimentation.

Base learner	Parameters
C4.5	Confidence factor used for pruning $= 0.25$. Minimum number of instances per leaf $= 2$. Number of folds used for reduced-error pruning $=$ 3. Pruning is performed after tree building.
JRip	Number of optimization runs $= 2$. Number of folds used for reduced-error pruning $=$ 3. Minimum total weight of the instances in a rule $=$ 2.0. Pruning is performed after tree building.
kNN	Number of neighbors $= 3$. Euclidean distance.

Table 2: Parameter specification for all base learners employed in the experimentation.

Dataset	#Instances	#Features	#Classes
automobile	159	15	2
appendicitis	106	7	2
australian	690	14	2
automobile	205	26	7
breast	286	9	2
bupa	345	6	2
chess	3196	36	2
contraceptive	1473	9	3
dermatology	358	34	6
ecoli	336	7	8
flare	1066	9	2
glass	214	9	7
haberman	306	3	2
heart	270	13	2
housevotes	435	16	2
iris	150	4	3
led7digit	500	7	10
lymph	148	18	4
mammographic	961	5	2
movement	360	90	15
page-blocks	5472	10	5
phoneme	5404	5	2
pima	768	8	2
ring	7400	20	2
satimage	6435	36	7
segment	2310	19	7
sonar	208	60	2
spambase	4597	57	2
spectheart	267	44	2
texture	5500	40	11
thyroid	7200	21	3
tic-tac-toe	958	9	2
titanic	2201	3	2
twonorm	7400	20	2
vehicle	846	18	4
vowel	990	13	11
wisconsin	683	9	2
wine	178	13	3
yeast	1484	8	10
zoo	101	17	7

Table 3: Brief description of datasets.

Tables 4-7 present the experimental results using 10%, 20%, 30% and 40% labeled ratio, respectively regarding all base learners.

Table 8 presents the number of wins of each one of the tested algorithms according to the supervised classifier used as base learner and utilized the ratio of labeled data in the training, while the best scores are highlighted in bold. It should be mentioned that draw cases between algorithms have not been encountered. Clearly, the presented results illustrated that EnSSL is the most effective method in all cases except the one using kNN as base learner with a labeled ratio of 30%. In this case, Tri-training performs better in 13 datasets, followed by EnSSL (9 wins). It is worth noticing that

- Depending upon the the ratio of labeled instances in the training set, EnSSL illustrates the highest classification accuracy in 46.2% of the datasets for 10% labeled ratio, 40% of the datasets for labeled ratio 20%, 44.4% of the datasets for labeled ratio 30% and 44.4% of the datasets for 40% labeled ratio. Obviously, EnSSL exhibits better classification accuracy for 10% and 40% labeled ratio.
- Regarding the base classifier, EnSSL (C4.5) presents the best classification accuracy in 14, 20, 21 and 19 of the datasets using a labeled ratio of 10%, 20%, 30% and 40%, respectively. EnSSL (JRip) prevails in 18, 14, 16 and 16 of the datasets using a labeled ratio of 10%, 20%, 30% and 40%, respectively. EnSSL (kNN) exhibit the best performance in 11, 9, and 17 of the datasets using a labeled ratio of 10%, 20%, 30% and 40%, respectively. Hence, EnSSL performs better using C4.5 and JRip as base learners.

Dataset	C4.5	Self (C4.5)	Co (C4.5)	Tri (C4.5)	EnSSL (C4.5)	JRip	Self (JRip)	Co (JRip)	Tri (JRip)	EnSSL (JRip)	kNN	Self (kNN)	Co (kNN)	Tri (kNN)	EnSSL (kNN)
automobile	64,21%	71,63%	71,58%	66,46%	69,79%	64,88%	69,08%	70,33%	64,63%	65,33%	61,75%	72,29%	64,13%	69,00%	74,13%
appendicitis	76,27%	81,09%	83,00%	82,00%	82,00%	83,91%	82,09%	81,00%	83,09%	83,09%	82,00%	85,82%	85,82%	85,82%	85,82%
australian	84,20%	85,80%	85,65%	87,10%	86,67%	85,22%	85,65%	85,36%	86,23%	86,38%	83,19%	83,91%	85,36%	83,77%	84,93%
banana	74,40%	74,58%	74,85%	75,00%	74,85%	73,19%	72,89%	73,15%	73,25%	73,30%	72,38%	72,89%	73,15%	73,25%	73,30%
breast	70,22%	75,87%	75,54%	73,82%	75,54%	68,45%	69,91%	67,81%	73,12%	69,56%	73,03%	72,41%	73,09%	73,45%	73,45%
bupa	56,24%	57,98%	57,96%	57,96%	58,57%	56,24%	58,57%	57,96%	57,96%	57,96%	56,24%	58,57%	57,96%	57,96%	57,96%
chess	98,97%	99,41%	97,62%	99,44%	99,41%	97,97%	99,09%	97,68%	99,09%	99,19%	93,90%	96,34%	90,02%	96,56%	96,40%
contraceptive	48,75%	49,69%	50,98%	50,37%	50,30%	43,04%	43,65%	46,64%	46,57%	46,77%	48,95%	50,84%	51,12%	51,59%	51,12%
dermatology	92,60%	94,54%	90,17%	94,54%	95,36%	85,76%	87,15%	86,06%	89,61%	91,00%	94,79%	97,25%	94,53%	97,24%	96,97%
ecoli	79,77%	80,37%	74,99%	80,97%	79,78%	78,83%	77,99%	75,88%	79,48%	78,88%	80,93%	80,97%	77,37%	82,15%	82,15%
flare	72,23%	74,66%	71,76%	73,73%	74,10%	68,38%	71,20%	67,18%	70,44%	70,36%	72,04%	74,95%	63,32%	73,92%	74,20%
glass	63,51%	67,81%	62,73%	64,48%	67,32%	61,21%	68,25%	62,64%	55,30%	64,09%	64,03%	72,51%	71,56%	72,97%	73,44%
haberman	71,90%	72,24%	70,24%	70,24%	70,24%	70,91%	71,57%	70,26%	70,56%	70,90%	71,55%	70,89%	73,88%	74,20%	74,20%
heart	78,54%	78,57%	76,89%	80,53%	81,52%	78,92%	80,89%	80,23%	80,90%	81,23%	80,87%	79,88%	80,86%	81,19%	80,20%
housevotes	96,52%	96,56%	94,84%	93,51%	95,69%	96,96%	96,56%	96,58%	93,51%	95,69%	91,34%	91,85%	91,85%	91,85%	91,85%
iris	92,67%	94,00%	95,33%	94,67%	94,00%	92,00%	93,33%	91,33%	90,00%	94,00%	92,67%	93,33%	93,33%	95,33%	94,67%
led7digit	69,80%	71,80%	58,60%	53,20%	69,40%	68,00%	70,60%	69,00%	34,20%	69,80%	72,60%	73,00%	56,00%	53,00%	69,40%
lymph	70,95%	74,38%	73,76%	73,71%	73,71%	72,90%	74,29%	75,05%	72,29%	74,38%	76,95%	78,48%	80,57%	81,19%	80,48%
mammographic	82,41%	83,49%	83,01%	84,22%	84,34%	82,41%	83,25%	82,29%	83,86%	83,73%	82,05%	82,65%	82,29%	83,73%	83,25%
movement	40,28%	56,94%	50,00%	35,83%	52,78%	29,44%	56,94%	49,17%	31,94%	48,89%	40,28%	65,00%	56,94%	59,72%	65,56%
page-blocks	95,39%	96,58%	95,71%	96,49%	96,71%	95,96%	96,09%	95,65%	96,36%	96,47%	96,05%	96,27%	95,34%	96,27%	96,16%
phoneme	80,33%	81,79%	80,13%	81,24%	81,98%	79,40%	81,35%	80,16%	80,46%	81,46%	80,26%	82,27%	81,25%	81,87%	82,14%
pima	74,47%	73,81%	73,81%	74,46%	74,20%	74,47%	73,29%	72,90%	73,81%	73,16%	72,69%	72,38%	73,03%	73,15%	73,54%
ring	80,41%	80,82%	80,91%	81,20%	83,54%	91,84%	92,47%	92,62%	92,61%	93,08%	62,15%	61,66%	60,51%	62,19%	61,05%
satimage	83,20%	84,38%	83,98%	84,65%	85,39%	83,31%	83,62%	84,15%	83,43%	84,80%	88,48%	89,25%	88,47%	89,03%	89,46%
segment	92,55%	94,42%	90,30%	93,90%	94,89%	91,82%	90,87%	86,15%	90,09%	92,77%	93,33%	93,12%	90,52%	93,29%	93,77%
sonar	67,43%	73,57%	68,67%	71,19%	71,19%	68,86%	77,05%	72,69%	74,71%	76,12%	70,69%	78,95%	74,10%	73,67%	76,05%
spambase	91,55%	92,72%	91,13%	92,79%	92,89%	90,68%	92,37%	91,55%	91,89%	92,83%	92,39%	93,02%	92,33%	93,22%	93,31%
spectheart	67,50%	68,75%	70,00%	70,00%	70,00%	63,75%	72,50%	70,00%	71,25%	71,25%	63,75%	66,25%	68,75%	68,75%	68,75%
texture	84,55%	87,87%	86,02%	86,65%	88,95%	84,73%	86,91%	86,33%	86,20%	89,64%	94,75%	96,07%	95,13%	95,78%	96,22%
thyroid	99,17%	99,32%	98,72%	99,24%	99,28%	98,89%	99,17%	98,42%	99,17%	99,24%	98,43%	98,76%	98,53%	98,69%	98,87%
tic-tac-toe	81,73%	83,60%	85,70%	85,27%	85,38%	97,08%	97,49%	97,91%	97,60%	97,49%	97,29%	99,06%	98,75%	98,64%	98,96%
titanic	77,15%	76,83%	77,60%	77,65%	77,82%	77,06%	77,19%	76,92%	77,65%	77,69%	77,06%	76,83%	77,69%	77,60%	77,65%
twonorm	78,99%	79,54%	79,50%	79,51%	82,19%	83,99%	84,82%	84,39%	84,19%	86,61%	93,39%	93,59%	93,69%	93,70%	94,61%
vehicle	66,55%	70,33%	66,78%	68,66%	70,44%	62,17%	60,87%	60,04%	61,34%	60,99%	64,90%	70,69%	67,97%	69,38%	70,33%
vowel	97,27%	98,28%	97,57%	98,28%	98,28%	96,96%	98,18%	97,17%	98,28%	98,28%	95,85%	97,57%	95,85%	97,47%	97,57%
wisconsin	94,57%	94,56%	93,57%	94,13%	94,56%	93,99%	95,85%	93,84%	94,98%	95,12%	96,42%	96,70%	96,28%	96,70%	96,70%
wine	84,28%	89,90%	78,01%	88,79%	89,90%	86,44%	89,28%	86,41%	89,87%	90,98%	93,20%	95,52%	94,97%	95,52%	95,52%
yeast	75,13%	74,93%	74,86%	74,86%	74,86%	75,07%	74,19%	75,74%	75,13%	75,20%	75,21%	74,19%	75,07%	75,27%	75,14%
zoo	93,09%	92,09%	89,18%	92,09%	92,09%	84,09%	86,09%	87,09%	86,09%	86,09%	90,09%	95,09%	84,27%	95,09%	95,09%

Table 4: Classification accuracy (labeled ratio 10%).

Dataset	C4.5	Self (C4.5)	Co (C4.5)	Tri (C4.5)	EnSSL (C4.5)	JRip	Self (JRip)	Co (JRip)	Tri (JRip)	EnSSL (JRip)	kNN	Self (kNN)	Co (kNN)	Tri (kNN)	EnSSL (kNN)
automobile	66,08%	77,29%	62,75%	73,50%	76,00%	65,42%	69,67%	64,67%	71,50%	74,04%	64,17%	68,46%	65,92%	72,25%	74,08%
appendicitis	80,09%	81,09%	83,00%	82,91%	82,91%	83,91%	82,09%	82,00%	82,91%	82,00%	83,09%	86,82%	86,73%	85,82%	85,82%
australian	86,09%	86,67%	86,23%	87,10%	87,68%	85,51%	86,09%	85,80%	86,23%	86,09%	84,93%	85,94%	83,04%	84,06%	85,07%
banana	74,62%	74,57%	75,23%	75,08%	78,26%	73,36%	72,75%	74,21%	73,79%	75,13%	74,55%	72,75%	74,21%	73,79%	75,13%
breast	70,23%	74,16%	71,31%	75,54%	75,64%	69,24%	72,07%	68,51%	71,70%	71,01%	73,12%	70,68%	71,69%	72,75%	72,75%
bupa	57,41%	58,27%	57,96%	57,96%	58,57%	57,10%	58,27%	57,96%	57,96%	57,96%	57,10%	57,41%	57,96%	57,96%	57,96%
chess	99,00%	99,41%	98,18%	99,37%	99,41%	98,87%	99,09%	98,15%	99,03%	99,06%	94,90%	95,99%	91,02%	96,71%	96,40%
contraceptive	50,44%	50,17%	50,84%	50,44%	50,71%	43,04%	42,57%	46,64%	46,36%	45,75%	50,51%	50,37%	51,93%	49,83%	50,71%
dermatology	93,41%	92,63%	89,32%	93,99%	94,81%	85,77%	88,52%	85,49%	89,05%	91,52%	94,79%	96,97%	95,32%	96,97%	97,24%
ecoli	80,02%	79,48%	76,79%	79,19%	80,06%	80,62%	78,89%	77,66%	78,01%	78,58%	80,94%	79,20%	80,07%	81,29%	81,58%
flare	73,17%	75,42%	72,70%	73,35%	74,29%	68,95%	73,17%	72,70%	71,85%	73,73%	72,51%	74,29%	68,48%	73,36%	73,45%
glass	65,52%	67,34%	63,70%	64,96%	70,24%	63,12%	64,94%	65,02%	62,21%	66,47%	67,81%	66,84%	71,58%	69,13%	72,97%
haberman	72,24%	70,24%	70,24%	70,24%	70,24%	71,27%	70,24%	70,27%	69,91%	70,24%	71,87%	70,59%	73,56%	73,56%	73,24%
heart	79,25%	77,89%	77,60%	79,22%	80,20%	80,88%	78,58%	76,89%	79,56%	79,57%	80,92%	81,53%	82,86%	80,86%	81,52%
housevotes	96,52%	96,56%	95,69%	93,51%	95,69%	96,96%	96,99%	96,99%	93,08%	94,38%	91,79%	91,85%	91,85%	91,85%	91,85%
iris	94,00%	94,00%	93,33%	93,33%	93,33%	93,33%	93,33%	91,33%	93,33%	93,33%	93,33%	93,33%	94,00%	93,33%	94,67%
led7digit	70,40%	71,00%	65,60%	68,00%	70,20%	69,60%	70,00%	70,80%	58,80%	70,40%	73,00%	73,80%	67,00%	69,40%	71,20%
lymph	71,57%	75,71%	72,43%	74,43%	76,43%	74,48%	72,43%	76,38%	73,76%	75,10%	79,19%	79,81%	83,24%	81,19%	81,14%
mammographic	83,61%	82,65%	82,65%	84,10%	83,37%	83,25%	83,37%	82,89%	83,73%	83,61%	83,01%	83,49%	82,29%	83,98%	83,25%
movement	50,00%	59,17%	47,50%	47,22%	57,50%	43,33%	54,17%	51,94%	21,39%	45,83%	57,22%	63,06%	55,83%	61,11%	65,00%
page-blocks	96,36%	96,75%	96,02%	96,58%	96,78%	96,22%	96,49%	95,74%	96,55%	96,71%	96,13%	96,40%	95,69%	96,18%	96,16%
phoneme	80,51%	81,33%	80,00%	81,20%	81,79%	79,94%	81,12%	80,11%	81,05%	81,55%	81,25%	82,12%	81,49%	81,81%	82,35%
pima	74,48%	74,33%	73,15%	73,29%	73,81%	74,62%	74,73%	73,41%	73,28%	73,67%	73,47%	74,07%	73,54%	73,68%	73,67%
ring	81,00%	80,69%	81,12%	80,91%	83,76%	92,28%	92,62%	92,16%	93,01%	93,14%	62,20%	61,36%	60,58%	62,38%	61,04%
satimage	83,29%	84,57%	84,27%	84,15%	84,90%	83,40%	83,23%	83,00%	83,73%	84,55%	88,90%	89,28%	88,50%	89,42%	89,65%
segment	93,46%	94,37%	91,17%	94,03%	94,59%	92,16%	91,21%	88,96%	90,48%	92,47%	92,34%	92,90%	91,21%	93,64%	93,55%
sonar	70,76%	71,24%	73,12%	73,62%	76,07%	70,71%	69,81%	75,07%	70,26%	69,83%	74,50%	75,98%	74,64%	78,86%	79,88%
spambase	92,28%	92,89%	91,87%	92,81%	92,85%	90,94%	92,55%	91,78%	92,52%	92,89%	92,85%	93,18%	92,81%	93,39%	93,70%
spectheart	71,25%	68,75%	71,25%	70,00%	68,75%	65,00%	71,25%	70,00%	71,25%	71,25%	66,25%	66,25%	66,25%	67,50%	68,75%
texture	86,36%	87,29%	86,29%	87,42%	88,76%	85,33%	86,53%	86,13%	86,51%	89,31%	94,49%	96,27%	95,58%	96,05%	96,56%
thyroid	99,21%	99,32%	98,96%	99,25%	99,31%	99,01%	99,17%	98,54%	99,13%	99,19%	98,58%	98,65%	98,96%	98,58%	98,79%
tic-tac-toe	82,36%	86,11%	85,28%	84,96%	87,47%	97,39%	97,70%	98,02%	98,01%	97,91%	98,12%	98,12%	97,07%	98,64%	98,33%
titanic	77,19%	77,06%	77,19%	77,65%	77,24%	77,15%	77,46%	75,69%	77,65%	77,65%	77,15%	76,92%	77,06%	77,33%	76,96%
twonorm	79,74%	79,58%	79,39%	79,64%	82,70%	84,11%	83,72%	84,16%	84,07%	86,62%	93,50%	93,73%	93,61%	93,73%	94,69%
vehicle	68,56%	71,26%	66,78%	70,09%	71,62%	62,54%	60,17%	59,92%	61,11%	60,63%	65,37%	67,50%	67,73%	70,21%	69,97%
vowel	97,87%	98,08%	98,48%	98,38%	98,58%	97,77%	98,18%	98,08%	98,18%	98,18%	96,76%	96,86%	96,66%	97,17%	97,47%
wisconsin	94,70%	94,28%	94,57%	94,13%	94,42%	94,42%	95,71%	95,56%	95,99%	95,70%	96,42%	96,85%	96,56%	96,85%	96,70%
wine	88,82%	89,90%	87,61%	85,42%	87,68%	89,90%	88,76%	84,15%	89,93%	89,90%	93,24%	95,52%	94,41%	95,52%	95,52%
yeast	75,34%	76,07%	74,39%	75,00%	74,73%	75,20%	75,80%	75,14%	74,80%	75,20%	75,47%	74,86%	75,34%	75,41%	75,20%
zoo	94,00%	92,09%	82,18%	89,09%	91,09%	86,09%	84,18%	89,00%	86,09%	86,09%	92,09%	95,09%	81,27%	94,18%	94,18%

Table 5: Classification accuracy (labeled ratio 20%).

Dataset	C4.5	Self (C4.5)	Co (C4.5)	Tri (C4.5)	EnSSL (C4.5)	JRip	Self (JRip)	Co (JRip)	Tri (JRip)	EnSSL (JRip)	kNN	Self (kNN)	Co (kNN)	Tri (kNN)	EnSSL (kNN)
automobile	74,21%	73,46%	72,92%	77,29%	79,21%	67,92%	63,42%	70,38%	71,54%	72,83%	65,50%	61,63%	69,17%	70,96%	70,33%
appendicitis	82,00%	83,09%	83,00%	84,82%	84,00%	83,91%	83,91%	84,82%	83,82%	83,82%	85,73%	86,73%	86,73%	84,91%	86,73%
australian	85,94%	86,52%	85,80%	86,81%	86,67%	85,65%	85,94%	85,65%	85,80%	85,51%	84,20%	83,91%	85,07%	84,06%	85,64%
banana	74,70%	74,58%	75,36%	74,70%	78,81%	73,45%	72,89%	73,70%	73,11%	76,11%	74,66%	72,89%	73,70%	73,11%	76,11%
breast	70,32%	75,20%	74,16%	75,54%	75,74%	69,54%	75,17%	69,95%	71,32%	72,03%	73,23%	73,09%	71,69%	73,09%	72,75%
bupa	57,10%	57,98%	57,96%	57,96%	58,57%	57,41%	57,98%	55,67%	57,96%	57,96%	57,41%	55,92%	57,96%	57,96%	57,96%
chess	99,12%	99,41%	98,28%	99,41%	99,44%	98,90%	99,00%	98,12%	99,22%	99,31%	94,96%	94,15%	92,49%	96,71%	95,93%
contraceptive	50,85%	49,82%	50,91%	50,17%	51,72%	46,50%	44,60%	47,39%	46,98%	46,43%	51,39%	49,21%	51,66%	52,20%	51,11%
dermatology	94,80%	93,15%	90,97%	94,53%	95,08%	87,67%	88,81%	86,35%	87,40%	89,08%	95,88%	96,43%	96,15%	97,24%	96,97%
ecoli	80,06%	79,15%	77,07%	78,87%	78,57%	80,66%	79,51%	79,79%	76,53%	77,12%	81,24%	79,80%	80,37%	80,70%	80,70%
flare	73,63%	74,48%	74,20%	73,45%	73,73%	69,13%	71,00%	70,64%	70,55%	71,95%	72,61%	73,35%	71,57%	74,11%	73,73%
glass	66,47%	61,19%	65,95%	69,74%	70,15%	63,16%	63,66%	65,06%	67,40%	68,83%	69,70%	63,68%	60,80%	71,99%	70,65%
haberman	72,24%	71,86%	70,24%	70,24%	70,24%	71,91%	71,86%	70,90%	70,24%	70,24%	72,89%	70,91%	72,57%	73,54%	72,56%
heart	79,90%	76,27%	79,87%	78,88%	80,22%	81,23%	79,59%	79,22%	82,22%	81,87%	82,22%	80,19%	83,84%	81,52%	81,84%
housevotes	96,52%	96,56%	96,99%	96,56%	96,56%	96,96%	96,99%	96,56%	96,99%	96,99%	92,21%	91,85%	91,85%	92,26%	91,85%
iris	94,00%	94,00%	94,00%	93,33%	94,00%	93,33%	93,33%	92,00%	94,00%	93,33%	93,33%	94,00%	94,00%	92,00%	93,33%
led7digit	71,20%	70,40%	69,20%	71,00%	71,00%	70,40%	69,20%	71,60%	69,00%	71,00%	73,20%	73,60%	70,80%	70,80%	71,80%
lymph	76,33%	73,62%	76,43%	72,38%	71,71%	74,90%	75,76%	79,76%	75,86%	77,14%	79,81%	79,14%	77,86%	81,19%	80,52%
mammographic	83,73%	83,98%	82,05%	84,22%	84,10%	83,61%	84,10%	82,29%	84,10%	84,22%	83,37%	83,86%	82,53%	83,73%	83,96%
movement	55,28%	58,89%	51,67%	50,56%	61,39%	51,39%	54,44%	50,00%	38,33%	53,06%	59,11%	63,06%	54,44%	58,06%	63,61%
page-blocks	96,38%	96,47%	96,38%	96,69%	96,87%	96,29%	96,36%	96,11%	96,38%	96,60%	96,20%	96,20%	95,92%	96,33%	96,34%
phoneme	81,05%	81,01%	80,11%	81,31%	81,42%	80,61%	80,55%	80,64%	80,88%	81,44%	81,68%	81,98%	81,35%	82,20%	82,14%
pima	75,53%	74,84%	73,68%	74,72%	75,24%	75,25%	73,80%	72,65%	72,37%	73,02%	74,48%	74,51%	74,20%	72,76%	74,71%
ring	81,23%	80,30%	81,43%	81,03%	83,15%	92,59%	92,88%	91,80%	92,59%	92,88%	62,36%	61,15%	60,65%	62,26%	60,80%
satimage	84,29%	84,48%	84,41%	84,69%	85,18%	83,43%	83,39%	83,36%	83,56%	84,91%	88,90%	89,08%	88,98%	89,45%	89,76%
segment	93,68%	94,03%	91,73%	94,37%	94,76%	92,64%	91,13%	87,88%	90,30%	92,77%	92,55%	92,51%	90,82%	93,55%	93,55%
sonar	72,62%	71,69%	74,57%	76,10%	74,17%	74,55%	74,14%	71,69%	74,10%	76,50%	74,52%	77,50%	76,43%	72,21%	74,10%
spambase	92,70%	92,70%	92,13%	92,92%	92,87%	92,15%	91,78%	91,83%	92,31%	92,44%	92,98%	92,55%	92,94%	93,37%	93,26%
spectheart	71,25%	71,25%	68,75%	67,50%	68,75%	68,75%	70,00%	71,25%	71,25%	71,25%	70,00%	71,25%	68,75%	67,50%	68,75%
texture	86,44%	87,80%	86,73%	86,76%	88,85%	86,25%	86,44%	87,45%	86,56%	88,95%	95,64%	95,89%	95,85%	96,16%	96,40%
thyroid	99,25%	99,17%	99,22%	99,32%	99,28%	99,07%	99,04%	99,17%	99,00%	99,13%	98,61%	98,33%	98,68%	98,63%	98,64%
tic-tac-toe	83,30%	84,96%	85,80%	85,38%	88,41%	97,81%	97,70%	97,60%	98,02%	97,70%	98,54%	96,45%	97,07%	98,85%	98,85%
titanic	77,15%	77,28%	77,46%	77,10%	77,24%	77,24%	77,24%	77,46%	77,51%	77,24%	77,17%	77,19%	77,46%	77,19%	77,06%
twonorm	79,85%	79,53%	79,68%	81,18%	83,59%	84,82%	83,93%	84,73%	84,91%	87,38%	93,72%	93,88%	93,73%	93,93%	94,89%
vehicle	68,68%	70,45%	69,15%	69,74%	71,75%	62,77%	58,52%	60,64%	60,76%	60,76%	67,73%	66,20%	67,86%	70,21%	69,04%
vowel	97,87%	97,47%	97,67%	97,98%	97,87%	97,77%	97,57%	97,98%	98,38%	98,28%	97,07%	96,86%	96,05%	97,97%	97,77%
wisconsin	94,99%	94,99%	94,13%	94,42%	94,85%	95,28%	96,42%	94,41%	94,99%	94,98%	96,57%	96,70%	96,56%	96,70%	96,70%
wine	89,35%	88,79%	87,61%	91,57%	91,57%	91,57%	87,58%	88,73%	88,79%	88,17%	94,35%	96,08%	96,63%	95,52%	96,08%
yeast	75,41%	74,73%	75,20%	75,20%	75,54%	76,08%	75,13%	75,00%	75,54%	76,21%	75,68%	74,53%	74,59%	75,20%	75,07%
zoo	94,00%	93,09%	88,09%	94,00%	95,00%	87,09%	87,09%	81,18%	86,09%	86,09%	93,01%	94,09%	88,27%	93,09%	93,09%

Table 6: Classification accuracy (labeled ratio 30%).

Dataset	C4.5	Self (C4.5)	Co (C4.5)	Tri (C4.5)	EnSSL (C4.5)	JRip	Self (JRip)	Co (JRip)	Tri (JRip)	EnSSL (JRip)	kNN	Self (kNN)	Co (kNN)	Tri (kNN)	EnSSL (kNN)
automobile	74,25%	72,33%	77,33%	75,46%	81,13%	70,88%	59,71%	68,46%	70,96%	71,58%	67,92%	65,33%	64,75%	67,21%	69,75%
appendicitis	83,82%	81,09%	85,73%	82,00%	82,00%	83,91%	81,09%	83,82%	83,00%	83,00%	85,81%	82,09%	85,82%	84,91%	85,82%
australian	86,23%	85,80%	86,09%	87,54%	87,10%	85,65%	85,36%	85,94%	86,38%	85,36%	85,38%	84,93%	84,06%	84,20%	86,78%
banana	74,79%	74,66%	75,77%	74,72%	80,53%	73,47%	72,74%	73,55%	72,81%	75,70%	74,94%	72,74%	73,55%	72,81%	75,70%
breast	70,95%	71,34%	75,20%	75,16%	75,16%	70,41%	70,68%	70,33%	71,70%	70,67%	73,04%	72,73%	72,38%	72,75%	73,08%
bupa	58,04%	54,75%	57,67%	57,96%	58,57%	57,44%	54,75%	57,67%	55,67%	57,96%	57,54%	55,34%	57,67%	57,96%	57,96%
chess	99,22%	99,25%	99,03%	99,41%	99,41%	99,00%	99,19%	98,62%	99,12%	99,16%	95,71%	93,55%	93,30%	96,65%	95,96%
contraceptive	51,41%	48,00%	51,73%	50,03%	51,52%	46,87%	42,84%	46,98%	47,05%	46,88%	51,96%	47,93%	51,11%	52,07%	51,93%
dermatology	95,08%	93,46%	92,05%	94,26%	95,38%	87,71%	87,98%	88,25%	89,08%	90,17%	96,14%	96,43%	95,59%	97,24%	97,24%
ecoli	81,84%	77,67%	80,63%	79,48%	80,34%	81,22%	79,49%	77,69%	80,37%	79,80%	82,04%	80,96%	79,46%	80,69%	82,47%
flare	73,82%	73,63%	73,07%	74,29%	74,10%	69,23%	68,86%	71,76%	69,79%	70,64%	73,27%	73,17%	72,32%	73,64%	73,36%
glass	70,65%	61,58%	67,38%	68,72%	72,01%	66,76%	55,13%	67,79%	61,77%	67,79%	73,42%	62,19%	70,17%	73,40%	74,78%
haberman	73,53%	73,53%	71,90%	70,24%	70,24%	72,20%	72,86%	70,94%	69,27%	69,27%	72,91%	72,22%	73,87%	74,20%	74,20%
heart	80,23%	74,94%	77,95%	77,90%	80,88%	81,55%	80,26%	82,47%	82,22%	83,52%	82,87%	81,53%	82,52%	80,86%	82,49%
housevotes	96,56%	94,82%	96,12%	96,56%	96,56%	96,96%	96,99%	96,56%	96,56%	96,56%	92,23%	91,85%	92,26%	91,85%	91,85%
iris	94,00%	94,00%	93,33%	93,33%	93,33%	94,00%	94,00%	86,67%	93,33%	93,33%	94,00%	94,00%	94,00%	92,67%	93,33%
led7digit	71,40%	68,60%	68,40%	70,40%	70,80%	70,80%	69,60%	68,80%	70,80%	71,00%	73,40%	74,00%	72,00%	71,80%	72,20%
lymph	76,33%	75,10%	74,29%	75,05%	75,05%	76,24%	76,43%	77,86%	75,76%	77,24%	80,52%	76,43%	79,81%	81,86%	81,86%
mammographic	83,73%	83,61%	82,29%	84,10%	84,10%	83,86%	83,61%	82,89%	84,22%	83,49%	83,37%	82,29%	82,29%	83,61%	83,13%
movement	55,83%	58,89%	51,11%	55,00%	59,17%	52,44%	50,28%	50,00%	49,17%	52,78%	61,39%	53,89%	58,89%	65,28%	62,78%
page-blocks	96,42%	96,56%	96,36%	96,77%	96,91%	96,34%	96,34%	96,29%	96,24%	96,34%	96,31%	96,27%	96,05%	96,31%	96,40%
phoneme	81,11%	80,51%	80,66%	81,20%	81,25%	80,90%	80,05%	80,48%	81,03%	81,18%	82,14%	81,61%	81,53%	82,11%	82,20%
pima	74,87%	73,54%	74,33%	73,16%	74,20%	76,05%	73,80%	73,81%	73,16%	74,33%	74,57%	74,19%	74,34%	73,02%	74,84%
ring	82,45%	80,91%	80,97%	81,16%	83,32%	92,69%	92,96%	91,64%	92,74%	93,19%	62,72%	60,47%	60,47%	62,32%	60,49%
satimage	84,38%	84,34%	84,55%	84,24%	85,10%	83,74%	84,48%	83,71%	83,73%	85,00%	88,92%	88,81%	89,20%	89,45%	89,73%
segment	94,20%	93,46%	92,03%	93,72%	94,20%	93,03%	90,35%	90,87%	90,26%	91,82%	92,99%	92,08%	92,12%	93,42%	93,07%
sonar	73,17%	71,74%	72,71%	72,69%	73,67%	76,00%	70,81%	72,71%	71,29%	76,26%	75,02%	77,00%	74,14%	75,57%	77,50%
spambase	92,81%	92,41%	92,11%	92,72%	92,76%	92,26%	91,87%	91,87%	92,05%	92,37%	93,02%	92,65%	93,22%	93,18%	93,41%
spectheart	72,50%	66,25%	71,25%	68,75%	68,75%	68,75%	72,50%	70,00%	70,00%	71,25%	70,00%	67,50%	70,00%	68,75%	68,75%
texture	87,05%	87,85%	87,05%	87,56%	88,89%	86,89%	86,42%	86,45%	87,24%	89,16%	95,91%	95,69%	95,84%	96,09%	96,31%
thyroid	99,25%	99,08%	99,25%	99,22%	99,25%	99,17%	99,07%	99,07%	99,17%	99,18%	98,69%	98,50%	98,54%	98,63%	98,78%
tic-tac-toe	83,51%	84,34%	85,90%	85,70%	88,93%	98,02%	97,49%	97,60%	97,70%	97,81%	98,64%	93,73%	97,29%	98,85%	98,43%
titanic	77,60%	77,46%	77,87%	77,51%	77,92%	77,60%	77,46%	77,96%	77,92%	77,92%	77,60%	77,65%	77,96%	77,19%	78,01%
twonorm	80,11%	80,04%	80,19%	80,22%	82,82%	84,89%	83,65%	84,18%	83,95%	86,07%	94,11%	94,03%	93,91%	93,84%	95,03%
vehicle	70,34%	69,25%	69,40%	68,45%	70,68%	64,88%	57,68%	60,88%	60,05%	60,88%	68,20%	67,60%	68,08%	70,09%	69,38%
vowel	98,08%	97,77%	97,98%	98,28%	98,18%	97,98%	98,28%	97,87%	98,18%	98,18%	97,57%	96,36%	97,67%	97,47%	97,37%
wisconsin	94,99%	94,28%	94,85%	94,99%	94,99%	95,99%	95,56%	94,70%	95,27%	95,27%	97,42%	96,42%	96,99%	96,70%	96,70%
wine	90,39%	88,79%	88,24%	88,79%	90,49%	91,57%	88,20%	85,39%	90,36%	88,73%	94,87%	94,97%	95,52%	95,52%	95,52%
yeast	75,35%	74,66%	75,20%	75,27%	75,60%	76,08%	73,91%	75,34%	74,93%	76,27%	76,08%	73,85%	75,40%	75,34%	75,40%
zoo	95,00%	90,09%	91,09%	93,00%	92,00%	87,09%	87,09%	85,09%	87,09%	87,09%	93,01%	90,18%	92,09%	92,09%	93,09%

Table 7: Classification accuracy (labeled ratio 40%).

	10%			20%			30%			40%		
SSL Algorithm	C4.5	JRip	kNN	C4.5	JRip	kNN	C4.5	JRip	kNN	C4.5	JRip	kNN
Self-Train	11	9	8	9	6	7	1	5	4	0	5	1
Co-Train	4	5	2	2	6	4	3	5	4	3	3	2
Tri-Train	4	3	8	2	4	7	7	5	**13**	3	4	8
Supervised	4	4	0	4	5	2	4	5	4	7	8	4
EnSSL	**14**	**18**	**11**	**20**	**14**	**15**	**21**	**16**	9	**19**	**16**	**17**

Table 8: Total wins of each SSL algorithm.

The statistical comparison of multiple algorithms over multiple data sets is fundamental in machine learning and usually it is typically carried out by means of a nonparametric statistical test. Therefore, the Friedman Aligned-Ranks (FAR) test [8] is utilized in order to conduct a complete performance comparison between all algorithms for all the different labeled ratios. Its application will allow us to highlight the existence of significant differences between the proposed algorithm and the classical SSL algorithms and evaluate the rejection of the hypothesis that all the classifiers perform equally well for a given level. Notice that FAR test is considered to be one of the most well-known tools for multiple statistical comparison tests when comparing more than two methods [10]. Furthermore, the Finner test is applied as a post hoc procedure to find out which algorithms present significant differences.

Ratio	Classifier (C4.5)	Friedman Ranking	Finner post-hoc test p-value	Null Hypothesis
10%	EnSSL	58.4375		
	Self-training	76.625	0.049750	rejected
	Tri-training	94.7875	0.037739	rejected
	Co-training	128.225	0.025321	rejected
	Supervised	144.425	0.012741	rejected
20%	EnSSL	56.6		
	Self-training	83.8	0.045583	rejected
	Tri-training	103.85	0.037739	rejected
	Supervised	115.4875	0.025321	rejected
	Co-training	142.7625	0.012741	rejected
30%	EnSSL	57.575		
	Tri-training	93.5375	0.044582	rejected
	Supervised	108.85	0.037739	rejected
	Self-training	109.2625	0.025321	rejected
	Co-training	133.275	0.012741	rejected

(continued).

Ratio	Classifier (C4.5)	Friedman Ranking	Finner post-hoc test p-value	Null Hypothesis
40%	EnSSL	58.475		
	Supervised	77.45	0.142611	accepted
	Tri-training	106.9625	0.000239	rejected
	Co-training	116.2	0.000016	rejected
	Self-training	143.4125	0.000000	rejected

Table 9: FAR test and Finner post hoc test (C4.5).

Tables 9, 10 and 11 present the information of the statistical analysis performed by nonparametric multiple comparison procedures for each base learner. The best(lowest) ranking obtained in each FAR test determines the control algorithm for the post hoc test. Moreover, the adjusted p-value with Finner's test (Finner APV) is presented based on the control algorithm, at $\alpha = 0.05$ level of significance. Clearly, the proposed algorithm exhibits the best overall performance, outperforming the rest SSL algorithms, since it reports the highest probability-based ranking, presenting statistically better results, relative to all labeled ratio.

6 Conclusions & future research

In this work, a new ensemble semi-supervised algorithm is proposed based on a voting methodology. The proposed algorithm combines the individual predictions of three SSL algorithms: Co-training, Self-training and Tri-training via a maximum-probability voting scheme. The numerical experiments and the presented statistical analysis indicate that the proposed algorithm EnSSL outperforms its component SSL algorithms, confirming its efficacy.

An interesting direction for future work is the development of a parallel implementation of the the proposed algorithm. Notice that the implementation of each one of its component based learners in parallel machines constitutes a significant aspect to be studied, since a huge amount of

Ratio	Classifier (JRip)	Friedman Ranking	Finner post-hoc test p-value	Null Hypothesis
10%	EnSSL	62.2625		
	Self-training	81.5375	0.136404	accepted
	Tri-training	100.2625	0.004429	rejected
	Co-training	121.0125	0.136404	rejected
	Supervised	137.425	0.000000	rejected
20%	EnSSL	69.25		
	Self-training	95.225	0.044749	rejected
	Tri-training	102.35	0.014031	rejected
	Supervised	116.7	0.000492	rejected
	Co-training	118.975	0.000488	rejected
30%	EnSSL	66.225		
	Supervised	99.9625	0.009140	rejected
	Tri-training	104.175	0.004484	rejected
	Self-training	109.25	0.001771	rejected
	Co-training	122.8875	0.000048	rejected
40%	EnSSL	64.925		
	Supervised	76.1	0.387887	accepted
	Tri-training	107.875	0.001206	rejected
	Co-training	121.175	0.000028	rejected
	Self-training	132.425	0.000001	rejected

Table 10: FAR test and Finner post hoc test (JRip).

Ratio	Classifier (kNN)	Friedman Ranking	Finner post-hoc test p-value	Null Hypothesis
10%	EnSSL	59.65		
	Tri-training	73.825	0.273404	accepted
	Self-training	89.3375	0.028959	rejected
	Co-training	129.8375	0.00000	rejected
	Supervised	149.85	0.00000	accepted
20%	EnSSL	59.5125		
	Tri-training	79.1625	0.128941	accepted
	Self-training	103.55	0.00089	rejected
	Co-training	130.075	0.00000	rejected
	Supervised	130.2	0.00000	accepted
30%	EnSSL	70.9625		
	Tri-training	86.9875	0.045642	rejected
	Supervised	101.175	0.026013	rejected
	Self-training	117.8625	0.000581	rejected
	Co-training	125.5125	0.0001	rejected
40%	EnSSL	61.9875		
	Supervised	74.3375	0.33996	accepted
	Tri-training	92.2625	0.02568	rejected
	Co-training	124.225	0.000003	rejected
	Self-training	149.6875	0.000000	rejected

Table 11: FAR test and Finner post hoc test (kNN).

data can be processed in significantly less computational time. Since the experimental results are quite encouraging, a next step could be the evaluation of the proposed algorithm in specific scientific fields applying real world datasets, such as the educational, health care, etc.

References

[1] David W. Aha. *Lazy Learning*. Dordrecht: Kluwer Academic Publishers, 1997. https://doi.org/10.1007/978-94-017-2053-3

[2] Jesús Alcalá-Fdez, Alberto Fernández, Julián Luengo, Joaquín Derrac, Salvador García, Luciano Sánchez, and Francisco Herrera. Keel data-mining software tool: data set repository, integration of algorithms and experimental analysis framework. *Journal of Multiple-Valued Logic & Soft Computing*, 17, 2011. https://doi.org/10.1109/nwesp.2011.6088224

[3] Ethem Alpaydin. *Introduction to Machine Learning*. MIT Press, Cambridge, 2nd edition, 2010. https://doi.org/10.1017/s0269888906220745

[4] Avrim Blum and Tom Mitchell. Combining labeled and unlabeled data with co-training. In *11th annual conference on Computational learning theory*, pages 92–100. ACM, 1998. https://doi.org/10.1109/icdm.2001.989574

[5] William W. Cohen. Fast effective rule induction. In *International Conference on Machine Learning*, pages 115–123, 1995. https://doi.org/10.1016/b978-1-55860-377-6.50023-2

[6] Bozidara Cvetkovic, Boštjan Kaluza, Mitja Luštrek, and Matjaz Gams. Semi-supervised learning for adaptation of human activity recognition classifier to the user. In *Proceddings of International Joint Conference on Artificial Intelligence*, pages 24–29, 2011.

[7] Asif Ekbal and Sivaji Bandyopadhyay. Named entity recognition using appropriate unlabeled data, post-processing and voting. *Informatica*, 34(1), 2010.

[8] Helmut Finner. On a monotonicity problem in step-down multiple test procedures. *Journal of the American Statistical Association*, 88(423):920–923, 1993. https://doi.org/10.2307/2290782

[9] Matjaž Gams. *Weak intelligence: through the principle and paradox of multiple knowledge*. Nova Science, 2001.

[10] Salvador García, Alberto Fernández, Julián Luengo, and Francisco Herrera. Advanced nonparametric tests for multiple comparisons in the design of experiments in computational intelligence and data mining: Experimental analysis of power. *Information Sciences*, 180(10):2044–2064, 2010. https://doi.org/10.1016/j.ins.2009.12.010

[11] Hristijan Gjoreski, Boštjan Kaluža, Matjaž Gams, Radoje Milić, and Mitja Luštrek. Context-based ensemble method for human energy expenditure estimation. *Applied Soft Computing*, 37:960–970, 2015. https://doi.org/10.1016/j.asoc.2015.05.001

[12] Tao Guo and Guiyang Li. Improved tri-training with unlabeled data. *Software Engineering and Knowledge Engineering: Theory and Practice*, pages 139–147, 2012. https://doi.org/10.1007/978-3-642-25349-2_19

[13] Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H. Witten. The WEKA data mining software: An update. *SIGKDD Explorations Newsletters*, 11:10–18, 2009. https://doi.org/10.1145/1656274.1656278

[14] Kyaw Kyaw Htike. Hidden-layer ensemble fusion of MLP neural networks for pedestrian detection. *Informatica*, 41(1), 2017.

[15] Ludmila I. Kuncheva. *Combining Pattern Classifiers: Methods and Algorithms*. McGraw Hill, John Wiley & Sons, Inc., second edition, 2014. https://doi.org/10.1002/9781118914564

[16] Jurica Levatić, Sašo Džeroski, Fran Supek, and Tomislav Šmuc. Semi-supervised learning for quantitative structure-activity modeling. *Informatica*, 37(2), 2013.

[17] Ming Li and Zhi-Hua Zhou. Improve computer-aided diagnosis with machine learning techniques using undiagnosed samples. *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, 37(6):1088–1098, 2007. https://doi.org/10.1109/tsmca.2007.904745

[18] Chang Liu and Pong C. Yuen. A boosted co-training algorithm for human action recognition. *IEEE transactions on circuits and systems for video technology*, 21(9):1203–1213, 2011. https://doi.org/10.1109/tcsvt.2011.2130270

[19] Ioannis E. Livieris, Ioannis Dimopoulos, Thedore Kotsilieris, and Panagiotis Pintelas. Predicting length of stay in hospitalized patients using ssl algorithms. In *ACM 8th International Conference on Software Development and Technologies for Enhancing Accessibility and Fighting Infoexclusion*, pages 1–8, 2018. https://doi.org/10.1145/3218585.3218588

[20] Ioannis E. Livieris, Konstantina Drakopoulou, Vassilis Tampakas, Tassos Mikropoulos, and Panagiotis Pintelas. Predicting secondary school students' performance utilizing a semi-supervised learning approach. *Journal of Educational Computing Research*, 2018. https://doi.org/10.1177/0735633117752614

[21] Ioannis E. Livieris, Konstantina Drakopoulou, Vassilis Tampakas, Tassos Mikropoulos, and Panagiotis Pintelas. *Research on e-Learning and ICT in Education*, chapter An ensemble-based semi-supervised approach for predicting students' performance, page 25-42. Springer, 2018. https://doi.org/10.1007/978-3-319-95059-4_2

[22] Ioannis E. Livieris, Andreas Kanavos, Vassilis Tampakas, and Panagiotis Pintelas. An ensemble SSL algorithm for efficient chest x-ray image classification. *Journal of Imaging*, 4(7), 2018. https://doi.org/10.3390/jimaging4070095

[23] Ioannis E. Livieris, Tassos Mikropoulos, and Panagiotis Pintelas. A decision support system for predicting students' performance. *Themes in Science and Technology Education*, 9:43–57, 2016.

[24] Christopher J. Merz. Using correspondence analysis to combine classifiers. *Machine Learning*, 36:33–58, 1999. https://doi.org/10.1023/A:1007559205422

[25] Vincent Ng and Claire Cardie. Weakly supervised natural language learning without redundant views. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1*, pages 94–101. Association for Computational Linguistics, 2003. https://doi.org/10.3115/1073445.1073468

[26] J. Ross Quinlan. *C4.5: Programs for machine learning*. Morgan Kaufmann, San Francisco, 1993. https://doi.org/10.1007/BF00993309

[27] Matteo Re and Giorgio Valentini. *Advances in Machine Learning and Data Mining for Astronomy*, chapter Ensemble methods: A review, pages 563–594. Chapman & Hall, 2012. https://doi.org/10.1201/b11822-34

[28] Lior Rokach. *Pattern Classification Using Ensemble Methods*. World Scientific Publishing Company, 2010. https://doi.org/10.1142/7238

[29] Moumita Roy, Susmita Ghosh, Ashish Ghosh. A novel approach for change detection of remotely sensed images using semi-supervised multiple classifier system. *Information Sciences*, 269:35–47, 2014. https://doi.org/10.1016/j.ins.2014.01.037

[30] S.K. Satapathy, A.K. Jagadev, and S. Dehuri. An empirical analysis of different machine learning techniques for classification of EEG signal to detect epileptic seizure. *Informatica*, 41(1), 2017.

[31] Sandeep Kumar Satapathy, Alok Kumar Jagadev, and Satchidananda Dehuri. Weighted majority voting based ensemble of classifiers using different machine learning techniques for classification of EEG signal to detect epileptic seizure. *Informatica*, 41(1):99, 2017.

[32] Gasper Slapničar, Mitja Luštrek, and Matej Marinko. Continuous blood pressure estimation from PPG signal. *Informatica*, 42(1), 2018.

[33] Shiliang Sun and Feng Jin. Robust co-training. *International Journal of Pattern Recognition and Artificial Intelligence*, 25(07):1113–1126, 2011. https://doi.org/10.1142/s0218001411008981

[34] Shiliang Sun and Qingjiu Zhang. Multiple-view multiple-learner semi-supervised learning. *Neural processing letters*, 34(3):229, 2011. https://doi.org/10.1007/s11063-011-9195-8

[35] Isaac Triguero, Salvador García, and Francisco Herrera. SEG-SSC: A framework based on synthetic examples generation for self-labeled semi-supervised classification. *IEEE Transactions on Cybernetics*, 45:622–634, 2014. https://doi.org/10.1109/tcyb.2014.2332003

[36] Isaac Triguero, Salvador García, and Francisco Herrera. Self-labeled techniques for semi-supervised learning: taxonomy, software and empirical study. *Knowledge and Information Systems*, 42(2):245–284, 2015. https://doi.org/10.1007/s10115-013-0706-y

[37] Isaac Triguero, José A. Sáez, Julián Luengo, Salvador García, and Francisco Herrera. On the characterization of noise filters for self-training semi-supervised in nearest neighbor classification. *Neurocomputing*, 132:30–41, 2014. https://doi.org/10.1016/j.neucom.2013.05.055

[38] Julius Venskus, Povilas Treigys, Jolita Bernatavičienė, Viktor Medvedev, Miroslav Voznak, Mindaugas Kurmis, and Violeta Bulbenkienė. Integration of a self-organizing map and a virtual pheromone for real-time abnormal movement detection in marine traffic. *Informatica*, 28(2):359–374, 2017.

[39] Xindong Wu, Vipin Kumar, J. Ross Quinlan, Joydeep Ghosh, Qiang Yang, Hiroshi Motoda, Geoffrey J. McLachlan, Angus Ng, Bing Liu, and Philip S. Yu, Zhi-Hua Zhou, Michael Steinbach, David J. Hand, and Dan Steinberg Top 10 algorithms in data mining. *Knowledge and information systems*, 14(1):1–37, 2008. https://doi.org/10.1201/9781420089653

[40] Qian Xu, Derek Hao Hu, Hong Xue, Weichuan Yu, and Qiang Yang. Semi-supervised protein subcellular localization. *BMC bioinformatics*, 10(1):S47, 2009. https://doi.org/10.1186/1471-2105-10-s1-s47

[41] David Yarowsky. Unsupervised word sense disambiguation rivaling supervised methods. In *Proceedings of the 33rd annual meeting of the association for computational linguistics*, pages 189–196, 1995. https://doi.org/10.3115/981658.981684

[42] Yan Zhou and Sally Goldman. Democratic co-learning. In *16th IEEE International Conference on Tools with Artificial Intelligence (ICTAI)*, pages 594–602. IEEE, 2004. https://doi.org/10.1109/ictai.2004.48

[43] Zhi-Hua Zhou. When semi-supervised learning meets ensemble learning. In *Frontiers of Electrical and Electronic Engineering in China*, volume 6, pages 6–16. Springer, 2011. https://doi.org/10.1007/s11460-011-0126-2

[44] Zhi-Hua Zhou and Ming Li. Tri-training: Exploiting unlabeled data using three classifiers. *IEEE Transactions on Knowledge and Data Engineering*, 17(11):1529–1541, 2005. https://doi.org/10.1109/tkde.2005.186

[45] Xiaojin Zhu. Semi-supervised learning. In *Encyclopedia of Machine Learning*, pages 892–897. Springer, 2011.

[46] Xiaojin Zhu and Andrew B. Goldberg Introduction to semi-supervised learning. *Synthesis lectures on artificial intelligence and machine learning*, 3(1):1–130, 2009. https://doi.org/10.2200/s00196ed1v01y200906aim006

https://doi.org/10.31449/inf.v43i2.2132 Informatica **43** (2019) 235–241 **235**

New Re-Ranking Approach in Merging Search Results

Vo Trung Hung
University of Technology and Education - The University of Danang
48 Cao Thang, Danang, Vietnam
E-mail: vthung@ute.udn.vn, http://www.udn.vn/english

Keywords: search engine, algorithm, merging, re-ranking, search result list

Received: January 3, 2018

When merging query results from various information sources or from different search engines, popular methods based on available documents scores or on order ranks in returned lists, its can ensure fast response, but results are often inconsistent. Another approach is downloading contents of top documents for re-indexing and re-ranking to create final ranked result list. This method guarantees better quality but is resource-consuming. In this paper, we compare two methods of merging search results: a) applying formulas to re-evaluate document based on different combinations of returned order ranks, documents titles and snippets; b) Top-Down Re-ranking algorithm (TDR) gradually downloads, calculates scores and adds top documents from each source into the final list. We propose also a new way to re-rank search results based on genetic programming and re-ranking learning. Experimental result shows that the proposed method is better than traditional methods in terms of both quality and time.

Povzetek: V prispevkih sta primerjana dva pristopa pri združevanju zadetkov iskanja: z enačbo in z algoritmom TDR, nato pa je primerjana še izvirna metoda.

1 Introduction

In the Internet, search engines like Google, Bing, Yahoo provide a convenient mechanism for users to search and exploit information on the Web. According to statistics of "Surface Web" in 2017[1], it shows that Google indexes about 50 billion web pages, Bing about 5 billion pages.

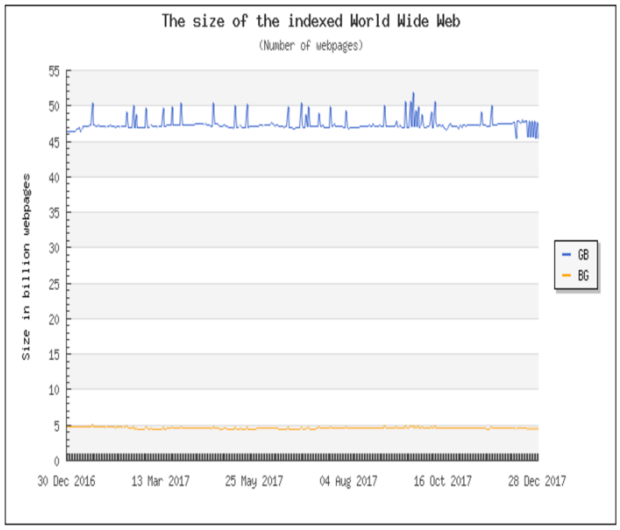


Figure 1: Size of the indexed webpages.

The "Surface Web" is only about 1% of the "Deep Web" - which is not indexed by popular search engines. Many websites do not allow search engines to crawl, instead offering themselves a separate query system such as PubMed or the US Census Bureau.

However, when searching on search engines such as Google, Yahoo or Bing users are not satisfied for two reasons. Firstly, each search engine has different corpus, searching and ranking methods so the returned results will be different. Secondly, search engines now perform monolingual searches (search only on the corresponding language for search keywords), so users can not find webpages in other languages.

To help users exploit the information effectively, there are some tools that combine search results from various sources. We can improve search results based on the available search engines by building a Meta Search Engines [1]. The nature of Meta Search Engines is to use techniques to exploit existing search engines and to process the results obtained from these search engines to generate a new search result that better matches user requirements. A Meta Search Engine needs to handle a variety of issues such as query processing, search on available search engines, processing returned results, re-ranking results found, and display results for users. In this study, we focused solely on re-ranking the results found by the search engines available.

There are two approaches to solve the problem. The first is to mix the search results (duplicate documents) of different search engines on the same information space. This method is often applied to "Surface Web". The second is to combine search results from independent sources (Federated Information Retrieval - FIR) [2], more in line with the exploitation of "Deep Web" information.

The research and development of a combination of search results from multiple sources focused on three main

[1] http://www.worldwidewebsize.com

issues: server description, server selection, and merging [1]. Server description is intended to estimate general information about the original search server such as the number of documents, terms; Frequency of search results returned, ... Server selection is made based on the server description information to determine the most suitable server to send the query. Mixed results are the main work of combining search results from multiple sources, evaluating, rearranging documents, creating final list of results returned to the user.

Merging techniques can be distinguished based on the types of information used for evaluating, re-ranking search results from sources [3]: server information search (total number of documents, results returned); Statistical information: the rank order of the document, the rating provided by the originator; basic information (title, abstract); or the content of the document itself. Research is aimed at improving the evaluation criteria such as accuracy, recall, data usage savings, response speed and bandwidth usage.

The innovation in this paper is using machine learning techniques and basic information returned from the original search engine for re-ranking. We propose solution of sequential mixing to balance the speed and quality of the results.

The rest of this paper is organized as follows. In the Session 2, we present an overview of re-ranking and focus on previous efforts on techniques of re-ranking as well as our analysis and remarks on pervious methods. Details of our proposal in using genetic programming for the re-ranking are presented in Section 3 and the experiment is presented in Session 4. We conclude important points in Section 5.

2 Overview on re-ranking

2.1 Ranking and re-ranking

In the information query, the ranking is usually done by calculating the score of fit between the document and the query, serving the goal of creating a list of documents in decreasing order of the score (shows the degree of suitability for user requirements).

After executing the initial query and receiving the results from a search engine, the data can be extracted including the query content itself, the text list, the ranking points corresponding to the text (some may be hidden from the user), some basic content for each text, such as title, abstract. On an interactive system, the search is performed repeatedly, and the system can store and analyse the contents of executed queries, found documents, read texts, declarations or manipulations by users. The above information may be exploited by the system to re-rank the result list in a variety of ways, distinguished by the type of data used as using the information of the available search engines, rating, or considering to user information.

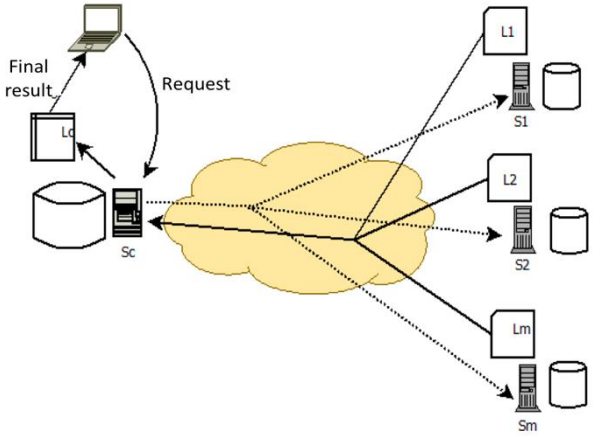


Figure 2: Mix model for search results.

Merging search results from multiple sources has the following process (Figure 2): The central server S_c receives the query from the user, sends the query to search servers from S_1 to S_m. From each S_i server, the list of L_i contains N best results created and returned to the central server. S_c re-evaluates the documents based on the content returned from the original search servers or the content themselves to create the final result list returned to the user.

2.2 Techniques of re-ranking

2.2.1 Combination available rating

The simplest method to merge ranking results is Raw-Score, which directly uses the rankings in each of the original search result listings [4]. The CombSUM method proposed by Fox and Shaw, takes the total score of the document in the various search engines to determine the CombSUM score for a document.

$$CombSUM = \sum_{i \in IR\ Servers} score_i$$

with IR Servers as the set of search engines, $score_i$ is the point of the document assigned by the i^{th} search engine.

The score assigned by a search engine can be normalized to a NormalizedScore score to avoid differences in searcher norms:

$$NormalizedScore = \frac{score - MinScore}{MaxScore - MinScore}$$

with *MinScore* and *MaxScore* being the smallest and largest values in the score of all documents assigned by the search engine.

The weakness of this method is the difference of search engines quality on ranking quality, scoring, presentation methods, ... To overcome the limitation, we can add a weighting for search engines. The WeightedCombSUM score for a document is calculated by the formula:

$$WeightedCombSUM = \sum_{i \in IR\ Servers} w_i \times NormalizedScore_i$$

Here, w_i is the weight assigned to the search engine i in the set of search engines IR Servers; $NormalizedScore_i$ is the normalization of being assigned by server i to the document as in the formula of *NormalizedScore.*

Similarly, some studies [5] suggest a linear function combining the ratings of search engines of the form:

$$M(d,q) = \sum_{i=1}^{n} \beta_i \times s_i(d,q)$$

Here $M(d, q)$ is the final ranking point, $s_i(d, q)$ is the ranking (normalized) of the search engine i, β_i is the weight assigned to the search engine i. The limitation of these methods lies in the need to identify values β_i by manual methods or based on observation of training data.

2.2.2 Ranking order information

The second solutions group uses ranking order information in the original search list. The Round Robin method [6] is the simplest method of mixing, which is performed as follows: We have the result list which is returned from L_1, L_2, ..., L_m; Firstly, we get the m first result as R_1 from the list of L_i, then take the m second result is R_2 from the list of L_i and so on. The final result of the mixing process is in the form of L_{1R1}, ..., L_{mR1}, L_{1R2}, ..., L_{mR2}, ... This is the right solution to ensure search speed when the source of quality information equivalent.

Borda mixing method [7] uses expert judgment scores. Each expert ranked a number of *c* documents. For each expert, the top document is *c*, the second document is *c-1* and so on. If there are some unrated documents, the remainder is divided equally among all unrated papers. Finally, the materials are ranked according to the total number of points assigned. Blending methods use useful ranking information in the absence of information about the search engine rankings. However, studies show that this method of mixing is not as effective as the combination of scores.

The LMS method (using result Length to calculate Merging Score) introduces the original search server counting formula based on the number of returned documents, then identifies new points for documents by multiplying the server point by original point [8].

2.2.3 Ranking learning

In a local search system, documents can be indexed in a variety of ways such as VSM, LSI, LMIR, ... The score of a document versus a query in different ways can be considered as different attributes of the document. Current information query systems tend to apply machine learning techniques to model or create ranking formulas based on these attributes.

The learning process consists of two steps: training and testing. The training input is *D* consisting of the set *{<q, d, r>}*, where *q* is the query, *d* is the document represented by the list of attributes $\{f_1, f_2, ..., f_m\}$, *r* is the relevancy of the document *d* versus the query *q*. The training step involves the construction of an *F* rating model, based on a training database that determines the relationship between the attributes of the document and the relevance of the document to the query. At the test step, the ranking model applied to the *T-dataset* is made up of the set $\{<q_{test}, d_{test}, r_{test}>\}$, the $r_{predict}$ value is the d_{test} document relevancy for the q_{test} query. - calculated by the *F-rating* model - will be compared to the r_{test} value for the rating quality of the rating model. Data for training *D* and experimental data *T* are usually generated by editing the search results in practice, and then manually evaluated by experts.

Ranking methods generally have the same approach by optimizing the objective function: find the maximum value of the gain function or find the minimum value of the loss function.

Ranking techniques are divided into three groups: *point-wise*, *pair-wise* and *list-wise* [9]. With a *point-wise* approach, each training object corresponds to an assigned document attached to the rating value. The learning process involves finding a model that maps each object to a rating close to its actual value. The *pair-wise* approach utilizes pairs of documents that are associated with rank order (before or after) as training subjects. In the *list-wise* approach, the training object is itself the list of ranked documents corresponding to the query.

The characteristic of the *point-wise* solution group is *PRank* introduced in [10] using a regression analysis.

In the *pair-wise* group, they constructed the *RankSVM* ranking algorithm with the aim of minimizing bias in the list of sorted pairs. This method is often referred to in studies as a basis for comparison. Freund applies boosting and introduces the *RankBoost* algorithm [11]. The advantage of this approach is that it is easy to deploy and can run in parallel for testing. Another example is *FRank* based on the probability ranking model.

In the *ListNet* method of the *list-wise* group, the document list itself is considered a training subject. The authors use a probability method to calculate the loss function for the list, which is determined by the difference between the expected sorting list and the correct sorting list. Neural network models and gradient descent are used in deployment algorithms to determine the ranking model.

While the presented methods may apply to mixing results from multiple search engines, the ranking learning methods apply to the case of the search system. Kits and documents are indexed in different ways. According to Yu-Ting and colleagues [12], ranking methods with training data (referred to as supervised ranking) were evaluated more effectively than others one (may be considered non-supervisor ranking).

2.2.4 Using user information

By default, traditional web search engines perform keyword-based queries. However, two different users, with different interests, can use same keywords with different search goals. In order to better meet the individual user's search needs, the user's declaration of behaviour and habits of the user during the search operation has become a research object. personalized ranking results or cooperative ratings [13].

Personalization of rankings results in querying and ranking results for users based on individual user interests and is carried out through two processes: (1) The

information that describes the user's interest and (2) the data collection reasoning to predict the content is close to the user's desires.

Initial data collection solutions require the user to disclose the information interest through the registration table, and the user may change this information [14]. The problem with this solution is that the user does not want to, or has difficulty in providing feedback about their search results as well as their concerns. Another direction, more popular, perform "learning", create user profiles through search history to classify, create groups of topics of interest to users with the aim of providing more information for the ranking. Based on the collected data, the authors build a model that describes and exploits relationships between users, queries, and Web pages, and serves search results matching the needs of user. In terms of characteristics, models may be limited to the exploitation of "two-way data" that exploits the user's interest in information topics, or "data in three directions" (three-way data) incorporates more information about the site.

In addition to the user-identified information solution, a number of solutions for exploiting user group information, created through the analysis of the already-searched content of the set User groups have the same characteristics (geographic location, occupation, interest) or have common search habits, such as Collaborative Filtering (CF). Web sites that meet a person's profile will be considered appropriate for others in the same group.

Due to the sparseness of the data sparsity, the latent semantic indexing algorithm is widely used as the primary technique for data modelling to optimize the layout as well as volume calculation [15].

2.3 Remarks

In the re-evaluation methods based on the rating of the original search engines, raw-score is the simplest method, which will compare directly the origin of the documents to the final result list. *CombSUM* is taking the total score of the document in the various search engines to determine the ranking in the final list. This score is standardized to avoid differences in the norms of each search engine, or to supplement the corresponding original server quality parameters in the Weighted *CombSUM*.

The second solutions group uses ranking order information in the original search list. This is the right solution to ensure search speed when the source of quality information equivalent.

The third solutions group uses the basic information (such as headings, excerpts, ...) of the original results in the scoring of documents. It compares the query with the title or footnote of the document, then applies the scoring formula based on ranking factors, title points, point lengths, lengths of title, and excerpts. In the news search system "News MetaSearcher" [16], in addition to the above factors, the time to update the document is also included in the rating formula.

The fourth solutions group performs the loading of the entire contents of the documents present in the original search result listings, then uses the indexing and scoring mechanism at the central server to perform the sorting, re-ordering the materials. It reviews the entire document to ensure a stable end result list, but takes a lot of time and bandwidth to load data from multiple servers.

The methods in the two first groups rely on the statistical information returned from the query (score, rank order) to perform calculations, so ensure a quick response to the final ranking result. However, some of the factors that make the quality of the endorsements are not good: Firstly, the search engines have large differences in data size, ranking algorithms that make the scoring formula based only on statistical information is not really relevant; Second, in reality the search server usually does not provide information about the document review point.

The third solutions group is usually chosen in practice because of its advantages in both speed and search quality compared to the two first groups. The final solution group has a stable ranking quality, but requires a lot of time for downloading the full content of the candidate materials as well as computational time for indexing and re-rating.

From here the requirement for a solution is guaranteed to make the most out of the basic information from the return lists, on the other hand requires the content of the documents in the final list to be consistent with the query and satisfactoriness on time and bandwidth costs.

3 Proposal solution

3.1 Idea

We propose a new solution to re-rank search results in using genetic programming.

Genetic Programming (GP) was first introduced by Angeline [17], based on genetic algorithms. In GP, each potential solution as a function is called an individual in the population set. GPs operate through the loop mechanism: at each generation, the dominant individual selectivity in the population is based on the content of the price; Perform hybrid, mutant, and spawn operations to create better individuals for later generations.

From randomness and irrelevance to the algorithmic principle of individual formation, in many cases genetic programming helps to overcome localized optimization errors. Although there is no assurance that the results identified by genetic programming are optimal, experimentation in different areas indicates that this result is generally better than the application of algorithms defined by the expert, in many cases, this result is close to the optimal solution [17].

An important element in the implementation of genetic programming is the definition of the individual, on the basis of which the content is determined, ensuring that the measurement accurately determines the quality of the solution. In addition, the complexity of the content, the number of individuals in the population, the rate of hybridization and mutation, the number of generations to be tested should be well defined to balance the ability to create a good solution, eliminate solutions that are not suitable for the calculation volume and time to solve the problem.

Previously, the practice of ranking methods was conducted independently, on different sets of data. This does not allow comparison of methods and hinders research. In 2007, Microsoft introduced the LETOR (LEARNING TO Rank) data set for the study of techniques in text search. In version 3.0 [18], the OHSUMED collection is edited from MEDLINE - a database of medical publications - for academic rankings. From the data of 106 queries, three files are created: the trainset contains 63 queries, the validation set contains 21 queries, and the testing set contains 22 queries. Each file contains records in the following format:

<lb> qid:<q> 1:<v1> 2:<v2> . ... 45:<v45>

where *<lb>* is the value of relevance; *<q>* is the query number; *<v1>, ... <v45>* are values that correspond to the features of the documents, which are calculated on the basis of common rankings for search. Some examples of attributes used include:

ID	Formula
1	$\sum_{q_i \in q \cap d} c(q_i, d)$ in the titles
5	$\sum_{q_i \in q \cap d} \log(\frac{C}{df(q_i)})$ in the titles
11	BM25 of the title
14	LMIR.JM of the title
16	$\sum_{q_i \in q \cap d} c(q_i, d)$ in the compendium
26	BM25 of the compendium
28	LMIR.JM of the compendium

Table 1: Example attribute of the OHSUMED collection.

In the above formulas, q_i is the query keyword i[th] in the query q, d is the document, $c(q_i, d)$ is the number of occurrences of q_i in the document d; C is the total number of documents in the corpus, $df(q_i)$ is the number of documents containing the keyword q_i. The BM25 and LMIR.JM scores are documented using the BM25 rating model and the Jelinek - Mercer smoothing language model [19].

3.2 Modelling application of genetic programming

The GP application solution for rating learning is as following model:

- **Input 1**: Training data set D with recording records in the form of the OHSUMED collection;

- **Input 2**: Parameters N_g is the number of generations, N_p is the number of individuals per generation, N_c is the hybrid speed, N_m is the speed of the mutation.

- **Output**: The rank function $F(q, d)$, which sets the value to a real number, corresponds to the relevance of the document d to the query q.

The training process consists of five steps as follows:

- *Step 1*: Randomly identify first generation individuals;

- *Step 2*: Determine the value of the content for each individual;

- *Step 3*: Perform hybrid and mutation operations;

- *Step 4*: Create a new generation and repeat steps from 2 to 4 until you have enough N_g;

- *Step 5*: Choose the best individual result.

Each individual (gene) is defined as a function $f(q, d)$ that measures the relevance of the document to the query, with the following options:

- Option 1: The linear function uses 45 attributes:

$$TF - AF = a_1 \times f_1 + a_2 \times f_2 + \cdots + a_{45} \times f_{45}$$

- Option 2: Linear function, using only a selective random attribute:

$$TF - RF = a_{i1} \times f_{i1} + a_{i2} \times f_{i2} + \cdots + a_{in} \times f_{in}$$

- Option 3: Apply function to attributes. Limit the use of functions x, $1/x$, $sin(x)$, $log(x)$, and $1/(1+e^x)$.

$$TF - FF = a_1 \times h_1(f_1) + a_2 \times h_2(f_2) + \cdots + a_{45} \times h_{45}(f_{45})$$

- Option 4: Create a *TF-GF* function similar to the one presented in [20], but retain the evaluation of non-linear functions. The function is binary tree, with inner vertices being operators, leaf vertices are constants or variables.

In the formulas, a_i are the parameters, f_i are the attribute values of the document, h_i are the function.

In options 1, 2 and 3, to hybridize two individuals $f_1(q, d)$ and $f_2(q, d)$, a random list of parameters has the same index of functions to be exchanged. The mutation operation for the individual, $f(q,d)$, is performed by swapping two random parameters of the function $f(q, d)$.

Comparison of search and ranking solutions is usually based on the measures *P@k, MAP, NDCG@k* [20] that is used to determine the value of the content. Here, we test the fitness function corresponding to the MAP value.

In the first two options, N_g, N_p, N_c, N_m are respectively 100, 100, 0.9, 0.1. For option 3, N_g, N_p are defined as 200,400. In option 4, N_g, N_p, N_c, N_m are respectively 1000, 100, 0.9 and 0.2. These values are determined by experiment. The N_g value, given in alternatives 3 and 4, is greater due to the complexity and diversity of individuals - the ranking function.

4 Experiment

The *TF-Ranking* experimental software, built on the basis of the *PyEvolve* library, was developed by Christian S. Perone[2], which enables the development of a genetic algorithm for development in the Python language.

In the OHSUMED collection, the data is divided into five directories, each containing the *train.txt*, *vali.txt* and *test.txt* files for training, re-evaluation, and experimentation. According to each directory, the training and experiment steps are as follows:

- The training module reads data from *train.txt* for best *pbest* selection, applying the scoring function to the text in *test.txt*.

- Microsoft's Eval-Score-3.0.pl tool is used to generate *P@k, MAP, NDCG@k* values ($k = 1,2,5,100$), evaluating the effect of the generated point function.

For each option, the mean value for each of the *P@k, MAP, NDCG@k* scores of the five directories was taken as the scores for the experimental option. The implementation of training and experiment was done 5

[2] http://pyevolve.sourceforge.net (access on 15/01/2016)

times, the average value for comparison and evaluation of results.

Table 2, Table 3 and Table 4 compare *MAP*, *P@k* and *NDCG@k* (with *k = 1, 2, 5, 10*) of the proposed solution against the baseline method, published in website of the LETOR[3] assessment data set. Bold cells contain the highest values in the corresponding column.

Method	MAP
Regression	0.4220
RankSVM	0.4334
RankBoost	0.4411
ListNet	0.4457
FRank	0.4439
TF-AF	0.4456
TF-RF	0.4467
TF-FF	**0.4468**
TF-GF	0.4427

Table 2: Comparison of MAP values

Method	K=1	K=2	K=5	K=10
Regression	0.4456	0.4532	0.4278	0.4110
RankSVM	0.4958	0.4331	0.4164	0.4140
RankBoost	0.4632	0.4504	0.4494	0.4302
ListNet	0.5326	0.481	0.4432	0.441
FRank	0.5300	0.5008	0.4588	0.4433
TF-AF	0.5506	0.4789	0.4476	0.4348
TF-RF	**0.5545**	0.4835	**0.4633**	0.4404
TF-FF	0.5294	**0.4957**	0.4600	**0.4437**
TF-GF	0.4997	0.4760	0.4507	0.4372

Table 3: Comparison of NDCG@k values

Method	P@1	P@2	P@5	P@10
Regression	0.5965	0.6006	0.5337	0.4666
RankSVM	0.5974	0.5494	0.5319	0.4864
RankBoost	0.5576	0.5481	0.5447	0.4966
ListNet	0.6524	0.6093	0.5502	0.4975
FRank	0.6429	0.6195	0.5638	**0.5016**
TF-AF	**0.6691**	0.6167	0.5499	0.4955
TF-RF	0.6642	0.6020	**0.5653**	0.4954
TF-FF	0.6619	**0.6279**	0.5612	0.4983
TF-GF	0.6220	0.6058	0.5520	0.4969

Table 4: Comparison of P@k values

Experimental results show that the *TF-AF*, *TF-RF* alternatives are good. *MAP*, *NDCG @ k* and *P @ k* values outperformed the corresponding Regression, RankSVM, and RankBoost methods, which were equivalent and slightly better than the ListNet and FRank methods. The TF-GF method was not very good: Despite the good results on the training set, the results on the experimental set were just average, sign of overfitting.

One-time training for 5 directories with *TF-AF*, *TF-TF*, *TF-FF*, and *TF-GF* options takes 150 minutes, 70 minutes, 200 minutes and 10 hours respectively on a dual-CPU computer. Core 3.30 GHz, 4 GB RAM installed Windows 7.

This result shows that the use of linear functions for ranking assures efficiency, both in terms of experimental quality and duration of training.

5 Conclusion

The paper introduces an overview on re-ranking. It evaluates the application of methods of mixing information retrieval results from multiple sources by re-calculating the scores based on the basic information returned from the original search engine and proposing a re-ranking method. sequentially, progressively download the best documents to create the final result list.

The innovation of this proposal is applying the machine learning method in using genetic programming. We experimented proposal solution on the LETOR experimental data set to develop a new ranking system with the objective of evaluating the effectiveness of this learning methodology. Experimental results suggest that the proposed method is better than traditional methods in terms of both quality and time.

Our next research is to integrate this re-ranking tool in multi-language and cross-language search systems. The systems are intended to allow users to find documents in languages other than the language of the search keywords.

Acknowledgement

This research is funded by Funds for Science and Technology Development of the University of Danang under project number B2019-DN06-18.

References

[1] Kurt I. Munson (2000), Internet Search Engines: Understanding Their Design to Improve Information Retrieval, *Journal of Library Metadata*, Volume 2, p.p. 47-60. https://doi.org/10.1300/J141v02n03_04

[2] M. Shokouhi and L. Si (2011), Foundations and Trends® in Information Retrieval, *Federated Search*, Volume 5 (No. 1), p.p. 101-107. https://doi.org/10.1561/1500000010

[3] J. Callan (2002), Distributed information retrieval, *The Information Retrieval Series: Springer*, INRE, Volume 7, p.p. 127-150. https://doi.org/10.1007/0-306-47019-5_5

[4] S. Wu, F. Crestani, Y. Bi (2006), Evaluating Score Normalization Methods in Data Fusion, *Information Retrieval Technology*, Proceedings of 3rd Asia Information Retrieval Symposium, AIRS 2006, Singapore, p.p. 642-648. https://doi.org/10.1007/11880592_57

[5] W. Shengli, B. Yaxin, Z. Xiaoqin (2011), The linear combination data fusion method in information retrieval, *Lecture Notes in Computer Science book series* (LNCS, volume 6861), pp. 219–233. https://doi.org/10.1007/978-3-642-23091-2_20

[3] http://research.microsoft.com/

[6] S. Wu, S. McClean (2005), Data Fusion with Correlation Weights, *Lecture Notes in Computer Science*, Volume 3408/2005, p.p. 275-286. https://doi.org/10.1007/978-3-540-31865-1_20

[7] B. Xu, S. Luo, K. Sun (2012), Towards Multimodal Query in Web Service Search, *19th International Conference on Web Services*, IEEE. https://doi.org/10.1109/icws.2012.42

[8] Y. Rasolofo, F. Abbaci, J. Savoy (2001), Approaches to collection selection and results merging for distributed information retrieval, *CIKM'01 Proceedings of the 10th international conference on Information and knowledge management*, ACM, p.p. 191 - 198. https://doi.org/10.1145/502585.502618

[9] L. Hang (2011), Learning to Rank for Information Retrieval and Natural Language Processing, *Synthesis Lectures on Human Language Technologies, Morgan & Claypool Publishers*, p.p. 1-113. https://doi.org/10.2200/s00348ed1v01y201104hlt012

[10] C. Koby, S. Yoram (2002), Pranking with Ranking, *Advances in Neural Information Processing Systems 14*, Volume 14, p.p. 641-647. https://doi.org/10.7551/mitpress/1120.003.0087

[11] M.R. Yousefi, T.M. Breuel (2012), Gated Boosting: Efficient Classifier Boosting and Combining, *Lecture Notes in Computer Science*, p.p. 262-265. https://doi.org/10.1007/978-3-642-33347-7_28

[12] L. Yu-Ting, L. Tie-Yan, Q. Tao, M. Zhi-Ming, L. Hang (2007), Supervised rank aggregation, *Proceedings of the 16th international conference on World Wide Web - WWW '07*, p.p. 481–490. https://doi.org/10.1145/1242572.1242638

[13] K. Veningston, R. Shanmugalakshmi (2012), Enhancing personalized web search re-ranking algorithm by incorporating user profile, *Third International Conference on Computing, Communication and Networking Technologies (ICCCNT'12)*. https://doi.org/10.1109/icccnt.2012.6396036

[14] P.A. Chirita, W. Nejdl, R. Paiu, C. Kohlschütter (2005), Using ODP metadata to personalize search, *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval - SIGIR '05*, p.p. 178--185. https://doi.org/10.1145/1076034.1076067

[15] T. Nasrin, H. Faili (2016), Automatic Wordnet Development for Low-Resource Languages using Cross-Lingual WSD, *Journal of Artificial Intelligence Research*, Volume 56, p.p. 61–87. https://doi.org/10.1613/jair.4968

[16] Y. Rasolofo, D. Hawking, J. Savoy (2003), Result Merging Strategies for a Current News MetaSearcher, *Information Processing & Management*, No 39(4), p.p. 581–609. https://doi.org/10.1016/s0306-4573(02)00122-x

[17] P.J. Angeline (1994), Genetic programming: On the programming of computers by means of natural selection, *Biosystems, MIT Press Cambridge*, p.p. 69-73. https://doi.org/10.1016/0303-2647(94)90062-0

[18] Q. Tao, L.T. Yan, X. Jun, L. Hang (2010), LETOR: A benchmark collection for research on learning to rank for information retrieval, *Information Retrieval*, Volume 13, No. 4, p.p. 346–374. https://doi.org/10.1007/s10791-009-9123-y

[19] C. Zhai, J. Lafferty (2001), *A study of smoothing methods for language models applied to Ad Hoc information retrieval*, Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval - SIGIR '01, p.p. 334–342. https://doi.org/10.1145/383952.384019

[20] T.G. Lam, T.H. Vo, C.P. Huynh (2015), Building Structured Query in Target Language for Vietnamese – English Cross Language Information Retrieval Systems, *International Journal of Engineering Research & Technology (IJERT)*, Volume 4, No. 04, p.p. 146–151. https://doi.org/10.17577/ijertv4is040317

Physical Match

Aaron E. Naiman, Eliav Farber and Yossi Stein
Department of Applied Mathematics, Jerusalem College of Technology-Machon Lev, Jerusalem, Israel
E-mail: naiman@jct.ac.il

Keywords: physical match, curve matching, pattern recognition, edge pixels, image processing, curve fitting, cyclic, longest common subsequence

Received: September 13, 2017

We present an approach to solving the problem of "physical match," i.e., reconnecting back together broken or ripped pieces of material. Our method involves correlating the jagged edges of the pieces, using a modified version of the longest common subsequence algorithm.

Povzetek: Predstavljena je izvirna metoda sestavljanja razbitih fizičnih predmetov.

1 Introduction

The problem of "physical match" spans many different applications, from whimsical (jigsaw) puzzle solving [9, 19, 20], to 3-D archeology [17, 14, 7, 6] (sometimes utilizing a priori shape knowledge, or rotational symmetry), to the forensic sciences [15, 16]. In the present paper we restrict ourselves to the 2-D problem, and the particular aspects of one-dimensional border which one can take advantage of, for this flat case, but not resorting to the rich information of 3-D surface-to-surface matching. Therefore, given a single piece of material, be it paper, clothing, glass, etc., once it is ripped or broken into many pieces, the challenge is to piece back together the parts, to the original shape.

Sometimes one can take advantage of text [18], color [9], orientation (e.g., lined paper), images [20, 10, 4], specific shapes (e.g., machine-shredded paper [18], or the jigsaw puzzle problem [19]) and/or texture. A survey of such methods can be found in [8]. We are making none of these assumptions, and therefore matching via shape alone.

Some approach this problem with gross polygon approximations of the pieces [1], whereas others solve with a multiscale of resolutions [3]. Our method is based on correlating the changes of direction along the edges of the pieces. In [21], the same information is used, however, whereas they build histograms of these changes of direction, we compare them directly, retaining their *order*, as described below. Finally, [16] present statistical findings (for the plausibility of the *Daubert* ruling in the court of law) for three different types of material. Our analysis is for generic materials, and we concentrate more on the algorithms involved.

2 Background and problem

We have chosen to, at least initially, model the pieces on the computer, rather than work with actually torn material. The reason for this is in order to steer away from possible parameter fixing based on a few physical cases, and instead deriving statistics and parameter values, based on thousands of computer-modeled Monte-Carlo simulations.

In order to accurately match along the edges of the pieces, we start by scanning in the pieces to determine the location of all of the edges. This has to be done with enough resolution to pick up the unique characteristics of each segment of the edge, and minimize the effects of the jagged nature of digitization of the edge pixel positions. On the other hand, the resolution cannot be too high, lest the computations be inordinately expensive.

Once the edges have been located, the edge slopes have to be calculated in such a fashion so that they can be compared to each other. The slope calculations require an *ordering* of the pixels, which is accomplished together with the previous step of locating the edge pixels. This is done for all of the edges of all of the pieces. Since the pieces have been translated and rotated with respect to each other, we actually stored away in arrays, one for each piece, the invariant *changes* in direction along the edge. This we are able to accomplish, as we "travel" along the edges (explained below), recording the rotations.

After the edges have been characterized, the core of our system is to find where the turns of the edges of separate pieces correctly match up. We use a modified version of the longest common subsequence (LCS) algorithm [5], adjusted to be appropriate for problems where the starting points of the sequences are not known.

Finally, the last step is to calculate from the matching turns, where exactly to "sew" the pieces back together. This stage requires further filtering of the matching algorithm, to dispose of spurious matches along the edges.

To summarize, the following algorithms are needed for us to model and verify our physical match system:

1) modeling original, non-trivial pieces of material, and breaking them further into pieces,

2) accurately scanning in the (edges of) the pieces, at the

required resolution, to locate and order the edge pixels,

3) calculating the edge slopes,

4) correlating the edge slope deviations to find the correct matches and

5) sewing the pieces back together.

3 Modeling random-shaped pieces and breaking them

For simplicity's sake, we start with pieces which are basically circular in shape, but we perturb the edge to obtain a polygon. Beginning with a center, we randomly choose a sequence of angles, as well as radii, ranging between two given extrema. These points are then connected with straight lines to obtain the polygon. See an example of this in Figure 1. A less-simple shape would be to connect the same points with splines, as seen in Figure 2. The ramifications of such a shape on the subsequent algorithms, will be the subject of future analysis.

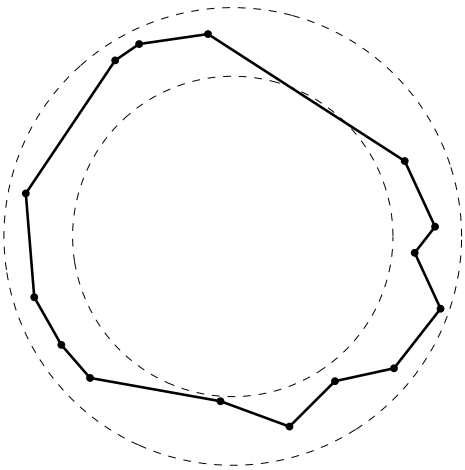


Figure 1: Polygonal edge

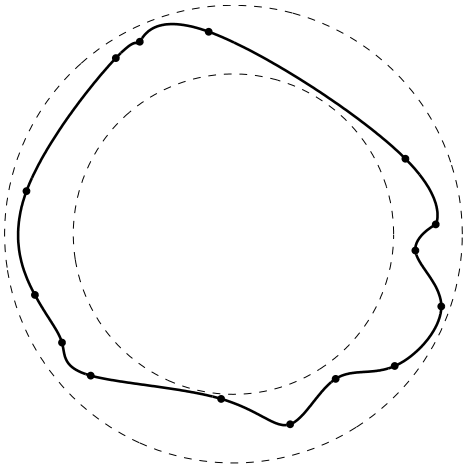


Figure 2: Spline edge

Given polygon, we must now simulate "breaking" it into smaller pieces. Starting at an interior point, we proceed in a straight line, randomizing both the direction and length of the step. This is repeated until a step exits the polygon, as seen in Figure 3. Continuing in the opposite direction, we can create the entire crack, defining our two sub-pieces, as in Figure 4.

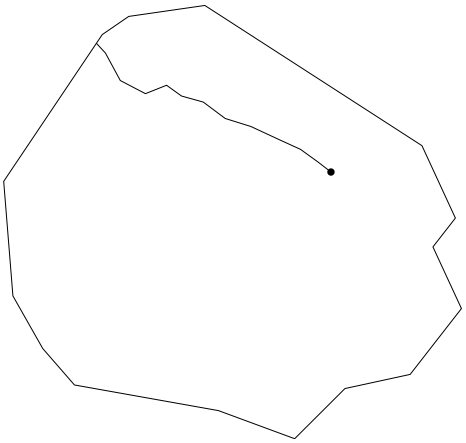


Figure 3: Break exiting polygon

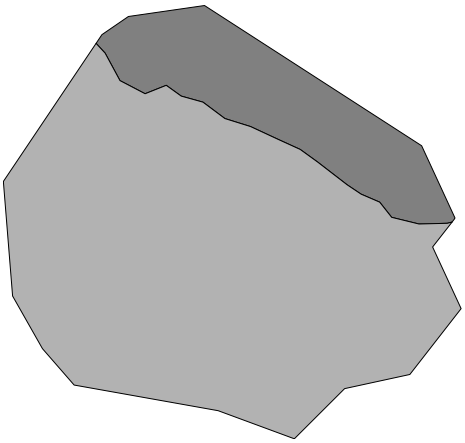


Figure 4: Two pieces

Treating each sub-piece separately, we can now recurse, breaking each piece further and further. Examples of 3 pieces and 100 pieces are shown in Figures 5 and 6.

When determining the values of the extrema, both for the random direction and the random length of the step we take, it is important to generate a break with similar "turn" characteristics to the original polygon edge.

The reason for this is the following. While it is true that, e.g., for a simply-shaped piece of material, the break may actually generate a different kind of edge, we are looking for a worst-case scenario, where we cannot easily tell where the ripped part is, and where the original polygonal edge is. Two counterexamples can be seen in Figures 7 and 8.

We note that calculating when the break "leaves" the piece, in not trivial. We cannot simply check whether the final end of the current "step" segment falls out of the poly-

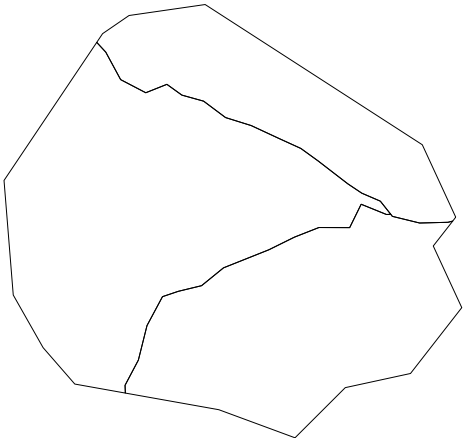


Figure 5: Three pieces

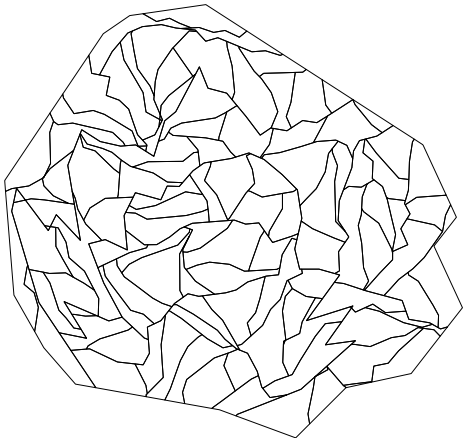


Figure 6: One hundred pieces

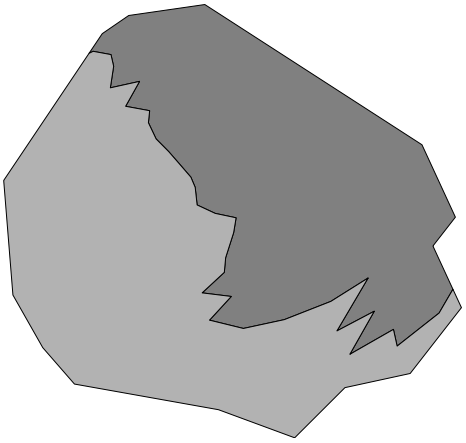


Figure 7: Complex break

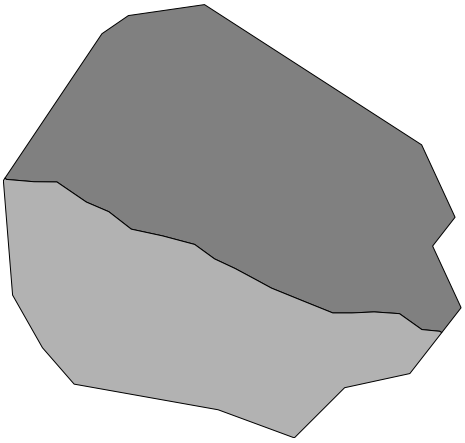


Figure 8: Simpler break

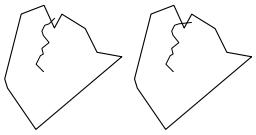


Figure 9: Breaking out of piece

gon, as is the case on the left hand side of Figure 9. This is because the segment may actually leave the polygon, but subsequently return back into it, as in the right hand side of the same figure.

We therefore check at each step of the break, whether the new step segment *intersects* any part of the edge of the polygon. If it does, we set the end of the break at the intersection point.

Finally, future analysis will study possibilities where the break is also generated with splines.

With the pieces now "virtually" created, we are ready to start our analysis. We will now discuss the rasterization of the images, in order to simulate their being scanned "back" into the computer, for running our physical match system.

4 Image scanning and edge pixel ordering

In real life, the entire process starts here, from scanning the pieces into the computer. At this point, a mesh of pixels representing each piece is available, with each pixel signifies where the piece is, or is not. For our analysis, this is a best-case segmentation of the material (not always easily obtainable), into foreground (where the material is) and background (where it is not). The next step is to discover which the *edge* pixels are, and to determine an ordering of them, for subsequent edge slope calculations.

In the model we built in the previous section, the situation is slightly different. There we already have a polygon defined, and we need to determine which of the pixels are edge pixels, as well as a proper ordering for them. To best simulate real life, we first rotate and translate each piece, as well as randomly choose whether or not to flip each piece over.

A standard method for detecting the edge, is found in [2]. The authors report in [13] a new, quick method for determining the edge (and interior) pixels of a polygon, for a large number of pixels. This is important for our problem, since with a high enough resolution of scanning (clearly effected by the smoothness of the curves, and in order to

avoid the need for smoothing and resampling), the number of pixels to check easily enters the tens of millions.

They present two methods for determining which pixels are interior, exterior or on the edge of a polygon. Both methods approach the problem by performing tests along the shape perimeter, differing in memory intensity and generalizability. Along with the complexity analysis, they show that a combination of the two methods, with a crossover from the first algorithm to the second, based on the number of pixels at each step (along a specific edge), works the best.

In addition, their method provides an ordering of the edge pixels, facilitating our subsequent necessary edge slope calculations. In this paper we take advantage of the edge pixel discovery and ordering.

5 Edge slope calculations

Given an ordering of edge pixels, we now need to calculate the edge slope at each pixel. This will subsequently be used for calculating the edge slope differences, needed for matching between pieces.

While an accurate edge slope calculation would be sufficient, it is not necessary. We need an approximation of the edge slope which will be both reproducible, and invariant (to a given degree of tolerance) under rotation and translation. With this characterization, we will be able to match the (changes of) slope of two different pieces, regardless of translation and rotation.

The authors demonstrate in [12] that the method of a linear least squares (LLS) fit to a parabola, accurately calculates this reproducable, rotation- and translation-independent characterization of the edge slope. (LLS fits to other orders of polynomials, were shown to be inferior.) Note that care needs to be taken so that the initial scanning supplies a high enough resolution. This will enable enough points to be supplied to the LLS fit so that no more than one "turn" will be present in each set of points. (The degree to which a "turn" is rendered significant, as well as the number of points per turn, are empirically derived in that paper.)

As they describe, the problem is not trivial, due to the element of rotation-invariance, where in the local neighborhood, a very different set of pixels represents the same edge segment. Nonetheless, after implementing their method, we have for each piece an array of edge slope values, one value for each edge pixel.

6 Matching edges

The matching between pieces is done on a one-to-one basis. That is, each of the pieces is compared with every other piece, in order to determine the best match.

Since the pieces are rotated vis-à-vis each other, there is no reason to find matches between edge slope values. However, the *changes* in edge slope are invariant to rotation. As described above, we have therefore stored away arrays of adjacent *differences* in edge slope values.

The basic method used for matching any two arrays which we have generated, is the longest common subsequence (LCS) algorithm. Since we are comparing real-valued numbers, we are not looking for exact matches, but numbers with are close enough. (In Section 8.2 we discuss values to quantify this closeness.) However, prior to applying the algorithm, we need to state a few geometric considerations.

6.1 Filtering out straight lines

Long stretches of edges might be straight lines (as in our case) or nearly straight lines. Therefore, the edge slope values for many adjacent edge pixels may be nearly equal, and their differences will be zero or close to it.

While these straight edge segments can represent important information, our current algorithm matches based on *changes* in edge slope directions. Therefore, with all of these zeros left in the arrays, too much "straight-edge" information will match, not uncovering the underlying turn-information.

Deleting all such zeros removes too much information, and therefore our current algorithm replaces multiple adjacent zeros with a single zero, as a place marker stating that there was a straight stretch of edge here.

Whereas one might think that quantitative information is lost here, since no *length* of the straight edges is retained, nonetheless cases of unequal length will properly be filtered out later on, when the pieces are attempted to be sewn together (Section 7).

6.2 Relative orientation

If the edge values of one piece are ordered in the exact opposite direction with respect to the other piece, then the changes in edge slope values of one piece are the negative values of the other piece, in addition to being in the opposite order.

For example, the edge on the left hand side of Figure 10 shows moving from one edge pixel to the next one is a 90°

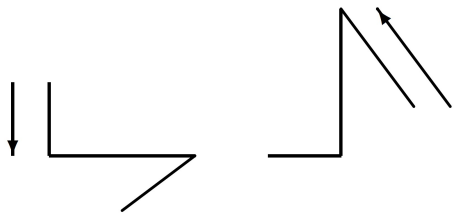


Figure 10: Pixels in reverse direction

turn to the left, followed by a 30° turn to the right—or a -30° turn. If the matching piece (on the right hand side of the figure) is ordered in reverse, this comes to a 30° turn, and then -90° one.

Similarly, if one piece is flipped over as compared to the other piece, then the values will be either in the opposite order $(-30°, 90°)$, as in Figure 11, *or* with reversed signs

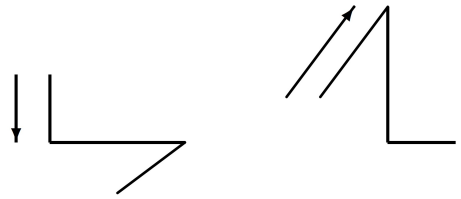


Figure 11: Flipped piece with opposite order

$(90°, -30°)$, as seen in Figure 12.

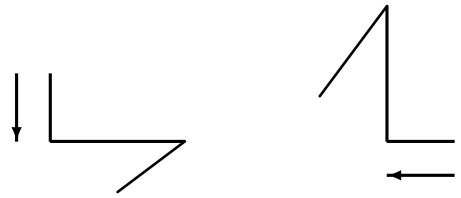


Figure 12: Flipped piece with reversed signs

Therefore, we need to compare the two array in four different fashions:

1) as is,
2) with one of the arrays reversed,
3) with one of the arrays with opposite signs and
4) with one of the arrays reversed and with opposite signs.

6.3 Cyclic correlation

Recall that each array represents the changes of edge slope values for the perimeter of a given piece. Since we randomly choose where along an edge to start the array, we do not know where a matching segment is for any two pieces. It could be that the match is in the middle of array A, but for array B, it is at the end of the array, and finishes subsequently back around at the beginning of the array.

The authors in [11] deal with this issue at length and demonstrate that initially four LCS algorithm invocations are necessary:

1) as is,
2) with one array rotated by 50%,
3) with the other array rotated by 50% and
4) with both arrays rotated by 50%.

(Rotating the array amounts to choosing a different starting point along the piece's edge.) Padding the array with the same content was avoided, in order to avoid possible spurious artifacts.

Note that these four possibilities are in addition to the four permutations mentioned previously in Section 6.2, with reversed arrays and negative values. Therefore, the composite set of possibilities includes a total of 16 possibilities. The best LCS (of the 16) is found, and then if deemed necessary, one or both arrays are reversed, sign-changed and 50% rotated.

In the same paper, the authors show that once the best LCS (so far) is found, and after performing a centering technique they devised, an additional invocation of the LCS algorithm is necessary to extract the LCS. They establish that for cases where the LCS in the two sequences are known to be clustered within the sequences, this final centering and subsequent LCS algorithm step provide optimal LCS results.

Our problem indeed exhibits this clustering behavior, as two pieces generally match along the adjacent side, and not all the way around. (A degenerate case is when one piece sits totally within another.)

With the two LCSs lined up and centered within their sequences, we are ready to proceed with the final stage of "reconnecting" the two pieces back together.

7 Sewing edges

As was noted in the previous section, a final invocation of the LCS algorithm was used in order to extract an LCS as long as possible. However, even with optimal results of obtaining the entire, appropriate LCS, we still need to concern ourselves with spurious, random matches which may enter the LCS.

7.1 Filtering random matches

Consider Figure 13. We see that although we have a good

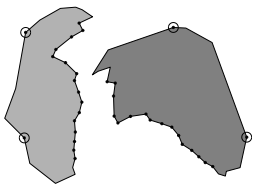


Figure 13: With spurious matches - both sides

match overall, if we are to consider the extra, spurious circled matches which are *outside* of our match, this might seriously degrade our subsequent calculations for reconnecting the pieces.

Note that random matches can occur *within* the LCS spans as well, although they would less effect the "sewing" process (described below), since in general they are geometrically closer to each other.

To dispose of the circled mis-matches, we return to the centering process mentioned in the previous section, regarding the final LCS algorithm invocation. We reuse this center to throw our support behind the 20–30% of the matches which are closest to the center, and discard the remaining matches.

Note as well, that the LCS algorithm only "rewards" for matches, and does not penalize for mismatches. These mismatches are possibly due to missing material along a matching edge, leading to intervening unmatching pairs. If the type of the material and/or situation are such that worn or missing pieces are unlikely, then penalties for mismatches can be considered as well.

7.2 Moving and rotating

We must first move the two pieces next to each other, in order to enable the final stitching. For simplicity's sake, we assume one piece to be stationary, and the other will be matched up to it. Since this second piece is both translated away, and rotated from the first piece, the following maneuver will return its coordinates $[x \;\; y]^T$ to their correct locations:

$$\begin{pmatrix} \cos\alpha & \sin\alpha \\ -\sin\alpha & \cos\alpha \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} \Delta x \\ \Delta y \end{pmatrix}$$

The challenge, therefore, is to find the optimal values of α, Δx and Δy, to juxtapose the two pieces.

Even though we have already established our matching positions within the arrays, along with their (x, y) coordinates, we acknowledge sources of initial measurement errors in the scanning, as well as possible internal mismatches, mentioned above. We therefore use a nonlinear least squares (LSQ) solver (in MATLAB: `lsqnonlin`) to find the triad of values which minimizes the sum of the square distances between our remaining LCS matches. Note: as many nonlinear solvers require, we initially seed the solver with a seat-of-the-pants approximation given the geometry and any three matching pairs. Also, it was crucial to adjust the maximum number of iterations allowed (in MATLAB: `MaxIter`) and the termination tolerance on x (MATLAB: `TolX`) in order to converge properly to the desired results.

7.3 Micro-stitching

With our two pieces lined up next to each other, we have the situation shown in Figure 14. We need to find the entire "seam" in order to properly stitch the two pieces together. A proper stitching is crucial for the success of further matching to more pieces. Note that now we would like to find the most number of pixels which can be considered part of the seam, with the remaining pixels comprising the edge of the combined piece. Therefore, matching between pixels of the two pieces:

1) entails geometric proximity, instead of matching slopes as before, and

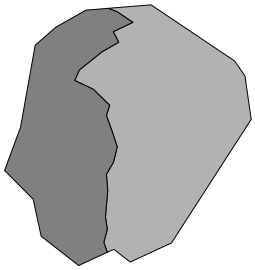


Figure 14: Pre-stitch

2) is done for *all* of the candidate pixels along the seam, and not only the LCS matches.

Starting from the LCS center, the algorithm proceeds in both directions along the seam, by adding the closest pixels (to each other) from the two pieces, as long as their distance is 0.4 of the average mean distance (calculated earlier in the nonlinear least squares procedure). We allow skipping pixels, as long as subsequent ones are part of the seam. When we hit three consecutive pairs which are too far apart, we consider the seam ended (in that direction). Note that the 0.4 value and the count of three consecutive pairs, were empirically derived.

7.4 Goodness of fit

Due to the digitized nature of the problem, we need metrics to measure the quality of the matches. The minimized sum squared error calculated to translate and rotate the pieces gives us one measure, but that was prior to the stitching.

We therefore measure the overlap between the two pieces, as well as gaps which may have been uncovered. See, e.g., in Figure 15, where the dark gray area near the

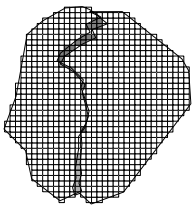


Figure 15: Overlap and gap

top shows us the overlap, and the thin light gray strip towards the bottom—where neither piece covers.

One way to quantify these values is to rescan the entire stitched area of the union of the two pieces. However, this process takes quite a bit of computation time, and is not necessary. Instead, we take a much quicker approach by analyzing the seam only. Starting at one end, we travel along the two edges, calculating where there are crossovers between the two pieces. At these points, we go from overlaps to gaps, or vice versa; we sum each of these separately.

There is one case where an entire scan of the union of the two pieces is helpful—and necessary. On the left hand side

of Figure 16 we see that although our eyes solve the prob-

Figure 16: Reconfigurable pieces

lem easily, moving the two pieces together, another possibility can be seen as well, on the right hand side of the figure. Note that the latter possibility suffers no overlaps or gaps, along the seam.

Clearly this is not the desired result. We therefore run a quick, low-resolution scan of the union of the two pieces, discarding this possibility by comparing the total area before and after stitching. Note that one cannot decide on a consistent ordering of the edge pixels based on concavity/convexity of the pieces, as even along matching edges, part of the shared edge might be concave, and part convex.

One final measure of fit, is the length of the seam. This is needed, as, e.g., if two pieces meet at just one point, then the sum squared error will be zero, there will be no overlap nor gap areas, and the area of the union will be exactly equal to the sum of the individual areas. However, the seam length will be zero too, indicating that no match at all occurred.

The gross overlap demonstrated in Figure 16, and the extremely short length of the seam, are not used statistically with the first two metrics, i.e., the lengths of overlaps and gaps. These last two measures of fit used in a binary fashion to throw away possible matches.

8 Results

We present here some sample cases, together with empirically-drawn heuristics regarding system parameter values.

8.1 More than two pieces

When we start with more than two pieces, we compare every piece to every other one. We take the "greedy" approach of choosing the candidate pairs which maximize the number of possible "sewings," for this round of the physical match. Others [10, 4] approach this by building graphs of matches, subsequently redivided into subgraphs by spectral clustering.) E.g., if the following pairs have been discovered: (1,2), (2,3) and (3,4), we will ignore the (2,3) match, first working with the other two pairs. Once that is finished, we iterate until all possible matches have been found. See Figures 17–21.

8.2 System parameter values

Our entire approach is subject to a number of interdependent, system-level parameters which need to be set to the correct values. We demonstrate below what can happen with the wrong set of values.

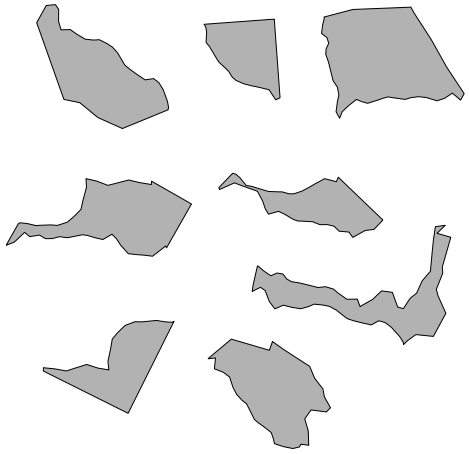


Figure 17: 8 pieces

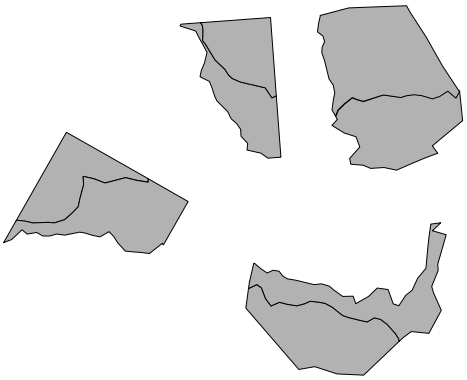


Figure 18: 8 pieces stitched

The parameters of interest are:

Resolution This is the scanning resolution of the various pieces into the computer, in MATLAB pixels. As mentioned in Section 5, care needs to be taken so that the resolution is high enough to capture the twists and turns of the edge, but not too high, as to make the computation inordinately long. Sample values (in MATLAB coordinates): 100, 200.

Edge characterization points Again, as discussed in the same section, this needs to be large enough to supply the linear least squares fit to a parabola algorithm with enough information to characterize the edge slope, but not too large, so that no more than one turn is present in the set of points. Sample values: 23–29 (resolution of 100), 31–37 (resolution of 200).

Straight corner maximum This determines the maximum value of change of direction at a pixel, such that we still consider there to be no change, as if a straight line was passing through. We addressed this in Section 6.1 to filter out straight lines. Sample values: 0.2–0.8.

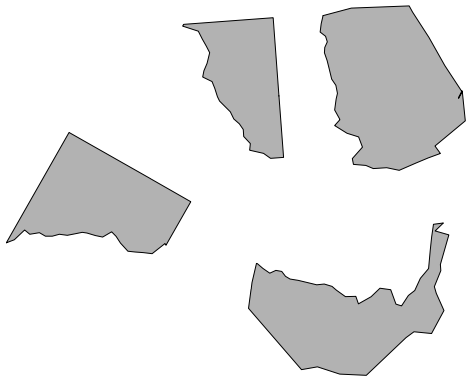


Figure 19: 4 pieces

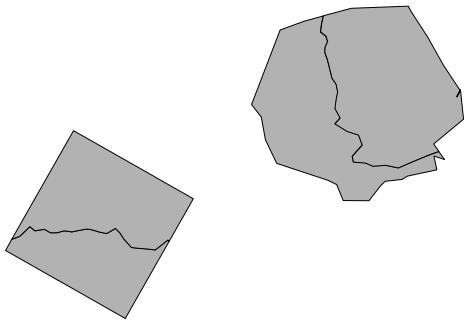


Figure 20: 4 pieces stitched

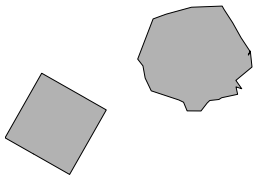


Figure 21: 2 pieces

LCS maximum match In the LCS matching algorithm, we stated in Section 6 that when comparing changes of slope, we are willing to have these numbers be different, up to this maximum value. Sample values: 0.1–0.4.

LSQ confidence in matches This value is used in Section 7.2 to supply the nonlinear least squares fit with matches, to translate and rotate the pieces together. It is a fraction of the matches in each direction, starting from the center of the LCS. Sample values: 0.10–0.25.

In addition to the conclusions we derive below, we found:

1) Overall it makes sense to scan at a resolution of 200, in order to obtain better fit statistics (preventing overlaps and gaps), which in turn provides better conditions for subsequent matches to other pieces.

2) The LSQ confidence in matches parameter was less important.

3) Of the three remaining parameter: edge characterization points, straight corner maximum and LCS maximum match, as long as at least two of them were within the nominal bounds prescribed below, then matching was successful approximately 80% of the time. If only one of them was within the nominal bounds (particularly, one of the first two parameters), then successful matching was achieved in about 50% of the cases.

4) The two parameters: straight corner maximum and LCS maximum match, were not particularly dependent on the resolution, as long as they were within the nominal bounds prescribed below.

8.3 Optimal values

In general, the values in Table 1 worked optimally for a scanning resolution value of 100. We show in Table 2 the optimal valus for a scanning resolution value of 200.

resolution of 100	
edge characterization points	27, 29
straight corner maximum	0.4–0.8
LCS maximum match	0.2–0.4
LSQ confidence in matches	0.10–0.25

Table 1: Resolution of 100 — optimal values

resolution of 200	
edge characterization points	35, 37
straight corner maximum	0.4–0.8
LCS maximum match	0.2–0.4
LSQ confidence in matches	0.10–0.25

Table 2: Resolution of 200 — optimal values

We present here a specific case, in order to illustrate what happens when the system parameters stray too far from the nominal values. In Table 3 are the system parameter values for this specific case. The subsequent matches are displayed in Figure 22, showing a good match up between the two pieces.

resolution of 100	
edge characterization points	25
straight corner maximum	0.8
LCS maximum match	0.2
LSQ confidence in matches	0.20

Table 3: Nominal values

If, however, the edge characterization points parameter is set too low, to 17, then the edge slope values are not properly calculated, and just low values of changes in the slope direction are matched. This can be seen in Figure 23, where we see that the matches are also not only along the

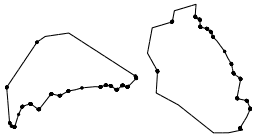


Figure 22: Nominal values

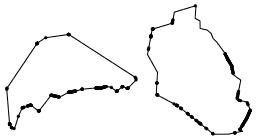


Figure 23: Edge characterization points too low

correct seam, but all around the edges.

Another problematic case is if the straight corner maximum is set too low. In Figure 24, with the value set to 0.3, we see that only *very* straight corners are considered to be modeling straight lines, and therefore we have a huge number of matches. The information at the corners of interest is lost in the noise.

Finally, we demonstrate what happens if the LCS maximum match system parameter is set too low. A value of 0.01 is too stringent to allow edge slope changes of the two pieces to match up with each other. This is seen in Figure 25, where very few matches have been established.

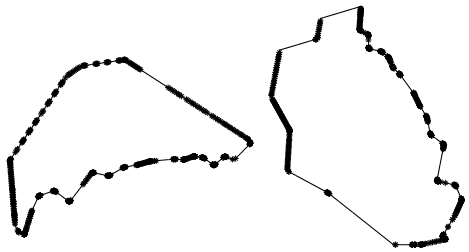


Figure 24: Straight corner maximum too low

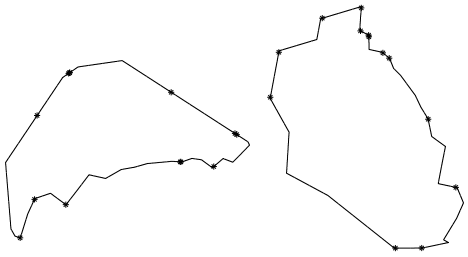


Figure 25: LCS maximum match too low

9 Future research

We list here a number of future directions for this research.

1) As mentioned in Section 3, we would like to build the initial piece, as well as generate the breaks, using the more general splines, instead of straight lines.

2) Clearly we would like to run the analysis for actual pieces of broken/torn material, in order to verify the optimal system parameter values.

3) "Holes" in the material might be due to missing pieces. How do the holes effect this process? Can the stitching algorithm of Section 7.3 easily be modified so as to be able to "jump" over such holes? Note that previous work of automating jigsaw puzzle reconstruction rarely addressed this.

4) In addition, we would like to understand how various types of material (glass, pottery, plastic, rubber, etc.) effect the system parameter values of choice.

5) Specifically, paper has more characteristics, due to the fibers jutting out along the torn edge. Can we take advantage of these? Are they "in the way?" Can we remove them, without effecting the success of the subsequent matching?

6) On the issue of paper, how does the fact that it is usually multi-ply, effect the analysis? Can we analyze each ply, and match on the composite picture?

7) This also brings us to the exciting extension to 3-D physical match, with applications to archeology and exploded object reconstruction.

8) Finally, there is the area of forensics. In order to claim: "This piece came from this object." and have this be admissable in court, we have to answer questions regarding confidence levels of the fit, as well as statistical probability that no other piece could likely fit there.

10 Summary

We have presented a method for solving the problem of physical match. The algorithm involves finding the edge pixels, ordering them, and using their positions to characterize the edge slope. The turns along the edges are then matched to each other using a modified version of the longest common subsequence algorithm.

Finally, the various pieces are translated, rotated and flipped (if necessary), and then stitched together for further matching to other pieces.

We discussed future directions for the research, particularly in the arena of extending the analysis to 3-D shapes and breaks.

References

[1] F. BORTOLOZZI, *Document reconstruction based on feature matching*, in 18th Brazilian Symposium on Computer Graphics and Image Processing, Oct. 2005, pp. 163–170.

[2] J. Canny, *A computational approach to edge detection*, Pattern Analysis and Machine Intelligence, IEEE Transactions on, 8 (1986), pp. 679–698.

[3] H. C. da Gama Leitao and J. Stolfi, *A multiscale method for the reassembly of two-dimensional fragmented objects*, IEEE Transactions on Pattern Analysis and Machine Intelligence, 24 (2002), pp. 1239–1251.

[4] A. C. Gallagher, *Jigsaw puzzles with pieces of unknown orientation*, in Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on, IEEE, 2012, pp. 382–389.

[5] M. R. Garey and D. S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*, W. H. Freeman, 1979.

[6] Q.-X. Huang, S. Flöry, N. Gelfand, M. Hofer, and H. Pottmann, *Reassembling fractured objects by geometric matching*, in ACM Transactions on Graphics (TOG), vol. 25, ACM, 2006, pp. 569–578.

[7] M. Kampel and R. Sablatnig, *3d puzzling of archeological fragments*, in Proc. of 9th Computer Vision Winter Workshop, vol. 2, Slovenian Pattern Recognition Society, 2004, pp. 31–40.

[8] F. Kleber and R. Sablatnig, *A survey of techniques for document and archaeology artefact reconstruction*, in Document Analysis and Recognition, 2009. ICDAR'09. 10th International Conference on, IEEE, 2009, pp. 1061–1065.

[9] D. A. Kosiba, P. M. D. and. S. Balasubramanian, T. L. Gandhi, and K. Kasturi, *An automatic jigsaw puzzle solver*, in Proceedings of the 12th IAPR International Conference on Pattern Recognition - Conference A: Computer Vision & Image Processing, vol. 1, 1994, pp. 616–618.

[10] H. Liu, S. Cao, and S. Yan, *Automated assembly of shredded pieces from multiple photos*, Multimedia, IEEE Transactions on, 13 (2011), pp. 1154–1162.

[11] A. E. Naiman, E. Farber, and Y. Stein, *CLCS—cyclic longest common subsequence*, 2018. submitted.

[12] ——, *Edge characterization of digitized images*, Oct. 2018. submitted.

[13] ——, *EdgeTrek—interior and boundary pixels for large regions*, Oct. 2018. submitted.

[14] G. Papaioannou and E.-A. Karabassi, *On the automatic assemblage of arbitrary broken solid artefacts*, Image and Vision Computing, 21 (2003), pp. 401–412.

[15] Scientific Working Group for Materials Analysis, *Glass fractures*, Forensic Science Communications, 7 (2005).

[16] Y. Shor, Y. Yekutieli, S. Wiesner, and T. Tsach, *Physical Match*, Elsevier, Academic Press, 2 ed., 2013, pp. 54–59. Encyclopedia of Forensic Sciences, ed.: Jay A. Siegel, Pekka J. Saukko, Max M. Houck.

[17] G. Üçoluk and I. HakkI Toroslu, *Automatic reconstruction of broken 3-d surface objects*, Computers & Graphics, 23 (1999), pp. 573–582.

[18] A. Ukovich, G. Ramponi, H. Doulaverakis, Y. Kompatsiaris, and M. G. Strintzis, *Shredded document reconstruction using MPEG-7 standard descriptors*, in Proceedings of the Fourth IEEE International Symposium on Signal Processing and Information Technology, Dec. 2004, pp. 334–337.

[19] H. Wolfson, E. Schonberg, A. Kalvin1, and Y. Lamdan, *Solving jigsaw puzzles by computer*, Annals of Operations Research, 12 (1988), pp. 51–64.

[20] F.-H. Yao and G.-F. Shao, *A shape and image merging technique to solve jigsaw puzzles*, Pattern Recognition Letters, 24 (2003), pp. 1819–1835.

[21] L. Zhu, Z. Zhou, J. Zhang, and D. Hu, *A partial curve matching method for automatic reassembly of 2d fragments*, in Intelligent Computing in Signal Processing and Pattern Recognition, Springer, 2006, pp. 645–650.

https://doi.org/10.31449/inf.v43i2.2090 Informatica **43** (2019) 253–262 **253**

The Permutable *k*-means for the Bi-partial Criterion

Sergey Dvoenko
Tula State University, 92 Lenin Ave., Tula, Russian Federation
E-mail: sergedv@yandex.ru

Jan Owsinski
Systems Research Institute, Polish Academy of Sciences, 6 Newelska, 01 447 Warsaw, Poland
E-mail: owsinski@ibspan.waw.pl, http://www.ibspan.waw.pl/glowna/en /

Keywords: distance, similarity, dissimilarity, cluster, *k*-means, objective function

Received: December 18, 2017

The bi-partial criterion for clustering problem consists of two parts, where the first one takes into account intra-cluster relations, and the second – inter-cluster ones. In the case of k-means algorithm, such bi-partial criterion combines intra-cluster dispersion with inter-cluster similarity, to be jointly minimized. The first part only of such objective function provides the "standard" quality of clustering based on distances between objects (the well-known classical k-means). To improve the clustering quality based on the bi-partial objective function, we develop the permutable version of k-means algorithm. This paper shows that the permutable k-means appears to be a new type of a clustering procedure.

Povzetek: Študija se ukvarja z gručenjem znotraj in med gručami, pri čemer izvirna metoda uporablja permutirano verzijo običajnega algoritma za gručenje.

1 Introduction and related works

1.1 Clustering by *k*-means

According to the basic idea of the classical *k*-means algorithm [1-5], a set $\Omega = \{\omega_1, \ldots \omega_N\}$ of N elements is divided into clusters Ω_k, $k = 1, \ldots K$, represented in a feature space by their "representative" objects $\tilde{\mathbf{x}}_k$, and/or "mean" objects $\bar{\mathbf{x}}_k$ (centers), where $\mathbf{x} = (x_1, \ldots x_n)^T$ is a vector in the $n-$dimensional space.

In this paper, we consider means as representatives and calculate new means as in the classical procedure.

The well-known respective clustering criterion minimizes average of squared distances to cluster centers

$$J(K) = \frac{1}{N}\sum_{k=1}^{K} N_k \sigma_k^2 = \sum_{k=1}^{K} \frac{N_k}{N} \sigma_k^2 , \qquad (1)$$

$$\sigma_k^2 = \frac{1}{N_k}\sum_{i=1}^{N_k} \| \mathbf{x}_i - \bar{\mathbf{x}}_k \|^2 = \frac{1}{N_k}\sum_{i=1}^{N_k} d^2(\mathbf{x}_i, \bar{\mathbf{x}}_k) ,$$

where σ_k^2 is the dispersion of the cluster Ω_k having size N_k, and $d(\mathbf{x}, \mathbf{y})$ is the Euclidean distance between vectors $\mathbf{x}$ and $\mathbf{y}$.

As it is well-known [6–10], cluster dispersions can be calculated without direct use of cluster means, based on pairwise distances between vectors

$$\sigma_k^2 = \frac{1}{2N_k^2}\sum_{i=1}^{N_k}\sum_{j=1}^{N_k} \| \mathbf{x}_i - \mathbf{x}_j \|^2 = \frac{1}{2N_k^2}\sum_{i=1}^{N_k}\sum_{j=1}^{N_k} d^2(\mathbf{x}_i, \mathbf{x}_j) . \quad (2)$$

Empirical data often appear in the form of a matrix of pairwise comparisons of elements of the set. Such comparisons can be nonnegative values of dissimilarity or similarity of objects from the set Ω [11].

This is important for our approach, since the permutable *k*-means, developed in this paper, uses only distance $D(N, N)$ or similarity $S(N, N)$ matrices. Therefore, cluster means are not presented in them, and we need to develop equivalent forms of (1) and (2).

The basis of our approach is the Torgerson's idea of the "gravity center", developed for multidimensional scaling problem [6] in the method of double centering for principal projections to get the appropriate feature space with the raw distance matrix, immersed in it.

Our goal is different: we do not want to restore a feature space itself, since it is sufficient to suppose that objects are immersed in some metric (more closely, Euclidean) space, as we show this later on.

Naturally, the two-component criteria, similar to the bi-partial one (Dunn, Calinski-Harabasz, Xie-Beni etc.), are used in cluster-analysis [9, 12]. They are mainly used to assess the proper number of clusters *K*. Such criteria are usually heuristic constructions, used to assess the results of some algorithms of quite different origins and properties.

Here we are interested in improving the results of the classical clustering problem with a predefined number of clusters *K*. Namely, we try to develop here the bi-partial objective function to build a homogeneous and strict metric criterion for standard *k*-means algorithm only for a predefined number *K*, and not to use any other idea of procedure than that of *k*-means.

1.2 The bi-partial criterion

In order to introduce here a general bi-partial objective function, we refer to an illustrative problem of dividing a

unidimensional empirical distribution of real values into a set of categories to get the "best" set in a definite sense [13–15]. This case serves merely the purpose of illustration, and assumptions made on data do not apply to the general bi-partial approach.

Let a sequence of N positive real observations $x_i, i = 1,... N$ be given in non-decreasing order, i.e. with $x_{i+1} \geq x_i$ for all of them. Any such sequence can be represented through a cumulative form, obtained via transformation $z_i = \sum_{p=1}^{i} x_p$, $i = 1,... N$.

As a result, we deal with a convex non-decreasing sequence $z_i, i = 1,... N$. This means that a straight line, connecting two observations, z_q and z_s, with $1 \leq q < s \leq N$ has all values not under the corresponding observations $z_i, i = q,... s$.

Obviously, for the sequence of constant values $x_1 = ... = x_N = c$ the convex cumulative form is the line $z_i = ic, i = 1,... N,$ with $z_1 = c, z_N = cN$, represented perfectly by the single linear piece.

Otherwise, for non-constant values, by increasing the number of linear segments from the single one (with $q = 1, s = N$), we steadily decrease the error of approximation of the original distribution $\{z_i\}$ by the broken line, composed of such segments, down to zero, when the maximal number $N-1$ of linear segments, corresponded to the number of observations *N*, is used to represent the distribution.

Under these conditions, the problem of obtaining the optimal piece-wise linear approximation of the cumulative sequence with the number of linear segments also being optimized was investigated in [13-15].

According to [13-15], the respective bi-partial objective function J_{DS} penalizes, first, deviation C_D of linear segments from the respective distribution, and, second, penalizes similarity C_S of linear segments to each other, and was represented in the form

$$J_{DS}(K) = (1-\alpha)C_D(K) + \alpha C_S(K) \to \min , \quad (3)$$

where $K \geq 1$ is the number of segments, $0 \leq \alpha \leq 1$ is the coefficient of linear combination of two parts of the criterion.

The criterion J_{DS}, investigated in [13, 14] for the above problem, is a particular case of the general bi-partial form, representing the fundamental "intra-cluster cohesion + inter-cluster separation" paradigm [15, 16].

It should be noted that the parameter α in (3) need not appear at all, if two parts of the objective function are assumed to reflect correctly the respective inner and outer measures. Note that by solving with respect to (3) we get both the cluster (segment) content and the number of clusters (segments). We can also represent (3) in different forms to obtain different data analysis problems as particular cases. So, e.g., (3) can be transformed to the linear regression problem for $K = 1,\ \alpha = 0$.

In other interesting cases, the problem (3) can be considered for other kinds of parameters than α, say, *K*. Thus, we can treat $K \geq 1$ as a hyper-parameter and find the optimal linear combination of parts in $J_{DS}(K)$.

Thus, in the context of the illustrative problem quoted, we would fix the number of line segments *K*, and look with (3) for the optimum weight α, meaning the significance we attach to accuracy of the approximation vs. distinctiveness of the consecutive segments.

In this paper, we investigate the single-parametric reduced form of (3) to find the optimal α for the predefined hyper-parameter *K* based on the direct implementing of the well-known k-means algorithm.

2 Distance and similarity *k*-means

In this paper, we use the specially developed *k*-means algorithm only for the case of distances or similarities between objects [17, 18].

A positive definite similarity matrix can be obtained as a matrix of pairwise scalar products of object descriptions in some metric space with the dimensionality of not more than a set cardinal number. This matrix of scalar products can be transformed into a distance matrix and vice versa. As a result, the dissimilarity matrix can be used as the distance matrix in the same space.

In this case, the mean object $\omega(\bar{\mathbf{x}}_k)$ cannot be defined in Ω by the distance matrix $D(N,N)$ as a center of a cluster. Usually, the object minimizing the sum of distances to the others in the cluster can be used as the center $\bar{\omega}_k$. Therefore, if representatives and centers coincide each with other, $\tilde{\omega}_k = \bar{\omega}_k$ for all clusters, then we get an unbiased clustering.

Nevertheless, if we immerse the set Ω in some feature space, we obtain in general the biased clustering, since the center $\mathbf{x}(\bar{\omega}_k)$ may not be the mean object $\bar{\mathbf{x}}_k$ in the unknown feature space.

The classical *k*-means algorithm was developed for distances and similarities in [17, 18]. Centers $\bar{\omega}_k$ provide the unbiased clustering with cluster dispersions $\sigma_k^2 = (1/N_k)\sum_{i=1}^{N_k} d^2(\omega_i, \bar{\omega}_k)$ minimizing $J(K)$. If the set Ω is immersed in a feature space, then two criteria

$$J^X(K) = \min_{\bar{\mathbf{x}}_1,... \bar{\mathbf{x}}_K} J(K) , \ J^D(K) = \min_{\bar{\omega}_1,... \bar{\omega}_K} J(K)$$

have not the same values, since $J^D(K) \geq J^X(K)$ in general. Yet, $J^X(K) = J^D(K)$, if objects $\mathbf{x}(\bar{\omega}_k)$ and $\bar{\mathbf{x}}_k$ are the same.

We would like to guarantee this condition. For some $\omega_l \in \Omega$, as a point of the origin and a pair ω_i, ω_j , the scalar product is $s_{ij} = (d_{li}^2 + d_{lj}^2 - d_{ij}^2)/2$, where distance is $d_{pq} = d(\omega_p, \omega_q)$ and $s_{ii} = d_{li}^2$ for $i = j$. Therefore, the main diagonal of the matrix $S_l(N,N)$ represents the squared distances from the origin $\omega_l \in \Omega$ to other objects.

According to [6], it is convenient to put the origin of the feature space in the center of all objects $\omega_i \in \Omega,\ i = 1,... N$. Therefore, we put the origin of the

feature space, cluster by cluster, in each center $\bar{\omega}_k$ to represent it by its distances to all other objects in the unknown feature space (N_k is the number of objects in Ω_k, $\omega_p, \omega_q \in \Omega_k$):

$$d^2(\omega_i, \bar{\omega}_k) = \frac{1}{N_k}\sum_{p=1}^{N_k} d_{ip}^2 - \frac{1}{2N_k^2}\sum_{p=1}^{N_k}\sum_{q=1}^{N_k} d_{pq}^2 , \quad (4)$$

where, according to (1), (4), the cluster dispersion is

$$\sigma_k^2 = \frac{1}{N_k}\sum_{i=1}^{N_k}\left(\frac{1}{N_k}\sum_{p=1}^{N_k} d_{ip}^2 - \frac{1}{2N_k^2}\sum_{p=1}^{N_k}\sum_{q=1}^{N_k} d_{pq}^2\right) = \frac{1}{2N_k^2}\sum_{p=1}^{N_k}\sum_{q=1}^{N_k} d_{pq}^2. \quad (5)$$

Hence, we develop the distance *k*-means algorithm based on the classical principle of the "minimum distance to a cluster center":

(a) **Step 0.** Determine in some way K centers $\bar{\omega}_k^1$ and put them as representatives $\tilde{\omega}_k^1 = \bar{\omega}_k^1$, $k = 1,... K$; $s = 1$.

Step *s*. Reallocate all objects between clusters:

1. $\omega_i \in \Omega_k^s$, if $d(\omega_i, \bar{\omega}_k^s) \le d(\omega_i, \bar{\omega}_j^s)$ for $\omega_i \in \Omega_{j\ne k}^s$, $j = 1,... K$, $i = 1,... N$.
2. Recalculate centers $\bar{\omega}_k^s$, $k = 1,... K$, represented by distances $d(\omega_i, \bar{\omega}_k^s)$, $i = 1,... N$.
3. Stop, if $\tilde{\omega}_k^s = \bar{\omega}_k^s$, $k = 1,... K$, else $\tilde{\omega}_k^{s+1} = \bar{\omega}_k^s$, $\bar{\omega}_k^{s+1} = \bar{\omega}_k^s$, $k = 1,... K$; $s = s+1$.

Based on the direct recalculation of the criterion (1), the equivalent realization is:

(b) **Step 0.** Determine in some way K centers $\bar{\omega}_k^1$ and put them as representatives $\tilde{\omega}_k^1 = \bar{\omega}_k^1$, $k = 1,... K$; calculate $J^1 = J^1(K)$ and put $\tilde{J}^1 = \tilde{J}^1(K) = J^1$ relative to representatives; $s = 1$.

Step *s*. Reallocate all objects between clusters:

1. $\omega_i \in \Omega_k^s$, if $J_{ik}^s \le J_{ip}^s$ for $\omega_i \in \Omega_{p\ne k}^s$, $p = 1,... K$, $i = 1,... N$.
2. Recalculate centers $\bar{\omega}_k^s$, $k = 1,... K$, represented by distances $d(\omega_i, \bar{\omega}_k^s)$, $i = 1,... N$; recalculate J^s.
3. Stop, if $\tilde{J}^s = J^s$, else $\tilde{J}^{s+1} = J^s$, $J^{s+1} = J^s$; $s = s+1$.

A positive definite similarity matrix $S(N,N)$ with elements $s_{ij} = s(\omega_i, \omega_j) \ge 0$ can be obtained as a matrix of scalar products in the positive quadrant of the feature space. Relative to some point $\omega_k \in \Omega$ as the origin, with $s_{ij} = (d_{ki}^2 + d_{kj}^2 - d_{ij}^2)/2$, $s_{ii} = d_{ki}^2$, distances are defined as $d_{ij}^2 = s_{ii} + s_{jj} - 2s_{ij}$. The cluster center $\bar{\omega}_k$ is represented by its similarities with other objects

$$s(\omega_i, \bar{\omega}_k) = \frac{1}{N_k}\sum_{p=1}^{N_k} s_{ip} , \ \omega_p \in \Omega_k , \omega_i \in \Omega, \ i = 1,... N . \quad (6)$$

The cluster compactness is the mean similarity of the cluster center with respect to other objects (6):

$$\delta_k = \frac{1}{N_k}\sum_{i=1}^{N_k} s(\omega_i, \bar{\omega}_k) = \frac{1}{N_k^2}\sum_{i=1}^{N_k}\sum_{p=1}^{N_k} s_{ip} \ ; \omega_i, \omega_p \in \Omega_k .$$

The unbiased clustering minimizes the cluster dispersion σ_k^2 and maximizes the compactness δ_k according to (5):

$$\sigma_k^2 = \frac{1}{2N_k^2}\sum_{i=1}^{N_k}\sum_{j=1}^{N_k}(s_{ii} + s_{jj} - 2s_{ij}) = \frac{1}{N_k}\sum_{i=1}^{N_k} s_{ii} - \delta_k ,$$

and for all clusters:

$$J(K) = \sum_{k=1}^{K}\frac{N_k}{N}\sigma_k^2 = \frac{1}{N}\sum_{i=1}^{N} s_{ii} - \sum_{k=1}^{K}\frac{N_k}{N}\delta_k = C - I(K) .$$

For similarity clustering, we maximize compactness $I(K)$, with $I(K) = C - J(K)$. The similarity *k*-means algorithm is the analogue of algorithms (a) and (b) relative to $I(K)$.

3 The bi-partial criterion for clustering

In this paper, we develop the bi-partial objective function like (3) for the dissimilarity *k*-means

$$J_\delta(K) = (1-\alpha)J(K) + \alpha\delta(K) , \quad (7)$$

so as to combine $J(K)$ for intra-cluster distances with the inter-cluster similarity $\delta(K)$. We define the inter-cluster similarity $\delta(K) = (1/K)\sum_{k=1}^{K} s(\bar{\omega}_k, \bar{\omega}_0)$ relative to the center of the whole set, being the object $\bar{\omega}_0$, represented by its similarities with respect to all other centers $s(\bar{\omega}_k, \bar{\omega}_0) = (1/K)\sum_{p=1}^{K} s(\bar{\omega}_k, \bar{\omega}_p)$; $\bar{\omega}_k, k = 1,... K$:

$$\delta(K) = \frac{1}{K^2}\sum_{k=1}^{K}\sum_{l=1}^{K} s(\bar{\omega}_k, \bar{\omega}_l) . \quad (8)$$

Unfortunately, the bi-partial criterion $J_\delta(K)$, as defined here, does not work for the classical *k*-means (b), since (8), as the second part of $J_\delta(K)$ in (7), cannot be changed for constant centers while attempting to transfer objects in step *s*.

Therefore, for any $0 \le \alpha < 1$, the clustering results are the same as for the classical case with $\alpha = 0$. And the algorithm does not work properly with $\alpha = 1$.

We develop here the new "permutable" version of the classical *k*-means (b) without direct calculation of cluster centers. Here, the new permutable *k*-means is the meanless clustering for the classical *k*-means (b).

As we can see in (5), the cluster dispersion is half of the average of squared distances between objects in the cluster. This representation does not contain centers themselves, and we calculate the criterion (1) without centers in the form

$$J(K) = \sum_{k=1}^{K}\frac{N_k}{N}\sigma_k^2 = \frac{1}{2N}\sum_{k=1}^{K}\frac{1}{N_k}\sum_{p=1}^{N_k}\sum_{q=1}^{N_k} d_{pq}^2 . \quad (9)$$

Next, we would like to calculate the similarity $s(\bar{\omega}_k, \bar{\omega}_l)$ between cluster centers in (8). According to (6),

the average similarity of the center $\bar{\omega}_k$ with the objects from the other cluster $\omega_i \in \Omega_l$ is

$$s(\Omega_l, \bar{\omega}_k) = \frac{1}{N_l}\sum_{i=1}^{N_l} s(\omega_i, \bar{\omega}_k) = \frac{1}{N_l N_k}\sum_{i=1}^{N_l}\sum_{p=1}^{N_k} s_{ip}\ ,\ \omega_p \in \Omega_k\ .$$

It is evident that $s(\Omega_l, \bar{\omega}_k) = s(\Omega_k, \bar{\omega}_l)$, as $s_{ij} = s_{ji}$. Therefore, we can use the suitable notation $s(\Omega_l, \bar{\omega}_k) = s(\Omega_k, \bar{\omega}_l) = s(\Omega_l, \Omega_k) = s(\bar{\omega}_l, \bar{\omega}_k)$. Hence, (8) is converted into ($\omega_p \in \Omega_k$, $\omega_q \in \Omega_l$):

$$\delta(K) = \frac{1}{K^2}\sum_{k=1}^{K}\sum_{l=1}^{K}\frac{1}{N_k N_l}\sum_{p=1}^{N_k}\sum_{q=1}^{N_l} s_{pq}. \qquad (10)$$

The goal of $J_\delta(K)$ is to produce clusters with possibly low dispersion and possibly dissimilar centers. We note that (10) is in a way an inconsistent function, since for $k = l$ it contains the cluster compactness δ_k. Hence, we modify (10) to get the inter-cluster similarities only and take into account the symmetry

$$\delta(K) = \frac{1}{2K(K-1)}\sum_{k=1}^{K}\sum_{l=1,l\neq k}^{K}\frac{1}{N_k N_l}\sum_{p=1}^{N_k}\sum_{q=1}^{N_l} s_{pq}, \quad (11)$$

for $\omega_p \in \Omega_k$, $\omega_q \in \Omega_l$.

We develop here the classical *k*-means (b) in the new form of the permutable *k*-means based on (9)–(11):

(c) **Step 0.** Determine in some way the sets $\Omega_l^1, l = 1,\dots K$; define α, calculate $J^1 = J_\delta^1(K)$; $s = 1$.

Step *s*. Reallocate all objects between clusters:

1. Remember, but do not move: $\omega_i \in \Omega_k^s$, if $J_{ik}^s \le J_{ip}^s$ for $\omega_i \in \Omega_{p\neq k}^s$, $p = 1,\dots K$, $i = 1,\dots N$.
2. Reallocate all objects $\omega_i, i = 1,\dots N$ at once; calculate J^{s+1}.
3. If $J^{s+1} = J^s$ then stop;
   If $J^{s+1} > J^s$ then: cancel last reallocations, $J^{s+1} = J^s$, stop;
   If $J^{s+1} < J^s$ then: $J^{s+1} = J^s$, $s = s+1$.

As we can see, in the step (*s*.1) we recalculate the criterion J^s in order to get its modified value J_{ip}^s. Let $\omega_i \in \Omega_j^s$. When trying to move ω_i from Ω_j^s to some other Ω_p^s, we try to change the respective sets to $\Omega_j^s \setminus \omega_i$ and to $\Omega_p^s \bigcup \omega_i$. Changes in the sets result in implicit changes of their centers, even though we do not calculate them. Consequently, this action differs from the same one in algorithms (a) and (b) for constant centers.

Algorithm (c) appears to be a new type of clustering procedures, since its result differs, in general, from those of the classical (a) and (b) procedures, both for the classical $(\alpha = 0)$ and the proper bi-partial $(\alpha > 0)$ cases. In addition, we can use some optimal initial clusters to enhance the quality of results, and optimal recalculations to improve performance of permutations.

As we can see, the algorithm (c) is the same as the classical ones (a) and (b) for the standard criterion $J(K)$ and differs (sometimes subtly and finely) from them for the bi-partial criterion $J_\delta(K)$.

It is clear that the new algorithm gives the classical result for non-intersecting clusters. Nevertheless, its result can be improved for intersecting clusters, since by means of the criterion $J_\delta(K)$ a cluster center can be shifted in some vicinity without changing the cluster itself. Such possibility depends on the gaps between real points in continuous feature space and the discrete cluster structure superimposed.

4 Redistribution of data dispersion by the bi-partial criterion

Here, we explain why by means of the criterion (7) it is possible to improve the classical clustering of *k*-means.

Consider the classical case. Let the set of size *N* be divided into *K* subsets (clusters). In our perspective, we consider balancing of total dispersion between its intra- and inter- parts. We know [19, 20] that $S_T = S_W + S_B$, where S_T is the total scatter matrix, S_W is the intra-cluster and S_B is the inter-cluster scatter matrices. Therefore, $trS_T = trS_W + trS_B$ for diagonal elements only. Since $trS_T = N\sigma_T^2$, $trS_W = N\sigma_W^2$, and $trS_B = N\sigma_B^2$, then finally $\sigma_T^2 = \sigma_W^2 + \sigma_B^2$.

Let the set $\Omega = \{\omega_1, \dots \omega_N\}$ be immersed in some metric space and represented by the distance matrix $D(N,N)$ only with elements $d_{ij} = d(\omega_i, \omega_j) \ge 0$. Let Ω be split into groups $\Omega_k, k = 1,\dots K$. Based on the Torgerson's formula, we define the following:

for single group dispersions

$$\sigma_k^2 = \frac{1}{2N_k^2}\sum_{p=1}^{N_k}\sum_{q=1}^{N_k} d^2(\omega_p, \omega_q),\ k = 1,\dots K\ ;$$

for the intra-group dispersion

$$\sigma_W^2 = \sum_{k=1}^{K}\frac{N_k}{N}\sigma_k^2 = \sum_{k=1}^{K}\frac{N_k}{N}\frac{1}{2N_k^2}\sum_{p=1}^{N_k}\sum_{q=1}^{N_k} d^2(\omega_p, \omega_q) = \frac{1}{2N}\sum_{k=1}^{K}\frac{1}{N_k}\sum_{p=1}^{N_k}\sum_{q=1}^{N_k} d^2(\omega_p, \omega_q)\ ;$$

for the total dispersion

$$\sigma_T^2 = \frac{1}{2N^2}\sum_{p=1}^{N}\sum_{q=1}^{N} d^2(\omega_p, \omega_q) = \frac{1}{2N^2}\sum_{k=1}^{K}\sum_{l=1}^{K}\sum_{p=1}^{N_k}\sum_{q=1}^{N_l} d^2(\omega_p, \omega_q)\ ;$$

for the inter-center dispersion

$$\sigma_{IC}^2 = \frac{1}{K}\sum_{k=1}^{K} d^2(\bar{\omega}_k, \bar{\omega}_0) = \frac{1}{2K^2}\sum_{p=1}^{K}\sum_{q=1}^{K} d^2(\bar{\omega}_p, \bar{\omega}_q)\ ,$$

where the center $\bar{\omega}_0$ of the set Ω is represented by its distances to other centers $\bar{\omega}_k$ through

$$d^2(\bar{\omega}_k, \bar{\omega}_0) = \frac{1}{K}\sum_{p=1}^{K} d^2(\bar{\omega}_k, \bar{\omega}_p) - \sigma_{IC}^2\ .$$

We remark that the classical inter-group dispersion is not given by the Torgerson's formula

$$\sigma_B^2 = \sum_{k=1}^{K} \frac{N_k}{N} d^2(\bar{\omega}_k, \bar{\omega}_0).$$

Therefore, the classical inter-group dispersion is

$$\sigma_B^2 = \sum_{k=1}^{K} \frac{N_k}{N} \left(\frac{1}{K} \sum_{p=1}^{K} d^2(\bar{\omega}_k, \bar{\omega}_p) - \sigma_{IC}^2 \right) =$$

$$\frac{1}{K} \sum_{k=1}^{K} \frac{N_k}{N} \sum_{p=1}^{K} d^2(\bar{\omega}_k, \bar{\omega}_p) - \sigma_{IC}^2 \sum_{p=1}^{K} \frac{N_k}{N} =$$

$$\frac{1}{K} \sum_{k=1}^{K} \frac{N_k}{N} \sum_{p=1}^{K} d^2(\bar{\omega}_k, \bar{\omega}_p) - \sigma_{IC}^2.$$

As shown above, we minimize the classical criterion $J(K)$ based on the distance matrix $D(N,N)$, and maximize the criterion in the dual form $I(K) = C - J(K)$ based on the similarity matrix $S(N,N)$.

Hence, in the dual form of the bi-partial criterion we try to maximize the classical part $I(K)$ and the new second part for the inter-center dispersion σ_{IC}^2, as based on the Torgerson's formula. Since the classical inter-group dispersion σ_B^2 is not based on the Torgerson's formula, we calculate it with distances $d^2(\bar{\omega}_k, \bar{\omega}_0)$. Such distances refer to distances between sets, not being a topic here.

Hence, in the dual form by maximizing $I(K)$, we minimize strictly equivalent classical $J(K)$ and maximize the inter-center dispersion σ_{IC}^2. Since $\sigma_T^2 = \sigma_W^2 + \sigma_B^2$, we have the decomposition

$$\sigma_T^2 = \sigma_W^2 + \frac{1}{K} \sum_{k=1}^{K} \frac{N_k}{N} \sum_{p=1}^{K} d^2(\bar{\omega}_k, \bar{\omega}_p) - \sigma_{IC}^2.$$

Let us denote $\sigma_{B \cup IC}^2 = \frac{1}{K} \sum_{k=1}^{K} \frac{N_k}{N} \sum_{p=1}^{K} d^2(\bar{\omega}_k, \bar{\omega}_p)$ and represent the classical inter-group dispersion in the form $\sigma_B^2 = \sigma_{B \cup IC}^2 - \sigma_{IC}^2$ without the contribution of the inter-center dispersion, where $\sigma_T^2 + \sigma_{IC}^2 = \sigma_W^2 + \sigma_{B \cup IC}^2$.

As we can see, the permutable *k*-means is targeted to minimize $J(K) = \sigma_W^2$. Since the total dispersion $\sigma_T^2 = const$, at the same time the classical inter-group dispersion $\sigma_B^2 = \sigma_{B \cup IC}^2 - \sigma_{IC}^2$ is maximized. Therefore, the balance $\sigma_T^2 = \sigma_W^2 + \sigma_B^2$ remains true. The decomposition $\sigma_T^2 + \sigma_{IC}^2 = \sigma_W^2 + \sigma_{B \cup IC}^2$ shows that the balance of two parts is maintained, while we increase both of them.

In this case, the bi-partial criterion influences σ_{IC}^2 only. Hence, by means of the bi-partial criterion we manipulate to maximize the inter-center dispersion σ_{IC}^2 with the other part $\sigma_{B \cup IC}^2$ being maximized "as is".

5 Experiments

5.1 Experimental setup

Experimental data are the original Fisher's *Iris data* [21]. We chose this data set as a simple illustration for the basic properties of the approach developed. Such data consist of 150 measurements of 50 plants, belonging to three varieties: *Iris setosa*, *Iris versicolor*, and *Iris virginica*. Four flower measurements are made: petal length and width, and sepal length and width.

It is known that the 1st class (*Iris setosa*) is well separated from other two classes (2nd class, *Iris versicolor*, and 3rd class, *Iris virginica*). The 2nd and 3rd classes intersect each other. Another peculiarity of *Iris data* is the coincidence of objects 102 and 143 from the 3rd class. *Iris data* are also included in Matlab.

There are also other available variants of the *Iris data*, differing from the classic set of [21]. Such differences usually concern corrections in some measurements.

Since the classification of data has been defined, we show that the bi-partial objective function $J_\delta(K)$, developed above, allows us to improve the classical clustering result. According to it, we separate as usual 1st class correctly from two others, and decrease the errors in separation of the 2nd and the 3rd class.

According to the formulation above, we investigate the problem

$$\alpha^* = \arg\min_{0 \le \alpha \le 1} J_\delta(K) = \arg\min_{0 \le \alpha \le 1} \left((1-\alpha) J(K) + \alpha \delta(K) \right).$$

As we can see, this formulation implies balancing of two parts of the criterion. Therefore, it would be good to measure $J(K)$ and $\delta(K)$ on the same scale.

The dispersion of standardized data is *n*, i.e. the number of features (*n*=4 for *Iris data*), and usually more than *n* for original (non-standardized) data. The clustering results for original and standardized data can differ.

In order to get rid of the potential scale bias, we normalize inter-cluster similarities $s'_{kl} = s_{kl} / \sqrt{s_{kk} s_{ll}}$ to get $s'_{kk} = 1,\ 0 \le s'_{kl} \le 1;\ k, l = 1, \ldots K$.

The last technical remark regarding the correctness of the criterion $J_\delta(K)$ is that in the case of usual standard multidimensional data, we need to move the origin out of the convex cover of the set relative to its center and provide positive scalar products as similarities between objects. This problem was discussed in [22].

Indeed, as it is mentioned above, all similarities in (6), (8), (10), (11) must be nonnegative for correct $I(K)$ and $\delta(K)$. According to (4), the origin is placed in the center of the data set in the feature space.

Unfortunately, it this case we cannot use scalar products $s_{ij} = (d_{ki}^2 + d_{kj}^2 - d_{ij}^2)/2$ in $J_\delta(K)$, since they can have negative values. Nevertheless, scalar products change to nonnegative values with respect to the origin placed out of the convex cover of the set, since all of them appear to be in the positive quadrant of the feature space. Hence, it does not matter at all for distances (they have been calculated and not changed for any place of the origin), but it is correct to represent nonnegative similarities by scalar products.

It is known that the *k*-means algorithm is the locally optimal procedure with results dependent on initial decisions (partition or choice of centers).

For all classes, we test three initial partitions: 50/50/50

(plant varieties as classes), 50/70/30 (20 plants from the 3rd class are wrongly placed in the 2nd class), 50/30/70 (20 plants from the 2nd class are wrongly placed in the 3rd class).

For just two intersecting classes (2nd and 3rd) we test also three initial partitions: 50/50 (plant varieties as classes), 70/30 (20 plants from the 3rd class are wrongly placed in the 2nd class), 30/70 (20 plants from the 2nd class are wrongly placed in the 3rd class).

In yet another case we investigate two classes of the entire set, organized as the small one (1st class) and the big one (2nd and 3rd classes). We test three initial partitions: 50/100 (plants from the 1st class versus all plants from the 2nd and 3rd classes together), 100/50 (all plants from the 1st and the 2nd classes together versus plants from the 3rd class), 30/120 (only first 30 plants from the 1st class versus all others).

In all experiments, we first get the classical result with $\alpha = 0$, starting from the predefined initial partitions as above. Second, starting, as well, from the predefined initial partitions characterized above, we vary the parameter $0 < \alpha \le 1$ with increment 0.01 to find the optimal α^* among the tested 100 points.

5.2 Results and discussion

In the first experiment with original *Iris data* for all initial partitions for three classes, we correctly separate the 1st class and decrease errors in separation of intersecting 2nd and 3rd classes (Table 1, Fig.1). For two intersecting classes only, we decrease errors in the separation of the 2nd and 3rd classes, too (Table 1, Fig. 2, 3). It can be seen that the optimal intervals for α^* depend on the number of clusters (Table 1), hence on data dispersion, and can slightly differ for different initial partitions. Error diagrams are not monotonic functions (Fig. 1–3).

As we can see, original *Iris data* are some sort of "well structured" data, since for different initial partitions we get the same 16 misclassified objects for the classical $(\alpha = 0)$ criterion and the same 15 misclassified objects for the bi-partial (α^*) criterion (Table 1). For the classical criterion, misclassified objects are generally from the 3rd class (Table 2). The object 135 is well classified and shown here, since it is misclassified for the bi-partial criterion.

Misclassified objects for the bi-partial criterion are from the 3rd class, too (Table 3). Here, objects 53 and 78 are well classified, and the object 135 is misclassified.

We repeat this experiment for standardized data (Table 4). Such data are more complicated. As we know, *Iris* classes are not so spherical ones in the original feature space, and that is why the *k*-means type of approach is not the best suited for them.

After data standardization, classes appear to be more spherical and contain more "mixed" objects from intersecting classes, usually giving more misclassifications in the classical case (Table 4).

Hence, for the classical criterion $(\alpha = 0)$ for standardized data, 25 misclassified objects are from two intersecting classes, 2nd and 3rd, with well classified all objects from the 1st class (Table 5). Objects 104, 109, 112, 126, 129 are well classified and shown too, since they are misclassified for the bi-partial criterion.

Misclassified objects for the bi-partial criterion are mainly from the 3rd class again (Table 6). Here, objects 52, 57, 66, 71, 76, 86, 87 are well classified and objects 104, 109, 112, 128, 129 are misclassified.

Table 1: Clustering results of original *Iris data*.

Initial partitions	Errors $(\alpha = 0)$	α^*	Errors (α^*)	Diagrams
50/50/50	16	0.6 ÷ 0.75	15	Fig. 1
50/70/30	16	0.6 ÷ 0.75	15	
50/30/70	16	0.6 ÷ 0.75	15	
50/50	16	0.81 ÷ 0.92	15	Fig. 2
70/30	16	0.81 ÷ 0.92	15	
30/70	16	0.81 ÷ 0.91	15	Fig. 3

Table 2: Classical 16 misclassifications of original *Iris data*.

$\alpha = 0$ 50/50/50 50/50 50/70/30 70/30 50/30/70 30/70	2nd cluster	3rd cluster
Iris versicolor 2nd class (51-100)		53 78
Iris virginica 3rd class (101-150)	102 120 128 147 107 122 134 150 114 124 139 115 127 143	Correct: 135

Table 3: Bi-partial 15 misclassifications of original *Iris data*.

α^* 50/50/50 50/50 50/70/30 70/30 50/30/70 30/70	2nd cluster	3rd cluster
Iris versicolor 2nd class (51-100)	**53** **78**	
Iris virginica 3rd class (101-150)	102 120 128 147 107 122 134 150 114 124 139 115 127 143 **135**	

Table 4: Clustering results of standardized *Iris data*.

Initial partitions	Errors $(\alpha = 0)$	α^*	Errors (α^*)	Diagrams
50/50/50	25	0.85	22	Fig. 4
50/70/30	25	0.85	22	
50/30/70	25	0.85	22	
50/50	17	0.94 ÷ 0.97	15	Fig. 5
70/30	17	0.92 ÷ 0.97	15	Fig. 6
30/70	17	0.83 ÷ 0.95	14	Fig. 7

Table 5: Classical 25 misclassifications of standardized *Iris data.*

$\alpha = 0$ 50/50/50 50/70/30 50/30/70	2nd cluster	3rd cluster
Iris versicolor 2nd class (51-100)		51 57 76 86 52 66 77 87 53 71 78
Iris virginica 3rd class (101-150)	102 120 134 147 107 122 135 150 114 124 139 115 127 143	Correct: 104 129 109 112 128

Table 6: Bi-partial 22 misclassifications of standardized *Iris data.*

α * 50/50/50 50/70/30 50/30/70	2nd cluster	3rd cluster
Iris versicolor 2nd class (51-100)	**52 71 86** **57 76 87** **66 77**	51 53 78
Iris virginica 3rd class (101-150)	102 122 139 **104** 107 124 143 **109** 114 127 147 **112** 115 134 150 **128** 120 135 **129**	

Table 7: Classical 17 misclassifications of standardized *Iris data.*

$\alpha = 0$		2nd cluster	3rd cluster
50/50	*Iris versicolor* 2nd class (51-100)		51 53 78
	Iris virginica 3rd class (101-150)	102 122 134 147 107 124 135 150 114 127 139 120 128 143	Correct: 112
70/30	*Iris versicolor* 2nd class (51-100)		51 53 78
	Iris virginica 3rd class (101-150)	102 122 134 147 107 124 135 150 114 127 139 120 128 143	Correct: 112
30/70	*Iris versicolor* 2nd class (51-100)	Correct: 52 71 87 57 77 66 86	51 53 78
	Iris virginica 3rd class (101-150)	102 122 134 147 107 124 135 150 114 127 139 120 128 143	

Table 8: Bi-partial misclassifications of standardized *Iris data.*

α *		2nd cluster	3rd cluster
50/50	*Iris versicolor* 2nd class (51-100)	**51** **53** **78**	
	Iris virginica 3rd class (101-150)	102 122 134 147 107 124 135 150 114 127 139 120 128 143 **112**	
70/30	*Iris versicolor* 2nd class (51-100)	**51** **53**	78
	Iris virginica 3rd class (101-150)	102 122 134 147 107 124 135 150 114 127 139 120 128 143	
30/70	*Iris versicolor* 2nd class (51-100)	52 71 87 57 77 66 86	51 **52 71 87** 53 **57 77** 78 **66 86**
	Iris virginica 3rd class (101-150)	107 114 120 135	102 128 147 122 134 150 124 139 127 143

For two intersecting classes of standardized data and for the classical $(\alpha = 0)$ criterion (Table 7), we get the same 3 misclassified objects from the 2nd class and 14 misclassified objects from the 3rd class.

We get different misclassified objects (Table 8) for different initial partitions in the bi-partial case (15 objects for the 50/50 and 70/30 initial partitions, 14 objects for the 30/70 initial partition).

For standardized data, we usually get different results for three and two classes relative to original data. As we can see, the best result with the minimum of 14 errors for the initial partition 30/70 differs in terms of objects from the results for other initial partitions (Table 8).

Even though standardization is a usual step in data processing, we can see that the clustering results for standardized *Iris data* are not so "natural" as for the original ones. This is the well known and unwanted effect of standardization.

Clustering results for *Iris data* by both classical and by bi-partial criteria are more "natural" for original data than for standardized data.

In the second experiment, we investigate the already mentioned general defect of the criterion (1). As it is well known, the classical *k*-means clustering tries to get clusters, which are approximately equal by size.

In case of classes that differ as to their sizes, the new permutable algorithm decreases usually the size of the bigger class (2nd and 3rd together) and increases the size of the smaller class (1st).

This is the classical result for $\alpha = 0$ with three errors for original *Iris data* (objects 58, 94, 99 were misclassified to the 1st class).

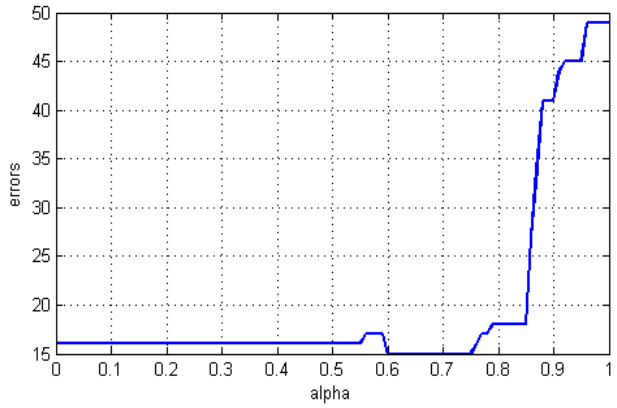


Figure 1. Clustering errors of original *Iris data* for Setosa/Versicolor/Virginica varieties (50/50/50, 50/70/30, 50/30/70) with 15 misclassified objects.

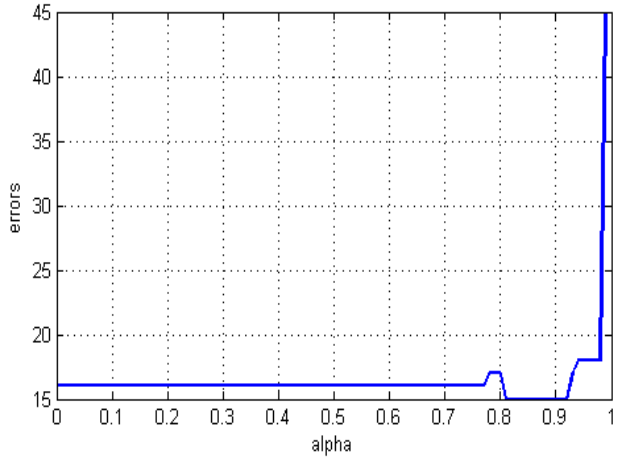


Figure 2. Clustering errors of original *Iris data* for Versicolor/Virginica varieties (50/50, 70/30) with 15 misclassified objects.

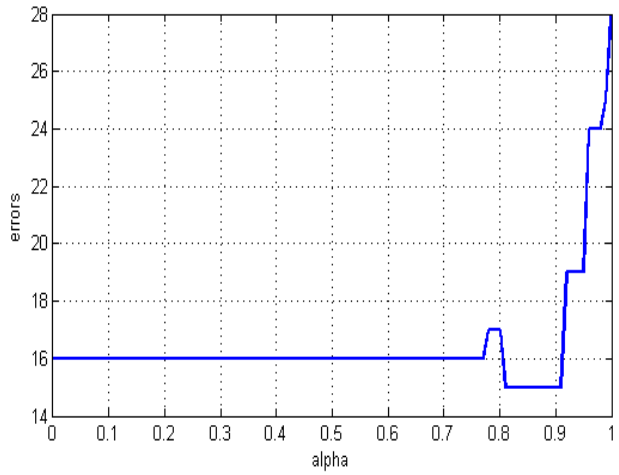


Figure 3. Clustering errors of original *Iris data* for Versicolor/Virginica varieties (30/70) with 15 misclassified objects.

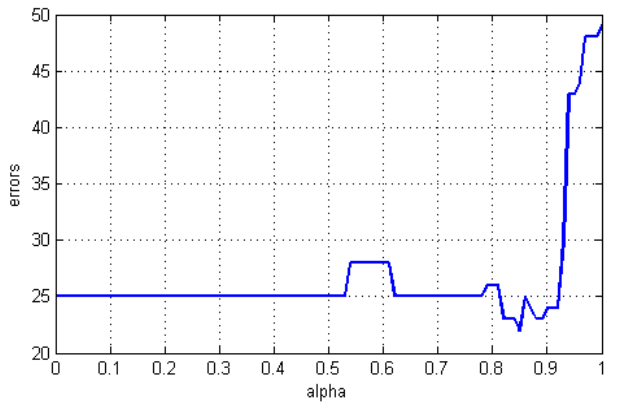


Figure 4. Clustering errors of standardized *Iris data* for Setosa/Versicolor/Virginica varieties (50/50/50, 50/70/30, 50/30/70) with 22 misclassified objects.

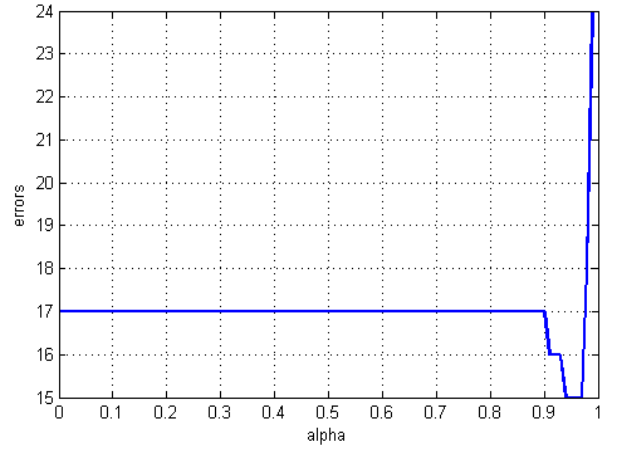


Figure 5. Clustering errors of standardized *Iris data* for Versicolor/Virginica varieties (50/50) with 15 misclassified objects.

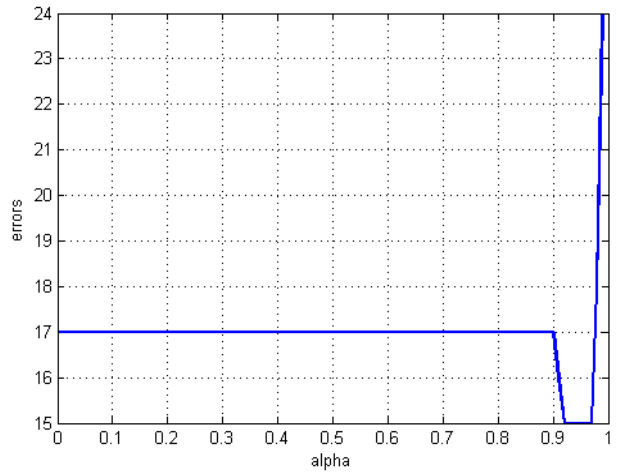


Figure 6. Clustering errors of standardized *Iris data* for Versicolor/Virginica varieties (70/30) with 15 misclassified objects.

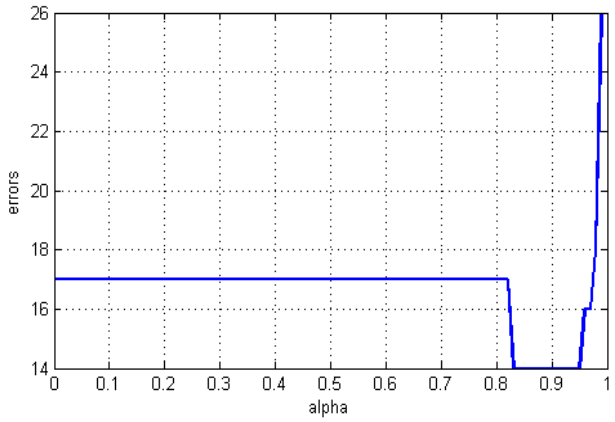


Figure 7. Clustering errors of standardized *Iris data* for Versicolor/Virginica varieties (30/70) with 14 misclassified objects.

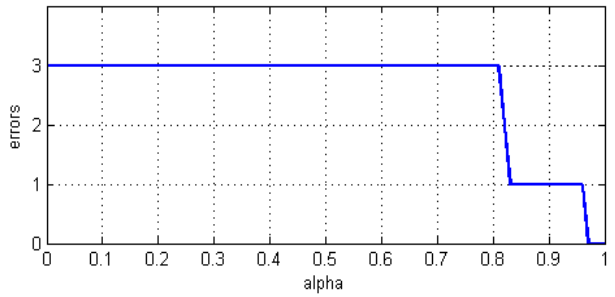


Figure 8: Clustering errors of original *Iris data* for Setosa versus Versicolor/Virginica varieties (50/100, 100/50, 30/120).

We reduce errors to zero (Fig. 8) and correctly separate the smaller 1st class from the bigger one (2nd and 3rd) in the optimal interval $0.97 \le \alpha^* \le 1$ for all initial partitions, i.e. 50/100, 100/50, 30/120. For standardized data, the result contains no errors at all for the whole interval $0 \le \alpha \le 1$ for all initial partitions.

6 Conclusion

The *k*-means procedure is very popular in machine learning and data mining fields. This procedure is very natural and understanding its principles and results is easy. Additionally, this procedure is deeply connected with other ideas, like the EM-algorithm, SOMs, etc.

On the other hand, the use of the bi-partial criterion can improve the classical clustering result. The bi-partial objective function consists of two parts, the first one supporting the best approximation of individual categories, and the second one supporting the appropriate separation among the categories. In the case of the *k*-means algorithm, the bi-partial objective function combines intra-cluster dispersions with the inter-cluster similarity, to be jointly minimized. In dual form, the bi-partial objective function combines cluster concentrations with the inter-cluster dispersion, to be maximized.

In this paper, we investigate the direct form of the bi-partial criterion function. The first part of this criterion provides the classical quality measure of *k*-means clustering, based on distances between objects.

As it is shown in this paper, the bi-partial criterion does not work directly through the standard procedure of the classical *k*-means, since the second part of the criterion cannot be changed within the classical procedure.

Therefore, to improve the clustering quality based on the bi-partial criterion, we develop here the new permutable version of the classical *k*-means algorithm.

As it is shown in this paper, the permutable *k*-means appears to be a new type of clustering procedures.

The permutable *k*-means uses distances and similarities only. Therefore, it does not need to use the feature-based representation of experimental data. To reduce the computational complexity of permutations we can use in further work the optimal iterative techniques.

It is easy to show that in the dual form the bi-partial objective function combines cluster concentrations with the inter-cluster dispersion, to be jointly maximized. The first part of both bi-partial objective functions provides the "standard" quality of clustering based on distances between objects (the classical *k*-means) or similarities between them in dual form (the similarity *k*-means).

As a result, what the algorithm have we built? It is clear, that we have merely shown the principle of developing a class of criteria and corresponding algorithms. As we can see in Figs. 1–7, error lines are not convex functions of α in general. The future study should, then, be oriented at defining conditions for convexity, on the one hand, and developing effective algorithms of extrema finding of the similar functions, on the other.

Acknowledgements

The work of the first author was supported by Russian Foundation for Basic Research under grant 17-07-00319.

References

[1] H. Steinhaus (1956). Sur la division des corps matériels en parties. *Bulletin de l'Academie Polonaise des Sciences* IV (C1.III), 801-804 (in French).

[2] M.I. Shlezinger (1965). Spontaneous discrimination of patterns. In: *Reading Automata*. Naukova Dumka, Kiev (in Russian).

[3] M.I. Shlezinger (1968). The interaction of learning and self-organization in pattern recognition. In: Kibernetika, 4(2), 81-88. http://irtc.org.ua/image/Files/Schles/non-supervised.pdf

[4] A.V. Milen'kii (1975). *Classification of signals in conditions of uncertainty*. Moscow, Soviet Radio (in Russian).

[5] E. Diday et al. (1979). *Optimisation en classification automatique*. INRIA, Domaine de Voluceau, Rocquencourt B.P. 105, 78150 Le Chesnay (in French).

[6] W.S. Torgerson (1958). *Theory and Methods of Scaling*. N.Y., Wiley.

[7] H. P. Friedman and J. Rubin (1967). On Some Invariant Criteria for Grouping Data. In: *J. of the American Statistical Association*, 62(320):1159-1178. https://doi.org/10.1080/01621459.1967.10500923

[8] H. Späth (1983). *Cluster-formation und -analyse: Theorie, FORTRAN-Programme und Beispiele*. R. Oldenbourg-Verlag, München — Wien.

[9] S.A. Aivazyan, et al. (1989). *Applied Statistics. Classification and reduction of dimensionality (Ch. 5. Basic concepts and definitions used in classification without training. 5.4. Classification quality functionals and extremal approach to cluster analysis problems)*. Finansy i statistika, Moscow (in Russian)

[10] H.-J. Mucha, U. Simon, R. Brüggemann (2002). *Model-based Cluster Analysis Applied to Flow Cytometry Data of Phytoplankton*. Tech. Report, Berlin. http://www.wias-berlin.de/techreport/5/ wias_technicalreports_5.pdf

[11] E. Pekalska, R.P.W. Duin (2005). *The Dissimilarity Representation for Pattern Recognition. Foundations and Applications*. W.S. Singapore.

[12] A.W.F. Edwards, L.L. Cavalli-Sforza (1965). A Method for Cluster Analysis. In: *Biometrics*, *21*, 362–375. https://www.jstor.org/stable/ 2528096 ?seq=1#page_scan_tab_contents

[13] Jan W. Owsinski (2012). On the optimal division of an empirical distribution (and some related problems). In: *Przegląd Statystyczny*, *special issue*, 1, 109-122.

[14] Jan W. Owsinski (2013). On dividing an empirical distribution into optimal segments. http://new.sis-statistica.org/wp-content/uploads/ 2013/09/RS12-On-Dividing-an-Empirical-Distribution-into.pdf

[15] Jan W. Owsinski (2011). The bi-partial approach in clustering and ordering: the model and the algorithms. In: *Statistica & Applicazioni*. *Special Issue*, 43–59.

[16] Jan W. Owsinski (1990). On a new naturally indexed quick clustering method with a global objective function. In: *Applied Stochastic Models and Data Analysis*, 6(3), 157-171. https://doi.org/10.1002/asm.3150060303

[17] S.D. Dvoenko (2009). Clustering and separating of a set of members in terms of mutual distances and similarities. In: *Transactions on MLDM*. IBaI Publishing 2, 2 (Oct. 2009), 80-99.

[18] S. Dvoenko (2014). Meanless *k*-means as *k*-meanless clustering with the bi-partial approach. In: *Proc. of 12th Int. Conf. on Pattern Recognition and Image Processing (PRIP'2014).* UIIP NASB, Minsk, Belarus, 50-54.

[19] R.O. Duda, P.E. Hart (1973). *Pattern Classification and Scene Analysis*. N.Y., Wiley.

[20] R.O. Duda, P.E. Hart, D.G.Stork (2000). *Pattern Classification*. Wiley-Interscience New York, NY.

[21] R.A. Fisher (1936). The use of multiple measurements in taxonomic problems. In *Ann. Eugenics*. 7, 2 (Sept. 1936), 179-188. https://doi.org/10.1111/j.1469-1809.1936.tb02137.x

[22] S. D. Dvoenko, D.O. Pshenichny (2016). A recovering of violated metric in machine learning. In: *Proceedings of 8th Int. Symposium on Information and Communication Technology (SoICT'2016)*. ACM NY, 15-21. https://doi.org/10.1145/3011077.3011084

https://doi.org/10.31449/inf.v43i2.2019 Informatica **43** (2019) 263–279 **263**

A CLR Virtual Machine Based Execution Framework for IEC 61131-3 Applications

Salvatore Cavalieri and Marco Scroppo
University of Catania, Department of Electrical Electronic and Computer Engineering (DIEEI), Italy
E-mail: salvatore.cavalieri@unict.it, marcostefano.scroppo@dieei.unict.it

Keywords: IEC61131-3, PLC, CLR VM, real-time industrial applications

Received: November 13, 2017

The increased need of flexibility of automation systems and the increased capabilities of sensors and actuators paired with more capable bus systems, pave the way for the reallocation of IEC 61131-3 applications away from the field level into so-called compute pools. Such compute pools are decentralised with enough compute power for a large number of applications, while providing the required flexibility to quickly adapt to changes of the applications requirements. The paper proposes a framework able to deploy IEC 61131-3 applications to multiple computing platforms based on CLR VM; it uses C# language as intermediate code. The software solution proposed by the authors does not require any modifications of the IEC 61131-3 applications. Current literature does not provide solutions like that here presented; due to the spread current use of C# language in the development of industrial applications, adoption of the proposed solution seems very attractive. The paper will deeply describe the software implementation and will also present an analysis about the capability of the proposed framework to respect real-time constraints of the industrial processes, mainly focusing on the periodic ones.

Povzetek: Prispevek predlaga okvir, ki omogoča uporabo aplikacij IEC 61131-3 za več računalniških platform, ki temeljijo na CLR VM.

1 Introduction

Programmable Logic Controllers (PLCs) are widely used for the control of automation systems. The standard IEC 61131-3 defines the execution model as well as programming languages for such systems [1]. According to IEC 61131-3, software development becomes independent of process mapping and device specific configuration files. Programmers can focus on the algorithm and control development. Device specific knowledge is outsourced into the block library and can be substituted, every time a new target PLC device should be programmed.

During these last years, the need to deploy IEC 61131-3 – based applications addressing multiple target platforms (also different from PLCs, e.g. based on general purpose computing architectures) became more and more urgent for the reason explained in the following. In a common factory automation scenario, actuators and sensors connect to the PLCs via automation buses; traditionally, bus based systems dominated the automation industry. Nowadays, more powerful and flexible automation networks appear and allow the connection of thousands of actuators and sensors to the same network, while still obtaining the required timing performance; interested readers are referred to [2] for a detailed overview.

Those changes in the communication technologies opens possibilities of computation further away from the field level, compared to how it is done in today's automation systems. On the other hand, many sensors and actuators are equipped with small microcontrollers, allowing them to do basic data processing; furthermore, they are able to connect directly to the new bus technologies.

Having basic data processing done at the lowest level (i.e., at the field level directly on sensors and actuators) and a connection to capable networks, allows the reallocation of applications away from the field level into so-called compute pools [3] [4]. Such compute pools are decentralised with enough compute power for a large number of applications, while providing the required flexibility to quickly adapt to changes of the applications requirements. This has several benefits. Changing control applications becomes merely a problem of reconfiguration in the compute pool. Costs will decrease as well as the need for physical PLCs will be decreased; in this new scenario, the PLC is migrated to the computing pool and can be also realised by general purpose computer architectures (e.g., a server or a cluster of servers).

Several requirements must be satisfied in order to reach this goal. The first one is the guarantee of the total compliance with IEC 61131-3; it is clear that moving an IEC 61131-3 application on a compute pool must be realised without any changes in the same application. Then, respect of real-time constraints of the IEC 61131-3 control applications must occur when migrating the application to the compute pool. Particular cares must be reserved for real-time applications requiring periodic executions; in these cases, executions of each process must occur exactly with the requested period.

Current literature presents several solutions in the direction just pointed out. For example, in [5], the use of the Java Virtual Machine (JVM) to deploy IEC 61131-3 applications to embedded devices has been proposed. In [6] different levels of an automation process are proposed and a cloud-based solution is presented. An example of virtual PLC is given by [7], where PLC systems are executed as applications within a legacy OS. Finally, [3] [4] present the use of a multi-core high performance computing architecture to realise the compute pool.

Based on what said, the aim of the paper is to contribute to find solutions able to deploy IEC 61131-3 – based applications to multiple computing platforms, mainly focusing on general purpose computer systems (e.g., single server or cluster of servers with common operating systems).

In the last years, the domain of factory and process automation features intense usage of languages (e.g., Java, C#) based on Virtual Machines (VMs), like JVM or Common Language Runtime (CLR) VM, as pointed out in [8]. A VM has some clear benefits: portability, security, Just-in-time Compiler to boost performance in time, ease of development in conjunction with a garbage collector, multi-threading and others; reader may refers to [8] in order to achieve a complete survey on this subject. On this basis, the authors believe that one of right possible directions to reach the aim of the paper is that to adopt languages supporting VMs for the deployment of IEC 61131-3 application on common computing platforms. This idea was already pointed out in [5], which proposed the deployment of IEC 61131-3 applications using Java bytecode as a common intermediate format, although the deployment was limited to the embedded devices.

To the best of authors' knowledge, literature does not provide solutions aimed to deploy IEC 61131-3 applications using languages based on CLR VM, like C# language, as intermediate code. Due to the spread current use of C# language in the development of industrial applications, adoption of C# language based on CLR VM to deploy IEC 61131-3 applications on computing platforms seems attractive. Typical candidate platforms are those based on general purpose computing architecture (on which CLR VM allows the use of common operating systems like Linux and Windows), but also all the embedded systems supporting a CLR VM may be considered.

For all the previous reasons, the authors propose a novel software solution made up by different features. First of all, it is able to translate a generic IEC 61131-3 application into C# code which could be executed in a general purpose CLR VM-based platform. Furthermore, the solution here proposed includes the definition of a framework which is able to realise the deployment of IEC 61131-3 applications on a compute poll based on CLR VM, using the C# code as intermediate one. The proposed solution does not require any modifications to the native IEC 61131-3 applications; all additional overhead is handled by the framework here defined. Applications in the automation domain often come with real-time requirements; in order to better allow their respect, the proposed framework features the use of a CLR VM on the top a real-time operating system who is in charge to schedule time-critical applications. Finally, the last feature of the proposed software solution is the use of open source environments; in particular, the implementation presented in the paper is based on the use of MONO [9] as CLR VM and a real-time Xenomai co-kernel [10] alongside a common Linux kernel. Choice of real-time Xenomai co-kernel has been based on a performance evaluation whose main results will be shown in the paper.

The paper will deeply describe the proposed software solution pointing out the main features. Then, results of a performance evaluation aimed to analyse its capability to respect real-time constraints of typical periodic industrial applications will be presented and discussed.

Some of the very preliminary results achieved at the first stages of the research carried out by the authors have been subject of publication [11][12]. This paper presents the full results of the research and gives a very deep analysis of the implementation realised and of the outcomes achieved by the authors.

2 PLC and IEC 61131-3 main features

The main feature of a PLC is the use of cyclic loops for the execution of programs; each loop is called Program Scan. As shown by Figure 1, in each Program Scan, PLC reads the real inputs copying them into an internal memory area called I. Then, PLC execute one or more programs and finally it updates all the output values found in the memory area Q into the real output devices. The program/s executed inside the Program Scan may use internal memory, called area M, for the temporary storage of information. Each program may have a task associated, whose main aim is to control the execution of the program itself; the most common task is the periodic one, triggering the program in such a way it should be iterated after a certain fixed time interval (i.e., the period of the task). Tasks may feature priorities and the execution of a program inside the Program Scan may be pre-empted by another program whose task associated features a higher priority. Generally, reading and writing operations shown by Figure 1, cannot be interrupted by other programs.

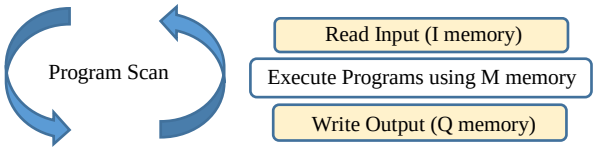


Figure 1: Program Scan.

IEC 61131-3 is the vendor independent standardised programming language for factory automation [1]. IEC 61131-3 allows users to write programs, choosing among five programming languages: Ladder Diagram (LD), Sequential Function Charts (SFC), Function Block Diagram (FBD), Structured Text (ST), Instruction List (IL).

IEC 61131-3 software development is independent of process mapping and device specific configuration files. IEC 61131-3 application is typically deployed on a PLC

device, whose specific knowledge is outsourced into the block library and can be substituted, every time a new target PLC device should be programmed.

In order to make the program itself independent on the device on which it must be deployed, the structure of an application written according to IEC 61131-3 standard is made up by at least two separate sections: Program and Configuration.

An IEC61131-3 Program provides a large re-usable software component. It is defined by a program type definition (starting with PROGRAM and ending with END_PROGRAM keywords) which has input, output and internal variables declaration and a body which contains software describing the behaviour of the program itself. As said before, one of the five languages specified above can be used to describe the program.

A Configuration defines the software for a complete PLC and will always include at least one but, in some cases, many Resources. A Configuration is specific to a particular type of PLC product and the arrangement of the relevant PLC hardware. It can be used to create software for another PLC if the hardware is identical. Configuration is introduced with the keyword CONFIGURATION and terminate with END_CONFIGURATION keyword.

A Resource describes a processing facility inside a PLC type that is able to execute an IEC 61131-3 Program. A Resource is defined within the Configuration using the keyword RESOURCE followed by an identifier and the type of the processor on which the Resource will be loaded (keyword ON is used before the type of processor). In the real cases, for each type of PLC a detailed description of the hardware and software features is associated (e.g., firmware version, number of inputs/outputs, internal memory). The resource definition contains a list of Global Variable declarations and task definitions that can be assigned to Programs. It terminates with the keyword END_RESOURCE.

As said, a task may be associated to a Program controlling its execution. Task may be single or periodic; in this last case, a period is specified for its execution. A priority value is assigned to each task in order to determine the order of their executions. Tasks are defined inside the RESOURCE section, as said. A Task declaration is introduced using the keyword TASK followed by the task identifier and optional values for the following parameters: SINGLE (if the task is not periodic), INTERVAL (period, if the task is periodic), PRIORITY (task priority value). After its definition, Task is associated to an instance of a Program using the keywords WITH.

Figure 2 shows a very simple IEC 61131-3 application using ST language; this same example will be used in the remainder of this paper.

As it can be seen, the simple IEC 61131-3 application is made up by only one PROGRAM section called MyProgram, which contains the definition of the local (i.e., VAR) and external (or global, i.e., VAR_EXTERNAL) variables. Furthermore, it contains the algorithm coded into ST language; it is made up by only two assignments, the first is relevant to a global variable (StepSizeVar) and the other to the local variable ComputedResult. CONFIGURATION section is named MyConfiguration and is made up by only one resource called MyResource; the type of PLC chosen for the execution of the software has been called PLC1 in the example. The RESOURCE section contains the declaration of the global variables used by the program (StepSizeVar, MaxValueVar and MinValueVar); the declaration includes the mapping of these variables into the internal PLC memory (called M memory, as shown by Figure 1) at the addresses 200, 204 and 208, respectively. RESOURCE section also contains the definition of two periodic Tasks, named MyTask1 and MyTask2; they differ for the period and the priority values. According to IEC 61131-3 standard, low priority values refer to high priority tasks.

```
PROGRAM MyProgram
  VAR
    ComputedResult : REAL;
  END_VAR
  VAR_EXTERNAL
    StepSizeVar : REAL;
    MaxValueVar : REAL;
    MinValueVar : REAL;
  END_VAR

  StepSizeVar := MaxValueVar-MinValueVar;
  ComputedResult := StepSizeVar;
END_PROGRAM

CONFIGURATION MyConfiguration
  RESOURCE MyResource ON PLC1
    VAR_GLOBAL
      StepSizeVar AT  %MD200 : REAL;
      MaxValueVar AT  %MD204 : REAL;
      MinValueVar AT  %MD208 : REAL;
    END_VAR
    TASK MyTask1(INTERVAL := T#100ms, PRIORITY := 1);
    TASK MyTask2(INTERVAL := T#150ms, PRIORITY := 2);
    PROGRAM MyInstance1 WITH MyTask1: MyProgram;
    PROGRAM MyInstance2 WITH MyTask2: MyProgram;
  END_RESOURCE
END_CONFIGURATION
```

Figure 2: IEC 61131-3 ST-based Program relevant to a simple algorithm.

Finally, two instances of the MyProgram Program are defined into the RESOURCE section; they are called MyInstance1 and MyInstance2 and are featured by the tasks MyTask1 and MyTask2 associated, respectively, controlling their execution.

3 Overview of Xenomai

The Xenomai project has the aim of providing real-time support for user applications [10].

It is a real-time development framework that cooperate with the Linux kernel in order to make possible the real-time management of tasks on any hardware with a Linux-based operating system. The project has a strong focus on embedded systems, although Xenomai can also be used over common desktop and server architectures. Xenomai has two modes of use:

- as co-kernel extension for a patched version of the original Linux kernel. This is the solution adopted in the paper.
- as libraries for native Linux kernel (features added in the version 3.0 in 2015)

In both modes, it is possible to use the Native Xenomai C language-based API functions to run real-time tasks [13].

To create and run a simple real-time task, three steps are needed:

1. Creation of the task and setting of its properties (e.g., priority) using the *rt_create_task()* API function. If the task is periodic, the *rt_task_set_periodic()* API function will be also used, in order to allow Xenomai to have knowledge of the task periodicity.
2. Creation of a C language-based procedure that the task will perform during its execution. If the task is not-periodic there are not particular constraints for the structure of this procedure. But, for periodic task, the C-language-based procedure must be featured by an infinite while loop inside which the *rt_task_wait_period()* API function must be present; it allows the procedure to be stopped after its conclusion and to resume its regular running after the task period previously set by *rt_task_set_periodic()* API function. Figure 3 shows how the C language-based procedure (called *task_function()* in the figure), must be written in the case of periodic task.
3. Association of the C language-based procedure to the Xenomai task created at the step 1 and starting the task using the *rt_task_start()* API function; in particular, the entry point of the C language-based procedure is passed to this function.

```
void task_function(){
  while (true) {
    //Code in C Language to be executed
    rt_task_wait_period();
  }
}
```

Figure 3: Structure of a C language-based procedure to be assigned to a periodic task.

4 Running C# programs over Xenomai

This section plays a strategic role inside the paper. Introduction pointed out that the aim of the paper is that to propose a framework able to translate an IEC61131-3 application into C# program; furthermore, the framework is able to allow real-time execution of the C# program using a Xenomai co-kernel.

Before the framework defined may be presented, this section has to point out how a C# program may be executed over a Xenomai real-time co-kernel and, most important, if execution of a C# program may actually exploit the real-time features of the Xenomai co-kernel.

The software solution presented in Figure 4 has been defined to allow execution of a C# program over Xenomai co-kernel. It is based on the use of a MONO Virtual Machine running on the top of a Linux OS with Xenomai co-kernel [9].

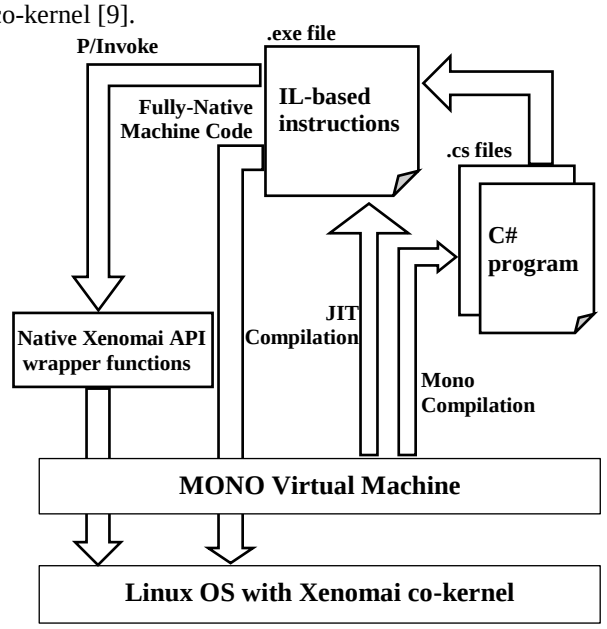


Figure 4: Software solution adopted for the Xenomai-based C# program execution.

As said in the previous section, Xenomai offers a set of native API functions to realise real-time mechanisms [13]; these API functions are callable inside a program written in C language. In order to allow a C# language-based program to call a particular real-time Xenomai API function, suitable wrapper functions had to be defined. Each wrapper function maps a C# function call to a particular Xenomai API function; this happens through the definition of a C function containing the call to the native Xenomai API. All the wrapper functions are pre-compiled and realise a run-time library named in the figure "Native Xenomai API wrapper functions". One of the following subsections will give an overview of the wrapper functions defined in the research here present.

C# programs (written inside .cs files) are compiled by Mono producing .exe files containing Intermediate Language (IL)-based instructions. At run-time, for each IL-based executable file, Just-In-Time (JIT) compilation is realised producing binary code. Native machine code is executed directly by Linux/Xenomai kernel. In order to execute the Native Xenomai API wrapper functions, P/Invoke procedure allows to call the unmanaged code produced by the compilation of the "Native Xenomai API wrapper functions". The unmanaged code is mapped on the Xenomai real-time system calls as the wrapper functions contains the calls to Xenomai API, as said before.

4.1 Native Xenomai API wrapper functions

The Xenomai wrapper functions defined according to the goal of the research here presented, are detailed in the following.

rt_task_create_wrap(). It calls the native Xenomai API *rt_task_create()* belonging to the Task Management Services. This service creates a new real-time task which is left in an innocuous state until it is actually started by the Xenomai service *rt_task_start()*. Among the parameters passed to the native Xenomai API function, the wrapper function specifies the priority of the new task in the range from [0 .. 99], where 0 is the lowest effective priority.

rt_task_set_periodic_wrap(). This wrapper function calls the native Xenomai API *rt_task_set_periodic()*, which makes a real-time task periodic, by programing its first release point and its period in the processor time line.

rt_task_start_wrap(). It allows to start execution of a Xenomai task that has been previously created. This wrapper calls the native Xenomai API *rt_task_start()*, which releases the target task from the dormant state. Among parameters passed to the native Xenomai API *rt_task_start()*, the wrapper function specifies the address of the procedure to be execute when the task is running.

rt_task_wait_period_wrap(). It makes the Xenomai task wait for the next periodic release point in the processor time line. A rescheduling of the task always occurs, unless the current release point has already been reached. In the latter case, the current task immediately returns from this service without being delayed.

rt_task_sleep_wrap(). It suspends the calling process for a certain amount of milliseconds passed as argument. This function calls the Xenomai *rt_task_sleep()* API function.

rt_sem_create_wrap(). It allows to create a Xenomai real-time semaphore, fully handled by Xenomai itself. It wraps the Native Xenomai API *rt_sem_create()*.

rt_sem_p_wrap(). It is used to acquire the semaphore or put on hold his release if already occupied. It is directly mapped to the native Xenomai API *rt_sem_p()*, which acquires a semaphore unit. If the semaphore value is greater than zero, it is decremented by one and the service immediately returns to the caller. Otherwise, the caller is blocked until the semaphore is either signalled or destroyed, unless a non-blocking operation has been required. Among the parameters passed to the native API function, there is the descriptor address of the affected semaphore.

rt_sem_delete_wrap(). It directly maps to the Xenomai API *rt_sem_delete()*, which destroys a semaphore and release all the tasks currently pending on it.

rt_sem_v_wrap(). This function allows to call the native Xenomai API *rt_sem_v()* inside a C# program. This service releases a semaphore unit; the parameters passed to the native Xenomai API function, specify the descriptor address of the affected semaphore.

4.2 Analysis of the real-time capabilities

Evaluation of the capability of the software solution shown by Figure 4 to respect real-time constraints of a generic C# program was considered of primary importance. Real-time feature has been evaluated observing the capability of a particular C# program to promptly react to a rising event; real-time capabilities have been measured checking that all rising events have been caught with the lowest delay.

The analysis has been carried out on an embedded system. Choice of an embedded system compared with a general purpose computing device like a server or a personal computer, had the advantage to allow an easier use of an oscilloscope to analyse the output produced upon the occurrence of an event realised by a digital input.

The embedded system is made up by a MPC8309 PowerQUICC processor [14] running at 333Mhz with 256MB RAM, a microcontroller PIC32MX, and two Serial Peripheral Interface (SPI) acquisition boards (each featuring 4 channels at 16 bit, sampling at 125 μs).

Figure 5 shows the general architecture of the embedded system. The PIC32MX receives the samples from SPI, forwarding them to MPC8309 through the MISO (Master data In/Slave data Out) bus. The SYNC line is used by MPC8309 processor to advise the PIC32MX that it is ready to start the acquisition of samples. After reception of this synchronization signal, the PIC32MX will start transmission of samples received from SPI, synchronizing them with a DRDY signal with a duration of 15 μs, sent with a period of 5ms.

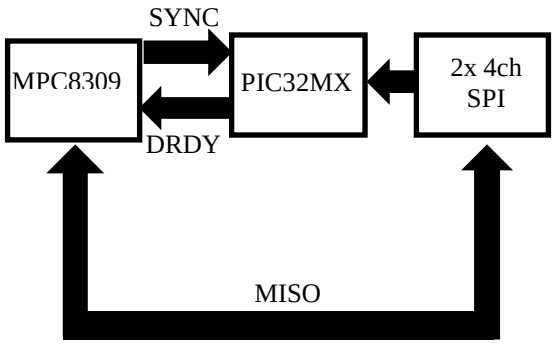


Figure 5: Architecture of the Embedded System.

A Linux Kernel 3.8.13 with co-kernel Xenomai version 2.6.4 has been installed in the MPC8309 embedded system. A Mono framework version 3.2.6 has been also installed.

A huge set of tests has been performed in order to explore the capability featured by the Xenomai-based software solution shown by Figure 4 to meet real-time constrains of a C# program running inside the MPC8309. A C# program realising the flow-chart described by Figure 6 has been defined. It reacts to the DRDY activation; on the receipt of this signal, the C# program set a particular General Purpose I/O, the GPIO #1, and maintains the value ON for 1 ms; this is achieved using the rt_task_sleep_wrap() describe before. After the sleep interval has passed, the GPIO #1 is put OFF. It is important to recall that DRDY is activated each 5 ms, as said at the beginning of this section.

Two other C# programs have been defined; they both realise the flow chart shown by Figure 7. Each program waits for the setting of the GPIO #1 (by the program shown by Figure 6); when this occurs, the GPIO #2 is set and suddenly reset. Then, each program calls a sleep function with a duration of 2 ms; one C# program realises

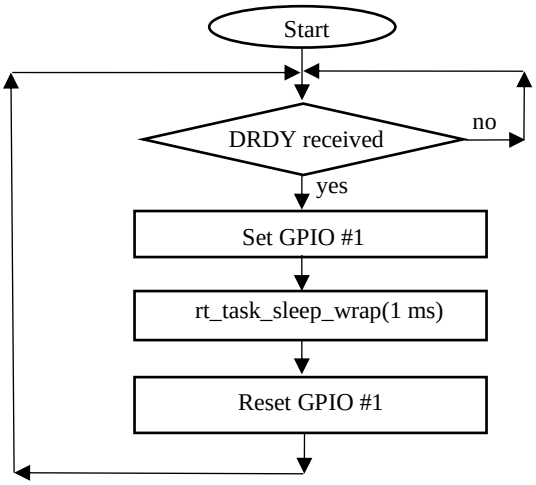


Figure 6: C# language-based program acting on receipt of DRDY and setting GPIO #1.

this call though the function rt_task_sleep_wrap(), whilst the other C# program uses the C# Thread.sleep(). The only difference between the two C# programs is that the first one foresees the real-time management of the sleep by Xenomai co-kernel, whilst the other one does not exploit the real-time features of Xenomai co-kernel, as the management of the sleep of the process is given to Linux OS.

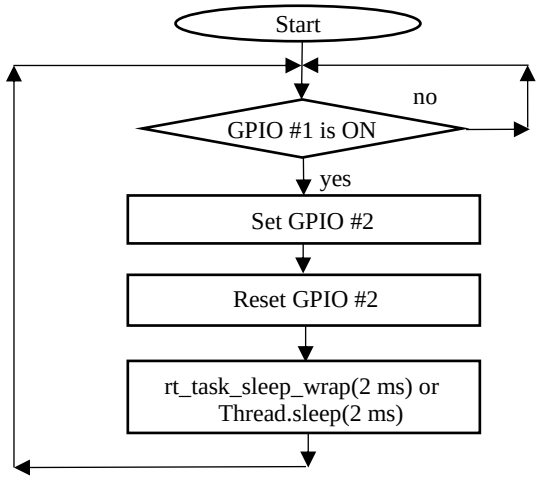


Figure 7: C# language-based programs acting on receipt of GPIO #1 signal.

Figure 8 points out the behaviour of the C# program described by Figure 7 using the C# Thread.sleep(). The signal number 1 (on the top) refers to the setting of the GPIO #1 done by the C# program shown by Figure 6; it easy to verify that the period of this signal is 5ms as it is synchronised with DRDY. Signal number 2 (on the bottom) refers to the GPIO #2 and is set/reset by the C# program shown by Figure 7 when C# Thread.sleep() is used.

Figure 9 refers to the C# program described by Figure 7 when rt_task_sleep_wrap() is used. Again, the signal number 1 (on the top) refers to the setting of the GPIO #1 done by the first C# program shown by Figure 6. Signal number 2 (on the bottom) refers to the GPIO #2 and is

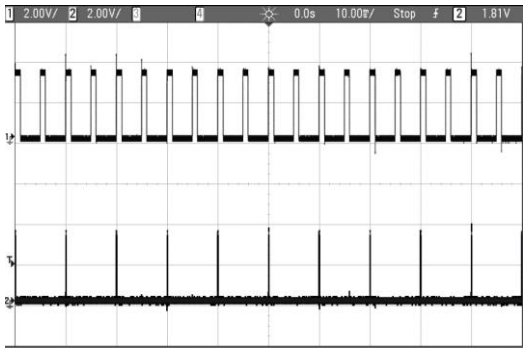


Figure 8: Performance achieved using C# Thread.sleep().

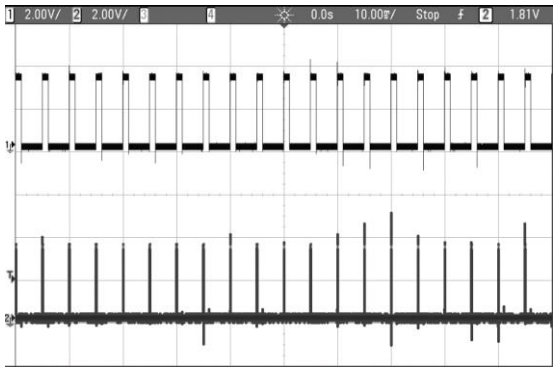


Figure 9: Performance using rt_task_sleep_wrap().

set/reset by the C# program shown by Figure 7 when rt_task_sleep_wrap() is used.

Comparison of the two Figures 8 and 9 points out that the C# Thread.sleep is not able to wake-up the process in time to catch each single setting/resetting of the GPIO #2. The use of Xenomai API allows total respect of real-time requirements here presented.

A huge set of other tests not shown here for space limitation, allowed to reach the same conclusions just pointed out: use of Native API Xenomai wrapper functions here defined according to the software solution shown by Figure 4, allows to fully exploit the real-time capabilities offered by Xenomai co-kernel and allows the respect of time-critical constraints. For these reasons, the software solution presented in Figure 4 will be used in the remainder of this paper.

5 Overview of the proposed framework

As pointed out in the Introduction, the main aim of this paper is that to present a framework based on the use of CLR-based virtual machine, able to deploy an IEC 61131-3 application on a computing system supporting CLR VM. The framework is made up by the two modules shown in Figure 10 with the grey coloured backgrounds: *C# Translator* and *PLC Framework*.

C# Translator is in charge to process a generic IEC 61131-3 application in order to produce C# language-based classes (contained in .cs files). These classes include all the information relevant to the different sections of the IEC 61131-3 application (e.g., program, configuration,

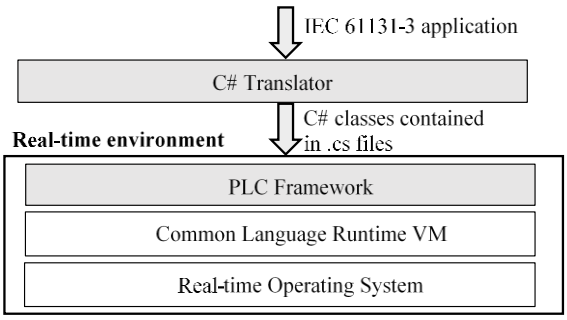


Figure 10: Architecture of the Framework proposed.

resource, including task definitions with periods and priorities). These classes will be used by the real-time environment shown by Figure 10, as explained in the remainder of this section.

A very important feature to be pointed out is that the C# Translator may be used stand-alone, i.e. not linked to the real-time environment shown by Figure 10. C# Translator allows to achieve a C# code that in principle may be executed by a common CLR-based platform, using a common C# language-based integrated development environment (IDE).

Introduction pointed out a main trend in automation, relevant to the reallocation of applications away from the field level into so-called compute pools. Real-time environment shown by Figure 10 is strictly linked to the concept of compute tools, as it will be shown better later. About C# Translator, no constraints exist for its installation; it may be installed in a compute poll or may be installed close to the system where the IEC 61131-3 development environment is running.

PLC Framework has the aim to realise a real-time environment implementing the same behaviour of a PLC. Mainly it allows the realisation of the Program Scan loop execution and allows the real-time scheduling of the tasks associated to the IEC 61131-3 Programs and the relevant executions.

On the basis of what said in the Introduction, and on the basis of the software solution shown by Figure 4, implementation of *PLC Framework* requires the presence of a CLR-based VM running on a real-time operating system, as shown by Figure 10. The CLR VM has been realised by MONO [9]. For the real-time operating system it has been assumed to adopt a Linux OS and a Xenomai co-kernel [10]; choice of Xenomai has been supported by the analysis of the relevant performances, shown in the previous section.

The remainder of this section will focus on the main technical details of the architecture shown by Figure 10.

5.1 C# translator

Given an IEC 61131-3 application, the main aim of the *C# Translator* module is to create a .cs file containing classes that can be used to execute a C# program which exactly behaves as the original IEC application. In the following, a detailed description of the procedure adopted by the *C# Translator* to map an IEC 61131-3 application to the C# classes, will be given.

Figure 11 shows the structure of a typical .cs file produced by *C# Translator*.

```
class ProgramName {
      public ProgramName(){}
      public void IECRoutine() {
      }
}

class ConfigurationName {
   class ResourceName {
         class GlobalDeclaration{
         }
         class TaskName {
               ProgramName instanceName;
               public TaskName(){
                  instanceName = new ProgramName();
               }
         }
   }
}

class ExternalProgramName {
}
```

Figure 11: File structure produced by *C# Translator*.

The .cs file produced contains three main classes: ProgramName, ConfigurationName and ExternalProgramName.

The class ProgramName is the translation of the PROGRAM section in the IEC61131-3 application; it mainly includes a method called IECRoutine() that translates the software describing the IEC61131-3 Program.

The class ConfigurationName is relevant to the CONFIGURATION section in the IEC61131-3 application and is made up by the class ResourceName, related to the RESOURCE section. Class ResourceName may include the declaration of Global Variables; in this case the class GlobalDeclaration is present. The other class contained in ResourceName, class TaskName, represents a single task (so many classes may be present if several IEC tasks have been defined); inside this class, the program to be associated with the task is specified.

Finally, class ExternalprogramName is a singleton class needed for the usage of the External Variables defined in the IEC61131-3 Program and declared as variables of the GlobalDeclaration class. This class has to contain a single instance of the GlobalDeclaration class, that IECRoutine can use so sharing the variables among all its instances. Furthermore, the class must implement suitable mechanisms able to guarantee that concurrent access to the shared variables must occur through use of critical section.

In order to clarify better the structure of the .cs file produced by the C# Translator, the IEC 61131-3 application shown by Figure 2 will be considered. After the processing operated by C# Translator, the .cs file shown by Figure 12 is achieved.

As it can be seen, the .cs file produced contains the three main classes: MyProgram, MyConfiguration and ExternalMyProgram.

The class MyProgram is the translation of the PROGRAM MyProgram section shown by Figure 2; it contains the local variable of the program, declared as

variables of the class (i.e., ComputedResult), and the method IECRoutine() that translates the algorithm of the PROGRAM MyProgram. Just like the IEC 61131-3 application, the C# algorithm uses global variables, as it will be explained in the following.

The class MyConfiguration refers to the IEC 61131-3 CONFIGURATION MyConfiguration section. As it occurs for the IEC 61131-3 application, it is made up by the class MyResource related to the RESOURCE section.

The class MyResource contains three classes: GlobalDeclaration, MyTask1 and MyTask2. Class GlobalDeclaration allows the definition of the global variables of the original IEC61131-3 application (i.e., StepSizeVar, MaxValueVar and MinValueVar). The classes MyTask1 and MyTask2 represent the IEC 61131-3 Tasks: inside these classes there are the variables that indicate the properties of the tasks as period and priority, a variable that indicates the program associated with the task (MyProgram) and a constructor with the aim of initializing these variables.

The class ExternalMyProgram contains a variable of type GlobalDeclaration (i.e., Gd), representing the External Variable of IEC61131-3 Program. This variable is used by the IECRoutine(), as shown by Figure 12. The ExternalMyProgram class implements a critical section using the lock mechanism, allowing a safe and unique instantiation of the single class instance and the safe concurrent access to it.

Again, it is important to point out that the classes produced by C# Translator could be directly used on a common CLR-based platform. They only requires a C# program using them; for example, execution of the IECRoutine() may be achieved through one or more instances of the class MyProgram.

In principle, implementation of *C# Translator* can be realised using whatever technology and language. In this work, the authors chosen to implement this module in C# as a CLR VM-based application. The implementation has been based on the use of the GOLD Parser system [15], by means of which parse tables have been created. In particular, its Builder component has been used to read a source grammar written in the GOLD Meta-Language and to produce the parse tables needed by the *C# Translator*.

5.2 PLC framework

During the design phase, it has been assumed that the *PLC Framework* had to comply with the following assumptions.

For each IEC 61131-3 periodic task, a Xenomai real-time task is created with the same period; priority value of the original IEC 61131-3 task is converted into the range 1 to 99, where 99 is given to the IEC 61131-3 tasks with the highest priority. It is important to recall that task priorities in Xenomai ranges from 0 to 99; as explained in the following, the value 0 has been reserved for a special purpose.

The native scheduling mechanism based on pre-emption adopted by Xenomai has been left unchanged; this means that execution of a Xenomai task is suspended when a higher priority Xenomai task has to be performed.

```
class MyProgram {
    double ComputedResult;
    public MyProgram(){}
    public void IECRoutine() {
      ExternalMyProgram.Gd.StepSizeVar=
                ExternalMyProgram.Gd.MaxValueVar-
                ExternalMyProgram.Gd.MinValueVar;
      ComputedResult = ExternalMyProgram.Gd.StepSizeVar;
    }
}

class MyConfiguration {
  class MyResource {
      class GlobalDeclaration{
          double StepSizeVar;
          double MaxValueVar;
          double MinValueVar;
      }
      class MyTask1 {
          double period;
          int priority;
          MyProgram MyInstance1;
          public MyTask1(){
            MyInstance1 = new MyProgram();
            period=100;
            priority=1;
          }
      }
      class MyTask2 {
          double period;
          int priority;
          MyProgram MyInstance2;
          public MyTask2(){
            MyInstance2 = new MyProgram();
            period=150;
            priority=2;
          }
      }
  }
}

class ExternalMyProgram {
    private static ExternalMyProgram instance = null;
    private static readonly object padlock = new object();
    private GlobalDeclaration Gd = new GlobalDeclaration();
    ExternalMyProgram() {}
    public static ExternalMyProgram Instance {
      get {
        lock (padlock) {
          if (instance == null) {
            instance = new ExternalMyProgram();
          }
          return instance;
        }
      }
    }
```

Figure 12: .cs file produced by *C# Translator* on the IEC 61131-3 application of Figure 2.

Program Scan loop has been realised through a Xenomai real-time task with the lowest priority, i.e. 0. This means that all the other tasks (to which priority values ranging from 1 to 99 have been assigned) can act pre-emption on the Program Scan task. This choice has been made in order to allow execution of a very urgent task, delaying the program scan loop.

The reading and writing operations shown by Figure 1, have been implemented in atomic way, in the sense that during their execution they cannot be interrupted by no other tasks. In order to make atomic the Program Scan execution, a semaphore-based mechanism has been

implemented. In particular, a Xenomai semaphore is created and locked when the Xenomai task implementing the Program Scan is started. The semaphore is deleted each time the Xenomai Program Scan task ends. All the other tasks, even if featuring higher priority cannot interrupt the Program Scan task if the Xenomai semaphore is locked.

Association of a C# program to a periodic Xenomai task has been realised according to the procedure described in Section 3. In particular, for each Xenomai task an instance of the task_function() method shown by Figure 13 is associated through the *rt_task_start_wrap()*.

```
class name {
// local variables
    void task_function () {
// local variables

        while (true) {
// local variables
            ScanProgram() or IEC61131Program();
            rt_task_wait_period_wrap (null);
        }
    }
}
```

Figure 13: task_function() method whose instance is associated to each Xenomai task.

Each task_function() method is made up by a while(true) cycle, inside which a particular C# code is defined; it may be the *ScanProgram()* or the *IEC61131Program()*, both described in the following. Then, the call to the wrapper function *rt_task_wait_period_wrap()* described before in this paper, is achieved. As said before, it forces the Xenomai task associated to the instance of task_function() to wait for the next periodic release point in the processor time line. Figure 13 points out that the local variables (if present) used by the *ScanProgram()* or by *IEC61131Program()*, could be defined at the three different scopes shown by the same figure; the definition of the proper scope will be discussed later in this paper. *ScanProgram()* is a C# program in charge to emulate the typical Program Scan of a PLC. It is shown by Figure 14, by means of a flow-chart graphical representation.

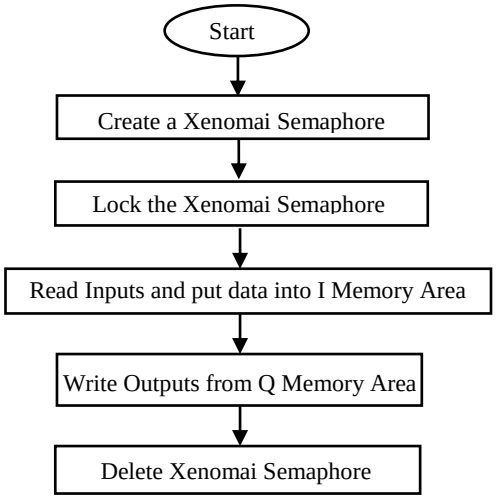


Figure 14: *ScanProgram()*.

At the beginning of the *ScanProgram()*, a semaphore is created by *rt_sem_create_wrap()*. Then, the C# program calls the *rt_sem_p_wrap()* in order to lock it. In this way, the *ScanProgram()* cannot be interrupted from this moment on; reading and writing operations are executed without pre-emption. Reading operations involve I memory, whilst writing operations are relevant to the Q memory (see Figure 1). When they are completed, the semaphore is deleted, by *rt_sem_delete_wrap()*; the relevant waiting queue on the same semaphore is deleted, so all the other task pending on it are released. These tasks may be scheduled for their execution by Xenomai co-kernel.

Figure 15 shows the algorithm implemented inside the *IEC61131Program()*. As it can be seen, it executes a program called IECRoutine(); during the description of the *C# Translator* module, it has been said that among the classes produced for each IEC 61131-3 application there is the class named ProgramName. As shown in Figure 11, this class contains a public void IECRoutine(). The program IECRoutine() executed inside *IEC61131Program* is made up by the same C# code extracted by the public void IECRoutine() contained in the class ProgramName. In the following, the extraction operation performed by the *PLC Framework* will be pointed out.

At the beginning, the *IEC61131Program()* checks the existence of Xenomai semaphore (by using the *rt_sem_p_wrap()*). As said before, this semaphore is created by the *ScanProgram()* and is deleted by the same routine when no more needed. If the *IEC61131Program()* does not find the semaphore, it runs the IECRoutine().

If semaphore exists, the *IEC61131Program()* must check if it is locked (e.g., by *ScanProgram()*). The check is again done using the *rt_sem_p_wrap()*, which locks the semaphore if it is found unlocked; this happens for example when the semaphore has been created by *ScanProgram()* but it was not already locked by it. In this case, the *IEC61131Program()* must suddenly unlock the semaphore (by using *rt_sem_v_wrap()*). This is needed as this task may be pre-empted by a higher priority task which must find the semaphore unlocked, otherwise it cannot be executed. Once the semaphore is unlocked, the

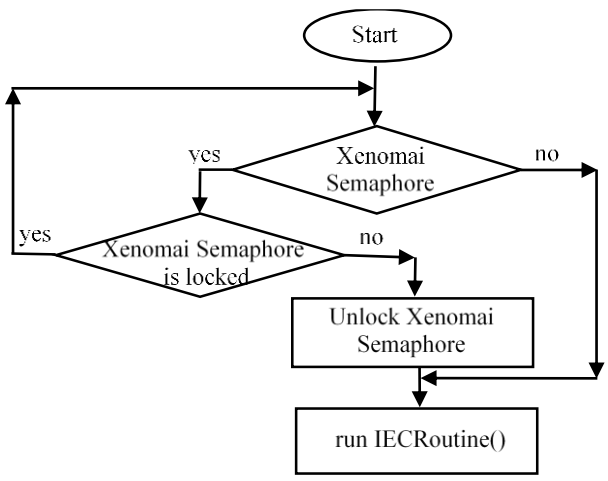


Figure 15: *IEC61131Program()*.

IEC61131Program() executes the IECRoutine(). Figure 15 points out that if the *IEC61131Program()* finds the semaphore locked, it will wait until it is deleted (by *ScanProgram*) or it is unlocked (e.g., by another Xenomai task).

On the basis of what said until know, for each IEC61131-3 application received by the *C# Translator* (in terms of C# classes contained by the .cs files already described), the *PLC Framework* has to performs two main activities.

The first is to produce the task_function() methods shown by Figure 13. One and only one method must contain the *ScanProgram()*, shown by Figure 14; each of the other task_function() methods must contain a *IEC61131Program()*, described by Figure 15.

The second important activity performed by *PLC Framework* is to associate a Xenomai task to each instance of task_function() method, and then activate them so they can be executed by Xenomai co-kernel.

These two activities are performed by two main modules running inside the *PLC Framework*: *ProgramsCreator* and *TasksCreator*. *ProgramsCreator* is in charge to produce the task_function() method on the basis of the C# classes received from the *C# Translator*. *TasksCreator* creates and activates the Xenomai tasks associated to these methods. Figure 16 shows them.

Figure 17 gives an overview of the main activities carried out by the *ProgramsCreator* on the reception of a .cs files produced by *C# Translator*. It analyses class ProgramName (see Figure 11) here contained. From this class, it extracts the class variables and the C# code contained in the IECRoutine() method; this method is placed into the *IEC61131Program()* as shown by Figure 15. Finally, the *ProgramsCreator* creates the class containing the task_function() method shown by Figure 13, placing the *IEC61131Program()* and placing the variables extracted as said before in one of the scopes shown by the same figure. Choice of the right scope will be discussed later in this paper, as said before. When no more ProgramName classes received from *C# Translator* are present, the *ProgramsCreator* will produce the class containing the task_function() method with the *ScanProgram()*, as shown by Figure 13.

The *TasksCreator* module has the main goal to create and run Xenomai tasks, starting from the task_function()

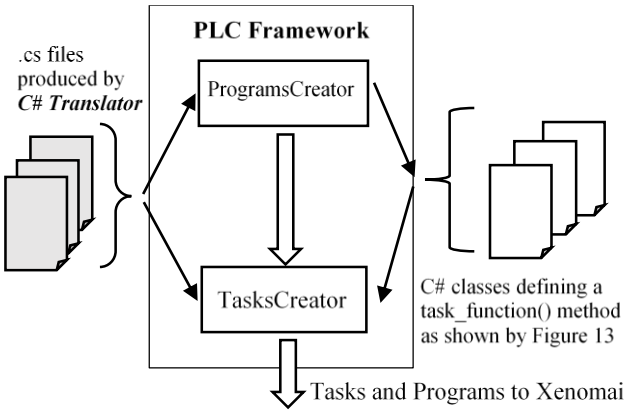


Figure 16: PLC Framework main components.

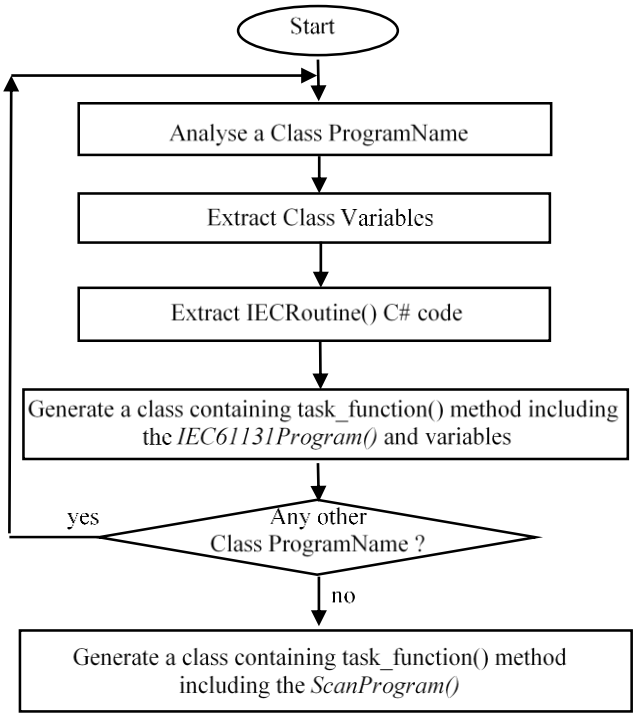


Figure 17: ProgramsCreator.

methods created by the *ProgramsCreator* module. Figure 18 shows the details of the algorithm implemented by this module.

At the beginning, the *TasksCreator* analyses each .cs file received from *C# Translator*. In particular, it focuses on the classes TaskName shown by Figure 11, extracting information (i.e., period and priority) of the entire set of IEC 61131-3 tasks. For each IEC 61131-3 task, a Xenomai real-time task is created through the *rt_task_create_wrap()*; a priority (ranging from 1 to 99) is specified for the Xenomai task to be created. As said

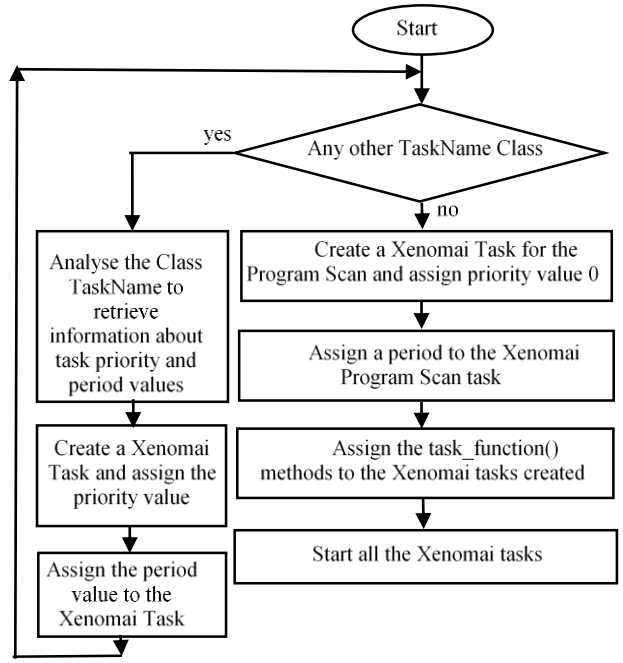


Figure 18: TasksCreator.

before, priority value is assigned according to the priority of the IEC 61131-3 task, mapping the highest priority IEC 61131-3 task with the highest Xenomai priority value (e.g., 99). Once a Xenomai task with a certain priority has been created, the same period of the relevant IEC 61131-3 task is assigned; this is done by the *rt_task_set_periodic_wrap()*.

When no more TaskName classes are present, a Xenomai task must be created in order to be subsequently associated to the task_function() method including the *ScanProgram()*. On the basis of the hypotheses explained in this section, the Xenomai task is created by using *rt_task_create_wrap()*, specifying the priority value 0 (the lowest Xenomai priority value). A period is assigned to this task according to the user settings; this is the period of the Program Scan loop the user has to apply. Period assignment is realised using again the *rt_task_set_periodic_wrap()*.

Finally all the previous Xenomai tasks are started using the *rt_task_start_wrap()*. This wrapper function calls the native Xenomai API *rt_task_start()*, passing the address of the instance of the task_function() method, created by the ProgramsCreator, to be associated to each Xenomai task with priority ranging from 1 to 99. Address of the instance of the task_function() method containing the *ScanProgram()*, is passed to the Xenomai task with priority 0.

In order to better understand the main activities performed by the *ProgramsCreator* and *TasksCreators* modules, let us consider the .cs file shown by Figure 12.

ProgramsCreator will produce the C# code shown by Figure 19; it includes the IEC61131Program() which in turns is made up by the IECRoutine() given by Figure 20.

As shown by Figure 19, the ExternalMyProgram class contained in the .cs file of Figure 12 is imported. The ExternalMyProgram class is made up by the instance Gd of GlobalDeclaration class inside which the external variables of the IEC 61131-3 MyProgram program (StepSizeVar, MaxValueVar, MinValueVar) are defined. Figure 20 points out that IECRoutine() accesses these external variables through the instance Gd contained in the ExternalMyProgram class.

```
import ExternalMyProgram;

class MyProgram {
  //double ComputedResult;
      void task_function () {
      //double ComputedResult;
        while (true) {
          //double ComputedResult;
          IEC61131Program();
          rt_task_wait_period_wrap (null);
        }
      }
}
```

Figure 19: C# code produced by ProgramsCreator.

The *TaskCreator* module extracts from the MyTask1 and MyTask2 classes produced by the *C# translator* the information about Xenomai tasks to be created; this information is about periodicity, priority and about the Program to be associated. On the basis of the information contained in the .cs file shown by Figure 12, it is clear that two Xenomai tasks must be created. Their period values are 100ms and 150ms, respectively; priority values of the original IEC 61131-3 tasks are 1 and 2, which could be mapped to Xenomai priority values 99 and 98, respectively, as priority 1 is the highest priority value according to IEC 61131-3 and must be mapped to the highest Xenomai priority value (i.e., 99). Finally, according to the information contained in the MyTask1 and MyTask2 classes, two instances of the same task_function() method shown by Figure 19 is associated to these two Xenomai tasks.

```
public void IECRoutine(){
  ExternalMyProgram.Gd.StepSizeVar=
              ExternalMyProgram.Gd.MaxValueVar-
              ExternalMyProgram.Gd.MinValueVar;
  ComputedResult = ExternalMyProgram.Gd.StepSizeVar;
}
```

Figure 20: IECRoutine().

As said many times until now, in the Figure 19 declaration of the local variable of the IEC 61131-3 MyProgram program (ComputedResult) is not defined; the figure points out only the three different scopes where declaration may occur. The following section will definitely clarify the position of the declaration of local variables, through a deep analysis of the impact of the possible choices on the run-time performance of the system.

6 Performance evaluation

It is well known that execution of a generic C# application on a CLR VM may be delayed by the activation of the Garbage Collection. When a collection starts, it causes the stop of all the tasks associated to the C# program, including the Xenomai real-time tasks in the real-time environment architecture shown by Figure 10. The main consequence is the increase of each single execution time of C# programs; furthermore, the periodicity of one or more tasks could be not respected if the time interval needed by the Garbage Collector to conclude its work is higher than the task period.

It is clear that a performance evaluation is strongly required in order to point out if what written can actually occur in the framework here defined, and, in in this case, suitable mechanisms to prevent performance deterioration must be proposed and evaluated.

During the description of the *PLC Framework* it has been left unsolved the problem relevant to the declaration of the local variables of a IEC 61131-3 program inside the class containing the task_function() method, shown by Figure 13. The possible scopes where this declaration could occur have been highlighted but no indication about the right choice has been given. Actually, this choice seems to play a very strategic role from the performance point of view of the entire real-time environment proposed. In fact, each of the three possible scopes shown by Figure 13 may led to a very different impact of the

Garbage Collector on the overall behaviour of the IEC61131-3 programs execution.

On account of what said until now, performance evaluation was carried out by the authors with the main aim to analyse the impact of the Garbage Collector on the behaviour of the real-time environment depicted by Figure 10, considering the effect of the three different scopes of local variables pointed out by Figure 13. As it will be shown, this analysis allowed to find the best choice, able to minimise the impact of the Garbage Collector over the framework here defined.

The performance evaluation has been carried out on two different architectures: the embedded system described in Section 4.1 and a general purpose computer.

Only one IEC 61131-3 ST-based Program has been considered for the performance evaluation. Several periodic tasks were associated to this Program. This choice has been done in order to make simpler the analysis of the results, removing their dependence from possible differences in the program codes executed.

Figure 21 shows the IEC 61131-3 ST language-based implementation of the Goertzel Algorithm [16] considered for the performance evaluation.

The PROGRAM section features several variables. *Q0*, *Q1* and *Q2* are output arrays used for the per-sample processing; the samples are stored in the *sbuffer* array. The *coeff* array is another basic variable of the Goertzel Algorithm; it sores some of the precomputed constants foreseen by the algorithm, needed during processing [16]. The *magnitude* array is used to store the magnitude values of the signals coming from the channels and relevant to different harmonics. *ch* and *num_ch* variables refer to the number of channels, whilst *i* and *k_max* refer to the number of harmonics. The number of samples is represented by variables n and *j*.

The first FOR cycle present in the ST code of the PROGRAM section, iterates for all the samples; the second FOR cycle allows iteration for all the harmonics to be analysed. Finally, the last cycle refers to all the channels producing the samples. The boolean condition (j = n-1) indicates the completion of the analysis of all the samples; when this condition occurs the algorithm calculates the magnitude relevant to a specific channel and to a specific harmonic, given by the value of index.

Figure 21 also shows the CONFIGURATION and RESOURCE sections containing the global constants and variables and an example of periodic task associated to the Program Goertzel. In particular, TASK MainTask1 defines a periodic task with period 100 ms and priority value 1; an instance of Program Goertzel (called MainInst1) is associated to the MainTask1. As said before, it was assumed to associate a huge number of periodic tasks to the Program Goertzel shown in Figure 21; only for reason of space limits, the other tasks are not shown in the RESOURCE section of Figure 21.

```
PROGRAM Goertzel
  VAR_EXTERNAL CONSTANT
    k_max: INT :=12;
    num_ch: INT :=8;
  END_VAR
  VAR_EXTERNAL
    coeff : ARRAY [0..k_max] OF REAL;
  END_VAR
  VAR
    count : INT;
    i : INT;
    j : INT;
    ch : INT;
    index : INT;
    n : INT;
    sbuffer : ARRAY [0..num_ch*n] OF REAL;
    Q0 : ARRAY [0..num_ch*k_max] OF REAL;
    Q1 : ARRAY [0..num_ch*k_max] OF REAL;
    Q2 : ARRAY [0..num_ch*k_max] OF REAL;
    magnitude : ARRAY [0..num_ch*k_max] OF REAL;
  END_VAR
  FOR j:= 0 TO n-1 DO
    FOR i:= 1 TO k_max DO
      count:= i * 8 - 8;
      FOR ch:= 0 TO num_ch-1 DO
        index := count + ch;
        Q0[index] := (coeff[i - 1] * Q1[index]) –
                     (Q2[index] + sbuffer[j + ch * n]);
        Q2[index] := Q1[index];
        Q1[index] := Q0[index];
        IF j = n-1 THEN
          magnitude[index] := SQRT(Q1[index] *Q1[index] +
                Q2[index] * Q2[index] –
                (Q1[index] * Q2[index] * coeff[i - 1]));
        END_IF;
      END_FOR;
    END_FOR;
  END_FOR;
END_PROGRAM

CONFIGURATION Config
  RESOURCE Resource1
    VAR_GLOBAL CONSTANT
      k_max: INT :=12;
      num_ch: INT :=8;
    END_VAR
    VAR_GLOBAL
      coeff : ARRAY [0..k_max] OF REAL;
    END_VAR
    TASK MainTask1(INTERVAL :=T#100ms, PRIORITY := 1);
    PROGRAM MainInst1 WITH MainTask1 : Goertzel;
  END_RESOURCE
END_CONFIGURATION
```

Figure 21: IEC 61131-3 ST-based application relevant to the Goertzel Algorithm.

5.3 Performance Evaluation on Embedded System

Figures 22 and 23 show the C# code produced by *ProgramsCreator* on the basis of the Goertzel algorithm shown in Figure 21, considering the local variables inside the while(true) cycle. Figure 23 details the C# code realising Goertzel's algorithm inside the IECRoutine(). As already explained in the previous section, the external variables of the IEC61131-3 PROGRAM Goertzel are defined inside class GlobalDeclaration and are used though the access to the class ExternalGoertzel, which contains the instance Gd of GlobalDeclaration. The ExternalGoertzel class contained in the .cs file is imported, as shown by Figure 22.

It has been assumed to execute the Goertzel's code with a number of harmonics equals to 6 (i.e., ExternalGoertzel.Gd.k_max constant was set to 6).

```
import ExternalGoertzel;

class Goertzel {
  void task_function () {
      double[] sbuffer =
            new double[ExternalGoertzel.Gd.num_ch * n];
      double[] Q0 = new double[ExternalGoertzel.Gd.num_ch *
                        ExternalGoertzel.Gd.k_max];
      double[] Q1 = new double[ExternalGoertzel.Gd.num_ch *
                        ExternalGoertzel.Gd.k_max];
      double[] Q2 = new double[ExternalGoertzel.Gd.num_ch *
                        ExternalGoertzel.Gd.k_max];
      double[] magnitude =
            new double[ExternalGoertzel.Gd.num_ch *
                        ExternalGoertzel.Gd.k_max];
      UInt16 count, n, index, i, j, ch;
      while (true) {
          IEC61131Program();
          rt_task_wait_period_wrap (null);
      }
  }
}
```

Figure 22: C# code produced by ProgramsCreator.

In order to analyse the execution of the Goertzel Algorithm though the use of an oscilloscope, the GPIO #2 is set at the beginning of the execution of the IECRoutine(). The GPIO #2 is reset at the conclusion of the execution of the same code. Set and reset operations are not shown in the code of Figures 22 and 23.

Figure 24 points out the GPIO #2 values during the time; the time interval during which GPIO #2 is on represents the single execution time of the Goertzel Algorithm. As pointed out by Figure 24, results achieved show that duration of the each algorithm execution maintains about the same value in time. But the figure highlights that execution of the Goertzel algorithm does not occur with the same frequency; the rt_task_wait_period_wrap() called in the in the C# code of Figure 22 is not able to guarantee that execution of the Goertzel algorithm occurred after a deterministic time interval.

```
Public void IECRoutine() {
  for (j = 0; j < n; j++) {
    for (i = 1; i < ExternalGoertzel.Gd.k_max + 1; i++) {
      count = (UInt16)(i * 8 - 8);
      for (ch = 0; ch < ExternalGoertzel.Gd.num_ch; ch++) {
        index = (UInt16)(count + ch);
        Q0[index] = (ExternalGoertzel.Gd.coeff[i - 1] *
              Q1[index]) – (Q2[index] + sbuffer[j + ch *n]);
        Q2[index] = Q1[index];
        Q1[index] = Q0[index];
        if (j == n - 1) {
          magnitude[index] = Math.Sqrt(Q1[index] *
            Q1[index] + Q2[index] * Q2[index] –
            (Q1[index] * Q2[index] *
            ExternalGoertzel.Gd.coeff[i - 1]));
      }
      }
    }
  }
}
```

Figure 23: Details of C# Goertzel Algorithm Code contained in the IECRoutine().

Figure 24: Execution of the Goertzel Algorithm shown by Figure 22 with k_max=6.

Utilisation of the CPU has been increased considering a higher number of harmonics (setting k_max to 12). Figure 25 shows the results achieved, pointing out that now the behaviour of the system is completely unpredictable. Both duration of each execution and repetition of the execution occur in an arbitrary fashion.

The behaviours depicted by Figures 24 e 25 are due to the intervention of the Garbage Collector whose execution has been forced by the choice to define the local Goertzel variables inside the while(true) loop of Figure 22. This means that, for each loop execution, these variables are de-allocated and re-allocated, producing garbage that must be collected, causing the intervention of the Garbage Collector which stops the real-time tasks producing the bad behaviour depicted by Figures 24 and 25.

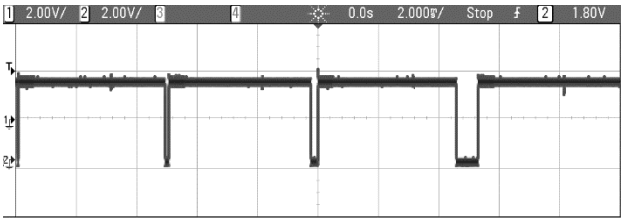


Figure 25: Execution of the Goertzel Algorithm shown by Figure 22 with k_max=12.

We proceeded to test the performance of the Goertzel algorithm by changing the scope of the local variables. The scope represented by Figure 26 has been considered; in this case, the Goertzel variable are global variable of the C# class Goertzel.

Figure 27 shows the executions of the algorithm during the time, considering a number of harmonics equal to 12 (k_max=12). As it is possible to see, now the duration of each execution is quite the same and the repetition in time of the Goertzel's algorithm is predictable because the impact of the Garbage Collector is much less than in the previous case. The variables are allocated only when the class is instantiated, and the Garbage Collector does not collect them until the deallocation of the class, that will occur only at the end of the associated task.

Another important performance improvement could be achieved defining the variable inside the task_function() as shown in Figure 28.

```
import ExternalGoertzel;

class Goertzel {
   double[] sbuffer = new double[ExternalGoertzel.Gd.num_ch * n];
   double[] Q0 = new double[ExternalGoertzel.Gd.num_ch *
                                    ExternalGoertzel.Gd.k_max];
   double[] Q1 = new double[ExternalGoertzel.Gd.num_ch *
                                    ExternalGoertzel.Gd.k_max];
   double[] Q2 = new double[ExternalGoertzel.Gd.num_ch *
                                    ExternalGoertzel.Gd.k_max];
   double[] magnitude = new double[ExternalGoertzel.Gd.num_ch
*
                                    ExternalGoertzel.Gd.k_max];
   UInt16 count, n, index, i, j, ch;

   void task_function () {
        while (true) {
             IEC61131Program();
             rt_task_wait_period_wrap (null);
        }
   }
}
```

Figure 26: C# code produced by ProgramsCreator.

In this case, it is well known that the variable access is faster than the scenario shown by Figure 16. In addition, the task_function() method is instanced only once before the creation and activation of the relevant task; this means that the so-defined variables are always active and never deallocated by Garbage Collector until the end of the task exactly like global variables. Figure 29 points out execution of the Goertzel algorithm in this scenario.

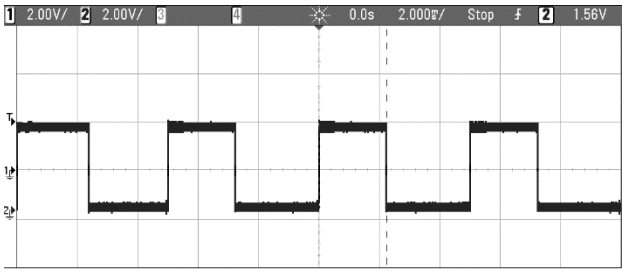


Figure 27: Execution of the Goertzel Algorithm shown by Figure 26 with k_max=12.

```
import ExternalGoertzel;

class Goertzel {
   void task_function () {
     while (true) {
        double[] sbuffer =
                  new double[ExternalGoertzel.Gd.num_ch * n];
        double[] Q0 = new double[ExternalGoertzel.Gd.num_ch *
                                    ExternalGoertzel.Gd.k_max];
        double[] Q1 = new double[ExternalGoertzel.Gd.num_ch *
                                    ExternalGoertzel.Gd.k_max];
        double[] Q2 = new double[ExternalGoertzel.Gd.num_ch *
                                    ExternalGoertzel.Gd.k_max];
        double[] magnitude=
                  new double[ExternalGoertzel.Gd.num_ch *
                                    ExternalGoertzel.Gd.k_max];
        UInt16 count, n, index, i, j, ch;

        IEC61131Program();
        rt_task_wait_period_wrap (null);
     }
   }
}
```

Figure 28: C# code produced by ProgramsCreator.

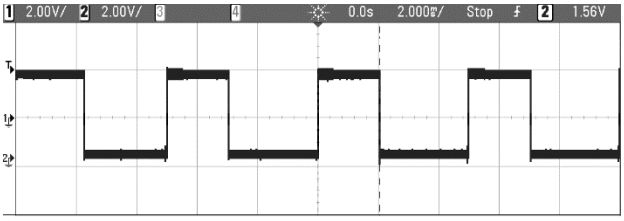


Figure 29: Execution of the Goertzel Algorithm shown by Figure 28 with k_max=12.

Comparing Figure 29 with Figure 27, it is possible to point out that the scope for the local variable considered in Figure 28 allows to improve performance of the system, as the execution time is now decreased.

5.4 Performance evaluation on general purpose computer

The algorithm shown by Figure 28 has been considered, as it allowed to achieve the best results in the performance evaluation on embedded system, as said before.

Performance evaluation has been carried out using a computer made up by a six-core Xeon processor (X5650 Intel) and 100 GB of RAM. The following software was installed on it: Ubuntu 16.04 (Kernel Linux 3.18.20), Xenomai co-kernel 3.0.2, and Mono 4.4.

Several periodic Xenomai tasks were associated to the task_function() method shown by Figure 28. It has been assumed to consider several groups of tasks; tasks belonging to each group share the same period and priority.

During execution of each Xenomai task, jitter values were measured. Figure 30 shows how jitter has been evaluated; each arrow represents a real execution of a Xenomai task.

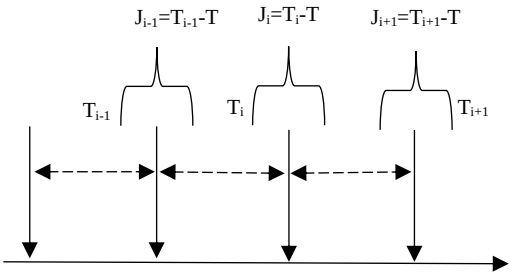


Figure 30: Jitter evaluation.

For each task, T_{i-1}, T_i, T_{i+1} are generic time intervals between consecutive Xenomai periodic task executions. Said T the period of the Xenomai task, J_{i-1}, J_i, J_{i+1} values shown in Figure 30 are the relevant jitter values. For each single task, the average absolute value of the jitters was calculated. It was said that tasks were divided into several group, each group sharing the same period and priority; for each group of tasks, the minimum and the maximum average absolute jitter values were pointed out.

Performance evaluation has been carried out considering different scenarios featured by different groups of tasks, different numbers of tasks for each group and different period values associated to a group. Scenarios were chosen in order to be comparable in terms

of bandwidth utilisation, otherwise comparison between their performances was meaningless.

For each group of tasks, the bandwidth utilisation has been defined by:

$$bandwidth\ utilisation = \frac{1}{T} * n * t \quad (1)$$

where *n* is the number of tasks belonging to the group and sharing the same period *T*, and *t* is the execution time of the Program associated to the task.

Use of the parameter given by (1) allowed to compare scenarios featured by the same bandwidth utilisation, during the performance evaluation carried out.

Tables 1 and 2 show two of the scenarios considered. Three groups of tasks have been considered for each scenario. Inside each scenario, the groups differs for the number of tasks and the relevant period. Comparing the different scenarios, they feature groups with the same bandwidth utilisation.

Tables 3 and 4 presents some of the results achieved. They show the minimum and maximum average absolute jitter values for each scenario and for each of the three groups. As it can be seen, the average absolute jitter values are always very close to zero.

Table 1: Scenario 1: Groups of Tasks with Period, Number of Tasks and Bandwidth Utilisation.

Group	Period (ms)	Number of Tasks	Bandwidth Utilisation
1	50	100	30%
2	30	100	50%
3	25	100	60%

Table 2: Scenario 2: Groups of Tasks with Period, Number of Tasks and Bandwidth Utilisation.

Group	Period (ms)	Number of Tasks	Bandwidth Utilisation
1	25	50	30%
2	15	50	50%
3	12.5	50	60%

Table 3: Scenario 1: Minimum and Maximum Average Absolute Jitters.

Group	Period (ms)	Min (ms)	Max (ms)
1	50	6.58 E-05	3.40 E-04
3	30	5.03 E-05	3.59 E-04
4	25	5.89 E-05	6.45 E-04

Table 4: Scenario 2: Minimum and Maximum Average Absolute Jitters.

Group	Period (ms)	Min (ms)	Max (ms)
1	25	5.27 E-05	1.89 E-04
2	15	5.39 E-05	3.41 E-04
3	12.5	5.86 E-05	6.96 E-04

These results seem to demonstrate that Garbage Collector does not affect at all the performance of the system. They confirm the same results achieved through the experiments carried out by the embedded system and shown in the previous subsection.

In order to verify this result, in the following an analysis will be presented in order to point out what could occur when the Garbage Collector intervenes. The C# code shown by Figure 22 has been considered. As said, in this scenario the local variables are mapped inside the while(true) cycle; this means that the entire set of variables are re-located for each cycle. This affects the heap memory capacity, going to fill it and forcing the Garbage Collector to intervene to free the unused variables, as each cycle uses another set of the same local variables.

Tables 5 and 6 show the minimum and maximum average absolute jitter values for the same scenarios seen before. It is important to point out the higher values of the jitter. Furthermore, it is important to compare the maximum average absolute jitter values with the period of each group; in some cases, the values are close to the same periods, pointing out the very bad performance achieved.

Time instants of each Xenomai task execution have been recorded during the performance evaluation, considering again the C# code shown by Figure 22. The entire set of the execution times for the tasks belonging to each group has been carefully analysed. Analysis pointed out that Xenomai task executions sometimes featured the behaviour depicted by Figure 31. Each vertical arrow in the figure represents a real execution; T_i is the time interval between two consecutive executions, and the dotted vertical arrows represents the instant at which a periodic execution is expected but does not occur.

Table 5: Scenario 1: Minimum and Maximum Average Absolute Jitters.

Group	Period (ms)	Min (ms)	Max (ms)
1	50	9.38	11.30
2	30	5.62	11.12
3	25	4.77	12.56

Table 6: Scenario 2: Minimum and Maximum Average Absolute Jitters.

Group	Period (ms)	Min (ms)	Max (ms)
1	25	2.12	2.56
2	15	1.30	2.49
3	12.5	1.10	2.62

As shown by Figure 31, during a task execution, jitter values greater than a multiple value of the task period may occur. It has been observed that the generic T_i may be greater than two or three times the task period, in the worst cases. The only event which could cause this behaviour is the running of Garbage Collector causing the stop of the

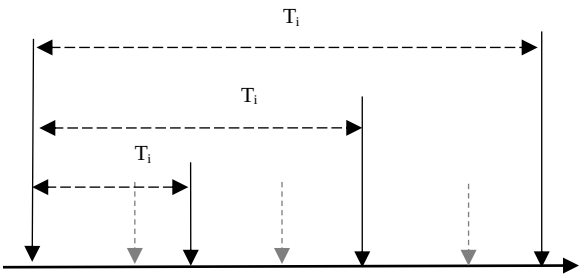


Figure 31: Xenomai task executions.

Xenomai tasks. Values of the time interval T_i clearly depends on the execution times of the Garbage Collector needed to collect all the garbage produced by the programs.

7 Final remarks

Paper has presented a software solution allowing the execution of IEC 61131-3 applications into computing systems based on a CLR VM. The software solution is made up by two main components.

The first component is a software able to realise translation of a generic IEC 61313-3 application into C# code. For each IEC 61131-3 application a .cs file is produced containing several classes relevant to the original IEC 61131-3 sections; these classes may be directly instantiated and used in a generic C# program running inside a CLR VM. Otherwise, the output produced may be passed to the second component here presented, which is a real-time execution environment. It is a framework able to realise the exact behaviour of a PLC (e.g., Program Scan loops and real-time task scheduling). It requires the presence of a CLR VM running on the top of a real-time operating system. The framework receives the C# classes produced by the first component described before, achieved for an IEC 61131-3 application. On the basis of these classes, it produces suitable C# programs and real-time tasks associated to the programs to be submitted to the underlying real-time operating system. In this paper, use of Xenomai real-time co-kernel has been presented.

After a description of both the software components, the paper focused on a performance evaluation of capability of the real-time environment to respect the periodic constraints of real-time tasks. As known, in a CLR VM-based environment, execution of a generic C# program may be delayed by the activation of the Garbage Collection. When a collection starts, it may cause the stop of all the tasks associated to the C# program and the increase of their execution time. The periodicity of one or more tasks could be not respected for the same reason. The results of the performance evaluation carried out by the authors, pointed out that although Garbage Collector may be a cause of performance degradation, its impact on the performance of the system may be drastically limited. This can be achieved by realising the right mapping between the IEC 61131-3 original local variables defined inside IEC 61131-3 PROGRAM section and the variables used by the C# classes generated by the real-time environment. Results presented in the paper, pointed out that the mapping choices operated by the authors avoid the intervention of the Garbage Collector. Under their adoption, performance evaluation allowed to demonstrate the capability of the real-time environment here presented to respect real-time constraints of periodic tasks.

To the best of authors' knowledge, current literature does not provide solutions aimed to deploy IEC 61131-3 applications on CLR VM, using C# language as intermediate code. Due to the spread current use of C# language in the development of industrial applications, adoption of the software solutions here presented seems attractive. Typical candidate platforms on which deployment may be achieved, are those based on general purpose computer architecture (on which CLR VM allows the use of common operating systems like Linux and Windows), but also all the embedded systems supporting a CLR VM may be considered.

Furthermore, the paper gives a contribution to a very spread research field currently present in literature; in particular it introduces a solution able to move computation further away from the field level into the so-called compute pools, which are decentralised and may be also realised inside cloud computing solutions. In the scenario proposed, the PLC is migrated to the compute pool which can be realised by a computer architectures based on CLR VM, as demonstrated by the research presented in the paper.

Although the paper has been presented considering Just-in-Time compilation, it is important to point out that the procedures presented in the paper and aimed to translate original IEC 61131-3 language-based programs may be applied also into the case the Ahead-of-Time (AOT) compilation was adopted.

References

[1] R.W.Lewis (1998). *Programming Industrial Control Systems Using IEC 1131-3*, IEE, ISBN-13: 978-0852969502, 1998.

[2] J. D. Decotignie (2009). The many faces of industrial ethernet [past and present]. *IEEE Industrial Electronics Magazine*, vol. 3, no. 1, pp. 8–19. https://doi.org/10.1109/mie.2009.932171

[3] M. Becker, K. Sandstrom, M. Behnam, T. Nolte (2015). A many-core based execution framework for IEC 61131-3. *Proceedings of 41st IEEE Annual Conference IECON 2015*, pp.4525-4530, https://doi.org/10.1109/IECON.2015.7392805.

[4] S. Mubeen, M. Becker, X. Zhao, L. Gan, M. Behnam, T. Nolte (2016). Towards automated deployment of IEC 61131-3 applications on multi-core systems. *Proceedings of IEEE World Conference on Factory Communication Systems (WFCS 2016)*, pp.1-4, https://doi.org/10.1109/WFCS.2016.7496531.

[5] M. Simros, S. Theurich and M. Wollschlaeger (2012). Programming Embedded Devices in IEC 61131-Languages with Industrial PLC Tools using PLCOPEN XML. *Proceedings of 10th Portuguese Conference on Automatic Control (2012)*, pp.51-56.

[6] O. Givehchi, H. Trsek, and J. Jasperneite (2013). Cloud computing for industrial automation systems - a comprehensive overview. *Proceedings of IEEE 18th Conference on Emerging Technologies Factory Automation (ETFA 2013)*, pp.1–4. https://doi.org/10.1109/etfa.2013.6648080

[7] O. Givehchi, J. Imtiaz, H. Trsek, and J. Jasperneite (2014). Control-as-a-service from the cloud: A case study for using virtualized PLCs. *Proceedings of 10th IEEE Workshop on Factory Communication Systems (WFCS 2014)*, pp.1–4. https://doi.org/10.1109/wfcs.2014.6837587

[8] I. Kühl and A. Fay (2011). A Middleware for Software Evolution of Automation Software. *Proceedings of IEEE 16th Conference on Emerging Technologies and Factory Automation (ETFA 2011)*, pp.1-9. https://doi.org/10.1109/etfa.2011.6059109

[9] Mono official website, available on http://www.mono-project.com/

[10] Xenomai official website, available on https://xenomai.org

[11] S. Cavalieri, L. Galvagno, M.S.Scroppo (2016). A Framework based on CLR Virtual Machine to deploy IEC 61131-3 programs. *Proceedings of 14th International Conference on Industrial Informatics (INDIN 2016)*, University of Poitiers, Poitiers, France,pp.126–131, https://doi.org/10.1109/INDIN.2016.7819146.

[12] S. Cavalieri, L. Galvagno, G. Puglisi, M.S. Scroppo (2016). Moving IEC 61131-3 applications to a computing framework based on CLR Virtual Machine. *Proceedings of 21th International Conference on Emerging Technologies and Factory Automation (ETFA 2016), Berlin, Germany*, pp.1-8, https://doi.org/10.1109/ETFA.2016.7733632.

[13] Xenomai official API reference website, available on https://xenomai.org/api-reference/

[14] Freescale Semiconductor, *MPC8309 (2014). PowerQUICC II Pro Integrated Communications Processor Family Hardware Specifications*, Data Sheet. Document Number MPC8309EC, Rev 4, 12/2014. Available on http://www.nxp.com/

[15] Gold parsing system official website, available on http://www.goldparser.org/

[16] G. Goertzel (1958). An algorithm for the evaluation of finite trigonometric series. *American Mathematical Monthly*, Vol. 65, No. 1, pp. 34-35, https://doi.org/10.2307/2310304

https://doi.org/10.31449/inf.v43i2.2133 Informatica **43** (2019) 281–289 **281**

A Comparative Study of Automatic Programming Techniques

Sibel Arslan and Celal Öztürk
Erciyes University, Engineering Faculty, Computer Engineering Department, Kayseri, Turkey
E-mail: sibel.arslan2@icisleri.gov.tr, celal@erciyes.edu.tr

Keywords: automatic programming, genetic programming, artificial bee colony programming, symbolic regression, prediction, feature selection

Received: January 3, 2018

Automatic programming, an evolutionary computing technique, forms the programs automatically and is based on higher level features that can be easily specified than normal programming languages. Genetic Programming (GP) is the first and best-known automatic programming technique that is applied to solve many practical problems. Artificial Bee Colony Programming (ABCP) is one of the latest proposed automatic programming method that combines evolutionary approach with swarm intelligence. GP is an extension version of Genetic Algorithm (GA) and ABCP is based on Artificial Bee Colony (ABC) algorithm. The main differences of these automatic programing techniques and their conventional algorithms (GA and ABC) are modeling solution. In ABC same as GA, the solutions are represented fixed code blocks. In GP and ABCP, the positions of food sources are expressed in tree structure that is composed of different combinations of terminals and functions that are specifically defined as problems. This paper presents a review on GP and ABCP and they are worked in symbolic regression, prediction and feature selection problems which are widely tackled by researchers. The results of the ABCP compared with results of GP show that this algorithm is a powerful optimization technique for structural design.

Povzetek: Predstavljena je primerjalna analiza tehnik avtomatskega programiranja.

1 Introduction

Computer programming is the process of obtaining a program that can be executed machine to use the necessary information to perform a task. Automatic programming is a computer programming technique which automatically generates the program code [1]. It provides practical solutions for many machine learning methods such as Artificial Neural Network (ANN), Decision Trees (DT), Support Vector Machines (SVM), Genetic Programming (GP), Artificial Bee Colony Programming (ABCP). GP, most well-known automatic programming method, was developed by Koza [2]. GP is an extension of Genetic Algorithm (GA) and the basic steps for the GP algorithm are similar to the steps of GA. ABCP is recently proposed automatic programming technique which is based on the Artificial Bee Colony Algorithm (ABC) [3]. The goal of this paper is to evaluate the success of the models obtained from these automatic programming methods in symbolic regression, prediction and feature selection problems and review papers related to the problems.

Symbolic regression is a type of regression problem aimed at finding the most appropriate mathematical model in terms of accuracy and complexity of data. There are works investigating the problem of symbolic regression with automatic programing techniques, mostly with GP [3-10]. ABCP was proposed for the first time as a new method for the symbolic regression problem and compared with GP [3]. Faris proposed solving the symbolic regression problem using GP model was compared to least square estimation, GA and particle swarm intelligence models based on estimating the parameters of the nonlinear regressive curve of the cutting tool [4]. According to the benchmarks in the paper, GP was found to be superior performance. In [5], two versions of GP (standard GP and multi-population GP) were compared with ANN on pharmaceutical formulation symbolic regression problem. Compared to successful ANN models, GP models provided a significant advantage, parametric equations that can be interpreted and analyzed more easily. Gene Expression Programming (GEP) [6], Immune Programming (IP) [7], Ant Colony Programming (ACP) [8-10] are other automated programming techniques that used in the problem of symbolic regression. GEP, proposed on both GA and GP, is flexible at genetic operations due to its linear code blocks and its parse trees [6]. IP is based on Artificial Immune System (AIS) and is a domain-independent approach [7] where antigens can be represented in the tree structure express solutions similar to GP. ACP [8, 9] and Dynamic Ant Colony Programming (DAP) which dynamic version of ACP [10] are the main ACP examples that are inspired on ant colony algorithm.

Automatic programming techniques have been solved prediction problems where most of applications are based on evolutionary optimization techniques [11-19]. Seidy proposed a new stock market prediction model using the Particle Swarm Optimization with Center of Mass Technique (PSOCoM) which was more successful results than the particle swarm optimization based models according to the prediction accuracy [11]. Manjusha et al. used Naive Bayes and J48 algorithms to diagnose potentially fatal dermatological diseases with similar

symptoms [12]. They developed the interface which the probability of recurrence of each disease was predicted. In [13] Box-Jenkins (BJ) and ANN were used together to model monthly water consumption in Kuwait. The input layer variables in the artificial neural network were obtained with BJ, the average error was considered, and the more successful results were obtained than the traditional methods. Ant Colony Optimization (ACO) technique was used to generate qualified bankruptcy prediction rules [15]. The Association Rule Miner (ARM) technique was used to group rules and eliminate irrelevant rules in the paper. Etemadi et al. compared Multiple Discriminant Analysis (MDA) and the GP in bankruptcy prediction modeling [18]. GP model was found to produce more accurate results than the MDA model, which is produced as an accurate bankruptcy forecasting model considering both the quality of the sample companies and the estimated duration. Searson et al. used multigene GP demonstrating it with an application in which a predictive symbolic Quantitative Structure Activity Relationship (QSAR) model of T. pyriformis aqueous toxicity was as successful as the QSAR models on the same data [19].

In the recent researches, the increase in the number of features in the data sets necessitated the use of feature selection methods. These methods are used to eliminate noisy and unnecessary features in collected data so that the data set can be expressed more reliably and at the same time classification achieve high success rates. Various optimization methods have been applied to solve feature selection problems [20-33]. Lujan et al. proposed an automatic programming technique called quadratic programming based on quadratic function optimization for multiclass problems [22]. The work was found more efficient than the Maximal Relevance (MaxRel) and Minimal-Redundancy-Maximum-Relevance (mRMR) on large data sets. In [23], statistical and entropy-based feature ranking methods were compared with different data sets. It was shown that the accuracy of the classifier was influenced by the choice of ranking indices. Brown et al. investigated the apparent statistical assumptions of feature selection criteria based on mutual knowledge [24]. They derived the objective function using the conditional probabilities of the training labels. When the results were evaluated, the Joint Mutual Information (JMI) criterion provided the best balance of accuracy, stability and flexibility criteria for small data sets. In [27], two-stage automatic programming technique called differential development based Named Entity Recognition (NER) was proposed. In the first stage, Conditional Random Field (CRF) and SVM classifiers were used in feature selection problems. In the second stage, classifiers according to F scale score were selected and combined using differential development based classifier collection technique. The technique was more successful than the other traditional methods. Yu et al. showed that the GP can be used as a feature selector and cancer classifier [30]. Selecting the discriminative genes of GP, expressing the relations between the genes as mathematical equations were proof that GP can be used in this field. In addition, training sets and GP classifiers obtained from the validation set in the work were tested GP successfully classified tumor classes and were more successful than various classification methods. The (k-Nearest Neighbor (k-NN) and GP based decision trees are applied to feature selection and were compared in terms of classification performance in [31]. Arqub et al. proposed an algorithm based on GA for the solutions of nonlinear systems of second-order boundary value problems [32]. The results show that the algorithm is very effective and convenient in linear and nonlinear cases with less computational generation and less time. Continuous Genetic Algorithm (CGA) is introduced solving systems of second-order boundary value problems [33]. The influence of different parameters, including the initialization method, the selection method, the rank-based ratio, the evolution of nodal values, the population size, the crossover and mutation probabilities, the step size, and the maximum nodal residual is studied in the paper. The algorithm had better performance than some modern methods.

GP and ABCP is a successful automatic programming techniques which were based on GA and ABC. In this paper, we have compared GP and ABCP on the main applications of automatic programming which are symbolic regression, prediction and feature selection problems. The organization of the paper is as follows: GP is described in Section 2, ABCP is introduced in Section 3, and Experimental Design is presented Section 4 and Results are discussed in Section 5. The paper is concluded in Section 6 with remarking the future work.

2 Genetic programming

GP, most well-known automatic programming method, expresses solutions as tree structures. The trees are randomly generated according to tree depth which is previously determined. The production of tree nodes is provided by terminals (constants or variables such as x, y, 5) and functions (arithmetic operators such as +, - /, max). The representation of the tree is shown in Fig. (1) [34]. The root node connects to the branches each of them consists of more than one component. In all cases the model of solution is found by analyzing the entire tree.

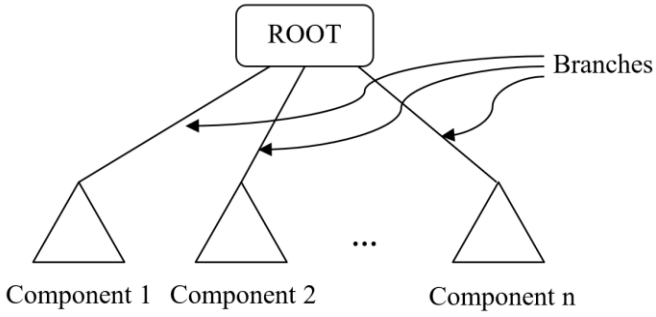


Figure 1: Representation of tree.

A flow chart of GP is given in Fig (2). Initial population is produced and the fitness of the solutions according to the determined fitness function is assessed. The production of the individuals in the initial population is based on full method, grow method, or ramped half-and-half method [35]. In the full method, nodes are selected from the function set until they reach the maximum tree

depth. In the grow method, nodes are randomly selected from a set of all terminals and functions. In the ramped half-and-half method, 50% full and 50% grow method are used to produce trees in various widths and depths [2]. GP aims to increase the number of individuals with high quality survival and decrease the number of low quality individuals. Individuals with high quality are more likely to pass on to the next generation. Almost all selection operators of GA can be used (mostly tournament selection methods) in GP [36]. Individuals the optimization of problems is developed them with exchange operators such as reproduction, crossover and mutation. The crossover operator allows the hybrid of two selected individuals to produce a new individual. The subtrees taken from the two randomly selected crossover points of the parent trees are crossed to obtain new trees. The mutation operator provides unprecedented and unexplored individuals. A randomly generated node or tree is usually exchanged in the mutation process instead of the node selected from the tree. The best individuals of the previous generation are transferred to the current generation with elitism. The stopping criterion is checked that the individuals reach a certain value or the predetermined number of generations and then, the program is terminated when the stopping criterion is satisfied.

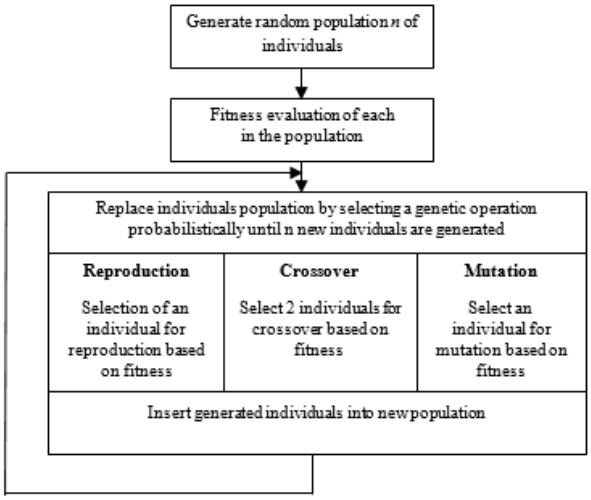


Figure 2: The flow chart of Genetic Programming.

3 Artificial bee colony programming

ABC is a swarm intelligence optimization algorithm that simulates the behavior of honeybees and provides a solution for multi variable problems [37]. ABCP, based on ABC algorithm, was introduced first time as a new method for symbolic regression [3]. In ABC, the positions of the food sources are represented with fixed size arrays. In the ABCP method, the positions of food sources are expressed in tree structure that is composed of different combinations of terminals and functions that are specifically defined for problems [41]. The mathematical relationship of the solution model in ABCP can be represented the individuals in Fig. (3) is described Eq. (1). In these notations, x is used to represent the independent, and $f(x)$ is dependent.

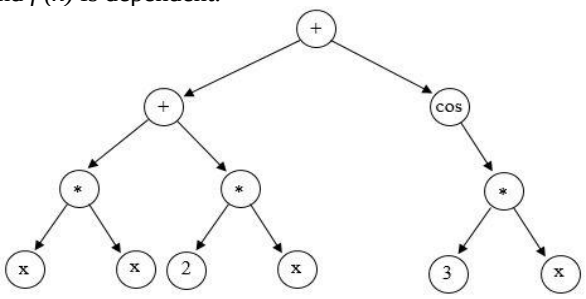


Figure 3: Representation of an example solution in ABCP with tree structure.

$$f(x) = [(x^2 + 2x)] + [cos3x] \quad (1)$$

There are three different types of bees in ABCP, each of which is responsible for a food source, employed bee, onlooker bee and scout bee. The position of a food source express a solution (a single parse tree). The number of employed bees is equal to the number of onlooker bees. Quality of the food source in terms of nectar is expressed through the fitness function of the solution. The employed bees search new food sources and shares information about food sources with the onlooker bees. They tend to be more inclined toward quality food sources in line with the information they receive from their employed bees. If a source is abandoned, the employed bee becomes a scout bee that starts to look for a new source randomly. The exhausted of food resources controls a parameter called "limit". For each source, the number of improving trials is kept, and in each cycle, it is checked to see whether the number of trials exceeds the "limit" parameter. If the food source is exhausted, the source is abandoned. Employed bees of an abandoned source turn into a scout bee and look randomly for a new source. Algorithmic steps for the ABCP are given in detail in Fig. (4).

ABCP starts with the production of food source in the initial phase. Similar to producing GP's solutions, food sources by full method, grow method or ramped half and half method are produced [2]. The main difference between ABCP and ABC is the neighborhood mechanism in generating candidate solutions [3]. When a candidate solution (v_i) is generated based on the node x_i which represents the current solution in the tree and a neighbor node solution x_k which is randomly taken from the tree considering predetermined probability p_{ip}. The node selected from the neighbor solution x_k determines what information will be shared with the current solution and how much it will be shared. This sharing mechanism is shown in Fig. (5). Figure 5a and 5b are: node x_i representing the current solution and neighbor node x_k, respectively; neighboring information and the generated candidate solution are given in Figure 5c and Figure 5d, respectively. If the quality of the candidate solution v_i is better than of the current solution x_i, v_i is selected on the other case x_i is going on.

Employed bees share the information they have gained after completing the research process in the sources they are related to with onlooker bees. They select source according to the probability values calculated by Eq. (2)

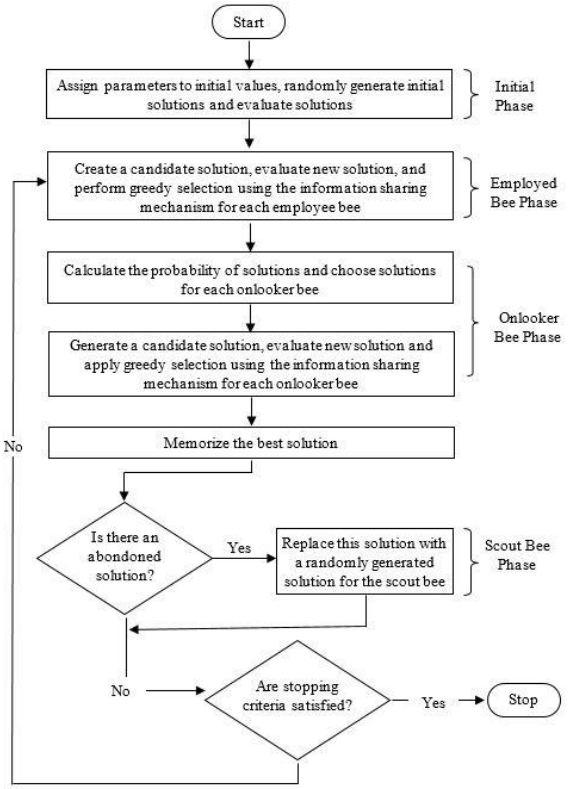


Figure 4: The flow chart of Artificial Bee Colony Programming.

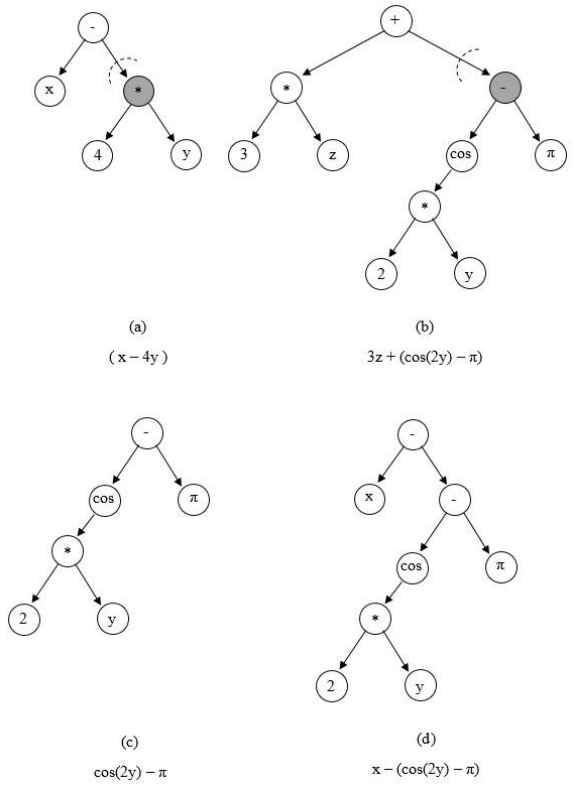


Figure 5: Example of information sharing mechanism.

depending on the nectar quantities of sources within the information they receive from employed bees' information.

$$p_i = \frac{0.9 * fit_i}{fit_{best}} + 0.1 \quad (2)$$

Where fit_i quality of the solution i, fit_{best} quality of the best solution current solutions. If a source is more qualified, the probability of selecting the source increases. After selecting the sources to be searched, the onlooker bees find new sources like employed bees. The amount of nectar of the new found source is checked. If the new source has a higher amount of nectar, the new source is taken to memory and the old source is deleted from memory. Therefore, the onlooker bees show a greedy selection like the employed bees.

In ABCP, the penalty point of the relevant sources is increased by one when no better sources can be found for each employed bee and onlooker bee. When a better source of any source is discovered, that source's penalty point is reset. Once all the employed bees and onlooker bees have completed the search operations in each cycle, the penalty points of sources are checked [42].

4 Experimental design

This section, three different experiments with benchmark data sets have been studied and performance values of GP and ABCP have been compared using similar parameter values and the results have been discussed.

4.1 Experiments

In the first experiment, the performance of models evolved by GP and ABCP are evaluated in the symbolic regression. Training data set for the 4-input non-linear Cherkassy function expressed in Eq. (3) [38]. The objective of the GP and ABCP are to evolve a symbolic function of x_1, x_2, x_3, *and x4* that closely approximates y.

$$y = \exp(2x_1 \sin(\pi x_4)) + \sin(x_2 x_3) \quad (3)$$

In the second experiment, the output values in the pH data set [39] are predicted. The data in this experiment is taken a simulation of a pH neutralization process with one output (pH), which is a non-linear function of the four inputs.

Concrete pressure compressive strength data [40] is taken to study feature selection performance of GP and ABCP in the last dataset. Concrete pressure compressive strength data is highly nonlinear of input values. The outputs being modelled are produced by automatic programming methods of the concrete data and the independent variables are: cement (x_1), blast furnace slag (x_2), fly ash (x_3), water (x_4), superplasticizer (x_5) ,Coarse aggregate (x_6), fine aggregate (x_7), age (x_8). The noise, which was added to concrete dataset, consist of 50 input variables (x_9, x_{10}, …, x_{58}) with random values in range [-500,500]. The number of inputs and instances of the problems are shown in Table 1.

4.2 Fitness function - parameters

The performance of models are evaluated by the Root Mean Square Error (RMSE) on both the training set and the test set. The fitness function is shown Eq. (4).

Name	#Inputs	#Total Instances	#Training Instances	#Test Instances	Noise
Cherkassy	4	500	400	100	-
pH	4	990	700	299	-
Concrete	8	1030	773	257	50 input variables [-500,500]

Table 1: Benchmark Problems.

$$fitness = \sqrt{\frac{\sum_{t=1}^{n}(y_{pred}-y_{actual})^2}{n}} \quad (4)$$

Where *n* define the data size, y_{actual} is y values from data set, y_{pred} is the predicted *y* value by obtained solution.

The complexity of the solution is calculated as in Eq. (5) in proportion to the depth of the tree and the number of nodes.

$$C = \sum_{k=1}^{d} n * k \quad (5)$$

Where *C* is tree complexity, *d* is the depth of the solution, and *n* is the number of nodes at depth. The parameters for GP and ABCP are summarized in Table 2. The *add3* function is the sum of three variables (x_1+ x_2+ x_3) and the *mult3* function is the multiplication of three variables (x_1* x_2* x_3). If the divisor value is equal to zero, the result is 1, otherwise the normal division is performed in the *rdivide* function. The ifbte and the iflte indicates the condition of the nodes. Eq. (6) and Eq. (7) describe how the functions operate condition expressions.

$$X = ifbte(A,B,C,D)$$
$$if(A \geq B) then\ X = C\ else\ X = D \quad (6)$$

$$X = iflte(A,B,C,D)$$
$$if(A < B) then\ X = C\ else\ X = D \quad (7)$$

The population size was taken high value according to the curse of the data sets in the literature. When experiments are evaluated, it is observed that the optimum results were obtained where population/ colony size was set to 100 for Cherkassy, 200 for pH and 300 for Concrete. Therefore, other parameters like generation number and maximum tree depth values was set to make fair test with same parameters like in the literature. In this work, the stop criterion is decided by using the maximum generation number for both GP and ABCP.

5 Results & discussions

This section demonstrate symbolic regression, prediction and feature selection abilities of ABCP and GP, set of experiments conducted.

5.1 Simulation Results

The experiments are run 30 times independently for ABCP and GP and the obtained results are demonstrated in Table 3 for the problems. The R^2 values of the best cases of GP and ABCP are also presented in the table for training and test sets.

Parameters	Problems					
	Cherkassy		pH		Concrete	
	GP	ABCP	GP	ABCP	GP	ABCP
Population / Colony size	100	100	200	200	300	300
Iteration size	100	100	200	200	500	500
Maximum tree depth	12	12	12	12	12	12
Tournament size	6	-	25	-	15	-
Mutation ratio	0.14	-	0.14	-	0.14	-
Crossover ratio	0.84	-	0.84	-	0.84	-
Direct reproduction ratio	0.02	-	0.02	-	0.02	-
Functions	+,-, *, tan, sin, cos, square, max, min, exp, ifbte, iflte		+, -, *, tanh, add3, mult3		+,-,*,rdivide,sin, cos,exp,rlog, add3,mult3	
Constans	[-10,10]		[-10,10]		[-10,10]	

Table 2: Parameters for GP and ABCP.

Metric	Criteria	Problems					
		Cherkassy		pH		Concrete	
		GP	ABCP	GP	ABCP	GP	ABCP
Mean	RMSE	0.07	0.03	0.92	0.77	11.64	10.50
Std	RMSE	0.03	0.02	0.14	0.07	2.38	1.51
Best	RMSE	0.02	0.01	0.60	0.59	8.83	8.26
Worst	RMSE	0.11	0.09	12.60	0.88	16.77	14.57
Best	R^2_{train}	0.97	1.00	0.90	0.96	0.72	0.76
Best	R^2_{test}	0.98	1.00	0.91	0.96	0.70	0.73

Table 3. Obtained results of GP and ABCP.

While obtaining the results of GP on symbolic regression problem, the GPTIPS (an open source symbolic regression solution toolbox) [39] is modified and used in this work. It indicates ABCP has much better training

performance than GP on these datasets. The best mean fitness value of all runs was found in the 0.0323 Cherkassy symbolic regression problem in ABCP. The standard deviation of ABCP is in 10% range where it is in 20% range at GP for pH dataset. However all criteria are worse than the other datasets due to the noise in the concrete dataset, ABCP has comparable performance than GP.

The curve fitting of the y_{actual} and the y_{pred} values for the training and test data set at this best fitness value is expressed in Figure 6 for Cherkassy, Figure 7 for pH, Figure 8 for Concrete dataset in ABCP.

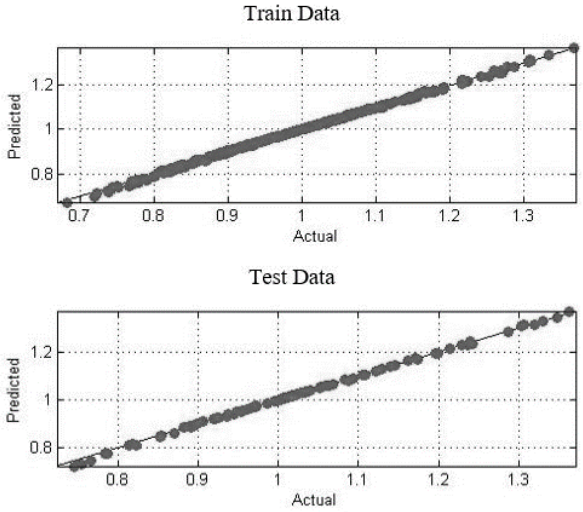


Figure 6: Predicted and actual data points on Cherkassy in best ABCP.

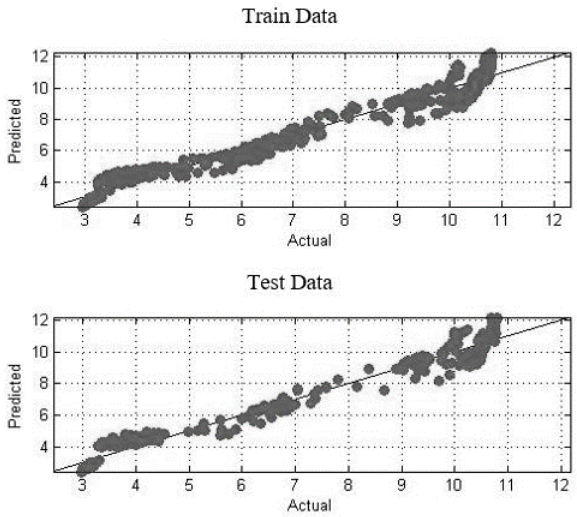


Figure 7: Predicted and actual data points on pH in best ABCP.

Figures 6-7 and 8 show the evolution plots for all datasets. The y_{actual} and the y_{pred} values are close to each other for training and test data as seen from the curves.

5.2 Analysis of evolved models

The evolved models of best solutions in ABCP of all runs are shown in Table 4. Mathematical models have been obtained using all inputs in the Cherkassy and pH datasets. The evolved model for Concrete has *Blast Furnace Slag, Age, Cement, Plastic* input parameters of 8 input parameters are selected. It should be noticed that only the x_{17} input is taken from the 50 added noise parameters. The presence of only one of the noise parameters in the equation is an indication that the ABCP is successful in feature selection.

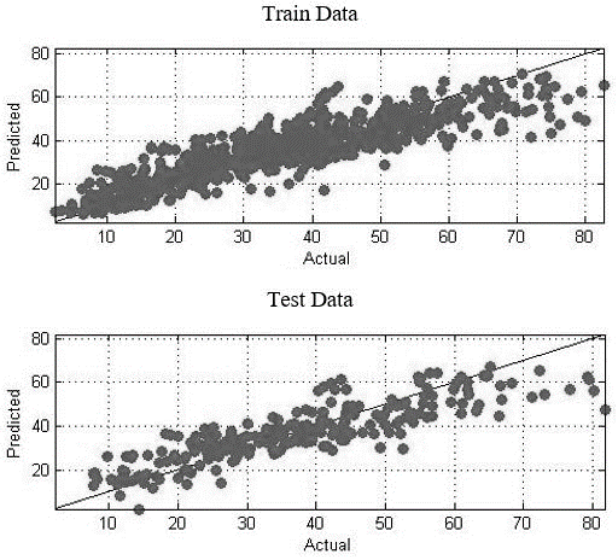


Figure 8: Predicted and actual data points on Concrete in best ABCP.

The total number of nodes tree has, tree depth, tree complexity of the best solution is given Table 5 for each dataset. As seen in Table 5, the noise parameters added to the inputs in the problems increase the difficulty of the problems. Increasing the difficulty enlarges the solution trees and increases their complexity. Since the Cherkassy function is easier than other problems, the complexity of the tree is the least problem.

Problem	ABCP		
	Total number of nodes	Depth of the best solution tree	Best solution tree complexity
Cherkassy	25	7	118
pH	72	11	467
Concrete	80	12	657

Table 5: Best solution tree information for each data set.

6 Conclusion

In this paper, the automatic programming methods have been examined on symbolic regression, prediction and feature selection. The results of the symbolic regression on Cherkassy function, prediction on pH and feature selection concrete datasets are used to compare GP and ABCP methods. In all three experiments, ABCP demonstrate higher performance than GP in terms of finding more accurate mathematical modeling in symbolic

Dataset	**Model of Best of Run Individual**	**Number of Inputs**
Cherkassy	$y = ((\exp(3x_4x_1) * (\exp(x_2x_3) + x_4x_1)) + x_4x_1) * \exp(x_4x_1))$	4
pH	$y = A + B + C$ $A = (\tanh(\tanh(x_4 + (x_3 * \tanh(x_2))) + 2x_3) * \tanh(\tanh(x_2)) - x_2)$ $B = x_2^2 * (\tanh\left(\left(x_4 - (x_1 - 8.935 + (x_1 + x_3 + x_4) * \tanh(\tanh(x_2)))\right) * \tanh(x_2)\right)) + x_2$ $C = \left(x_4 + (x_2 + \tanh(\tanh(\tanh(x_2))) + \tanh(\tanh(2x_2 + x_4))) + \left((x_2^2 * x_3) + x_2\right)\right) + \tanh(x_2) + \tanh\left(\tanh(x_2) - (x_1 - x_3)\right) - x_4$	4
Concrete	$y = \left\lvert \log\left(\frac{A}{B}\right) + (C) \right\rvert$ $A = \left\lvert (\tanh\left(\tanh\left(sqrt\left(sqrt(Age)\right)\right)\right) + \left(\frac{sqrt(\log(Cement))}{(-2.997)} + Slag\right) * Age * \frac{\tanh\left(\tanh\left(\log\left(\frac{Plastic}{-3.877}\right)\right)\right)}{\frac{\tanh(\tanh(x_{17}))}{(-3.877)}}) \right\rvert$ $B = (sqrt(Cement))^2 + \lvert(\lvert Cement\rvert * (-3.877))^2\rvert$ $C = (-3.877) + \left\lvert \left(sqrt(\log(Age)^2) * sqrt(Cement) - \left(\frac{\tanh(\tanh(Plastic))}{\tanh\left(\frac{\tanh(Cement)}{(-3.877)}\right)}\right)\right)\right\rvert - \tanh\left(\tanh\left(\log\left(\frac{Plastic}{(-3.877)}\right)\right)\right) / \tanh(\tanh\left(\tanh\left(\frac{\tanh(Cement)}{(-3.877)}\right)\right))$	5

Table 4: Models of Best Run ABCP and GP.

regression, better fitting in prediction ability and effective in finding important features along with the presence of redundant features.

In the future, we are intending to investigate several interesting researches. Simulation works will be done to model the fundamental classification problems (cancer, diabetes, heart, gene diseases etc.) with GP and ABCP to get performance results. In addition, we are planning to work the Multi-Gen Genetic Programming and Multi-Hive ABCP and compare the results with standard GP and ABCP to enhance the symbolic regression, prediction and future selection abilities of solutions.

Acknowledgement

This project was supported by Scientific Research Project Foundation of Erciyes University (Project ID: FBA-12-4029).

References

[1] Alan.W. Biermann. *Automatic Programming: A Tutorial on Formal Methodologies*, J. Symbolic Computation, pp. 119-142, 1985. https://doi.org/10.1016/s0747-7171(85)80010-9

[2] John. R. Koza. *Genetic Programming: On the Programming of Computers by Means of Natural Selection*, MIT Press, Cambridge, MA, USA, 1992. https://doi.org/10.1007/BF00175355.

[3] Dervis Karaboga, Celal Ozturk, Nurhan Karaboga and Beyza Gorkemli. *Artificial bee colony programming for symbolic regression*, Information Sciences, 209, pp. 1–15, 2012. https://doi.org/10.1016/j.ins.2012.05.002

[4] Hossam Faris. *A Symbolic Regression Approach for Modeling the Temperature of Metal Cutting Tool*, International Journal of Control and Automation. 6(4), 2013.

[5] Panagiotis Barmpalexis, Kyriakos Kachrimanis, Athanasios Tsakonas, E. Georgarakis. *Symbolic regression via genetic programming in the optimization of a controlled release pharmaceutical formulation*, Chemometrics and Intelligent Laboratory Systems, 107, pp. 75–82, 2011. https://doi.org/10.1016/j.chemolab.2011.01.012

[6] Xin Li. *Self-Emergence of Structures in Gene Expression Programming*, Ph.D. Thesis, University of Illinois at Chicago, 2006.

[7] Petr Musilek, Adriel Lau, Marek Reformat and Loren Wyard-Scott, *Immune programming*, Information Sciences, 176, pp. 972–1002, 2006.

https://doi.org/10.1016/j.ins.2005.03.009.

[8] Mariusz Boryczka, *Ant colony programming: application of ant colony system to function approximation*, Intelligent Systems for Automated Learning and Adaptation: Emerging Trends and Applications, pp. 248–272, 2010. https://doi.org/10.4018/978-1-60566-798-0.ch011.

[9] Olivier Roux, Cyril Fonlupt . *Ant* programming or, how to use ants for automatic *programming*, Proceedings of ANTS'2000, pp. 121– 129, 2000.

[10] Shinichi Shirakawa, Shintaro Ogino, Tomoharu Nagao. T. Nagao, *Dynamic ant programming for automatic construction of programs*, IEEE Transactions on Electrical and Electronic Engineering, pp. 540–548, 2008. https://doi.org/10.1002/tee.20311.

[11] Essam.El. Seidy. *A New Particle Swarm Optimization Based Stock Market Prediction Technique*, International Journal of Advanced Computer Science and Applications (IJACSA), Vol. 7, No. 4, 2016. https://doi.org/10.14569/ijacsa.2016.070442.

[12] K.K. Manjusha, K. Sankaranayanan, P. Seena. *Data Mining in Dermatological Diagnosis: A Method for Severity Prediction*, International Journal of Computer Applications, (0975 – 8887) Vol. 117 – No.11, 2015.

[13] Sana BuHamra, Nejib Smaoui, Mahmoud Gabr. *The Box–Jenkins analysis and neural networks: prediction and time series modelling*, Applied Mathematical Modelling, Vol: 27,pp. 805–815, 2003. https://doi.org/10.1016/s0307-904x(03)00079-9.

[14] Gianluca Bontempi. *Machine Learning Strategies for Time Series Prediction, Machine Learning Group*, Computer Science Department Boulevard de Triomphe - CP 212, Hammamet, 2013. Retrieved from: http://www.ulb.ac.be/di.

[15] A. Martin, V. Aswathy, V.Prasanna Venkatesan. *Framing Qualitative Bankruptcy Prediction Rules Using Ant Colony Algorithm*, International Journal of Computer Applications, 0975 – 8887 ,Volume 41– No.21, 2012. https://doi.org/10.5120/5827-8143.

[16] Shuzhan Wan, Shengwu. Xiong and Yi Liu, *Prediction based multi-strategy differential evolution algorithm for dynamic environments*, Evolutionary Computation (CEC), 2012 IEEE Congress, pp. 10-15, 2012. https://doi.org/10.1109/cec.2012.6256628.

[17] Dandan .Li, Wanxin. Xue, Yilei Pei . *A high-precision prediction model using Ant Colony Algorithm and neural network*, International Conference on Logistics, Informatics and Service Sciences (LISS), 2015. https://doi.org/10.1109/liss.2015.7369696.

[18] Hossein Etemadi, Ali Asghar Anvary Rostamy, Hossan Farajzadeh Dehkordi . *A genetic programming model for bankruptcy prediction: Empirical evidence from Iran*, Expert Systems with Applications, Vol: 36, pp. 3199–3207, 2009. https://doi.org/10.1016/j.eswa.2008.01.012.

[19] Pamela Dominic, David Edward Leahy, Mark J. Willis. *Predicting the toxicity of chemical compounds using GPTIPS: a free open source genetic programming toolbox for MATLAB*, Intelligent Control and Computer Engineering, Lecture Notes in Electrical Engineering, Vol. 70, Springer, pp. 83-93, 2011. https://doi.org/10.1007/978-94-007-0286-8_8.

[20] Yudong Zhang, Shuihua Wanga, Preetha Phillips, Genlin Ji. *Binary PSO with mutation operator for feature selection using decision tree applied to spam detection*, Knowledge-Based Systems, Vol: 64, pp. 22–31,2014. https://doi.org/10.1016/j.knosys.2014.03.015.

[21] Mark A. Hall , *Correlation-based Feature Selection for Machine Learning*, PhD Thesis, The University of Waikato, 1999.

[22] Irene Rodriguez Lujan, Ramon Huerta, Charles Elkan, Carlos Santa Cruz. *Quadratic Programming Feature Selection*, Journal of Machine Learning Research, 11, pp. 1491-1516, 2010.

[23] Jasmina Novaković, Perica Strbac, Dusan Bulatović. *Toward Optimal Feature Selection Using Ranking Methods And Classification Algorithms*, Yugoslav Journal of Operations Research, Vol: 21, Number 1, pp. 119-135, 2011. https://doi.org/10.2298/yjor1101119n.

[24] Gavin Brown, Adam Pocock, Ming-Jie Zhao, Mikel Luj´an. *Conditional Likelihood Maximisation: A Unifying Framework for Information Theoretic Feature Selection*, Journal of Machine Learning Research, Vol: 13, pp. 27-66, 2012.

[25] Riyaz Sikora, Selwyn Piramuthu . *Framework For Efficient Feature Selection In Genetic Algorithm Based Data Mining*, European Journal of Operational Research, Vol:180, pp. 723–737, 2007. https://doi.org/10.1016/j.ejor.2006.02.040.

[26] Shital C. Shah, Andrew Kusiak. *Data mining and genetic algorithm based gene/SNP selection*, Artificial Intelligence in Medicine, Vol: 31, pp. 183—196, 2004. https://doi.org/10.1016/j.artmed.2004.04.002.

[27] Utpal Kumar Sikdar, Asif Ekbal, Sriparna Saha. *Differential Evolution based Feature Selection and Classifier Ensemble for Named Entity Recognition*, Proceedings of COLING 2012: Technical Papers. COLING 2012, Mumbai, December, pp. 2475–2490, 2012. https://doi.org/10.1007/s10032-011-0155-7.

[28] Yuanning Liu, Gang Wang, Huiling Chen, Hao Dong, X.iaodong Zhu, Sujing Wang . *An Improved Particle Swarm Optimization for Feature Selection*, Journal of Bionic Engineering, Vol: 8, 2011. https://doi.org/10.1016/s1672-6529(11)60020-6.

[29] Bing Xue, Mengijie Zhang, Will N. Browne. *Particle Swarm Optimization for Feature Selection in Classification: A Multi-Objective Approach*, IEEE Transactıons on Cybernetıcs, Vol. 43, No. 6, 2013. https://doi.org/10.1109/tsmcb.2012.2227469.

[30] Jianjun Yu, Jindan Yu, Arpit A. Almal, Saravana M.Dhanasekaran ,Debashis Ghosh, William P.Worzel, Arul M.Chinnaiyan. *Feature Selection and Molecular Classification of Cancer Using Genetic Programming*, Neoplasia, Vol. 9, No:4, pp. 292 – 303, 2007. https://doi.org/10.1593/neo.07121.

[31] Jacques-Andre Landry, Luis Da Costa and Thomas Bernier. *Discriminant Feature Selection by Genetic Programming: Towards a domain independent multi-class object detection system*, Systemics Cybernetics and Informatics, Vol: 3(1), pp. 76-81, 2006.

[32] Omer Abu-Arqub, Zaer Abo-Hammour, Shaher Mohammad Momani. *Application of Continuous Genetic Algorithm for Nonlinear System of Second-Order Boundary Value Problems*, Applied Mathematics& Information Sciences, 8, No.1, pp. 235-248, 2014. https://doi.org/10.12785/amis/080129.

[33] Omer Abu-Arqub, Zaer Abo-Hammour. *Numerical solution of systems of second-order boundary value problems using continuous genetic algorithm*, Information Sciences, Vol: 279, pp. 396-415, 2014. https://doi.org/10.1016/j.ins.2014.03.128.

[34] Riccardo Poli, William B. Langdon, Nicholas F. McPhee, John R. Koza, *A Field Guide to Genetic Programming*, 2016. http://cswww.essex.ac.uk/staff/rpoli/gp-field-guide/.

[35] Zhaohui Gan, Tommy W.SChow, W.N.Chau. Clone selection programming and its *application to symbolic regression*, Expert Systems with Applications, Vol: 36, 2009, pp. 3996–4005, 2009. https://doi.org/10.1016/j.eswa.2008.02.030.

[36] Hajira Jabeen, Abdul Rauf Baig. *Review of Classification Using Genetic Programming*, International Journal of Engineering Science and Technology, Vol: 2, pp. 94-103, 2010.

[37] Beyza Gorkemli. *Study of Artificial Bee Colony Programming (ABCP) to Symbolic Regression Problems*, PhD Thesis, Erciyes University, Engineering Faculty, Computer Engineering Department, 2015.

[38] Vladimir Cherkassky, Don Gehring, Filip Mulier. *Comparison of adaptive methods for function estimation from samples*, IEEE Transactions on Neural Networks, Vol: 7 (4), 1996, pp. 969- 984, 1996. https://doi.org/10.1109/72.508939

[39] Dominic P. Searson, *GPTIPS 2: an open-source software platform for symbolic data mining*. Chapter 22 in Handbook of Genetic Programming Applications, A.H. Gandomi et al., (Eds.), Springer, New York, NY, 2015., https:// sites.google.com/site/gptips4matlab/file-cabinet. https://doi.org/10.1007/978-3-319-20883-1_22

[40] UCI, *Machine Learning Repository*, Concrete Compressive Strength Data Set. https://archive.ics.uci.edu/ml/datasets/Concrete+Compressive+Strength, Access Date: 15.10.2016.

[41] Sibel Arslan, Celal Ozturk, *Multi Hive Artificial Bee Colony Programming for high dimensional symbolic regression with feature selection*, Applied Soft Computing Journal 78, 515–527, 2019. https://doi.org/10.1016/j.asoc.2019.03.014.

[42] Sibel Arslan, Celal Ozturk, *Artificial Bee Colony Programming Descriptor for Multi-Class Texture Classification*, Applied Sciences, 9(9), 2019. https://doi.org/10.3390/app9091930.

https://doi.org/10.31449/inf.v43i2.2303 Informatica **43** (2019) 291–298 **291**

Evolving Neural Network CMAC and its Applications

Oleg Rudenko, Oleksandr Bessonov and Oleksandr Dorokhov
Kharkiv National University of Economics, Nauka Ave 9a, 61166 Kharkiv, Ukraine
E-mail: aleks.dorokhov@meta.ua

Keywords: neural network, training, nonlinear object, identification, adaptive control

Received: April 21, 2018

The conventional neural network (NN) CMAC (Cerebellar Model Articulation Controller) can be applied in many real-world applications thanks to its high learning speed and good generalization capability. In this paper, it is proposed to utilize a neuro-evolutional approach to adjust CMAC parameters and construct mathematical models of nonlinear objects in the presence of the Gaussian noise. The general structure of the evolving NN CMAC (ECMAC) is considered. The paper demonstrates that the evolving NN CMAC can be used effectively for the identification of nonlinear dynamical systems. The simulation of the proposed approach for various nonlinear objects is performed. The results proved the effectiveness of the developed methods.

Povzetek: Razvit je postopek za evolucijsko iskanje najbolj prilagojene CMAC (Cerebellar Model Articulation Controller) nevronske mreže za probleme z Gaussovim šumom.

1 Introduction

Using a mathematical model of the cerebellar cortex developed by D. Marr [1] in 1975 J. Albus proposed a model describing the motion control processes that occur in the cerebellum, which was subsequently implemented in the neural network controller for controlling the robot - arm, which he called CMAC - Cerebellar Model Articulation Controller [2, 3]. Ease of implementation and a good network of approximating properties have ensured its wide usage not only in the tasks of controlling the robotic arm in real time, but also to solve many other practical problems [4-14].

However, it should be noted that in designing a network CMAC a number of difficulties in the selection of parameters such as the number of levels and the quantization levels, the shape of the receptive field, the type of applied information hashing algorithm and training. These parameters have a significant impact on the accuracy and speed of CMAC network, and therefore, the determination of the optimal values of these parameters is an important practical problem. In this article, for eliminating the drawbacks of traditional methods of synthesis and functioning ANN CMAC we provide the use of a new class of networks - evolving ANN (EANN) in which, in addition to traditional learning it is used another fundamental form of adaptation - evolution, realized by applying the evolutionary computation [15-18].

The use of two forms of adaptation in EANN - evolution, and training, allowing to change the network structure, its parameters and learning algorithms without external intervention, make the network data most suitable for work in non-stationary conditions and uncertainty about the properties of the object under study and the conditions of its functioning.

The main advantage of using evolutionary algorithms (EA) as learning algorithms is that many ANN parameters can be encoded in the genome and determined in parallel. Moreover, unlike most optimization algorithms designed to solve a problem, EA operate with a multitude of solutions - the population, which allows reaching a global minimum, without getting stuck in the local ones. In this case, information about each individual of the population is encoded in a chromosome (genotype), and the solution (phenotype) is obtained after evolution (selection, crossing, mutation) by decoding.

Among EAs that are stochastic and include evolutionary programming, evolutionary strategies, genetic algorithms, genetic programming, in particular, programming with gene expression, genetic algorithms (GA) are the most common [19,20]. GA abstract the fundamental processes of Darwinian evolution: natural selection and genetic changes due to recombination and mutation.

2 Neural network CMAC

The modification of the network proposed by Albus is shown in Figure 1. The network consists of the input, hidden and output layers, labeled L1, L2, L3, respectively, and uses two basic conversions:

$$S: X \Rightarrow A, \quad (1)$$

$$P: A \Rightarrow y, \quad (2)$$

where X - N-dimensional space of continuous input signals; A - n-dimensional space associations; y - a one-dimensional output.

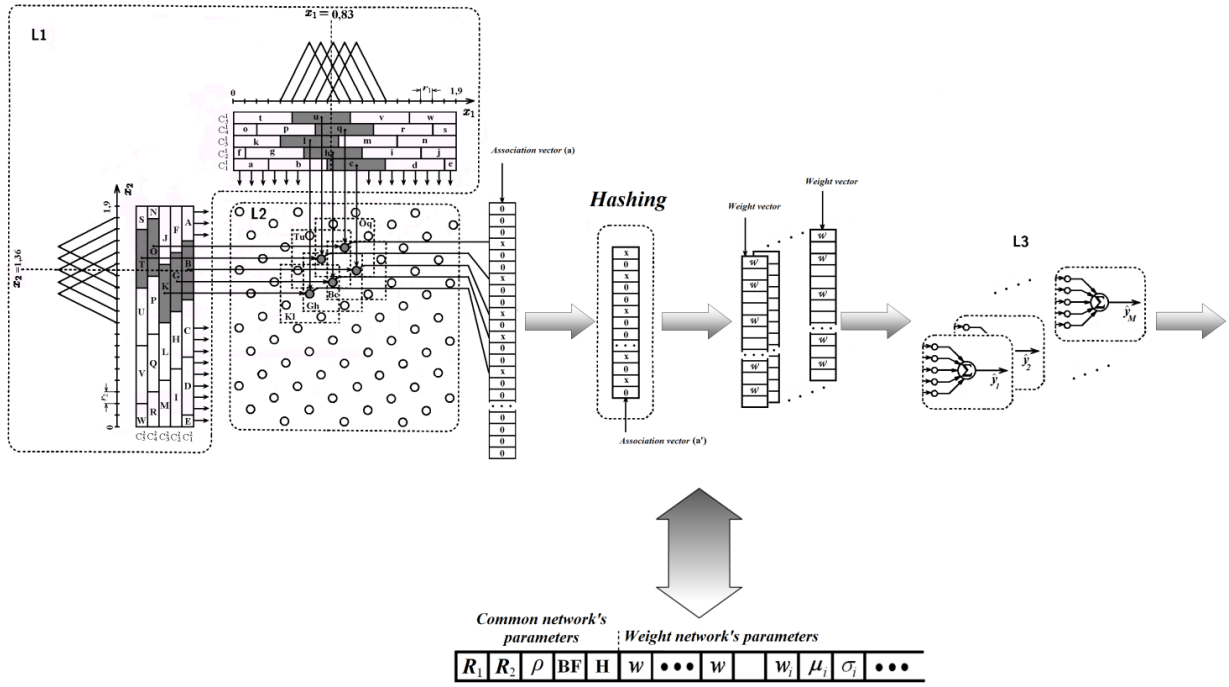


Figure 1: Albus network.

Converting $S \Rightarrow A$, in turn, consists of two transformations:

$$X \Rightarrow M \quad (3)$$

$$M \Rightarrow A, \quad (4)$$

where M - the space of binary variables.

The principle of the network operation as an associative memory is as follows. Approximated function *y = f (x)* is given to a limited number of points (argument values) **x** constituting N-dimensional space of the input signals. This space is divided into subspaces M formed the input signals **x**(*i*) ($i = \overline{1,M}$).

A number of subspaces M impacts the accuracy of the network and number of utilized memory cells. Therefore, on the one hand, side it should be big enough to ensure good approximation capabilities of the network and on the other hand side, it should be not too big to save some memory. In constructing the cerebellum model Albus proceeded from the fact that the appearance of the excitation signal activates its a certain area of the cerebellum, or receptive field, characterized by a parameter ρ.

Therefore, storage of values of **y**(*i*) (network output signal) corresponding to **x**(*i*) ($i = \overline{1,M}$), used ρ memory cells, the number of which is constant for all vectors of the input signals on the network. At receipt of the input signal ***x***(*i*) a signal ***y***(*i*) appears at network output, which is the sum of ρ addressable cells content.

Associative CMAC properties manifest themselves in the form of used addressing, which is based on a special coding input information and called hash coding or hashing [21-23].

3 Encoding information in CMAC

Information coding in the network means that to each N-dimensional input vector x(i) an n-dimensional association vector a(i), is assigned and stored in virtual memory.

Elements of **a**(*i*) can take the values from the interval [0, 1] (in the papers cited above it is assumed that these elements take the values 0 or 1). Thus only ρ << n elements of the vector have non-zero values, i.e. only ρ memory elements are active.

A continuous plurality of input signals by sampling (at the level of quantization) is converted into discrete. Thus to represent the *i*-th input signal components R_i quantization levels used with the appropriate quantization step r_i ($i = \overline{1,N}$). It should be noted that the accuracy of the system identification depends substantially on the size of the quantization step, and loss of stability is possible in digital automated control systems with an incorrect choice of this parameter.

Each stage is characterized by a corresponding association matrix A_i $\left(i = \overline{1,\rho}\right)$, only one element of which is different from zero.

Construction associations vector as follows. For a given total number of input signals association matrix A_i of each quantization stage ($i = \overline{1,\rho}$) are formed. The columns of these matrixes form association vectors $\boldsymbol{a}_i$ ($i = \overline{1,\rho}$).

The dimension of these vectors, *n*, equal to the sum of all elements of the matrices A_i ($i = \overline{1,\rho}$) and can be calculated by the formula:

$$n = \left\rfloor \rho\left(\frac{R-1}{\rho}+1\right)^{N} \right\lfloor, \quad (5)$$

where R - the number of used levels for quantizing input signals; N - the dimension of the input vector; $\rfloor\bullet\lfloor$ - means rounded to the nearest whole number.

Since all ρ matrices A_i ($i=\overline{1,\rho}$) have only one non-zero element, from the n components of the vector $\boldsymbol{a}(i)$ only ρ are non-zero.

The quantization region is arranged in such a way, that any of them relating to the adjacent stages have not more than (ρ - 1)-th connection. This corresponds to a restriction on the maximum total number of cells equal to (ρ - 1) used for storing two different vectors of the input signals in which their recognition is still possible.

4 Selecting the basic functions of neurons

Selecting the basic functions of neurons in L1 layer significantly affects the approximating properties of CMAC network.

As already noted, the traditional CMAC performs piecewise constant approximation, that is a consequence of the usage of neurons with a rectangular activation function.

When choosing rectangular basis functions computational cost will be minimal. Also, CMAC networks widely use B-splines as basis functions.

B-splines undoubted advantage is the possibility of recurrent calculating in both the splines in accordance with the formula [24-27]:

$$B_{n,j}(x) = \left[\frac{x-\lambda_{j-n}}{\lambda_{j-1}-\lambda_{j-n}}\right] B_{n-1,j-1}(x) + \left[\frac{\lambda_j - x}{\lambda_j - \lambda_{j-n+1}}\right] B_{n-1,j}(x) \quad (6)$$

and their derivatives δ -order:

$$^{(\delta)}B_{n,j}(x) = \frac{(n-1)}{(n-\delta-1)}\left[\frac{x-\lambda_{j-n}}{\lambda_{j-1}-\lambda_{j-n}}\right]^{(\delta)} B_{n-1,j-1}(x) + \frac{(n-1)}{(n-\delta-1)}\left[\frac{\lambda_j - x}{\lambda_j - \lambda_{j-n+1}}\right]^{(\delta)} B_{n-1,j}(x) \quad (7)$$

Here:

$$B_{0,j}(x) = \begin{cases} 1, \text{if } \ x \in [\lambda_{j-1}, \lambda_j); \\ 0, \text{otherwise;} \end{cases}$$

$$^{(0)}B_{0,j}(x) = \begin{cases} 1, \text{if } \ x \in [\lambda_{j-1}, \lambda_j); \\ 0, \text{otherwise;} \end{cases}$$

$$^{(\delta)}B_{n,j}(x) = \left[\frac{^{(\delta-1)}B_{n-1,j-1}(x)}{\lambda_{j-1}-\lambda_{j-n}}\right] - \left[\frac{^{(\delta-1)}B_{n-1,j}(x)}{\lambda_j - \lambda_{j-n+1}}\right] \quad (8)$$

where λ_j - j-th spline's node (center of the quantization field).

Thus, after determining the active slot $(\lambda_{j-1}, \lambda_j]$ for the zero order B-spline, these expressions can be used to obtain the values of all nonzero B-spline of higher order and, if appropriate, their derivatives.

Note that traditional CMAC uses zero order B-spline. Selection of the first order B-spline leads to the triangular membership function, and selection of the fourth-order B-spline leads to membership function similar to the Gaussian.

The CMAC network also uses Gaussian activation function of the form:

$$\Phi_i(x_j) = \exp\left\{-\frac{(\boldsymbol{x_j} - \boldsymbol{\mu}_i)^2}{\sigma^2}\right\} \quad (9)$$

As a basis one can use trigonometric functions, for example, cosine:

$$\Phi_i(x_j) = \begin{cases} \cos\left(\frac{\pi}{\rho r_j}(x_j - \lambda_i)\right) \text{if } \ x_j \in (\lambda_i - \frac{\rho r_j}{2}, \lambda_i + \frac{\rho r_j}{2}]; \\ 0 \qquad\qquad \text{otherwise,} \end{cases} \quad (10)$$

where λ_i - i-th center of the quantization field; r_j - quantization step of j-th component of the input signal.

However, it should be noted that although the most commonly used Gaussian membership functions also allow a very simple calculation of derivatives and have the property of a local activation, it is difficult to allocate clearly enough their activation boundary, which is often important for the implementation of the network that used, for example, scaling basis functions.

In order to eliminate this disadvantage, one can use a modified Gaussian function that has the following form:

$$\Phi_i(x) = \begin{cases} \exp\left\{-\frac{(\lambda_2 - \lambda_1)^2/4}{(x-\lambda_1)(\lambda_2 - x)}\right\} \text{if } \ x \in (\lambda_1, \lambda_2); \\ 0 \qquad\qquad \text{otherwise.} \end{cases} \quad (11)$$

As seen from the expression (11), the function is strictly defined in the range (λ_1, λ_2), which simplifies the process of scaling basis functions when the network parameters such as R and ρ are changing.

5 Network training

Defining the network parameters, i.e. in the general case defining a vector $\theta(k)$, that includes all network parameters (weights, parameters of basis functions, etc.) is accomplished by training with the teacher.

The training criterion can be presented as follows:

$$F[e(k)] = \sum_{i=1}^{k} \rho(e(i)), \quad (12)$$

where $\rho(e(i))$ - some loss function.

Gradient network training algorithm has the following form:

$$\hat{\theta}(k) = \hat{\theta}(k-1) + \gamma(k)\frac{\partial F(e(k))}{\partial \theta_j}, \quad (13)$$

or

$$\hat{\theta}(k) = \hat{\theta}(k-1) + \gamma(k)\rho'\left(e(k)\right)\frac{\partial e(k)}{\partial \theta_j} \quad (14)$$

where $\gamma > 0$ – parameter that affects the training speed and which can be selected differently for different network parameters.

Training of traditional CMAC that uses $\rho\left(e(i)\right) = 0,5e^2(i)$ and rectangular basis functions, occurs on each step after the presentation of training pairs $\{\boldsymbol{x}(k), y(k)\}$, where $y(k)$ – function's value, that corresponds $\boldsymbol{x}(k)$, and consists in the correction of only those of its ρ weights that correspond to the single components of the association vector for a given vector $\boldsymbol{x}(k)$.

In this case, the training algorithm for all i, j, for which $a_i(k) = a_j(k) = 1$, is the following:

$$w_j(k+1) = w_j(k) + \gamma\left(y(k) - \frac{1}{\rho}\sum_{i=1}^{n} w_i(k)\right), \quad (15)$$

where $\gamma \in (0,1]$ – parameter that affects the speed of training.

When membership functions with a form other than rectangular are used, this algorithm can be written as follows:

$$w(k+1) = w(k) + \gamma(k)\left(\frac{y(k) - a^T(k)\Phi(x)w(k)}{\left\|\Phi(x)a(k)\right\|^2}\Phi(x)a(k)\right), \quad (16)$$

where $\gamma(k)$ - in general case a variable parameter.

Multistep network training algorithms were considered in [7,8].

6 Evolving ANN CMAC

During switching from ANN to EANN for all types of networks the common evolutionary procedure (initialization population, an estimation of the population, selection, cross-breeding, mutations) is used. Differences are only in the method of encoding the structure and parameters of a particular form of ANN in the chromosome.

At the beginning of EA functioning, a population P_0 that consisting of N individuals (ANN): $P_0 = \{H_1, H_2, ..., H_N\}$ is randomly initialized. The proper choice of the N's value is very important as this parameter significantly affects the speed of the algorithm and its selection is critical for real-time systems.

Each individual in the population at the same time gets its own unique description, encoded in the chromosome $H_j = \{h_{1j}, h_{2j}, ...h_{Lj}\}$, which consists of L gene, wherein $h_{ij} \in [w_{\min} w_{\max}]$ - *i*-th value of *j*-gene chromosome ($w_{\min}$ - the minimum and $w_{\max}$ - maximum allowable values, respectively).

Figure 1 shows an example ECMAC chromosome's format and the correspondence between genes and network parameters stored in the chromosome. It should be noted that chromosome length depends on the dimensionality of the problem and the maximum amount of memory.

As seen from the drawing, it consists of a chromosome gene in which information about corresponding network parameters is stored. At the beginning of the chromosome, there are genes that contain information about the parameters of the noise and they are active only in case of the noisy measurements. Next gene's block encodes the number of levels and the quantization steps, the shape of the receptive field of neurons and type of algorithm that is used for hashing information.

Due to the large amount of the BF that can be used in CMAC, there is a special gene in its chromosome BF, that is responsible for coding the type of the used functions. There is also a gene H in the chromosome, that encodes a type of the hashing algorithm (If its value is set to 0 then hashing is not used).

Then, in the chromosome, there is a group of genes encoding weighting parameters directly relevant to the associative neurons. During the initialization phase, initial values are assigned to all these parameters by using a random number generator.

Since during evolution mutation may occur in the parameters affecting the amount of used associative neurons, the length of the chromosome can vary. The use of variable length chromosomes occur individuals with specific genetic code segments (introns) which are not used for coding characteristics [28-29].

Typically, introns are used in the EA:

- as noncoding bits that are uniformly added to the genetic code (in this case, introns only fill the space between the active genes of the chromosome);

- as the nonfunctional parts of the genetic code, i.e., parts of the decision which do not actually do anything, thus not affect the fitness of the chromosome (this usually occurs in the genetic programming and in the chromosomes, which are subject to the cycle of development after birth);

-as posterior useless part of the chromosome, which do not participate in the calculation of its fitness (usually it manifests itself in some types of competitive-trained neural networks, in which only neurons-winners in contrast to other neurons that are a posteriori useless affect network performance results).

Introns appear in other types of neural networks, where they are called potentially useful waste.

The control of introns amount in the population carried out by:

- the use of special operators, that alter the length of the chromosome and add or remove introns

(experimental results show that some number of introns improves overall properties EA);

- the use of the selection operator: depending on the number of introns, the value of individual fitness function will increase or decrease.

Once the initial population is formed, the fitness of each individual part in it evaluates by some defined fitness function.

Conventionally, as such a function the quadratic one is used:

$$F(x)=\frac{1}{M}\sum_{i=1}^{M}\left(y^{*}(x_i)-\hat{y}(x_i)\right)^2, \qquad (17)$$

where $y^{*}(k)$ - the desired network response; $\hat{y}(k)$ - real output signal; M – sample size.

The next step is the selection of individuals, the chromosomes of which are involved in the formation of the new generation, and subsequent hybridization.

The task of crossing operator (crossover) is the transfer of genetic information from the parent individuals to their offspring.

After completion of the operator's work, any gene of any individual in the new population may mutate, i.e. change its value.

Since chromosome uses hybrid coding, during the mutations various operations must be performed for different encoding methods.

For example, in the case of the gene that is responsible for neuron's activation and uses binary encoding, inverse mutation should be used.

For coding the BF and weighting parameters, that uses real values, different types of mutations may be used.

Thus ECMAC algorithm can be represented as follows:

- create an initial population (initialization of each individual chromosome, estimation of the initial population);

- the stages of evolution - the construction of a new generation (selection of candidates for mating /breeding, hybridization, i.e. causing by each pair of selected candidates some new individuals, mutation, evaluation of the new population);

- check the completion criterion, if not satisfied - go back to the stages of evolution.

7 First Modeling Experiment

It is considered the problem of nonlinear dynamic object identification that is described by the equation:

$$y(k)=\max\left[e^{-10u^2(k)};\ e^{-50y^2(k-1)};\right.$$
$$\left.;\ 1.25e^{-5\left(u^2(k)+y^2(k-1)\right)}\right]+\xi(k), \qquad (18)$$

where $u(k)$ - the input signal, that is a stationary random sequence with the random uniform distribution in the interval [-1, 1].

During the study of this object, the population of CMAC networks consisting of 100 individuals was used.

The population evolved during 500 epochs. All configurable network parameters (including R and ρ) were determined by EA.

It should be noted that the values of R and ρ determine the amount of memory that is used for storing network parameters and significantly affect the accuracy of the approximation.

Graphs of the network-winner fitness function and the required amount of memory to store its parameters are shown in Figures 2 and 3.

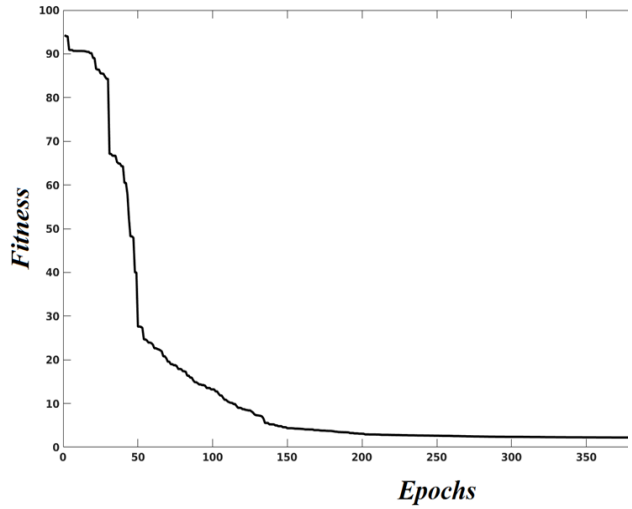


Figure 2: The network-winner fitness function.

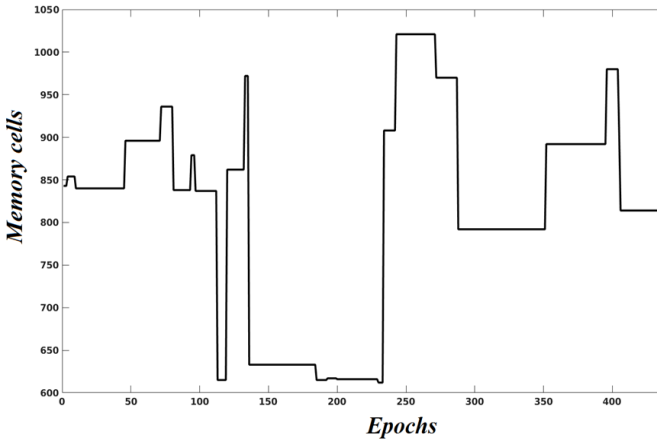


Figure 3: Required amount of memory.

Results of the stationary object (18) identification are shown in Figures 4 and 5.

So, Figure 4 shows the surface itself, according to the equation described, and Figure 5 - the surface recovered by ECMAC with ξ (k) = 0.

The winning network comprises 814 weighting parameters with R = 186 and ρ = 95, and rectangular activation function was chosen.

Figure 6 shows the results of the object (18) identification in the presence of the random noise ξ (k) that is normally distributed in the interval [-0.3, 0.3].

In this case, the winning network used cosine activation functions (10).

$y(k)$

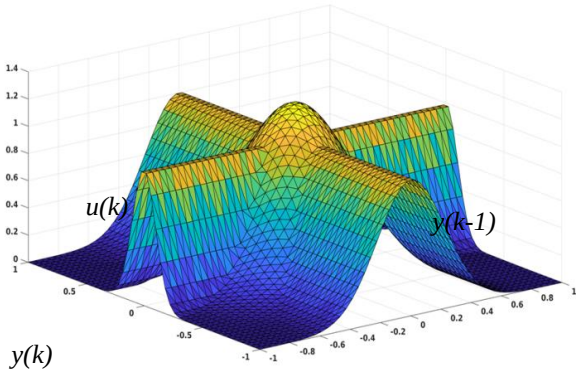


Figure 4: The surface itself, according to the equation described.

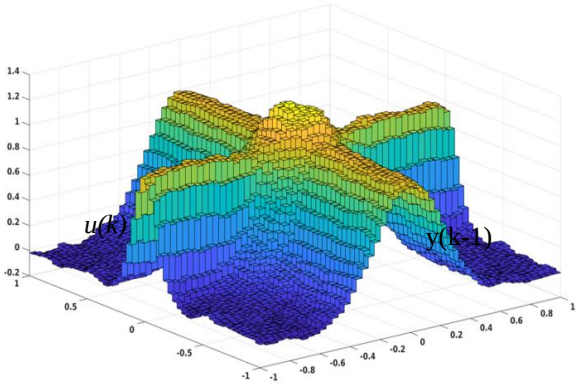


Figure 5: The surface recovered by ECMAC.

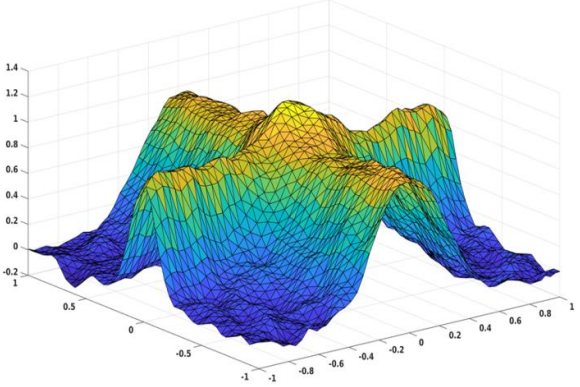


Figure 6: The results of the object (13) identification.

8 Second Modeling Experiment

We solved the problem of the identification of a dynamic object described by the equation:

$$y(k)=\frac{y(k-1)y(k-2)y(k-3)y(k-4)(y(k-3)-1)}{1+u(k)^2+y(k-2)^2}+$$

$$+\frac{u(k)}{1+u(k)^2+y(k-2)^2} \qquad (19)$$

To solve this problem, we used a population of evolving CMAC networks comprising 250 individuals.

After reaching the required accuracy of identification, to assess the quality of the resulting model, to the object and the winner network the same control actions were given:

a) $u(k)=\sin(\pi k/200)+\cos(\pi k/400)$;

b) $u(k)=-1.5+0.001k$.

The experimental results for the cases a) and b) are shown in Figure 7 and 8.

The solid line shows the output signal of the object and the dashed - neural network model output.

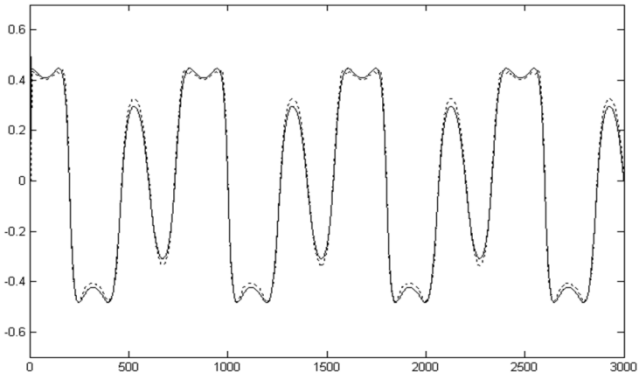


Figure 7: The experimental results for the a)-case.

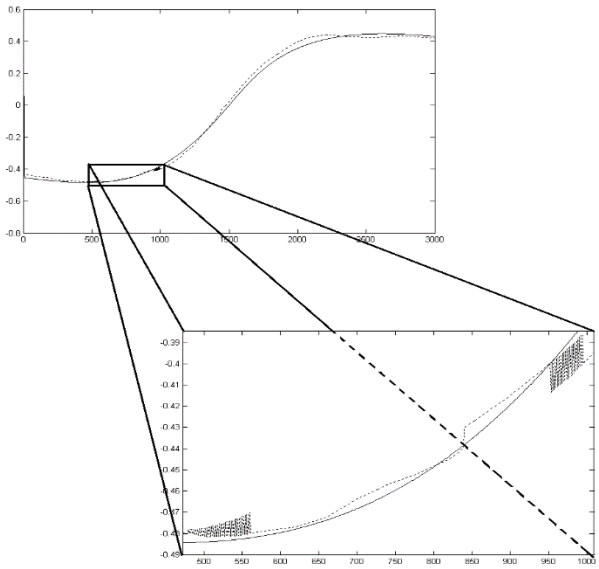


Figure 8: The experimental results for the b)-case.

As can be seen from the simulation results, the accuracy of identification of a multidimensional object (19) via evolving networks CMAC is sufficiently high and error – is inessential.

In Figure 8 there are some minor oscillations that appeared due to rounding off calculations.

9 Third Modeling Experiment

It is considered the problem of controlling a multidimensional nonlinear dynamic object described by the following equations:

$$y_1(k)=\frac{y_1(k-1)}{1+y_2(k-1)^2}+u_1(k-1);$$
$$y_2(k)=\frac{y_1(k-1)y_2(k-1)}{1+y_2(k-1)^2}+u_2(k-1). \quad (20)$$

Uncorrelated random sequences with a uniform distribution law in the interval [–1, 1] were used as inputs of the network during the training.

Training was carried out on the basis of the presentation of a network of 10000 training pairs and the following parameters of the CMAC neural network: activation functions - trigonometric; R=100; ρ=40.

The required memory capacity for such parameters is 5818 memory cells. The size of the population was 300 individuals.

The required values of the output signals were set in the following way:

$$y_1^*(k)=\sin(\pi k/100);$$
$$y_2^*(k)=\begin{cases}0.5, k=\overline{1,500};\\ \sin(\pi k/300)+\sin(\pi k/500), k=\overline{501,1000}.\end{cases} \quad (21)$$

The results of the neural control are shown in Figures 9 and 10.

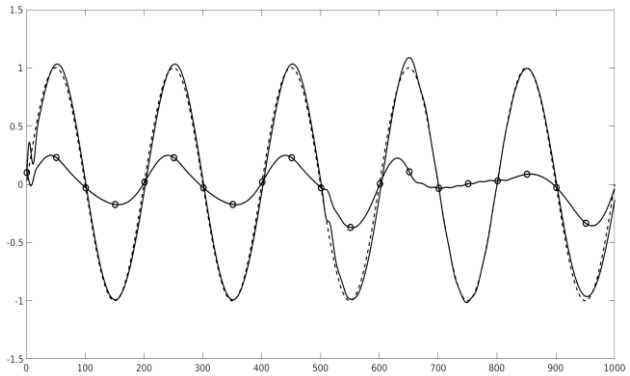


Figure 9: The experimental results for the $y_1(k)$

In all these figures, the dotted line shows the required output signal $y_i^*(k)$, solid line – real $\hat{y}_i(k)$, and the line with the circles - the corresponding change of the control signal $u_i(k)$ $(i=1,2)$.

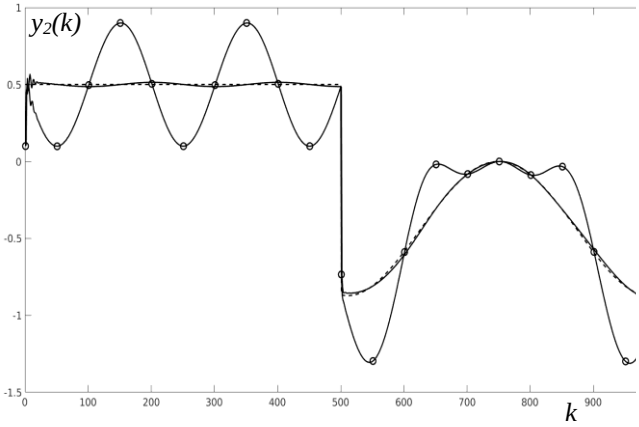


Figure 10: The experimental results for the $y_2(k)$.

10 Conclusions

The results showed that the evolving neural network CMAC is quite effective and convenient in solving practical problems (identification of nonlinear objects, control, etc).

Substantial savings of required memory in combination with evolutionary training algorithms make it particularly attractive for implementation in real complex dynamic object control systems in the presence of noisy measurements.

An additional advantage of the evolutionary approach to CMAC network training is the solution of the problem of choice the receptive field form of the associative neurons that is affecting the method and the accuracy of the approximation of the studied functions.

In the case ECMAC this problem is solved automatically.

References

[1] Marr, D. (1969). Theory of Cerebellar Cortex. *Journal Physiology*, Vol. 202, 437-470.

[2] Albus, J. (1975). A new approach to manipulator control: the cerebellar model articulation controller (CMAC). *J. Dynamic Systems, Measurement, and Control*, Vol. 97, №3, 220-227. https://doi.org/10.1115/1.3426922

[3] Albus, J. (1975). Data storage in cerebellar model articulation controller (CMAC). *J. Dynamic Systems, Measurement and Control*, Vol. 97, №3, 228-233. https://doi.org/10.1115/1.3426923

[4] Miller, W., Glanz, F., Kraft, L. (1990). CMAC: An associative neural network alternative to backpropagation. *Proc. of the IEEE*, Vol. 78, №10, 1561–1567. https://doi.org/10.1109/5.58338

[5] Miller, T., Hewes, R., Glanz, F., Kraft, L. (1990). Real-time dynamic control of industrial manipulator using a neural-network-based learning controller. *IEEE Trans. Robot. Automat.*, Vol. 6, 1-9. https://doi.org/10.1109/70.88112

[6] Iigumi, Y. (1996). Hierarchical image coding via cerebral model arithmetic computers. I*EEE Trans. Image Processing*, Vol. 5, 1393-1401. https://doi.org/10.1109/83.536888

[7] Avdeyan, E., Hormel, M. (1991). The increase of the rate of convergence of the learning process in a special system of associative memory. *Automation and telemechanics*, Vol. 6, 1-9.

[8] Rudenko, O., Bessonov, A. (2005). CMAC Neural Network and Its Use in Problems of Identification and Control of Nonlinear Dynamic Objects. *Cybernetics and Systems Analysis*, Vol.41, Issue 5, 647–658. https://doi.org/10.1007/s10559-006-0002-x

[9] Li, H.-Y., Yeh, R.-G., Lin, Y.-C., Lin, L.-Y., Zhao, J., Rudas, I. (2016). Medical Sample Classifier Design Using Fuzzy Cerebellar Model Neural Networks. *Acta polytechnica Hungarica*, Vol. 13, №6, 7-24.

https://doi.org/10.12700/aph.13.6.2016.6.1

[10] Shafik, A., Abdelhameed, M., Kassem ,A. (2014). CMAC Based Hybrid Control System for Solving Electrohydraulic System Nonlinearities. *Int. Journal of Manufacturing, Materials, and Mechanical Engineering,* Vol.4(2), 20-26. https://doi.org/10.4018/ijmmme.2014040104

[11] Lee, C.-H., Chang, F.-Y. (2014). An Efficient Interval Type-2 Fuzzy CMAC for Chaos Time-Series Prediction and Synchronization. *IEEE Transactions on Cybernetics*, Vol. 44, №3, 329-341. https://doi.org/10.1109/tcyb.2013.2254113

[12] Chung, C.-C., Lin, C.-C. (2015). Fuzzy Brain Emotional Cerebellar Model Articulation Control System Design for Multi-Input Multi-Output Nonlinear. *Acta Polytechnica Hungarica*, Vol. 12, № 4. 39-58. https://doi.org/10.12700/aph.12.4.2015.4.3

[13] Dorokhov, O., Chernov, V., Dorokhova, L., Streimkis, J. (2018). Multi-criteria choice of alternatives under fuzzy information, *Transformations in Business and Economics*, Vol. 2, 95-106.

[14] Malyarets L., Dorokhov, O., Dorokhova L. (2018). Method of constructing the fuzzy regression model of bank competitiveness. Journal of Central Banking Theory and Practice, Vol. 7, №2, 139–164. https://doi.org/10.2478/jcbtp-2018-0016

[15] Xu, S., Jing, Y. (2016). Research and Application of the Pellet Grate Thickness Control System Base on Improved CMAC Neural Network Algorithm. *Journal of Residuals Science & Technology*, Vol. 13, № 6, 1501-1509.

[16] Floreano, D., Mattiussi, C. (2008). *Bio-Inspired Artificial Intelligence Theories, Methods, and Technologies.* The MIT Press Cambridge, Massachusetts-London, England.

[17] Andries, P. (1997). *Engelbrecht Computational Intelligence An Introduction.* John Wiley & Sons.

[18] Yao, X. (1993). A Review of Evolutionary Artificial Neural Networks. *Int. J. Intell. Syst.*, №8 (4), 539-567.

[19] Yao, X. (1999). Evolving Artificial Neural Networks. *Proc. of the IEEE*, Vol. 87, №9,1423-1447. https://doi.org/10.1109/5.784219

[20] Holland, J. (1975). *Adaptation in Natural and Artificial Systems. An Introductory Analysis With Application to Biology, Control and Artificial Intelligence.* University of Michigan.

[21] Goldberg, D. (1989). *Genetic Algorithms in Search, Optimization and Machine Learning.* Addison-Wesley, MA.

[22] Knuth, D. (1973). *Sorting and Searching, in the Art of Computer Programming*. Menlo Park, Calif., Addison Wesley.

[23] Wang, Z.-Q., Schiano, J., Ginsberg, M. (1996). Hash-Coding in CMAC Neural Networks. *IEEE Int. Conf. on Neural Networks*, Vol. 3, 1698-1703. https://doi.org/10.1109/icnn.1996.549156

[24] Rudenko, O., Bessonov, O.(2004). Hashing information in a neural network CMAC. *Control Systems and Machines*, №5, 67-73.

[25] Chiang, C.-T., Lin, C.-S. (1996). CMAC with General Basis Functions. *Neural Networks*, Vol. 7, №7, 1199-1211.

[26] Lane, S., Handelman, D., Gelfand, J. (1992). Theory and Development of Higher-Order CMAC Neural Networks. *IEEE Control Systems*, Vol. 12, № 2, 23-30. https://doi.org/10.1109/37.126849

[27] Rudenko, O., Bessonov, O. (2004). On the Choice of Basis Functions in a Neural Network CMAC. *Problems of Control and Informatics*, № 2, 143–154.

[28] Wu, A.(1995). Empirical Studies of the Genetic Algorithm with Non-Coding Segments. *Evolutionary Computation*, Vol. 3(2), 121-147.

[29] Castellano, J. (2001). *Scrapping or Recycling: the Role of Chromosome Length-Altering Operators in Genetic Algorithms.* GeNeura Group, Department of Architecture and Computer Technology, University of Granada. (2001).

https://doi.org/10.31449/inf.v43i2.2783 Informatica **43** (2019) 299–303 **299**

Design of Intelligent English Writing Self-evaluation Auxiliary System

Man Liu
School of Foreign Languages, Changchun Institute of Technology, Changchun, Jilin 130012, China
E-mail: manliu_lm@aliyun.com

Keywords: English writing, self-evaluation, auxiliary system, writing teaching

Received: May 23, 2019

Since the reform and opening up, the exchanges between China and the world have become more and more frequent. English, as a widely used international language, plays an important role in international exchanges. English teaching includes five aspects, listening, speaking, reading, writing and translation. Writing teaching is very important but difficult. In order to improve students' autonomous writing ability, this paper briefly introduced the real-time multi-writing teaching mode and designed an automatic scoring algorithm of writing self-evaluation auxiliary system, random sampling based Bayesian classification and combinational algorithm. One thousand CET-4 and CET-6 compositions from Chinese Learner English Corpus (CLEC) were evaluated, and the scoring effect of Bayesian classification algorithm was also tested. The results showed that the accuracy rate, recall rate and F value of the proposed algorithm was better than that of Bayesian classification algorithm under 150 feature extraction dimensions, the two algorithms had improved scoring effect under the optimal feature extraction dimensions, and the improvement amplitude of the algorithm proposed in this study was larger. In summary, the random sampling based Bayesian classification and combinational algorithm is effective and feasible as an automatic scoring algorithm of writing self-evaluation auxiliary system.

Povzetek: Za boljše učenje angleščine je za kitajske študente razvita vrsta pripomočkov v obliki informacijskih storitev.

1 Introduction

As the economic globalization deepens, the communication between China and other countries is more and more frequent, and the most frequently used language is English. English teaching includes five aspects, listening, speaking, reading, writing and translation, among which writing teaching is the most important and difficult part [1]. English writing ability can be improved through a large number of writing exercises and comments of teachers. However, the ratio of the number of teachers to the number of students is very small in China. Teachers can not provide guidance to all students. If students lack guidance about writing exercises, the improvement effect will be greatly reduced. Because of the rapid development of computer and natural language research, a computer intelligence based self-correction system has been developed [2]. Despite the fact that the writing language is changeable and the effect is not ideal in practice, it is still of great help to lighten the burden on teachers and improve students' English writing level. Deng [3] put forward a design method of cloud service platform based intelligent English writing auxiliary teaching system, constructed the overall design model of English writing teaching system, improved the intelligence level of the English writing teaching system using Cloud-P2P fusion model, and found that the English writing teaching system had favourable adaptability, learning ability and reliability. Li [4] selected two parallel classes as the object, one was taught by the traditional business English writing teaching mode and the other was guided by computer assisted technology. The teaching quality was evaluated after one academic term, and it was found that computer assisted technology had positive effect on business English writing. Tsai [5] applied the blackboard course management system in English essay writing teaching and found that the teaching result of the experimental group was superior to that of the control group after two academic years. The questionnaire result suggested that most of the students had positive learning result, indicating the teaching mode could improve the effectiveness of English writing learning. This paper briefly introduced the real-time multi-writing teaching mode and designed the automatic scoring algorithm of the writing self-evaluation assistant system. One thousand CET-4 and CET-6 compositions from Chinese Learner English Corpus (CLEC) were scored. The scoring effect of the Bayesian classification algorithm was also tested and compared with the automatic scoring algorithm.

2 English writing teaching mode

The traditional teaching mode in China is mostly "duck-stuffing", which is similar to assembly line. Most colleges and universities regard English writing teaching as a part of English teaching, or as a subsidiary part, and moreover it is not paid much attentions to because of time and energy waste. The traditional teaching of English writing is usually conducted in the classroom, but teachers usually do not pay attention to whether students understand or not and only provide students with template sentences and simple explanation [6].

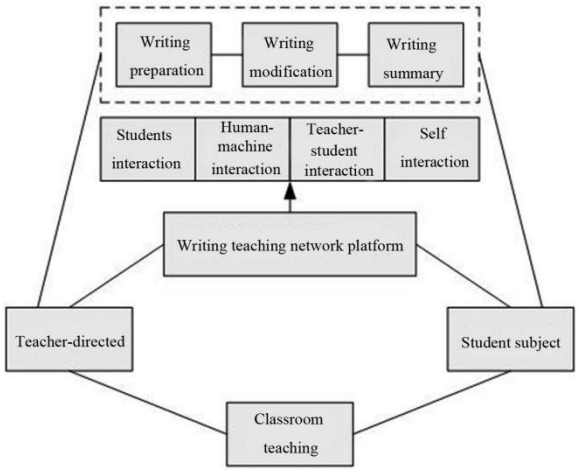


Figure 1: The flow of real-time multi-writing teaching.

Guided by the theory of constructivism [7], a new teaching mode, real-time multi-writing teaching mode [8] has been proposed, and its flow chart is shown in Figure 1. The whole writing process is divided into 3 parts, writing preparation, writing modification, and writing summarization. Teachers and students participate in the whole process. Teachers take students as the center to teach in the classroom and on the network platform. Classroom teaching follows the principle of student-oriented to implement the traditional teaching mode. On the network platform, student interaction, human-computer interaction, teacher-student interaction and self-interaction can be achieved because of the convenient Internet. Besides the Internet, the achievement of the above interaction also relies on the writing self-evaluation system. Through the objective evaluation of computer and based on the evaluation of students and teachers, comments and review comments are obtained.

3 Writing self-evaluation system

3.1 General structure of the system

The general structure of the system is shown in Figure 2. The system in this paper is a Web system based on B/S mode [9]. The client used by user runs in the browser, while the business function of the system runs on the server. The overall structure of the system is divided into 3 parts, user interface layer, business logic layer and data layer. The user interface is the web browser; the business logic layer contains all the functions of the system, and the data layer contains the data needed to run the system. Automatic scoring is the main function of the system.

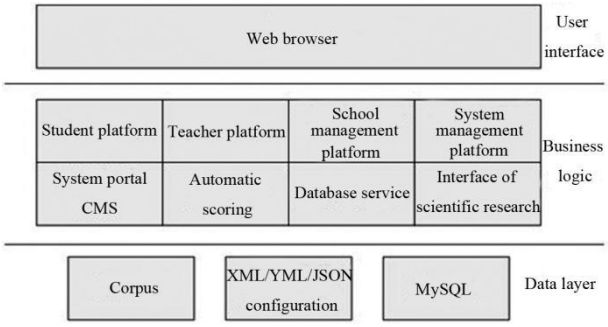


Figure 2: The overall structure of the system.

3.2 Automatic scoring

3.2.1 Feature extraction

The features of compositions with different scores needed to be extracted through training corpus before use to facilitate the classification of compositions to be tested [10]. In this study, the information gain method was used to extract the features of compositions. The expression of the information gain method [11] is:

$$IG(t) = -\sum_{i=1}^{k} P(a_i)\log(P(a_i)) + P(t)\sum_{i=1}^{k} P(a_i|t)\log P(a_i|t) + P(\bar{t})\sum_{i=1}^{k} P(a_i|\bar{t})\log P(a_i|\bar{t}) \quad (1)$$

where t is the feature of adjacent binary phrase, a_i is the set of compositions with the i-th score, $\bar{t}$ stands for the condition in case of absence of feature t, $P(a_i)$ stands for the possibility of score a_i in the training corpus, $P(t)$ stands for the possibility of composition containing feature t in the training corpus, $P(a_i|t)$ stands for the possibility of composition containing feature t and with score a_i, $P(\bar{t})$ stands for the possibility of composition not containing feature t, $P(a_i|\bar{t})$ stands for possibility of composition which is scored as a_i but not contains feature t, k stands for the number of score grade, 4 here (grade 1: 1 ~ 5 points; grade 2: 6 ~ 9 points; grade 3: 10 ~ 13 points; grade 4: 13 ~ 14 points).

3.2.2 Random sampling and Bayesian classification based composition scoring algorithms

The flow of the scoring algorithm [12] is shown in Figure 3. $Y = \arg\max \sum_{1\le i\le m} I(\alpha = y_i)$ (2), where α refers to different score grades, y_i refers to the classification result of a random sampling, $I(\bullet)$ refers to indicator function, 1 if the parameter is true and 0 if not.

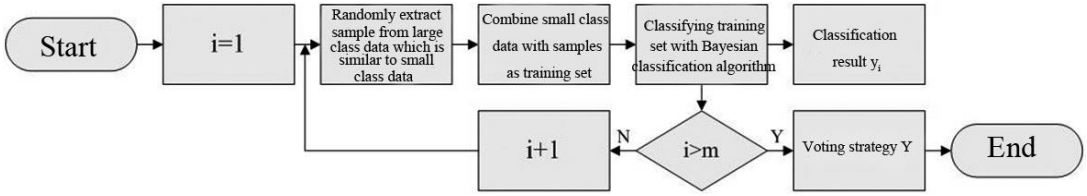


Figure 3: The flow of random sampling and Bayesian classification based composition scoring algorithms.

The calculation formula of Bayesian classification based algorithm [13] is:

$$P(b|a_i)=\prod_{k=1}^{V}\frac{P(\theta_k|a_i)^{N_k}}{N_k!} \quad (3)$$

$$P(\theta_j|a_i)=\frac{N_{ji}+1}{N_{a_i}+|V|} \quad (4)$$

where b refers to the spatial vector of an English composition, $b=\{\theta_1,\theta_2,\cdots,\theta_i,\cdots\theta_m\}$, m refers to the number of features of adjacent binary phrase, θ_i refers to the weight of the i-th feature in composition b, V refers to the number of binary phase features, N_k refers to the number of times of the k-th feature appearing in b, $P(b|a_i)$ refers to the possibility of a composition obtaining some score, $P(\theta_j|a_i)$ refers to the possibility of feature θ_j in a composition which is scored as a_i, N_{ji} refers to the number of times of the j-th feature appearing in a composition which is scored as the j-th feature, and N_{a_i} refers to the number of all the features of a composition which is scored as a_i.

4 Automatic scoring test of the writing self-evaluation system

4.1 Testing method

Five hundred of CET-4 compositions which involved two themes and 500 CET-6 compositions with which involved two themes were selected from CLEC [14]. Before the test, the compositions were scored and classified into four score grades according to the scoring criteria of CET-4 and CET-6 compositions. Based on the binary phrase features, the compositions which involved four themes were scored under 150 feature extraction dimensions, i.e., the 1000 compositions were classified into four score grades using the algorithm proposed in this study. Then the compositions were scored after the automatic calculation of optimal feature dimensions. The whole system program ran on a server in a lab. The server was configured with Windows 7 system, I7 processor and 16G memory. In order to increase the persuasiveness of the results, the test results of the Bayesian classification algorithm based scoring system was selected for comparison. The test method was the same as the system proposed in this study.

4.2 Evaluation criteria

The automatic scoring effect of the writing self-evaluation system was evaluated in the aspects of accuracy rate, recall rate and F value [15]. The accuracy rate could be calculated using the following formula: the accuracy rate = correctly recognized number/total recognized number, where correctly recognized number is the number of correctly classified compositions based on above scoring algorithm and the total recognized number is the total number of compositions identified by the scoring algorithm.

The recall rate could be calculated using the following formula: recall rate = correctly recognized number/actually existed number, where actually existed number refers to the number of compositions which actually existed and ought to be recognized.

F value (comprehensive evaluation index) could be calculated using the follow formula: F value = 2 × accuracy rate × recall rate /(accuracy rate + recall rate).

4.3 Testing results

As shown in Table 1, the accuracy rate, recall rate and F value of the scoring algorithm were about 14%, 28% and 22% higher than those of Bayesian classification algorithm in scoring different themes of compositions. The accuracy rate, recall rate and F value of the Bayesian classification algorithm were 0.739, 0.661 and 0.690 respectively; the accuracy rate, recall rate and F value of the algorithm proposed in this study were 0.845, 0.850 and 0.844 respectively. It indicated that the algorithm proposed in this study was superior to the Bayesian classification algorithm in scoring compositions, and all

Theme	Bayesian classification algorithm			Random sampling based Bayesian classification algorithm		
	Accuracy rate (P)	Recall rate (R)	F value	Accuracy rate (P)	Recall rate (R)	F value
Theme 1 (CET-4)	0.719	0.626	0.660	0.849	0.881	0.862
Theme 2 (CET-4)	0.750	0.679	0.708	0.838	0.833	0.832
Theme 3 (CET-6)	0.759	0.710	0.730	0.836	0.831	0.831
Theme 4 (CET 6)	0.726	0.630	0.661	0.857	0.853	0.851
Average value	0.739	0.661	0.690	0.845	0.850	0.844

Table 1: The scoring test results of the two algorithms under 150 feature extraction dimensions.

Theme	Optimal feature dimension number	Bayesian classification algorithm			Random sampling based Bayesian classification algorithm		
		Accuracy rate (P)	Recall rate (R)	F value	Accuracy rate (P)	Recall rate (R)	F value
Theme 1 (CET-4)	390	0.856	0.846	0.838	0.990	0.989	0.989
Theme 2 (CET-4)	350	0.784	0.778	0.763	0.989	0.988	0.988
Theme 3 (CET 6)	710	0.878	0.866	0.857	1	0.999	0.999
Theme 4 (CET 6)	530	0.844	0.795	0.739	0.987	0.986	0.986
Average value		0.841	0.821	0.799	0.992	0.991	0.991

Table 2. The scoring test results of the two algorithms under optimal feature extraction dimensions.

the indicators were above 80%. Based on the binary phrase features extracted from the compositions, the algorithm could accurately classify the tested composition into the corresponding score grade and make accurate and reasonable evaluation on the content of compositions based on the score grade.

As shown in Table 2, the optimal feature extraction dimension of theme 1, 2, 3 and 4 compositions was 390, 350, 710 and 530 respectively. Considering the evaluation index values of the four themes, the accuracy rate, recall rate and F value of Bayesian classification algorithm were 0.841, 0.821 and 0.799 respectively under the optimal feature extraction dimension, and the accuracy rate, recall rate and F value of the algorithm proposed in this study were 0.992, 0.991 and 0.991 respectively.

As shown in Figure 4, the accuracy, recall rate and F value of the algorithm under the optimal feature extraction dimensions were better than those under 150 feature extraction dimensions. Moreover the three indicators of the algorithm proposed in this study were always the best, and the improvement of the algorithm was greater after changing feature extraction dimensions. That is to say, after the application of optimal feature extraction dimension, the algorithm proposed in this study could classify the compositions more accurately according to the binary phrase features, and make more accurate and reasonable evaluation according to the score grade.

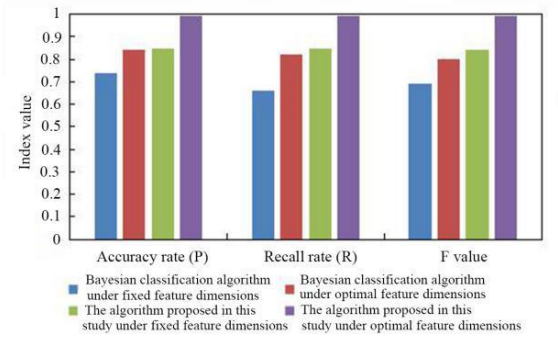


Figure 4: The comparison of testing results of the two algorithms under the fixed and optimal feature extraction dimensions.

5 Conclusion

This paper briefly introduced the real-time multi-writing teaching model and designed an automatic scoring algorithm for the writing self-evaluation system. Then 1000 CET-4 and CET-6 compositions in CLEC were scored according to four grades. As a comparison, the scoring effect of the Bayesian classification algorithm was tested. The results showed that the accuracy rate, recall rate and F value of the Bayesian classification algorithm were 0.739, 0.661 and 0.690 respectively, and the corresponding data of the algorithm proposed in this study were 0.845, 0.850 and 0.844 respectively, indicating that the scoring effect of the algorithm proposed in this study was superior to that of the Bayesian classification algorithm. Under the optimal feature extraction dimensions, the accuracy rate, recall rate and F value of the Bayesian classification algorithm were 0.841, 0.821 and 0.799 respectively and those of the algorithm proposed in this study were 0.992, 0.991 and 0.991 respectively, which were improved compared to under the fixed feature extraction dimensions. Moreover the improvement amplitude of the algorithm proposed in this study was larger.

Acknowledgement

This study was supported by General Project of Higher Education Department of Education Department of Jilin Province in 2018, Research on English “Four-in-One” Teaching Reform in College English under the Background of Internet+, Key Project of Teaching Reform of Changchun University of Engineering, June 2018 to June 2020, under researching.

References

[1] Nattapong J, Rattanavich S (2015). The effects of computer-assisted instruction based on top-level structure method in English reading and writing abilities of Thai EFL students. *English Language*

Teaching, 8(11), pp. 231. https://doi.org/10.5539/elt.v8n11p231

[2] Xia M, Zhang Y, Zhang C (2017). A TAM-based approach to explore the effect of online experience on destination image: A smartphone user's perspective. *Journal of Destination Marketing & Management*. https://doi.org/10.1016/j.jdmm.2017.05.002

[3] Deng L (2018). Design of English writing assisted instruction teaching system based on intelligent cloud service platform. *International Conference on Intelligent Transportation, Big Data & Smart City*. IEEE Computer Society, pp. 287-290. https://doi.org/10.1109/ICITBS.2018.00080

[4] Li X (2018) Influence of computer-aided instruction model on business English writing teaching effect. *International Journal of Emerging Technologies in Learning*, 13(3), pp. 197.

[5] Tsai Y R (2015). Applying the technology acceptance model (TAM) to explore the effects of a course management system (CMS)-assisted EFL writing instruction. *Calico Journal*, 32(1). https://doi.org/10.1558/cj.v32i1.153-171

[6] Tarhini A, Hassouna M, Abbasi M S, et al. (2015). Towards the acceptance of RSS to support learning: an empirical study to validate the technology acceptance model in Lebanon. *Electronic Journal of e-Learnin*g, 13(1), pp. 30-41.

[7] Li L (2016). Design and implementation of higher vocational English writing and training testing system. *International Conference on Materials Engineering, Manufacturing Technology and Control*.

[8] Chapela M E G (2016). A corpus-based analysis of post-auxiliary ellipsis in modern English: methodological and theoretical issues.

[9] Ohta R (2017). The impact of an automated evaluation system on student-writing performance. *Kate Bulletin*, 22, pp. 23-33.

[10] Ahsen M E, Ayvaci M, Raghunathan S (2017). When algorithmic predictions use human-generated data: a bias-aware classification algorithm for breast cancer diagnosis. *Social Science Electronic Publishing*. https://doi.org/10.2139/ssrn.3087467

[11] Alcázar V, Fernández S, Borrajo D, et al. (2015). Using random sampling trees for automated planning. *Ai Communications*, 28(4), pp. 665-681.

[12] Myttenaere A D, Golden B, Grand B L, et al. (2015). Study of a bias in the offline evaluation of a recommendation algorithm. *Computer Science*, 79(3), pp. 263-72. https://doi.org/10.1016/0002-9416(81)90074-9

[13] Hao J, Niu K, Meng Z, et al. (2017). A collaborative filtering recommendation algorithm based on score classification. *International Conference on Security, Privacy and Anonymity in Computation, Communication and Storage. Springer, Cham*, pp. 435-445. https://doi.org/10.1007/978-3-319-72395-2_40

[14] Santos D P D, André C P L F D C (2017). Automatic selection of learning bias for active sampling. *Intelligent Systems*. IEEE, Recife, Brazil, pp. 55-60. https://doi.org/10.1109/BRACIS.2016.021

[15] Zoph B, Yuret D, May J, et al. (2016). Transfer learning for low-resource neural machine translation. *Conference on Empirical Methods in Natural Language Processing*, pp. 1568-1575.

JOŽEF STEFAN INSTITUTE

Jožef Stefan (1835-1893) was one of the most prominent physicists of the 19th century. Born to Slovene parents, he obtained his Ph.D. at Vienna University, where he was later Director of the Physics Institute, Vice-President of the Vienna Academy of Sciences and a member of several scientific institutions in Europe. Stefan explored many areas in hydrodynamics, optics, acoustics, electricity, magnetism and the kinetic theory of gases. Among other things, he originated the law that the total radiation from a black body is proportional to the 4th power of its absolute temperature, known as the Stefan–Boltzmann law.

The Jožef Stefan Institute (JSI) is the leading independent scientific research institution in Slovenia, covering a broad spectrum of fundamental and applied research in the fields of physics, chemistry and biochemistry, electronics and information science, nuclear science technology, energy research and environmental science.

The Jožef Stefan Institute (JSI) is a research organisation for pure and applied research in the natural sciences and technology. Both are closely interconnected in research departments composed of different task teams. Emphasis in basic research is given to the development and education of young scientists, while applied research and development serve for the transfer of advanced knowledge, contributing to the development of the national economy and society in general.

At present the Institute, with a total of about 900 staff, has 700 researchers, about 250 of whom are postgraduates, around 500 of whom have doctorates (Ph.D.), and around 200 of whom have permanent professorships or temporary teaching assignments at the Universities.

In view of its activities and status, the JSI plays the role of a national institute, complementing the role of the universities and bridging the gap between basic science and applications.

Research at the JSI includes the following major fields: physics; chemistry; electronics, informatics and computer sciences; biochemistry; ecology; reactor technology; applied mathematics. Most of the activities are more or less closely connected to information sciences, in particular computer sciences, artificial intelligence, language and speech technologies, computer-aided design, computer architectures, biocybernetics and robotics, computer automation and control, professional electronics, digital communications and networks, and applied mathematics.

The Institute is located in Ljubljana, the capital of the independent state of S**love**nia (or S♡nia). The capital today is considered a crossroad between East, West and Mediterranean Europe, offering excellent productive capabilities and solid business opportunities, with strong international connections. Ljubljana is connected to important centers such as Prague, Budapest, Vienna, Zagreb, Milan, Rome, Monaco, Nice, Bern and Munich, all within a radius of 600 km.

From the Jožef Stefan Institute, the Technology park "Ljubljana" has been proposed as part of the national strategy for technological development to foster synergies between research and industry, to promote joint ventures between university bodies, research institutes and innovative industry, to act as an incubator for high-tech initiatives and to accelerate the development cycle of innovative products.

Part of the Institute was reorganized into several high-tech units supported by and connected within the Technology park at the Jožef Stefan Institute, established as the beginning of a regional Technology park "Ljubljana". The project was developed at a particularly historical moment, characterized by the process of state reorganisation, privatisation and private initiative. The national Technology Park is a shareholding company hosting an independent venture-capital institution.

The promoters and operational entities of the project are the Republic of Slovenia, Ministry of Higher Education, Science and Technology and the Jožef Stefan Institute. The framework of the operation also includes the University of Ljubljana, the National Institute of Chemistry, the Institute for Electronics and Vacuum Technology and the Institute for Materials and Construction Research among others. In addition, the project is supported by the Ministry of the Economy, the National Chamber of Economy and the City of Ljubljana.

Jožef Stefan Institute
Jamova 39, 1000 Ljubljana, Slovenia
Tel.:+386 1 4773 900, Fax.:+386 1 251 93 85
WWW: http://www.ijs.si
E-mail: matjaz.gams@ijs.si
Public relations: Polona Strnad

INFORMATICA

AN INTERNATIONAL JOURNAL OF COMPUTING AND INFORMATICS

INVITATION, COOPERATION

Submissions and Refereeing

Please register as an author and submit a manuscript at: http://www.informatica.si. At least two referees outside the author's country will examine it, and they are invited to make as many remarks as possible from typing errors to global philosophical disagreements. The chosen editor will send the author the obtained reviews. If the paper is accepted, the editor will also send an email to the managing editor. The executive board will inform the author that the paper has been accepted, and the author will send the paper to the managing editor. The paper will be published within one year of receipt of email with the text in Informatica MS Word format or Informatica LaTeX format and figures in .eps format. Style and examples of papers can be obtained from http://www.informatica.si. Opinions, news, calls for conferences, calls for papers, etc. should be sent directly to the managing editor.

SUBSCRIPTION

Please, complete the order form and send it to Dr. Drago Torkar, Informatica, Institut Jožef Stefan, Jamova 39, 1000 Ljubljana, Slovenia. E-mail: drago.torkar@ijs.si

Since 1977, Informatica has been a major Slovenian scientific journal of computing and informatics, including telecommunications, automation and other related areas. In its 16th year (more than twentyfive years ago) it became truly international, although it still remains connected to Central Europe. The basic aim of Informatica is to impose intellectual values (science, engineering) in a distributed organisation.

Informatica is a journal primarily covering intelligent systems in the European computer science, informatics and cognitive community; scientific and educational as well as technical, commercial and industrial. Its basic aim is to enhance communications between different European structures on the basis of equal rights and international refereeing. It publishes scientific papers accepted by at least two referees outside the author's country. In addition, it contains information about conferences, opinions, critical examinations of existing publications and news. Finally, major practical achievements and innovations in the computer and information industry are presented through commercial publications as well as through independent evaluations.

Editing and refereeing are distributed. Each editor can conduct the refereeing process by appointing two new referees or referees from the Board of Referees or Editorial Board. Referees should not be from the author's country. If new referees are appointed, their names will appear in the Refereeing Board.

Informatica web edition is free of charge and accessible at http://www.informatica.si.
Informatica print edition is free of charge for major scientific, educational and governmental institutions. Others should subscribe.

Informatica WWW:

http://www.informatica.si/

Referees from 2008 on:

A. Abraham, S. Abraham, R. Accornero, A. Adhikari, R. Ahmad, G. Alvarez, N. Anciaux, R. Arora, I. Awan, J. Azimi, C. Badica, Z. Balogh, S. Banerjee, G. Barbier, A. Baruzzo, B. Batagelj, T. Beaubouef, N. Beaulieu, M. ter Beek, P. Bellavista, K. Bilal, S. Bishop, J. Bodlaj, M. Bohanec, D. Bolme, Z. Bonikowski, B. Bošković, M. Botta, P. Brazdil, J. Brest, J. Brichau, A. Brodnik, D. Brown, I. Bruha, M. Bruynooghe, W. Buntine, D.D. Burdescu, J. Buys, X. Cai, Y. Cai, J.C. Cano, T. Cao, J.-V. Capella-Hernández, N. Carver, M. Cavazza, R. Ceylan, A. Chebotko, I. Chekalov, J. Chen, L.-M. Cheng, G. Chiola, Y.-C. Chiou, I. Chorbev, S.R. Choudhary, S.S.M. Chow, K.R. Chowdhury, V. Christlein, W. Chu, L. Chung, M. Ciglarič, J.-N. Colin, V. Cortellessa, J. Cui, P. Cui, Z. Cui, D. Cutting, A. Cuzzocrea, V. Cvjetkovic, J. Cypryjanski, L. Čehovin, D. Čerepnalkoski, I. Čosić, G. Daniele, G. Danoy, M. Dash, S. Datt, A. Datta, M.-Y. Day, F. Debili, C.J. Debono, J. Dedič, P. Degano, A. Dekdouk, H. Demirel, B. Demoen, S. Dendamrongvit, T. Deng, A. Derezinska, J. Dezert, G. Dias, I. Dimitrovski, S. Dobrišek, Q. Dou, J. Doumen, E. Dovgan, B. Dragovich, D. Drajic, O. Drbohlav, M. Drole, J. Dujmović, O. Ebers, J. Eder, S. Elaluf-Calderwood, E. Engström, U. riza Erturk, A. Farago, C. Fei, L. Feng, Y.X. Feng, B. Filipič, I. Fister, I. Fister Jr., D. Fišer, A. Flores, V.A. Fomichov, S. Forli, A. Freitas, J. Fridrich, S. Friedman, C. Fu, X. Fu, T. Fujimoto, G. Fung, S. Gabrielli, D. Galindo, A. Gambarara, M. Gams, M. Ganzha, J. Garbajosa, R. Gennari, G. Georgeson, N. Gligorić, S. Goel, G.H. Gonnet, D.S. Goodsell, S. Gordillo, J. Gore, M. Grčar, M. Grgurović, D. Grosse, Z.-H. Guan, D. Gubiani, M. Guid, C. Guo, B. Gupta, M. Gusev, M. Hahsler, Z. Haiping, A. Hameed, C. Hamzaçebi, Q.-L. Han, H. Hanping, T. Härder, J.N. Hatzopoulos, S. Hazelhurst, K. Hempstalk, J.M.G. Hidalgo, J. Hodgson, M. Holbl, M.P. Hong, G. Howells, M. Hu, J. Hyvärinen, D. Ienco, B. Ionescu, R. Irfan, N. Jaisankar, D. Jakobović, K. Jassem, I. Jawhar, Y. Jia, T. Jin, I. Jureta, Đ. Juričić, S. K, S. Kalajdziski, Y. Kalantidis, B. Kaluža, D. Kanellopoulos, R. Kapoor, D. Karapetyan, A. Kassler, D.S. Katz, A. Kaveh, S.U. Khan, M. Khattak, V. Khomenko, E.S. Khorasani, I. Kitanovski, D. Kocev, J. Kocijan, J. Kollár, A. Kontostathis, P. Korošec, A. Koschmider, D. Košir, J. Kovač, A. Krajnc, M. Krevs, J. Krogstie, P. Krsek, M. Kubat, M. Kukar, A. Kulis, A.P.S. Kumar, H. Kwaśnicka, W.K. Lai, C.-S. Laih, K.-Y. Lam, N. Landwehr, J. Lanir, A. Lavrov, M. Layouni, G. Leban, A. Lee, Y.-C. Lee, U. Legat, A. Leonardis, G. Li, G.-Z. Li, J. Li, X. Li, X. Li, Y. Li, Y. Li, S. Lian, L. Liao, C. Lim, J.-C. Lin, H. Liu, J. Liu, P. Liu, X. Liu, X. Liu, F. Logist, S. Loskovska, H. Lu, Z. Lu, X. Luo, M. Luštrek, I.V. Lyustig, S.A. Madani, M. Mahoney, S.U.R. Malik, Y. Marinakis, D. Marinčič, J. Marques-Silva, A. Martin, D. Marwede, M. Matijašević, T. Matsui, L. McMillan, A. McPherson, A. McPherson, Z. Meng, M.C. Mihaescu, V. Milea, N. Min-Allah, E. Minisci, V. Mišić, A.-H. Mogos, P. Mohapatra, D.D. Monica, A. Montanari, A. Moroni, J. Mosegaard, M. Moškon, L. de M. Mourelle, H. Moustafa, M. Možina, M. Mrak, Y. Mu, J. Mula, D. Nagamalai, M. Di Natale, A. Navarra, P. Navrat, N. Nedjah, R. Nejabati, W. Ng, Z. Ni, E.S. Nielsen, O. Nouali, F. Novak, B. Novikov, P. Nurmi, D. Obrul, B. Oliboni, X. Pan, M. Pančur, W. Pang, G. Papa, M. Paprzycki, M. Paralič, B.-K. Park, P. Patel, T.B. Pedersen, Z. Peng, R.G. Pensa, J. Perš, D. Petcu, B. Petelin, M. Petkovšek, D. Pevec, M. Pičulin, R. Piltaver, E. Pirogova, V. Podpečan, M. Polo, V. Pomponiu, E. Popescu, D. Poshyvanyk, B. Potočnik, R.J. Povinelli, S.R.M. Prasanna, K. Pripužić, G. Puppis, H. Qian, Y. Qian, L. Qiao, C. Qin, J. Que, J.-J. Quisquater, C. Rafe, S. Rahimi, V. Rajkovič, D. Raković, J. Ramaekers, J. Ramon, R. Ravnik, Y. Reddy, W. Reimche, H. Rezankova, D. Rispoli, B. Ristevski, B. Robič, J.A. Rodriguez-Aguilar, P. Rohatgi, W. Rossak, I. Rožanc, J. Rupnik, S.B. Sadkhan, K. Saeed, M. Saeki, K.S.M. Sahari, C. Sakharwade, E. Sakkopoulos, P. Sala, M.H. Samadzadeh, J.S. Sandhu, P. Scaglioso, V. Schau, W. Schempp, J. Seberry, A. Senanayake, M. Senobari, T.C. Seong, S. Shamala, c. shi, Z. Shi, L. Shiguo, N. Shilov, Z.-E.H. Slimane, F. Smith, H. Sneed, P. Sokolowski, T. Song, A. Soppera, A. Sorniotti, M. Stajdohar, L. Stanescu, D. Strnad, X. Sun, L. Šajn, R. Šenkeřík, M.R. Šikonja, J. Šilc, I. Škrjanc, T. Štajner, B. Šter, V. Štruc, H. Takizawa, C. Talcott, N. Tomasev, D. Torkar, S. Torrente, M. Trampuš, C. Tranoris, K. Trojacanec, M. Tschierschke, F. De Turck, J. Twycross, N. Tziritas, W. Vanhoof, P. Vateekul, L.A. Vese, A. Visconti, B. Vlaovič, V. Vojisavljević, M. Vozalis, P. Vračar, V. Vranić, C.-H. Wang, H. Wang, H. Wang, H. Wang, S. Wang, X.-F. Wang, X. Wang, Y. Wang, A. Wasilewska, S. Wenzel, V. Wickramasinghe, J. Wong, S. Wrobel, K. Wrona, B. Wu, L. Xiang, Y. Xiang, D. Xiao, F. Xie, L. Xie, Z. Xing, H. Yang, X. Yang, N.Y. Yen, C. Yong-Sheng, J.J. You, G. Yu, X. Zabulis, A. Zainal, A. Zamuda, M. Zand, Z. Zhang, Z. Zhao, D. Zheng, J. Zheng, X. Zheng, Z.-H. Zhou, F. Zhuang, A. Zimmermann, M.J. Zuo, B. Zupan, M. Zuqiang, B. Žalik, J. Žižka,

Informatica

An International Journal of Computing and Informatics

Web edition of Informatica may be accessed at: http://www.informatica.si.

Subscription Information Informatica (ISSN 0350-5596) is published four times a year in Spring, Summer, Autumn, and Winter (4 issues per year) by the Slovene Society Informatika, Litostrojska cesta 54, 1000 Ljubljana, Slovenia.
The subscription rate for 2019 (Volume 43) is
– 60 EUR for institutions,
– 30 EUR for individuals, and
– 15 EUR for students
Claims for missing issues will be honored free of charge within six months after the publication date of the issue.

Typesetting: Borut Žnidar, borut.znidar@gmail.com.
Printing: ABO grafika d.o.o., Ob železnici 16, 1000 Ljubljana.

Orders may be placed by email (drago.torkar@ijs.si), telephone (+386 1 477 3900) or fax (+386 1 251 93 85). The payment should be made to our bank account no.: 02083-0013014662 at NLB d.d., 1520 Ljubljana, Trg republike 2, Slovenija, IBAN no.: SI56020830013014662, SWIFT Code: LJBASI2X.

Informatica is published by Slovene Society Informatika (president Niko Schlamberger) in cooperation with the following societies (and contact persons):
Slovene Society for Pattern Recognition (Vitomir Štruc)
Slovenian Artificial Intelligence Society (Mitja Luštrek)
Cognitive Science Society (Olga Markič)
Slovenian Society of Mathematicians, Physicists and Astronomers (Marej Brešar)
Automatic Control Society of Slovenia (Nenad Muškinja)
Slovenian Association of Technical and Natural Sciences / Engineering Academy of Slovenia (Mark Pleško)
ACM Slovenia (Borut Žalik)

Informatica is financially supported by the Slovenian research agency from the Call for co-financing of scientific periodical publications.

Informatica is surveyed by: ACM Digital Library, Citeseer, COBISS, Compendex, Computer & Information Systems Abstracts, Computer Database, Computer Science Index, Current Mathematical Publications, DBLP Computer Science Bibliography, Directory of Open Access Journals, InfoTrac OneFile, Inspec, Linguistic and Language Behaviour Abstracts, Mathematical Reviews, MatSciNet, MatSci on SilverPlatter, Scopus, Zentralblatt Math

Volume 43 Number 2 June 2019 **ISSN 0350-5596**

Informatica

An International Journal of Computing and Informatics

Informatica **43** (2019) Number 2, pp. 151–305