Volume 36 Number 3 September 2012 ISSN 0350-5596

Informatica

An International Journal of Computing and Informatics

1977

Editorial Boards, Publishing Council

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.

The State-of-the-Art in Visual Object Tracking

Anand Singh Jalal
Department of Computer Engineering & Applications
GLA University, Mathura, India - 281406
E-mail: anandsinghjalal@gmail.com

Vrijendra Singh
Indian Institute of Information Technology, Allahabad, India - 211012
E-mail: vrij@iiita.ac.in

Overview paper

Keywords: object tracking, object modelling, moving object detection

Received: May 27, 2011

There is a broad range of applications of visual object tracking that motivate the interests of researchers worldwide. These include video surveillance to know the suspicious activity, sport video analysis to extract highlights, traffic monitoring to analyse traffic flow and human computer interface to assist visually challenged people. In general, the processing framework of object tracking in dynamic scenes includes the following stages: segmentation and modelling of interesting moving object, predicting possible location of candidate object in each frame, localization of object in each frame, generally through a similarity measure in feature space. However, tracking an object in a complex environment is a challenging task. This survey discusses some of the core concepts used in object tracking and present a comprehensive survey of efforts in the past to address this problem. We have also explored wavelet domain and found that it has great potential in object tracking as it provides a rich and robust representation of an object.

Povzetek: Podan je pregled metod vizualnega sledenja objektov .

1 Introduction

As the technology is advancing with rapid pace, the cost of video cameras and digital media storage is affordable. In recent years, we have seen a remarkable increase in the amount of video data recorded and stored around the world. In order to process all these video data, there is a growing demand of automatically analyze and understand the video contents. One of the most fundamental processes in understanding video contents is visual object tracking, which is the process of finding the location and dynamic configuration of one or more moving objects in each frame (image) of a video [1].

There is a broad range of applications of object tracking that motivate the interests of researchers worldwide. Video surveillance is a very popular one. Surveillance systems are not only for recording the observed visual information, but also extracting motion information and, more recently, to analyze suspicious behaviors in the scene [2]. One can visually track airplanes, vehicles, animals, micro-organisms or other moving objects, but detecting and tracking people is of great interest. For instance, vision-based people-counting applications can provide important information for public transport, traffic congestion, tourism, retail and security tasks. Tracking humans is also an important step for human-computer interaction (HCI) [3]. Video can also be processed to obtain the story, to group similar frames into shots, shots into scenes or to retrieve information of interest.

The objective of this overview paper is to explore the different approaches and provide comprehensive descriptions of different methods used for object detection and tracking. Our aim is to introduce recent advances in visual object tracking as well as identifying future trends. The remainder of the paper is structured as follows. Section 2 discusses the approaches related to object modeling. Section 3 presents motion detection including modeling of environments and shadow removal. Section 4 describes the different prediction methods. Section 5 reviews the work related to object tracking. Section 6 discusses the evaluation measures and datasets used for evaluation and comparison of object tracking methods. Finally, section 7 presents concluding remarks and future directions.

1.1 Problem Definition

Object tracking itself is the task of following one or more objects in a scene, from their first appearance to their exit [4]. An object may be anything of interest within the scene that can be detected, and depends on the requirements of the application. Given a sequence of image frames to trace a set of objects, which are subimages, in each frame. In general, in a dynamic environment both background and object are allowed to vary. In principle, to solve this general unconstrained problem is hard. One can put a set of constraints to make this problem solvable. The more the constraints, the problem is easier to solve. Some of the constraints that generally imposed during object tracking are:

- Object motion is smooth with no abrupt changes
- No sudden changes in the background
- Gradual changes in the appearance of object
- Fixed camera
- Number and size of objects
- Limited amount of occlusion

Let denotes the image at time. Then, a video can be defined as the concatenation of images during different times, as following:

$$\upsilon = \{I_t : t = 1..T\}$$

where T is the time frame when the video stop.

Figure 1 shows the general structure and representation of a video. In a color video, each frame consists of three components: R, G and B, while in a grayscale video, each frame has a single component.

There are two important facts which require attention for object tracking in video:

1. Video is a temporal sequence of image frames and video coding generally encode the temporal relationship. However, when image frames are regenerated, each frame exists independently of each other and their temporal relationships can be seen visually and are to be derived again, if needed.
2. Objects are embedded in the background and both are part of the image (frame), which is generally represented as an array of pixels. The spatial relationship, which groups pixels as object, are to be derived explicitly, may be in each frame.

Deviation of these spatio-temporal relationships forms the core of all object tracking algorithms.

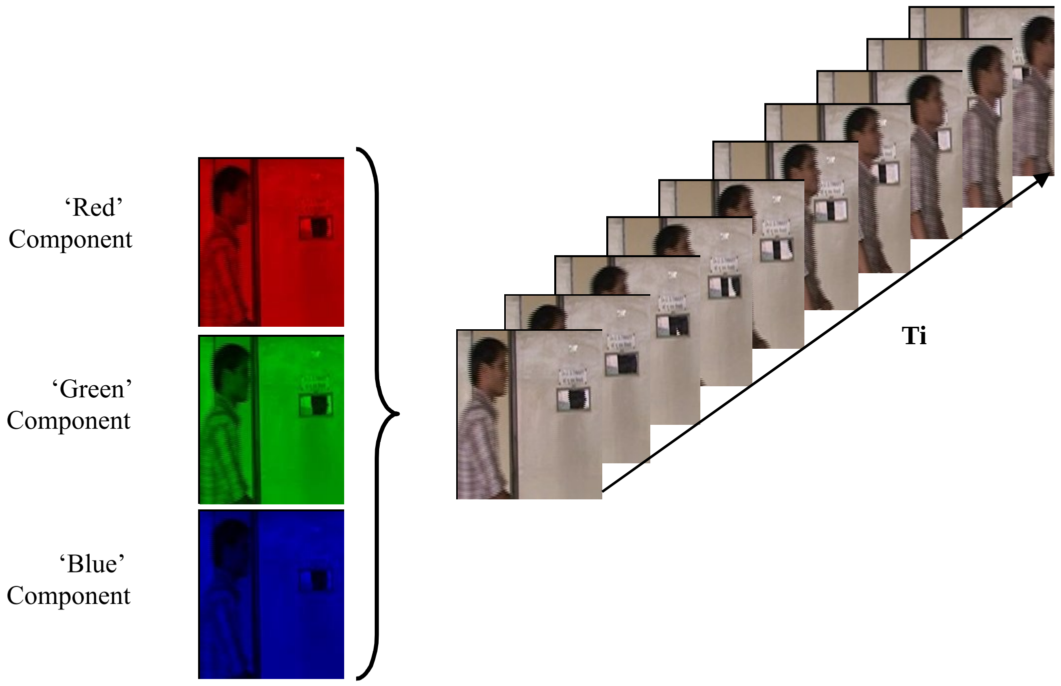


Figure 1: Video structure and representation.

1.2 Major Issues in Object Tracking

Many approaches have been proposed in the literature for object tracking. These approaches can be distinguished based on the way they handle the issues: i) segmentation algorithm is to extract moving objects in a video; ii) object representation for robust object tracking; iii) image features used to detect object in the feature space iv) handling of occlusion and v) the motion modeling. Some of the fundamental open problems in object tracking are abrupt object motion, noise in the image sequences, changes in scene illumination, changing appearance patterns of the object and the scene, object-to-object and object-to-scene occlusions, non-rigid object structures, camera motion and real time processing requirements. There are a number of issues involved in the development of a robust object tracking system, which needs to be understood.

Object Modeling is an important issue in visual object tracking. One of the major tasks of object modeling is to find an appropriate visual description that makes the object distinguished from other objects and background.

Changes in appearance and shape are issues that should also be considered during visual object tracking. The appearance of an object can vary as camera angle changes. Deformable objects such as human can change their shape and appearance during different video frame sequences. The appearance and shape can also change due to perspective effect i.e. objects farther from the camera appears smaller than those near to the camera.

Handling ***illumination changes*** is also one of the challenging issues for visual object tracking. The appearance of an object can largely affected by illumination changes. An object may look different in indoor environment (artificial light) than outdoor environment (sun light). Even the time of day (morning, afternoon, evening) and weather conditions i.e. cloudy, sunny etc. can be the causes of illumination changes.

Shadows and reflections are also difficult to handle during object tracking. Some of the features such as motion, shape and background are more sensitive for a shadow on the ground which behaves and appears like the object that casts it. Same kind of problem can be caused by reflections of moving objects on smooth surfaces.

Occlusion is also very important issue for visual object tracking. Occlusion occurs either due to one object is occluded by another object or an object is occluded by some component of the background. During occlusion, an ambiguity occurs in the objects and their features. The tracking methods must be capable to resolve the individuality of the objects involved in the occlusion, before and after the occlusion takes place.

The issues mentioned above are significant to both single-object tracking and multi-object tracking. However, multi-object tracking also requires to resolve some other issues e.g. modeling the multiple objects. Tracking method should be able to distinguish different objects in order to keep them consistently labeled. Although during the last few years, there has been a substantial progress towards moving object detection and tracking. But tracking an object in an unconstraint, noisy and dynamic environment still makes this problem a central focus of research interest.

1.3 Typical Object Tracking Architecture

The visual object tracking field relies on three modules that interact with each other to perform robust object tracking.

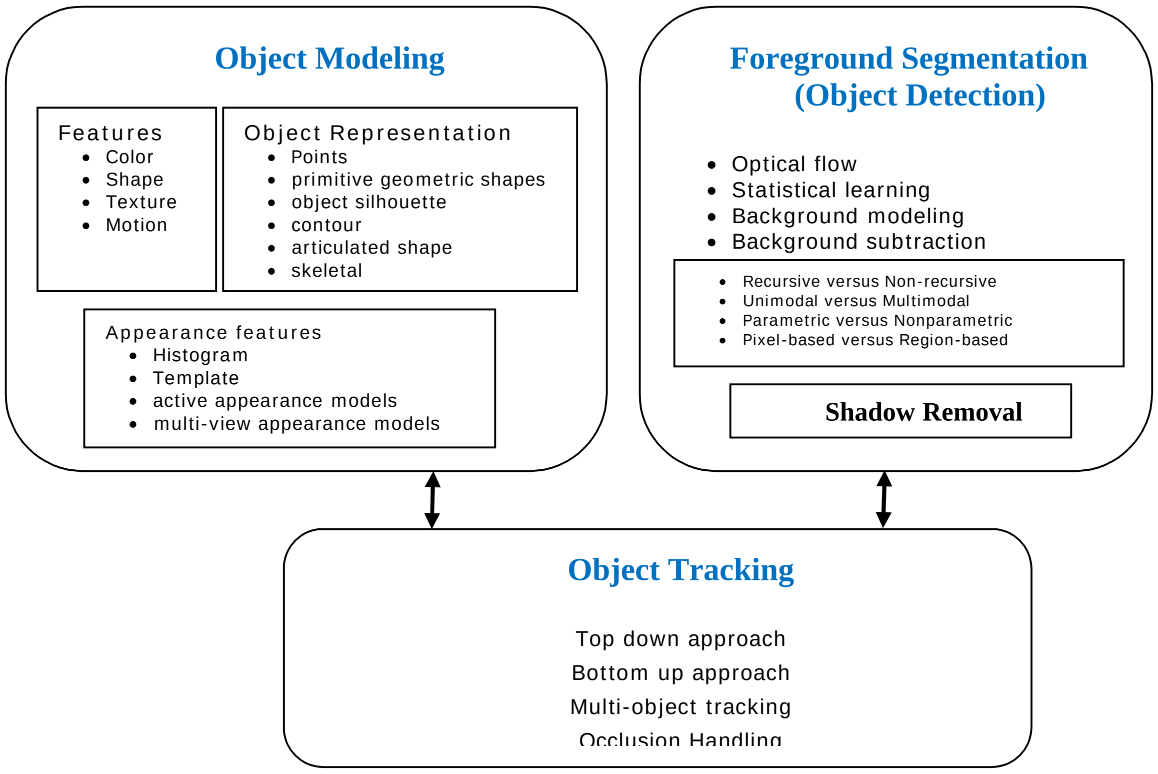


Figure 2: Functional architecture of visual object tracking,

2 Object Modeling

Object modeling plays a crucial role in visual tracking because it characterizes an object of interest. Selecting an effective object model plays a critical role in object tracking. Only the feature defined by the object model is used to maintain the estimate of the track. Object modeling therefore consists of two attributes: the representation of the object, which describes its span in the frame, and the features, which characterize it. Consequently, a poor choice of object model inevitably leads to poor tracking. The range of object representations encompasses various types of models and is application dependent. Some applications only require a simple model, while others require accurate and complex object models to achieve tracking.

2.1 Object Representation

Two important aspects that determine the performance of the tracking algorithms are object representation and object localization. Object representation refers to how the object to be tracked is modeled and object localization deals with how the search of the corresponding object in the following frame is accomplished. From the object representation point of view amongst the wide variety of approaches adopted it can distinguish those that use a minimum amount of information extracted from the object, like color [5], intensity [6], feature points [7], spatialized color histograms [8]. Also, it can be used the integration of multiple number of features to have a better representation of the object [9]. There are also the approaches that use a very specific model of an object; this basis is useful when the goal is to track solid models. These approaches are based mostly on the contour, edges or a more detailed representation of an image curve using a parameterization like B-splines [10].

Object shape representations generally used for tracking are: points, primitive geometric shapes (e.g. rectangle, ellipse), object silhouette, contour, articulated shape and skeletal models [11] as shown in Figure 3.

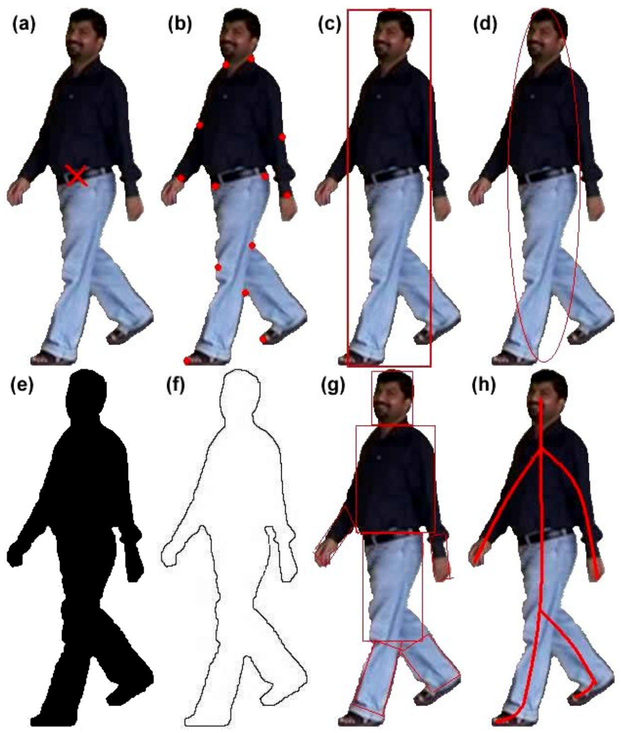


Figure 3: Object shape representations (a) point (b) multiple points c) primitive geometric shape (rectangular) d) primitive geometric shape (elliptical) e) silhouette f) contour (g) articulated shape (h) skeletal model.

Point Representation: In visual object tracking, the trivial shape is the point. An object is represented with a pixel location representing either some statistics on the object, such as the centroid, or a particular characteristic of interest. Point representation has been used in a plethora of applications due to its processing simplicity and the ease of point manipulation with complex algorithms [12]. For instance, it has been used for point tracking in radar imagery [13], distributed point tracking [14] or for Monte Carlo techniques where the number of samples prohibits heavy calculations [15], [16], [17]. Point tracking also alleviates the uncertainty regarding the position of the object of interest in the frame since it is based on a single point. It can be complemented with various order moments describing the distribution of the shape, such as the variance of pixels in the object of interest [18], [19]. Points have also been used to generate heuristics on some characteristics of the object. They are also used in the calculation of optical flow: due to the large number of vectors to estimate, only the point representation can be afforded [20] [21].

Primitive geometric shapes: The point representation of an object is a simple model. However, it does not grasp the entire dynamics of the object. For instance, rotation is not catered for with point representation. More advanced parametric shapes are, therefore, necessary to address these types of problems. The popular parametric shapes are primitive geometric shape such as rectangle, square, ellipse and circle. They are more appropriate for representing simple rigid objects. However using adaptive methods they can also be used for non-rigid objects. The rectangle representation is ubiquitous in geometric object tracking such as cars [22], [23] or in low-distortion object tracking such as people [24]. An adaptive square shape has been used for object representation in [25]. The ellipse offers the advantage of "rounding" the edges compared to the rectangle when the object does not have sharp edges [26]. In [8], [27], the author used an elliptical shapes to represent the moving object.

Articulated shape models: Articulated shapes are employed for tracking if different portions of the object of interest are to be described individually (*e.g.* legs, arms and head). This kind of representation is much suitable for a human body, which is an articulated object with head, hands, legs etc. These constituent parts should be related by a kinematic model. The constituent parts can be represented by any primitive geometric shape such as rectangles, circles and ellipses. Ramanan and Forsyth developed an articulated shape model to describe the body configuration and disambiguate overlapping tracks [28] In [29] the position of the different body limbs to analyze the behavior of people.

Skeletal models: In this representation a skeleton of object can be extracted to model both articulated and rigid objects. We can define the skeleton as a set of articulations within an object that describes the dependencies and defines constraints between the representations of the parts. In [30] the author utilized the skeletal model for automatic segmentation and recognition of continuous human activity.

Object silhouette: The silhouette is also called 'Blobs'. A blob is a dense, non-disjoint, binary mask that represents an object of interest. Blobs are of particular importance for pixel-wise processing. For instance, background subtraction provides blobs identifying the foreground or the moving objects in a scene [31], [32] [33].

Contour: In this representation the boundary of an object is defined as a contour. It provides a convenient non-parametric trade-off between an exhaustive description of the object and storage requirements. Instead of storing the entire silhouette, contours only describe the edges enclosing the object. A non-rigid object shape can be better represented by these representations [34].

The appearance features of objects can also be characterized by a number of methods including probability densities of object appearance, templates and active appearance models [11].

Histogram approach is the most popular probability density estimates of the object appearance. The color histogram is relatively unaffected by pose change or motion, and so is also a reliable metric for matching after occlusion. However, one limitation of histograms is that they do not contain any position information. Two objects that have very similar color histograms may have dramatically different appearances due to the distribution of the colors. The color correlogram [35] is a variant of the color histogram, where geometric information is encoded as well as color information according to predefined geometric configurations.

Templates are formed using simple geometric shapes or silhouettes. A comprehensive description of the use of templates in computer vision can be found in [36]. Templates aim to represent objects with a set of predefined models. In that sense, templates can be categorized as semi-parametric representations. The predefined models are *a priori* non-parametric and can be of arbitrary form, providing single or multiple views of the object of interest. However, the matching of the model is performed by projection, distortion, scaling, etc., which are parametric transforms. One of the main tasks concerning templates is to maintain the set of models to minimize their number and maximize their relevance to the scene. First, if the appearance of the object is assumed to be static, the set of templates can be generated at initialization and updates are not necessary [37]. If the object changes appearance but is limited to a pre-defined range, the set of templates can be learnt off-line [38], thereby limiting its size. Another approach is on-line update and pruning of the set throughout time [39]. Templates are simple non-parametric representations to manipulate due to the restriction in the set of models and the parametrization of transforms used for matching. Nguyen et al. [40] performed a normalized correlation template tracking in the modulation domain. For each frame of the video sequence, they compute a multi-component AM-FM image model that characterizes the local texture structure of objects and backgrounds.

An *Active Appearance Models (AAMs)* contains a statistical model of the shape and grey level appearance of the object of interest [41]. They incorporate both shape and texture into their formulation; hence they enable us to track simultaneously the outline of an object as well as its appearance. It is therefore easy to use the parameters provided by an AAM tracker in other applications. Stegmann [42] demonstrated that AAMs can be successfully applied to perform object tracking. In his deterministic approach, the AAM search algorithm is applied successively to each frame.

2.2 Object Features

The object can be modeled by their shapes and appearances. Figure 4 illustrates that the object can be represented at different level of abstraction. At the low level the object can be represented simply by intensity value of its pixels. At the middle level it can be represented by some features like color, texture etc. At the highest level it can be represented by a global feature vector which can be boosted from many features. In general, a tracking framework exploits a global feature vector to measure the similarity between target and candidate object.

The major issue is finding an appropriate visual description for an object so that it can be uniquely define in the feature space and easily distinguished from others. The ideal feature for object tracking is an invariant of the object, i.e. at least robust to any type of transform, any change of illumination, any degradation. Some of the features such as color, shape, texture, and motion can be used to describe objects.

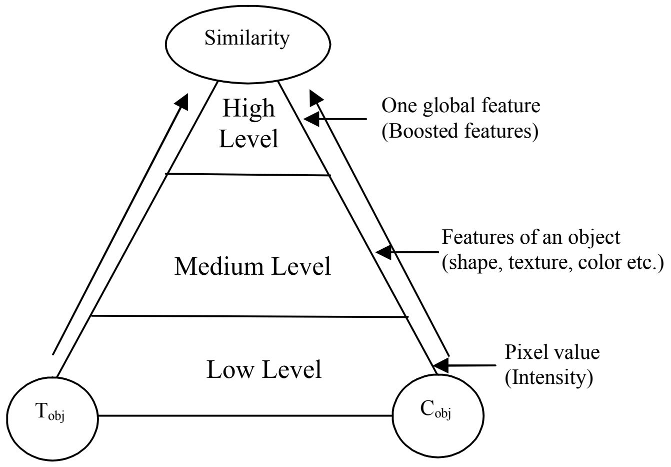

Similarity
High Level
Medium Level
Low Level
One global feature (Boosted features)
Features of an object (shape, texture, color etc.)
Pixel value (Intensity)
T_{obj}
C_{obj}

Figure 4: Object representation and matching in feature space (T_{obj} : Target object, C_{obj} : Candidate object).

Color Modeling: Color is most fundamental feature to describe an object. Due to its strong descriptive power, color is a good choice for representing an object [43]. RGB color space is usually used to represent images; however, the RGB color model is perceptually not a uniform color model. HSV is an approximately uniform color space and used intensively in literature. Hue, saturation and value are the three components of a HSV color space. In general, color spaces are sensitive to illumination change and noise. In [25], a single channel (hue) is considered in the color model.

Shape Modeling: The shape features are used as a powerful cue to detect object in video frame sequences. The shape of an object can be represented by a set of control points on the spline [44]. Edge is also used as feature where boundary of the objects is used to track the object. For more detail review on edge detection, the reader is referred to [45]. The advantage of edge feature over color feature is that edge is less sensitive to illumination changes.

Texture Modeling: Texture is also an important identifying characteristic of images. It is used to measure the intensity variation of a surface and concerned with representing regular patterns in an image [4]. The texture representations can be classified into two classes: structural and statistical. Morphological operator and adjacency graph are two structural methods used to describe texture. Statistical methods include 1-D grey-level histograms, co-occurrence metrics, grey-level differences and multi-resolution filtering methods. As compared to color, texture features are also less sensitive to illumination changes.

Motion Modeling: Motion detection is vital part of the human vision system [44]. It is one of the functions of rod cells of our eyes. Optical flow is the most widespread depiction of motion. Optical flow represents motion as a displacement vectors which defines the movement of each pixel in a region between subsequent frames [46]. Horn and Schunk [46] computed displacement vectors using brightness constraint, which assumes brightness constancy of corresponding pixels in consecutive frames. Lucas-Kanade [47] proposed a method that computes optical flow more robustly over multiple scales using a pyramid scheme.

3 Foreground Segmentation

Detection of object to be tracked is the primary step in object tracking process. The object can be detected either once in the first frame or in every frame in the video. The goal of segmentation is to find out the semantically meaningful regions of an image and cluster the pixels belonging to these regions. It is very expensive to segment all static objects in an image. However, it is more practical to segment only the moving objects from video using spatio-temporal information in sequence of images (frames). A segmentation method should be generic and should not depend on other factors like color, shape and motion. Also, segmentation method should not be computationally intensive and should require less memory.

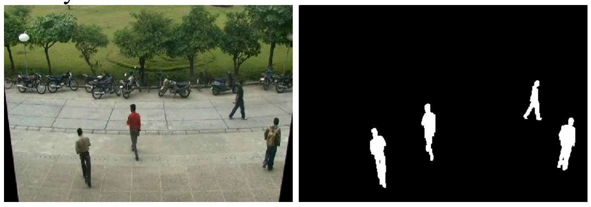


Figure 5: Foreground mask for an outdoor scene (Manual segmentation).

Foreground objects are defined as objects that are moving or involve in some activity. To track these objects, they have to be separated from background scene. A background scene is assumed as temporally stationary component of the video-frame such as roads, buildings and furniture. The Figure 5 shows an example of a foreground mask, where white and black colors represent the foreground and background pixels respectively.

Although a lot of studies have been conducted in recent years, the subject is still challenging. Some of the popular approaches proposed in the literature include background subtraction method, optical flow method and statistical learning method (non-parametric kernel density estimation). Algorithmic complexity is the major disadvantage of optical flow method. It requires higher time span than other methods. The non-parametric kernel density estimation method stores color values of multiple frames and estimates the contributions of a set of kernel functions using all of the data instead of iteratively updating background models at each frame [33]. The requirement of training samples and higher computational complexity make these methods infeasible for real time processing [48].

The following are some usually referred classifications discussed in the literature [49].

Recursive versus Non-recursive - Recursive techniques [31], [32], [50] use a single background model that is periodically updated. Non-recursive methods [51], [52], [53], [54] estimate a background model using statistical properties of previous frames by keeping a buffer. So non-recursive technique requires higher memory in comparison to recursive technique.

Unimodal versus Multimodal – In unimodal methods [31], a single modality is used to model the intensity of a pixel. On the other hand, multimodal methods [32] are used to represent the multi-modality of the underlying scene background. Although these methods cope with multi-modal distributions caused by shadows, waving tree branches, flickering monitor etc., but at the cost of higher complexity.

Parametric versus Nonparametric – Parametric methods [31], [32] require a tricky parameter initialization. The nonparametric models [33], [48] are more flexible and not require any assumptions about the underlying distributions. However, nonparametric models are memory and time consuming.

Pixel-based versus Region-based – Pixel-based methods [31], [32] use the statistical properties of individual pixel to model the background. While region based methods [33], [54] assume that foreground pixels have a tendency to appear in sets of connected points as a region.

3.1 Background Subtraction

The background subtraction method is one of the very simple and promising approaches for extracting moving objects from video sequences [55]. In background subtraction approach, we compare current frame with a reference frame known as background image. A significant difference indicates the presence of moving objects. However, if the reference frame is not modeled or updated adequately, this approach can be highly vulnerable to environmental conditions like illumination and structural background changes. So background modeling is one of the primary and challenging tasks for background subtraction. The background subtraction algorithm should be robust against environmental changes i.e. capable to handle changes in illumination conditions and able to ignore the movement of small background elements.

In recent years, several methods for performing background modeling and subtraction have been proposed. Frame differencing [55] is a simple and easy way to extract moving object from a video sequence. In this approach, the image difference between consecutive frames is used and considerable difference in pixels value is considered as foreground region. However, Frame differencing methods suffer from fat boundary and thresholding problem. Another background modeling method is the use of temporal median filter, proposed by Lo and Velastin [51]. In this the median value of the pixels in the last 'n' frames is taken as the background model. In [52], the author extended this model for color images. Cucchiara et al. [53] proposed a mediod filtering approach, in which the mediod of the pixels can be computed from the buffer of image frames. However, this approach does not produce a measure of variance.

Wren et al. [31] proposed a method to model the background independently at each pixel location using a single Gaussian distribution. A recursive updation using a simple linear filter is used to estimate the Gaussian parameter. This technique is very simple and having fast implementation. However, it fails whenever some kind of variations occurs in the background.

Stauffer and Grimson [32], [50] proposed a method known as Gaussian Mixture Model (GMM), to handle multi-modal distributions using a mixture of several Gaussians. The GMM is the most representative approach and has been widely used [56]. The weight (w), mean (μ) and the standard deviation (σ) of a Gaussian component is updated recursively to imitate the new observations for pixel value. For the unmatched distributions the mean and variance remain unchanged but weights decrease exponentially. The matched components are updated by using a set of equations. These equations boost the confidence in the matched component by increasing w, decreasing σ, pushing μ towards the pixel value. A component is considered a matched component if the difference of mean and pixel's intensity value is less than a scaling factor (D) of a background component's standard deviation σ. A confidence metric (w/σ) is used to decide which components are parts of the background model. This is useful to select 'M' most confident guesses. M is the maximum number of modes one expects in the background probability distribution function. The first M components whose weight w is larger than a specified threshold become background model. Those pixels that don't match with any Gaussian components are treated as foreground pixels. The GMM can deal with multimodal distribution. However, the major disadvantages of GMM are that it is computationally intensive and require a tricky parameter optimization.

Elgammal et al. [33] exploited a nonparametric kernel density estimation to build a background PDF. The probability density estimation is performed using the recent historical samples without any assumption about background and foreground. The model is robust and has good model accuracy as compared to Gaussian mixture model in more complex scenes. However, the high computation cost, limits its scope.

Recently, a method based on texture is proposed to model the background and extract moving objects [57]. A binary pattern calculated around the pixel in a circular region is used to model each pixel. The binary pattern indicates whether the neighbouring pixel is smaller or larger than the central pixel. A modified local binary pattern (LBP) operator is used to extract features to make the method invariance to monotonic gray-scale change. However, the method can cause poor performance on flat image areas, where the intensity values of the neighbouring pixels are similar. Though this texture based method belongs to nonparametric methods, but it is fast due to simplification of LBP computation.

In [56], the author exhibits that the output of a background segmentation algorithm in the form of foreground segmentation masks can be significantly improved by applying post-processing techniques. This post-processing includes noise removal, morphological opening, closing operation, area thresholding etc.

3.2 Shadow Removal

There are many processes that are used to improve the performance of foreground segmentation. These processes include suppression of shadow, reflection and handling ghosts. Among these processes shadow suppression is most crucial task, which helps in improving detection accuracy and avoids analysis failure. After background subtraction, we get pixels correspond to objects as well as shadows. This is because shadow pixels are also detected different from the background and adjacent to object pixels and merge in a single blob as shown in Figure 6(b). Shadows occur when an object exists between a source of illumination and the surface on which it rests. These are natural phenomena and are easily perceptible to the human eye. However, shadows cause a lot of complications in various computer vision algorithms like object segmentation, object recognition, object tracking, scene understanding, etc. This is because the shadows tend to move in similar patterns and directions as an object being tracked, thus getting detected as a part of the object [58]. The shadow areas appear as surface features and corrupt the original object area, resulting in misclassification of object of interest and bias in estimation of object parameters. It is clearly depicted in Figure 6(c) that due to shadow the bounding box representation of the object becomes incorrect and contains a large portion of background. As a result any features computed for higher level analysis give an incorrect end results.

Shadows are mainly of two types, self shadow and cast shadow [59]. A 'self-shadow' is one which occurs on the object itself and is perceived as the darker regions of the body in the direction opposite to the direction of illumination. These are usually obscure and gradually change in intensity, with no definite boundaries. On the other hand, 'cast-shadow' is the dark region projected on the ground by occlusion of light due to the object. These tend to have hard, distinct shapes with sharp boundaries. Cast shadows are a major issue in object tracking and object recognition tasks. A number of approaches are proposed in literature to suppress cast shadow. A comparative evaluation of shadow detection methods is given by Prati et al. in [58]. They proposed a two layer classification to highlight differences between different shadow removal algorithms. In the first layer, the approaches are classified as statistical and deterministic. In statistical methods, a probabilistic function is used to classify the shadow pixels whereas deterministic methods use an on/off decision process. The statistical approaches are further classified in parametric and non-parametric classes. The deterministic approaches are also further divided in model based and non-model based approaches. In model based techniques prior knowledge can be used to represent the model. On the other hand, non-model based methods use spectral and temporal properties to detect shadows. It is concluded that in case of noisy environment a statistical approach outperform as compared to deterministic model. It is also suggested that fewer assumptions should be defined to handle shadow problem in more generalized way.

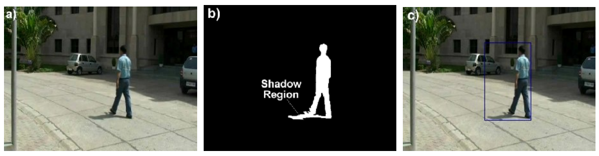


Figure 6: Moving object segmentation for an outdoor scene a) Original frame b) foreground mask c) bounding box representation of object.

Hsies et al. [60] proposed a shadow elimination method based on statistical model using Gaussian shadow modeling. They used a coarse-to-fine shadow modeling approach. At the coarse stage, an orientation of the detected moving object mask is computed using central moment. Then shadow is detected by computing the rough boundaries between the cast shadow and the moving object, using difference of histogram (orientation, vertical) and silhouette features. A Gaussian shadow modeling is used to further refine the rough approximation of the shadow area. Parameters such as orientation, illumination and position are used for this purpose.

Early researches for shadow removal were typically focused on identifying dark areas on plain and flat surfaces. In [61], Nadimi et al. proposed physics based approach of shadow detection. A Gaussian mixture model is used for background modeling. They followed a multistage approach. At each stage the pixels are filtered out, which cannot be shadow pixels. The stages are as follows:

1. Initial shadow pixel reduction- In a training phase, the body color of surface that may come under shadow in the scene, is calculated. Only pixels having attenuated intensity than their background (in R, G, B) are considered as shadow candidates.
2. Blue ratio test - A blue ratio test exploits the fact that the illumination due to blue sky is responsible for outdoor shadows, so there is a higher ratio of blue.
3. Albedo ratio segmentation – An albedo ratio is used to extract regions of uniform reflectance. The albedo ratio is computed by combining two components. The first is the ratios of difference between two neighboring pixels and second is the ratios of differences between foreground and background pixels.
4. Ambient reflection correction – In this step foreground pixel values are subtracted from the background pixels, to suppress the effect of sky illumination.
5. Body color segmentation – In this step dichromatic reflection model is utilized to compute the true color of the object.
6. Verification – Finally verification is used to match the various surfaces with their expected body colors and determines which regions lie in shadow.

No spatial assumptions are considered in Nadimi's approach. However, the approach is supervised and mainly suited for outdoor situations.

The Dichromatic reflection model is also used in [59]. Initially a candidate shadow regions is identified using the hypothesis that shadow darken the surface on which it is cast upon. Then a verification stage is applied based on photometric invariant color features and geometric properties. Hue is used as the color invariance feature which is expected to be unchanged between shadows and object regions.

In [53], Cucchiara et al. proposed a general-purpose approach to extract the moving objects, ghosts, and shadows. They used a HSV color space and exploit the statistical assumptions that in a shadow region, the brightness and saturation properties are reduced while hue properties remain same. The ratio of reduction lies in the range α to β [53]. The first range (α) represents a maximum value for the darkening effect and depends on the intensity of light source, while the second range (β) is imposed to avoid detecting points that have been slightly altered by noise. The ghost object can be separated by analyzing the optical flow. As ghost object does not represent any motion, so they have an optical flow either zero or inconsistent.

A number of methods use the spectral property of shadow to indentify it [53], [59], [61]. However, the spectral properties fail to resolve the candidate shadow region accurately when the object body is darker than the background. In such cases geometrical properties can provide valuable information for shadow segmentation. In [59], [62], geometry properties such as shadow-background boundary and shadow-object boundary are used as an aid to verify the existence of shadows.

In [63], Leone and Distante exploited texture analysis for shadow detection with the assumption that textural properties remain same in shadow-regions. A Gabor function is used to perform texture analysis. Although the texture based method are simple, however, these methods mostly good for the identification of weak shadows, indicating its use for indoor environments. Also these methods are computationally intensive.

4 Prediction Methods

For tracking objects in a video, we have to find the position of object's instances in two consecutive frames. One of the brute force methods is to exploit the matching technique on the whole image of every incoming frame. But this puts an overhead of exhaustive search. In general, the tracking algorithm assumes that in a few consecutive frames the trajectory of object does not change abruptly. Therefore, to increase the efficiency of the algorithm, the matching technique is not exploited in whole image frame. Rather the reference template is matched in a search space, which will be somewhere in the surrounding of the region where last time the object was detected. Predicting possible location of candidate object in each frame will also help in improving tracking accuracy and minimizing the search space. The robustness of the system to handle abrupt motion and occlusion is also influenced by this prediction. The better the prediction of the object location is, the search space become smaller. Thus the accurate algorithm can additionally speed up the whole tracking process. There are three common approaches to predict an object's position.

- Motion Model
- Kalman Filter
- Particle Filter

The simplest type of predictor is the *motion model*. Motion model exploits past observation to predict the next position [64]. A simple motion model can be formulated as,

$$l(t+1) = l(t) + v(t)$$

where $l(t)$, $l(t+1)$ represents the current location and the predicted location at t and $t+1$ time step respectively, and $v(t)$ is the velocity at t time and is defined as,

$$v(t) = l(t) - l(t-1)$$

Acceleration can also be used in motion model.

Predictive filtering can also be used as one of tracking approach. The *Kalman filter* is a linear predictive filter and widely explored in the vision community for tracking [65]. It is proved to be good to predict the state, in the presence of noise. Kalman filtering consists of two steps, prediction and correction. The prediction step estimates the process state at the next time step, using previous state variables. While the correction step incorporates the new observations into the system to update the object's state. A detailed explanation of the mathematical equations is discussed in [65]. However, Kalman filter restricted its application to only linear dynamic and measurements models with additive Gaussian noise. To handle the non linear relationships, an extension is proposed such as Extended Kalman filter (EKF) [65].

In [66] the author proposed an algorithm which adaptively predicts possible coordinate transform parameters for the next frame and selects them as the initial searching point when looking for the real transform parameters. An adaptive Kalman filter is used, but instead of directly filtering the values of transform parameters, the Kalman filter is applied on the changing rate of those parameters to effectively predict their future values.

Another major problem with Kalman filter is that it can model only a single hypothesis. Due to the unimodal Gaussian assumption, it is not feasible to represent multiple hypotheses simultaneously using the Kalman filter. This limitation can be overcome by using *particle filtering*, which is based on Monte Carlo integration methods [67]. In particle filtering, a set of random samples with associated weights are used to compute the current density of the state (which can be location, size, speed, boundary etc.). These samples and weights are also used to compute the new density.

5 Object Tracking

The goal of an object tracker is to create the trajectory of an object over time by locating its position in every frame of the video. Tracking can also be defined as detecting the object and maintain correspondence between object instances across every frame of the video. The features of a good tracking algorithm are as follows;

1. The tracking algorithm should detect all the objects that enter or moved in the scene.
2. The tracking algorithm should differentiate between multiple objects that are present in the scene at the same time.
3. To monitor and extract the trajectory of all objects the unique label assigned to each object must be maintained for all the tracked objects.
4. The motion or lack of motion of the object should not lead to change of object label.
5. The tracking algorithm should handle occlusion and exposure without object labels changing.

There are two distinct methodologies to approach the tracking problem, top down (forward tracking) and bottom-up (back-tracking). Top down methods are goal oriented and find the positions of the object in the current frame using a hypothesis generated at the start of the tracking based on parametric representation of the target. On the other hand, in a bottom-up approach, the moving objects are detected in every frame and then a correspondence is established with the objects those were detected in the previous frame. A representative of top down approach is many model based and template matching approach [27], [25]. While a blob based tracking represents a bottom up approaches [31].

5.1 Top down Approach

The top down approaches often rely on external input to initialize the tracking process. These tracking methods use different object characteristics, such as color, texture, shape and motion. One of the popular method in this category is mean-shift based tracking [8], [27], [25]. Mean shift tracker exploits the concept of non-parametric density gradient estimator that iteratively executed within the local search kernels [68]. It uses the color histogram to model object probability density and moves the object region in the largest gradient direction.

In [25], Bradski proposed an adapted version of mean-shift called CAMShift (Continuously Adaptive Mean-shift). A histogram based on known hue value in color image sequences is used to track the head and face movement. Mean shift algorithm is used with adaptive region sizing step. The kernel having simple step function is applied to a skin probability map. The mean location i.e. centroid at each iteration with the search window is computed as zero and first order moment. Due to the consideration of a single channel (hue), the algorithm is supposed to consume less CPU time. However, the algorithm may fail to track objects having multiple hue value or objects, where hue value is not sufficient to discriminate the object from background.

Comaniciu and Meer [27] proposed a kernel based tracking algorithm. A weighted color histogram is used as feature to represent the target object in an ellipse. The weighted histogram is computed using Epanechnilov kernel profile which assigns smaller weights to pixels farther from center. The author used Mean shift to find the location of target model in the current frame. Bhattacharyya coefficient has been used as a measure of comparability between the target object and the candidate object. The location of the target object in previous frame is used as a starting point for Mean shift procedure in current frame. The Mean shift procedure maximizes the value of similarity measure i.e. Bhattacharyya coefficient iteratively. Although the kernel based method is computationally simple, however, the method fails as soon as the color distribution of object becomes similar with any other region in image frame. This is because color histogram does not contain any position information. Two objects that have very similar color histograms may have dramatically different appearances due to the distribution of the colors. For example, one person may be wearing a white shirt and black pants whilst a second is wearing a black shirt and white pants. Whilst these people may have quite distinct appearances, they would have very similar histograms.

One of the problems of Mean shift is that it is designed to find local maxima for tracking objects. As a result Mean shift tracker may fail in case of large target movement between two consecutive frames. In [69], the author proposed a multibandwidth procedure to help conventional MS tracker to reach the global mode of the density function using any staring points.

There are also a number of literatures on the problem based on fragments of object tracking. In [70], Adam et al. presented fragment-based tracking which accounts for partial occlusions. It uses a computationally and memory expensive technique called integral histograms. Through exhaustively searching, there is no formal framework by which we can selectively use certain fragments. Some significant improvements in Mean shift tracking are suggested in [71], where a fast target updation scheme using foreground separation to tackle appearance change is proposed. To improve robustness, the object to be tracked is divided into more than one fragment. Whenever a new frame is extracted from video sequence, candidate model is built for each such fragment and Mean shift is used to find a pair of target object. An enhanced kernel based object tracking system is developed that uses background information and edge. Color marginal histogram is exploited to tackle the scale change problem. The coordinates of the winning fragment of the object location using the most confident estimate is used to obtain the object position. However, there is no self driven ways by which we can get the most confident estimate of the object. Moreover, the intrinsic feature selection would result in a tracking drift or decreasing the weights of some target blobs. In [72], a fusion scheme has been proposed to fuse multiple spatially distributed fragments. Under the fusion scheme, a mean shift type algorithm which allows efficient target tracking with very low computational overhead. However, the weight of each fragment in occlusion will result in draft. A Mean shift based multiple model tracking algorithm is proposed in [73]. The author exploits several connected regions to incorporate spatial information into object representation. Multiple models are used to adapt changes in object appearance during the tracking process. Switching between multiple models has been done using Bayes probabilistic rule.

In [74], the author proposed a multi-kernel approach to handle fast motion area and improves the convergence problem by integrating two likelihood terms. In [75], Fang et al. proposed an efficient and robust fragments-based multiple kernels tracking algorithm. A feature, which is based on the separation of the foreground/background likelihood function, is used for tracking.

Recently, many methods exploited multiple features to improve reliability and tracking performance [76], [77]. In [76], the authors integrate the shape-texture and color features in the Mean shift tracking framework. The shape-texture feature is represented by an orientation histogram. Ning et at. [77] extended the mean-shift tracking algorithm by combing a LBP texture feature with color histogram features. In [78], geometric features are used for real-time vehicle tracking. Texture patterns give the spatial structure of an object. However, texture feature become ineffective, where objects having large smooth regions. Hu et al. [79] extracted three histograms from each person, one each for the head, torso and legs, to not only allow for matching based on color, but also on distribution of color. Shen et al. [80] extend the use of statistical learning algorithms for object localization and tracking. In contrast to building a template from a single frame, a probabilistic kernel-based SVM is used to represent object model from a large amount of samples.

In designing an appearance model, the crucial properties that a tracker needs to meet are robustness and adaptability to changes in target appearance (e.g.,pose, illumination). Recently, many tracking methods are developed to achieve these goals by incorporating an adaptive appearance model. Ross et al. [81], proposed a Incremental Visual Tracker (IVT) and represented a target as a low-dimensional subspace that captures the principal components of possible appearance variations, where the subspace is updated adaptively using the image patches tracked in the previous frames. Unlike many non-adaptive approaches that employ fixed appearance template models, this method alleviates the burden of constructing a target model prior to tracking with a large number of expensive offline data, and tends to yield higher tracking accuracies. However, the model is restricted to characterizing only texture-rich objects. A robust tracking algorithm based on the adaptive pixel-wise appearance model is proposed in [82]. Intensity value of each pixel in appearance model is modeled by a mixture of Gaussian density whose parameters are updated using sequential kernel density approximation.

5.2 Bottom up Approach

The bottom up approach covers those methods which uses the background modeling and subtraction approach to extract foreground objects and then track the objects by establishing a unique correspondence with the previously detected targets over time. In [31], the author proposed a system named as 'Pfinder', for detecting and tracking human body. The background is modeled separately for each pixel location. It uses a Gaussian probability density function in the YUV space on the last '*n*' pixel's value. In each new frame the statistics is updated using running Gaussian average. Multiple blobs are used to model the person's various body parts. Each blob is represented by spatial information, color component and the corresponding Gaussian distributions. In each new frame the scene and the person model is dynamically changing. A Kalman filter is used to predict spatial distribution for current frame. The log likelihood method is used to resolve the class membership i.e. decision about the pixel assigning to the background scene or one of the blobs. Then iterative morphological operations are used to produce a single connected region and the statistical models for the blob and scene texture model are updated. The person body parts like head, hands and feet are labeled using a 2-D contour shape analysis. The skin color is used to initialize hand and face blobs. The method shows good results in indoor environment, however, its success in outdoor scenes is not explored much.

Zhixu Zhao et al. [83], proposed a texture based multi-target tracking algorithm. A local binary patterns (LBP) is used as texture descriptor. A single Gaussian model is used for background modeling to perform foreground segmentation. A Kalman filter is used as a motion predictor. A LBP histogram distance is used to distinguish blob in case of occlusion.

In [84], the author proposed an integrated framework for object detection and tracking. A Support Vector Regression (SVR) is used to model background. The background model is updated online over time. A confidence coefficient computed using shape, color and motion information is used to improve target-to-target correspondences over time.

5.3 Tracking in Wavelet Domain

Most of the algorithms discussed in previous sub-sections are unable to track objects in the presence of noise, variations in illumination, appearance and camera jittering, as most of these algorithms working in spatial domain use features which are sensitive to these variations [85]. In recent years, the wavelet feature based techniques have gained popularity in object tracking. It provides a rich and robust representation of an object [86], due to the following characteristics:

- Wavelet transform provides powerful insight into an image's spatial and frequency characteristics [87], [88].
- Provides an efficient framework for representation and storage of images at multiple levels [89].
- Provides noise resistant ability [90].
- High frequency sub-bands of wavelet transform represent the edge information [91].
- Wavelets are also a core technology in the next generation compression methods [92].

One of the features of discrete wavelet transform (DWT) is that the spatial information is retained even after decomposition of an image into four different frequency coefficients. In this decomposition, the high frequency sub images (horizontal coefficient, vertical coefficient and diagonal coefficient) contain the detailed information. In [93], a rule based method is proposed to track object between video images sequences. First the moving object is isolate from background in each wavelet transformed frame. Then, a feature is computed based on the positions, the size, the grayscale distribution and presence of textures of objects. However, the method is computationally expensive and unable to deal with occlusion. A wavelet based vehicle tracking system was proposed in [94]. A frame difference analysis of two consecutive frames has been used to extract the moving object. After the removal of shadow from extraction object, a wavelet-based neural network is used recognized the moving vehicles. The centroid and wavelet features difference measures are used to track the identified objects. They highlight that decomposing an image at lower level using WT reduces the computational complexity. However, still the recognition step acquires a major computational cost. In [95] the author used highest energy coefficients of Gabor wavelet transform to model the object in the current frame and 2D mesh structure around the feature points to achieve global placement of the feature point. The 2D golden section algorithm is used to find the object in the next frame.

A real-time multiple object tracking algorithm is proposed in [96]. In this algorithm, wavelet coefficients is not used as object features, rather the original frame is only preprocessed using a 2-level discrete wavelet transform to suppress the fake background motions. The difference image of successive frames is computed using the approximation band of the wavelet transform. Then, the object is identified using the concept of connected components in the difference image. The identified objects are then represented by a bounding box in the original approximation image. This bounding box representation is used to compute some color and spatial features. In the successive frames these features are then used to track the objects. Chang et al. [97] proposed a tracking system based on discrete wavelet transform to track a human body. A CCD camera is used, which is mounted on a rotary platform for tracking moving objects. The background subtraction method is used for object detection. The YIQ (Y: luminance, I & Q values jointly describe the hue and Saturation) color coordinate system is used as a feature for tracking. The second level Haar DWT is used to pre-process the images for reducing computation overhead. However only single human object can be tracked and also need a background model (background don't having object).

In object tracking we require any object feature which remains invariant by translation and rotation of the object. A real wavelet is used in most of the papers discussed above. However, one of the major problems with real wavelet transform is that it suffers from shift-sensitivity [98]. In [85], [99] authors used an undecimated wavelet packet transform (UWPT) to overcome the problem of shift sensitivity. Amiri et al. [99] proposed an object tracking algorithm based on a block matching method in using Undecimated wavelet packet tree (UWPT). For block matching in wavelet domain they used the motion vector for each pixel of reference block. The method for finding best match among FVs of the reference block and FVs of the search window is called "Dispersion of Minimums (DOM)". A object tracking algorithm for crowded scenes based on pixel features in the wavelet domain and a adaptive search window updating mechanism based on texture analysis have been proposed in [85]. An adaptive feature vector generation and block matching algorithm in the UWPT domain is used for tracking objects in crowded scenes in presence of occlusion and noise. In addition, an inter-frame texture analysis scheme is used to update the search window location for the successive frames. However, the UWPT expansion is redundant and computationally intensive.

A Daubechies complex wavelet transform [100] can be a better solution which is also approximately shift-invariant. Not many researchers have explored complex wavelet transform (CxWT) application to tracking problems. Recently [88] has shown the applicability of CxWT to denoising and deblurring. Recently an object tracking method based on Daubechies complex wavelet transform domain is proposed [101]. A complex wavelet domain based structural similarity index is used, which is simultaneously insensitive to small luminance change, contrast change and spatial translation. The reference object in the initial frame is modeled by a feature vector in terms of the coefficients of Daubechies complex wavelet transform. They illustrated that the proposed algorithm has good performance even in noisy video with significant variations in object's pose and illumination. Figure 7 shows that the Daubechies complex wavelet transform based method gives accurate results even in the presence of noise.

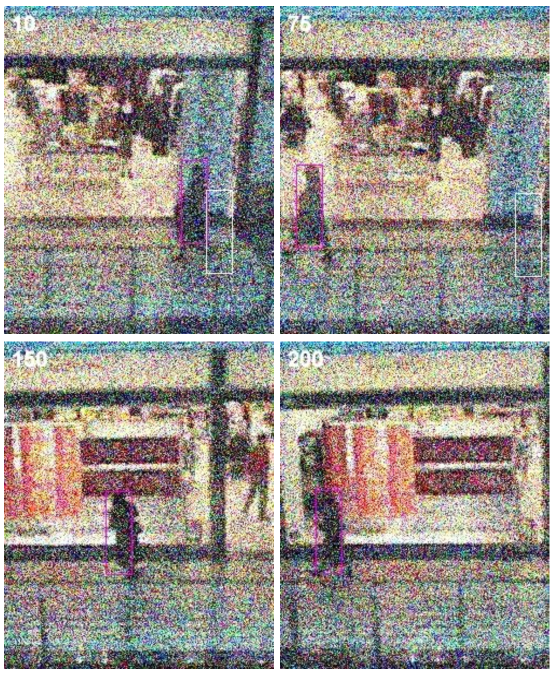


Figure 7: Tracking in noisy video (Gaussian Noise with) using CWT tracker [101].

5.4 Multi-Object Tracking and Occlusion Handling

Mutli-object tracking algorithms should be able to establish unique correspondences between objects in each frame of a video. Tracking multiple objects of the same class implies that the tracking method must be able to discriminate objects, especially when the objects are similar in appearance. Occlusion may be of different types: self occlusion, inter object occlusion, object to background occlusion [102] as shown in figure 8. Self occlusion takes place when an object occludes itself e.g. the face of a person can be occluded by its hand. On the other hand, inter object occlusion occurs due the partial or full overlapping of more than one objects. While a background occlusion occurs when the tracked object is occluded by some component of the background.

Split may occur due to merged objects or because of errors in the segmentation method. An error in the split may mislead the tracker. A good multi-object tracking method should be able to detect changing numbers of objects in the scene, adding and removing objects when appropriate and also able to handle both occlusion and split events.

Although object tracking has been explored a lot but very little work has been done to resolve the issues of multiple object tracking. Kalman filtering is an efficient solution to track multiple objects [103]. However, mistakes become more frequent and are difficult to correct as the number of objects increases. The problem can be solved using particle filtering by exploiting the multiple hypotheses [104]. In [105], the authors formulate the multi-object tracking as a Bayesian network inference problem and explore this approach to track multiple players. In [106], the author proposed a probabilistic framework based on HMM to describe a multiple object trajectory tracking. The framework was able to track unknown number of multiple objects. The association problem has been represented as a bipartite graph in [107]. A method was proposed to maintain hypotheses for multiple associations. They also resolved the problem of objects entering and exiting, and handled the error due to merging and splitting objects. However, particle filter-based tracking algorithms having not enough samples that are statistically significant modes, faced difficulty to track multiple objects. They are only capable to handle partial short duration occlusion.

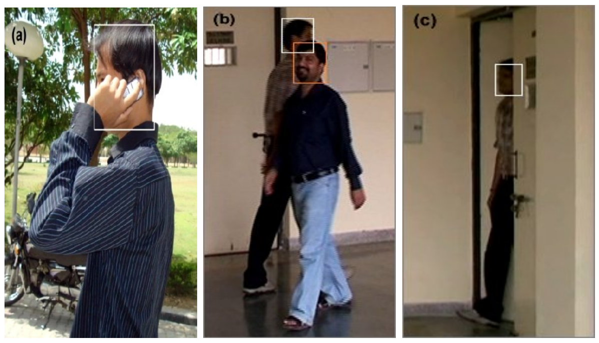


Figure 8: The different types of occlusion states: a) Self Occlusion b) Inter object occlusion c) Background occlusion.

Okuma et al. [108] incorporates an Adaboost learning machine into a single particle filter to differentiate multiple targets. The problem with a joint state is the high computational cost since generic importance sampling becomes less efficient without exploiting the intrinsic correlation among the targets as the number of targets increases. Cai et al. [109] extend the work [108] by proposing an individual mean-shift embedded particle filter for each hockey player, while the association problem is solved by an extra nearest neighbor (NN) algorithm. By moving sampled particles to stabilized positions using mean-shift, the posterior probability is better approximated with fewer particles. The overhead is the mean-shift procedure for each particle.

Senior et al. [110] used an appearance template having a RGB color model with associated probability mask, to represent each pixel of the object. The probability mask represents the probability of the corresponding pixel belonging to that template. The appearance model is continuously updated. The background is statistically modeled and a background subtraction method is then used to extract the foreground regions. A distance matrix is used to establish the correspondence between each foreground region with one of the active objects. Using learned appearance model and the predicted object positions, the occluded foreground region is separated in corresponding objects. However, in case of similar colored/textured objects, the approach becomes vulnerable to distinguish the objects.

In [111], Rad and Jamzad discussed three criteria to predict and detect occlusion of vehicles in a highway. The first criterion is based on examining the trajectory of each vehicle. Occlusion can be predicted, if the centroid of two foreground regions is too closed to each other. Second and third criteria are based on size of foreground region. If a drastic change is found in the size between two consecutive frames then there may be possibility of a merge or split event. A bounding contour of the motion mask is used to resolve the occlusion state.

In [112], the author proposed a multi-object tracking system to track pedestrians using a combined input from RGB and thermal cameras. The motion in the scene is modeled using a particle filter based Bayesian framework. Benezeth et al. [113] proposed a vision-based system for human detection and tracking in indoor environment using a static camera. The moving objects are extracted using a background subtraction approach exploiting a single Gaussian model. Multiple cascades of boosted classifiers based on Haar-like filters are used to know the nature of various objects

In [114], a two stage nonlinear feature voting strategy based method is used for tracking multiple moving objects. The first stage i.e. object voting is used to match two objects. Whereas the second stage i.e. corresponding voting is used to resolve confusion in matching in case of multiple matches. A rule based approach with no appearance model is used to track objects in presence of noise, occlusion and split problem. The method is able to track multiple objects in presence of heavy occlusion and split using the plausibility rules. However, it fails when the objects suddenly disappears or change its direction.

Lao and Zheng [115] discussed special problem of multi-target tracking, where a group of targets are highly correlated, usually demonstrating a common motion pattern with individual variations. They proposed a algorithm which explore the correlations among the targets in a statistical online fashion and embeds both the correlation and the most recent observations into sampling to improve the searching efficiency.

6 Evaluation Measures and Data Sets

6.1 Evaluation Measures

Evaluation of the performance of moving object detection and tracking algorithm is one of major task to validate the correctness and robustness of the tracking algorithm. In March 2000, the first initiative was taken in form of a workshop known as *Performance Evaluation of Tracking and Surveillance* (PETS). Since then, several such PETS workshops have been organized, including most recently a workshop in which the focus was on event level task. The consortium on *CLassification of Events, Activities, and Relationships* (CLEAR) established in 2006 as a collaboration of the European CHIL (Computers in the Human Interaction Loop) project and US National Institute of Standards and Technology (NIST). The goal was to establish a universal evaluation framework for tracking and other related tasks.

The evaluation of different object detection and tracking methods can be performed in two way i.e.

qualitatively and *quantitatively*. Qualitative evaluation approaches are performed on visual interpretation, by looking at processed image yield by the algorithm. On the other hand, quantitative evolution requires a numeric comparison of computed results with ground truth data. Due to the necessity of computing a valid “ground truth” data, the quantitative (experimental) evaluation of object detection and tracking algorithms are highly challenging. Figure 9 illustrates an example of qualitative evaluation and it is obvious that tracker_1 has better tracking accuracy.

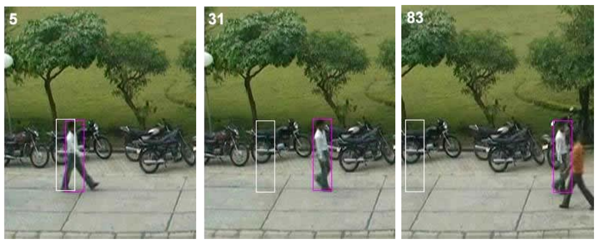


Figure 9: Qualitative result of tracking [101]: tracker_1 (indicated with magenta box), tracker_2 (represented by white box).

In [110] the author proposed a number of metrics for evaluating performance of tracking. These measures includes #track false positives, #track false negatives, average position error, average area error, average detection lag, and average track incompleteness. However, the dependency on input data is the major limitation of these measures. Tracker Detection Rate (TDR), False Alarm Rate(FAR), Object Tracking Error (OTE), Track Fragmentation (TF) and Occlusion Success Rate (OSR) are the evaluation measures proposed in [116] to finds the correspondence between ground truths and tracked objects to compute true positive and false positive matches. The measures defined in [116] are as follows:

$$\text{Tracker Detection Rate}\left(\text{TRDR}\right)=\frac{\text{Total True Positives}}{\text{Total Number of Ground Truth Points}}$$

$$\text{False Alarm Rate}\left(\text{FAR}\right)=\frac{\text{Total False Positives}}{\text{Total True Positives}\ +\ \text{Total False Positives}}$$

$$\text{Track Detection Rate}\left(\text{TDR}\right)=\frac{\text{Number of true positives for tracked object}}{\text{Total number of ground truth points for object}}$$

$$\text{Occlusion Success Rate}\left(\text{OSR}\right)=\frac{\text{Number of successful dynamic occlusions}}{\text{Total number of number of dynamic occlusions}}$$

$$\text{Tracking Success Rate}\left(\text{TSR}\right)=\frac{\text{Number of non}-\text{fragmented tracked objects}}{\text{Total number of number of ground truth objects}}$$

$$\text{Object Tracking Error}\left(\text{OTE}\right)=\frac{1}{N_{rg}}\sum_{\exists i\ g(t_i)\Lambda r(t_i)}\sqrt{(xg_i-xr_i)^2+(yg_i-yr_i)^2}$$

where N_{rg} is the number of frames used for tracking and $(xg_i,xr_i)(yg_i,yr_i)$ is the location of the ground truth and result track at frame *i* respectively. Ground truth points that reside inside the bounding box are referred to as a true positive. While a ground truth point that is not located within the bounding box is referred as false negative.

Similar to OTE, in [117] the author measure the performance of tracking algorithm by finding the localization error, which can be calculated as $\text{SQRT}((Gx_i-X_i)^2 + (Gy_i-Y_i)^2)$, where (Gx,Gy) is the centroid representing ground truth and (X_i,Y_i) is the centroid of computed tracked object. The graph is depicted in figure 10.

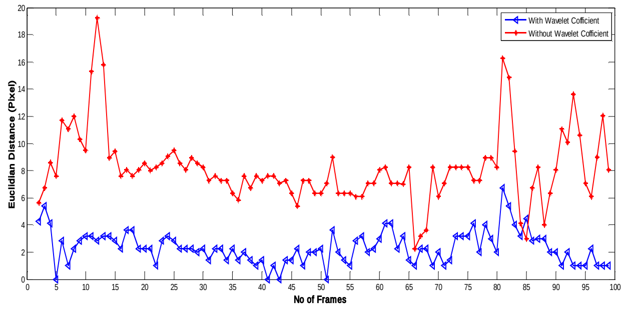


Figure 10: Resulting graph used in [117] to show error of object location using the Euclidian distance (in number of pixels) in each frame.

In [118] the author presented an evaluation method that determines the threshold from the distance matrix between the centroid of the bounding box for the ground truths and the result of the tracking algorithm. False Positive Track Error, False Negative Track Error, Average Area Error, and Task Incompleteness Factor measures are computed using the correspondence between the result of the tracking algorithm and the ground truth data.

Yin et al. [119] presented a new set of metrics to assess different aspects of performance of motion tracking. They proposed statistical metrics, such as Track matching Error (TME), Closeness of Tracks (CT) and Track Completeness (TC) that indicate the accuracy of estimating the position, the spatial and temporal extent of the objects respectively. They also discussed metrics, such as Correct Detection Track (CDT), False Alarm Track (FAT) and Track Detection Failure (TDF) to show the overview of the algorithm performance.

Smith et al. [104] attempted to describe an objective procedure to measure multiple object tracker performance. They have evaluate the configuration and identification performance of multi-object tracking systems by measuring the configuration errors as False Positive (FP), False Negative (FN), Multiple Trackers(MT), Multiple Objects(MO) and identification errors as Falsely Identified Tracker(FIT), Falsely Identified Object (FIO), Tracker Purity(TP), Object Purity(OP).

In [120] the author proposed two intuitive and general metrics to allow for objective comparison of tracker characteristics, focusing on their precision in estimating object locations, their accuracy in recognizing object configurations and their ability to consistently label objects over time. They defined two very intuitive metrics as multiple object tracking precision (MOTP) and the multiple object tracking accuracy (MOTA). The MOTP describe the total error in estimated position for matched object-hypothesis pairs over all frames, averaged by the total number of matches made. The MOTA compute the accuracy in terms of the number of missed detects, false positives, and switches in the system output track for a given reference ground truth track.

The *Multiple Object Tracking Accuracy (MOTA)* was defined as:

$$MOTA = \frac{1 - \sum_{t=1}^{N_{frames}} (c_m(m_t) + c_f(fp_t) + c_s(ID - SWITCHES_t))}{\sum_{t=1}^{N_{frames}} N_G^{(t)}}$$

where, m_t is the number of misses, fp_t is the number of false positives, and $ID - SWITCHES_t$ is the number of ID mismatches in frame t considering the mapping in frame $t-1$. Therefore, during tracking, if there was a track split or merge, one would still consider the contribution of the new track but penalized it by counting it as an $ID - SWITCHES$. The values used for the weighting functions in this evaluation were $c_m = c_f = 1$ and $c_s = \log_{10}$. ID-SWITCH count is started from 1 because of the log function.

The *Multiple Object Tracking Precision (MOTP)* was defined as:

$$MOTP = \frac{\sum_{i=1}^{N_{mapped}} \sum_{t=1}^{N_{frames}^{(t)}} \left[\frac{\left|G_i^{(t)} \cap D_i^{(t)}\right|}{\left|G_i^{(t)} \cup D_i^{(t)}\right|} \right]}{\sum_{t=1}^{N_{frames}} N_{mapped}^{(t)}}$$

where N_{mapped} refers to the mapped system output objects over an entire reference track taking into account splits and merges and N_{mapped}^{t} mapped refers to the number of mapped objects in the t_{th} frame.

A comprehensive overview of object detection and tracking evaluation measures has been given in [121]. The author systematically address the challenges of object detection and tracking through a common evaluation framework that permits a meaningful objective comparison of detection and tracking techniques used for face, text and vehicle objects in video. They discussed the need of a large development and evaluation corpus that can be used to support machine learning approaches and statistically differentiate differences in system performance. A comprehensive description of evaluation measures is presented, which permit system performance differences to be discerned via a minimal number of measures. Overall they discussed the necessary infrastructure (source video, task definitions, metrics, ground truth, and scoring tools) to perform formal evaluations of face, text, and vehicle detection and tracking tasks.

In order to provide a quantitative perspective about the quality of foreground detection two of the measures are very popular among the researcher [122]. These measures are, the false negative rate (FNR) and false positive rate (FPR) and defined as

$$\text{FNR}=\frac{\text{the number of foreground pixels wrongly classified}}{\text{the number of foreground pixels in the ground truth}}$$

$$\text{FPR}=\frac{\text{the number of background pixels wrongly classified}}{\text{the number of background pixels in the ground truth}}$$

In [123] it was discussed that FNR, FPR measures not accurate enough, when averaging the measures over various environments. The author proposed a new formulation to evaluate the foreground segmentation. The new similarity measure was:

$$S(A, B)=\frac{A \cap B}{A \cup B}$$

where A and B represent detected region and ground truth region respectively. The value of S(A, B) lie between 0 (lest similarity) to 1. This measure integrates the FNR and FPR into a single measure. However, it is a nonlinear measure. Another measures used to quantify the performance of moving object segmentation are recall and precision [124].

$$\text{Recall}=\frac{\text{Number of foreground pixels correctly identified by the algorithm}}{\text{Number of foreground pixels in ground truth}}$$

$$\text{Precision} = \frac{\text{Number of foreground pixels correctly identified by the algorithm}}{\text{Number of foreground pixels detected by the algorithm}}$$

The value of recall and precision fall within the range of 0 and 1. A good background algorithm should attain as high a recall value as possible without sacrificing precision. In order to quantify the performance of the shadow detection method, two measures are proposed in [125], which are shadow detection rate (η) and shadow discrimination (ξ) and defined as:

$$\eta = \frac{\text{TP}_S}{\text{TP}_S + \text{FN}_S};\ \xi=\frac{\overline{\text{TP}_F}}{\text{TP}_F + \text{FN}_F}$$

Where TP represent the number of true positives, the $\overline{\text{TP}_F}$ is the number of ground truth points of the foreground objects minus the number of points detected as shadows, but belonging to foreground objects. The subscript S stands for shadow and F for foreground.

6.2 Data Sets

Data set is one of key components of any system. Evaluating the algorithm against a standard dataset is one of the challenging tasks in object tracking. In the recent years a number of common data set is available by different communities. The Performance Evaluation of Tracking and Surveillance (PETS) series of workshops was unique in the tracking community in that a common data set was used by all of the workshop participants. In the following section we are giving details of these data sets.

- PETS'2000: Outdoor people and vehicle tracking (single camera)
  ftp://ftp.pets.rdg.ac.uk/pub/PETS2000/

- PETS'2001: Outdoor people and vehicle tracking (two synchronised views; includes omnidirectional and moving camera) (annotation available)

ftp://ftp.pets.rdg.ac.uk/pub/PETS2001
http://www.cvg.cs.rdg.ac.uk/PETS2001/pets2001-dataset.html

- PETS'2002: Indoor people tracking (and counting) and hand posture classification data (annotated).

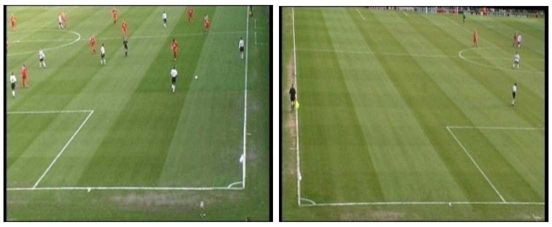
ftp://ftp.pets.rdg.ac.uk/pub/PETS2002
http://www.cvg.cs.rdg.ac.uk/PETS2002/pets2002-db.html

- PETS ICVS'2003: Annotation of a smart meeting (annotation available). Includes facial expressions, gaze and gesture/action.

ftp://ftp.pets.rdg.ac.uk/pub/PETS-ICVS
http://www.cvg.cs.rdg.ac.uk/PETS-ICVS/pets-icvs-db.html

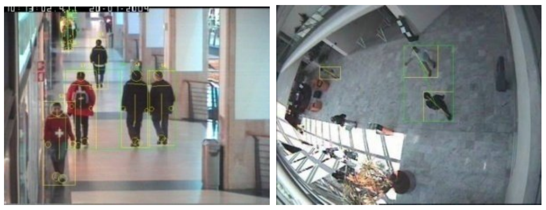
- VS PETS 2003: Outdoor people tracking - football data (two views). The datasets consists of football players moving around a pitch.

ftp://ftp.pets.rdg.ac.uk/pub/VS-PETS
http://www.cvg.cs.rdg.ac.uk/VSPETS/vspets-db.html



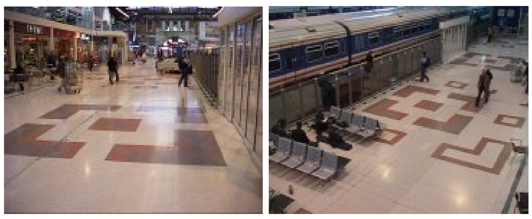
- PETS ECCV'2004: CAVIAR people scenarios

http://groups.inf.ed.ac.uk/vision/CAVIAR/CAVIARDATA1/



- PETS'2006: Multi-sensor sequences containing left-luggage scenarios with increasing scene complexity.

ftp://ftp.pets.rdg.ac.uk/pub/PETS2006/

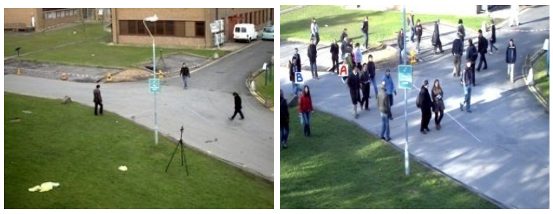


- PETS'2007: The datasets are multisensor sequences containing the following 3 scenarios, with increasing scene complexity: 1. loitering, 2. attended luggage removal (theft), 3. Unattended luggage.

ftp://ftp.pets.rdg.ac.uk/pub/PETS2007/
http://pets2007.net/

- PETS'2009: The datasets are multisensor sequences containing different crowd activities.

http://www.cvg.rdg.ac.uk/PETS2009/a.html



- SPEVI (Audiovisual people dataset): This is a dataset for uni-modal and multi-modal (audio and visual) people detection tracking. The dataset consists of three sequences recorded in different scenarios with a video camera and two microphones.

http://www.elec.qmul.ac.uk/staffinfo/andrea/spevi.html

7 Conclusion and Future Directions

Over the past one decade moving object detection and tracking has continued to be a booming area of research. The demands of potential mass-market applications in surveillance, human computer interaction and video retrieval makes this area favorite among the researcher. Increased activity in this research area has been driven by both academia and industry.

In this overview paper, we have discussed some of the core concepts used in object tracking and present a comprehensive survey of efforts in the past to address this problem. In the recent year, an important issue that has been got attention is the integration of knowledge (contextual information) in the design of tracking methods. In [126], an unsupervised data mining approach has been integrated to reduce the uncertainty in the tracking process. A set of auxiliary objects are found during the process, which gives extra information to help the tracking process. The occlusion can be resolved by exploiting the motion correlations among the auxiliary object and the target. A multi-object tracking framework exploiting the scene contextual information (target births, spatially persistent clutter) as feedback was proposed in [127]. A GMM is used to model birth and clutter data.

Knowledge of multi modality can also help in tracking and specially occlusion handling, because each modality may introduce new information that compensate the weaknesses of other. In [128] the author proposed a framework for object tracking using joint statistical characteristics of the audio-video data. The two modalities were fused at semantic level to help the tracking task.

Although during the last few years there has been a substantial progress towards object detection and tracking. But tracking an object in an unconstraint, noisy and dynamic environments are still makes this problem a central focus of research interest. Developing a background model robust and efficient to environmental changes is still a challenging task. Exploitation of prior and contextual knowledge in tracking is still in its initial phase. Tracking objects in noisy and compressed video data is also required a serious attention. Most of the past researches have focused independently on object detection and object tracking. Thus there is great demand of developing a robust and efficient approach that can incorporate the task of object detection, tracking and analysis in a single framework. Although in the recent years a number of benchmark data set (e.g. PETS data set) and evaluation measures are proposed by different communities but still the development of a common benchmark data and evaluation measures, which can cover all kind of scenario, is a critical requirement for object detection and tracking. Future work can also focus on development of parallel version of tracking algorithm so that one can utilize the processing capability of network resources.

References

[1] T. Acharya and A. K. Ray (2005). *Image processing: principles and applications*, New Jersey: Wiley-Interscience.

[2] M. Shah, O. Javed and K. Shafique (2007). Automated visual surveillance in realistic scenarios. *IEEE MultiMedia*, vol.14, no.1, pp.30-39.

[3] W. M. Hu, T. N. Tan, L. Wang and S. Maybank (2004). A survey on visual surveillance of object motion and behaviors. *IEEE Transactions on Systems, Man and Cybernetics*, vol.34, no.3, pp. 334-352.

[4] D. A. Forsyth and J. Ponce. (2003) *Computer vision: a modern approach*, Prentice Hall.

[5] P. Perez, C. Hue, J. Vermaak and M. Gagnet (2002). Color-Based Probabilistic Tracking. *In Proceedings of ECCV*, pp. 661–675.

[6] K. Pahlavan and J.O. Eklundh (1992). A head-eye system- analysis and design. *CVGIP: Image Understanding*, vol. 56, pp. 41–56.

[7] P. Tissainayagam and D. Suter (2005). Object tracking in image sequences using point features. *Pattern Recognition*, vol. 38, pp. 105–113.

[8] D. Comaniciu, V. Ramesh and P. Meer (2000). Real-time tracking of non-rigid objects using mean shift. *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, pp.142-149.

[9] D. Serby, E. K. Meier and L.V. Gool (2004). Probabilistic object tracking using multiple features. *In Proceedings of International Conference on Pattern Recognition*, pp. 184–187.

[10] R. Cipolla and M. Yamamoto (1990). Stereoscopic tracking of bodies in motion. *Image and vision computing*, vol. 8, no. 1, pp. 85–90.

[11] A. Yilmaz, O. Javed and M. Shah (2006). Object tracking: a survey. *ACM Journal of Computing Surveys*, vol. 38, no.4, Article 13.

[12] C. Veenman, M. Reinders, E. Backer (2001). Resolving motion correspondence for densely moving points. *IEEE Trans. Patt. Analy. Mach. Intell.*, vol. 23, no. 1, pp. 54–72.

[13] L. M. Novak (1981). Optimal target designation techniques. *IEEE Transactions on Aerospace and Electronic Systems*, vol. 17, no.5, pp. 676–684.

[14] N. K. Kanhere and S. T. Birchfield (2008). Real-time incremental segmentation and tracking of vehicles at low camera angles using stable features. *IEEE Transactions on Intelligent Transportation Systems*, vol. 9, no. 1, pp.148–160.

[15] D. Angelova and L. Mihaylova (2008). Extended object tracking using monte carlo methods. *IEEE Transactions on Signal Processing*, vol. 56, no.2, pp.825–832.

[16] F. Gustafsson, F. Gunnarsson, N. Bergman, U. Forssell, J. Jansson, R. Karlsson, and P. J. Nordlund (2002). Particle filters for positioning, navigation, and tracking. *IEEE Transactions on Signal Processing*, vol. 50, no.2, pp.425–437.

[17] C. Kwok, D. Fox, and M. Meila (2004). Real-time particle filters. *Proceedings of IEEE*, vol. 92, no.3, pp.469–484.

[18] Y. Chen, G. Liang, K. K. Lee, and Y. Xu (2007). Abnormal behavior detection by multi-svm-based bayesian network. *In Proceedings of the International Conference on Information Acquisition*, pp. 298–303.

[19] X. Wu, Y. Ou, H. Qian, and Y. Xu (2005). A detection system for human abnormal behavior. *In Proceedings of the International Conference on Intelligent Robots and Systems*, pp. 1204–1208.

[20] D. Kragic and H. I. Christensen (2000). Tracking techniques for visual servoing tasks. *In Proceedings of the IEEE International Conference on Robotics and Automation*, vol. 2, pp. 1663–1669.

[21] G. Unal, H. Krim, and A. Yezzi (2005). Fast incorporation of optical flow into active polygons. *IEEE Transactions on Image Processing*, vol. 14, no. 6, pp. 745–759.

[22] J. Melo, A. Naftel, A. Bernardino, and J. Santos-Victor (2006). Detection and classification of highway lanes using vehicle motion trajectories. *IEEE Transactions on Intelligent Transportation Systems*, vol. 7, no.2, pp.188–200.

[23] C. Shu-Ching, S. Mei-Ling, S. Peeta, and Z. Chengcui (2003). Learning-based spatio-temporal vehicle tracking and indexing for transportation multimedia database systems. *IEEE Transactions on Intelligent Transportation Systems*, vol. 4, no. 3, pp.154–167.

[24] C. Yang, R. Duraiswami, and L. Davis (2005). Fast multiple object tracking via a hierarchical particle filter. *In Proceedings of the IEEE International Conference on Computer Vision*, vol. 1, pp. 212–219.

[25] G. Bradski (1998). Computer vision face tracking for use in a perceptual user interface. *Intel Technology Journa*l, vol.2, no.2, pp.1-15.

[26] C. Chang, R. Ansari, and A. Khokhar (2005). Multiple object tracking with kernel particle filter. *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, vol. 1, pp. 566–573.

[27] D. Comaniciu, V. Ramesh and P. Meer (2003). Kernel-based object tracking. *IEEE Transactions Pattern Analysis and Machine Intelligence*, vol.25, no.5, pp.564-575.

[28] D. Ramanan and D. A. Forsyth (2003). Finding and tracking people from the bottom up. *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, vol. 2, pp. 467–474.

[29] I. Haritaoglu, D. Harwood, and L. S. Davis (2000). W4: Real-time surveillance of people and their activities. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no.8, pp.809–830.

[30] A. Ali and J. Aggrawal (2001). Segmentation and recognition of continuous human activity. *In IEEE Workshop on Detection and Recognition of Events in Video*. Pp. 28–35.

[31] C. Wren, A. Azarbayejani, T. Darrell and A.P. Pentland (1997). Pfinder: Real time tracking of the human body. *IEEE Transactions Pattern Analysis and Machine Intelligence*, vol.19, no.7, pp.780-785.

[32] C. Stauffer and W.E.L. Grimson (1999). Adaptive background mixture models for real-time tracking. *Proc. IEEE Int'l Conf. Computer Vision and Pattern Recognition*, pp.246-252.

[33] A. Elgammal, R. Duraiswami, D. Harwood and L. S. Davis (2002). Background and foreground modeling using non-parametric kernel density estimation for visual surveillance. *Proceedings of the IEEE*, pp.1151-1163.

[34] A. Yilmaz, X. Li and M. Shah (2004). Contour based object tracking with occlusion handling in video acquired using mobile cameras. *IEEE Trans. Patt. Analy. Mach. Intell.* vol.26, no. 11, pp. 1531–1536.

[35] A. Rao, R. Srihari, and Z. Zhang (2000). Geometric histogram: a distribution of geometric configuration of color subsets. *in SPIE: Internet Imaging*. vol. 3964, pp. 91-101.

[36] R. Brunelli (2009). *Template matching techniques in computer vision: theory and practice*. Wiley.

[37] R. Bastos and J. M. S. Dias (2005). Fully automated texture tracking based on natural features extraction and template matching. *In Proceedings of the 2005 ACM SIGCHI International Conference on Advances in Computer Entertainment Technology*, pp.180–183.

[38] F. Jurie and M. Dhome (2001). A simple and efficient template matching algorithm. *In Proceedings of the IEEE International Conference on Computer Vision*, vol. 2, pp. 544–549.

[39] L. J. Latecki and R. Miezianko (2006). Object tracking with dynamic template update and occlusion detection. *In Proceedings of the IEEE International Conference on Pattern Recognition*, vol. 1, pp. 556–560.

[40] C.T. Nguyen, J.P. Havlicek, M. Yeary (2007). Modulation domain template tracking. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1–8.

[41] T.F. Cootes, G. J. Edwards, and C. J. Taylor (1998). Active appearance models. *ECCV*, vol. 2, pp. 484–498.

[42] M. B. Stegmann (2001). Object tracking using active appearance models. *in Proc. 10th Danish Conference on Pattern Recognition and Image Analysis*. vol. 1, pp. 54–60.

[43] R. Gonzalez and R. Woods (2002). *Digital Image Processing*, 2nd edition, Prentice Hall.

[44] M. Sonka, V. Hlavac and R. Boyle (2008). *Image Processing, Analysis and Machine Vision*, 3rd edition, Singapore: Thomson Asia Pvt. Ltd.

[45] D. Ziou and S. Tabbone (1998). Edge detection techniques - an overview. *International Journal of Pattern Recognition and Image Analysis*, vol. 8, pp. 537–559.

[46] B.K.P. Horn and B.G. Schunk (1981). Determining optical flow. *Artificial Intelligence*, vol.17, no.1, pp.185-203.

[47] B.D. Lucas and T. Kanade (1981). An iterative image registration technique with an application to stereo vision. *Proc. of the Workshop on Image Understanding*, pp.674-679, 1981.

[48] A. Tavakkoli, M. Nicolescu, G. Bebis and M. Nicolescu (2009). Non-parametric statistical background modeling for efficient foreground region detection. *Machine Vision and Applications, Springer*, vol. 20, no.6, pp.395-409.

[49] L. Maddalena and A. Petrosino (2008). A self-organizing approach to background subtraction for visual surveillance applications. *IEEE Transactions on Image Processing*, vol.17, no.7, pp.1168-1177.

[50] C. Stauffer and W.E.L. Grimson (2000). Learning patterns of activity using real time tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol.22, no.8, pp.747-757.

[51] B.P.L. Lo and S.A. Velastin (2001). Automatic congestion detection system for underground

platforms. *Proc. Int'l Symp. Intelligent Multimedia, Video, and Speech Processing*, pp.158-161.

[52] S. Calderara, R. Melli, A. Prati and R. Cucchiara (2006). Reliable background suppression for complex scenes. *ACM international workshop on Video surveillance and sensor networks*, pp.211-214.

[53] R. Cucchiara, M. Piccardi and A. Prati (2003). Detecting moving objects, ghosts, and shadows in video streams. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol.25, no.10, pp.1337-1342.

[54] K. Toyama, J. Krumm, B. Brumitt and B. Meyersv (1999). Wallflower: principles and practice of background maintenance. *Proc. 7th IEEE Conf. Computer Vision*, vol.1, pp.255-261.

[55] M. Piccardi (2004). Background subtraction techniques: a review. *Proc. IEEE International Conference on Systems, Man and Cybernetics*, vol.4, pp.3099- 3104.

[56] D. H. Parks and S. S. Fels (2008). Evaluation of background subtraction algorithms with post-processing. *Proc. IEEE Int'l Conf. Advanced Video and Signal-based Surveillance*, pp.192-199.

[57] M. Heikkila and M. Pietikainen (2006). A texture-based method for modeling the background and detecting moving objects. *IEEE Transactions Pattern Analysis Machine Intelligence*, vol.28, no.4, pp.657-662.

[58] A. Prati, I. Mikic, M.M. Trivedi and R. Cucchiara (2003). Detecting moving shadows: algorithms and evaluation. *IEEE Transactions. Pattern Analysis Machine Intelligence*, vol.25, no.6, pp.918-923.

[59] E. Salvador, A. Cavallaro and T. Ebrahimi (2004). Cast shadow segmentation using invariant color features. *Computer Vision and Image Understanding*, vol.95, no.3, pp.238-259, 2004.

[60] J.W. Hsieh, W.F. Hu, C.J. Chang and Y.S. Chen (2003). Shadow elimination for effective moving object detection by Gaussian shadow modeling. *Image and Vision. Computing*, vol.21, no.6, pp.505-516.

[61] S. Nadimi and B. Bhanu (2004). Physical models for moving shadow and object detection in video. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol.26, no.8, pp.1079-1087.

[62] Liu Zhi Fang, Wang Yun Qiong and You Zhi Sheng (2008). A method to segment moving vehicle cast shadow based on Wavelet transform. *Pattern Recognition Letters, Elsevier*, vol.29, no.16, pp.2182-2188.

[63] A. Leone, C. Distante, and F. Buccolieri (2007). Shadow detection for moving objects based on texture analysis. *Pattern Recognition, Elsevier*, vol.40, pp.1222-1233.

[64] T. Broida and R. Chellappa (1986). Estimation of object motion parameters from noisy images. *IEEE Transactions. Pattern Analysis Machine Intelligence*, vol.8, no.1, pp.90-99.

[65] G. Welsh and G. Bishop (1995). An introduction to the kalman filter. Technical Report TR95-041, University of North Carolina, Chapel Hill, NC, 1995.

[66] Jiyan Pan, Bo Hu, and Jian Q. Zhang (2006). An efficient object tracking algorithm with adaptive prediction of initial searching point. *Proc of the IEEE Pacific-Rim Symposium on Image and Video Technology (PSIVT'06), Lecture Notes in Computer Science*, vol. 4319, pp. 1113-1122.

[67] L. Mihaylova, P. Brasnett, N. Canagarajah and D. Bull (2007). Object tracking by particle filtering techniques in video sequences. *Advances and Challenges in Multisensor Data and Information Processing*, vol. 8, pp.260-268.

[68] D. Comaniciu and P. Meer (2002). Mean shift: a robust approach toward feature space analysis. *IEEE Transactions Pattern Analysis and Machine Intelligence*, vol.24, no.5, pp.603-619.

[69] A. Dargazany, A. Soleimani and A. Ahmadyfard (2010). Multi-bandwidth kernel-based object tracking. *Journal of Advances in Artificial Intelligence, Hindawi Publication*, Article ID.175603.

[70] A. Adam, E. Rivlin, I. Shimshoni (2006). Robust fragments-based tracking using the integral histogram. *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 798–805.

[71] J. Jeyakar, R. V. Babu, and K.R. Ramakrishnan (2008). Robust object tracking with background-weighted local kernels. *Computer Vision and Image Understanding, Elsevier*, vol.112, no.3, pp.296-309.

[72] F. L. Wang, S. Y. Yu, J. Yang (2009). Robust and efficient fragments-based tracking using mean shift. *International Journal of Electronics and Communications*, pp. 1-10, 2009.

[73] M. Lucena, J. M. Fuertes, N. Blanca, Manuel J and M. Jimenez (2010). Tracking people in video sequences using multiple model. *Multimedia Tools and Applications*, Springer, vol.49, no.2, pp.371-403.

[74] F. Porikli and O. Tuzel (2005). Multi-kernel object tracking. *Proc. IEEE Int'l Conf. on Multimedia and Expo.*, pp.1234-1237.

[75] J. Fang, J. Yang, H. Liu (2011). Efficient and robust fragments-based multiple kernels track. *International Journal of Electronics and Communications, Elsevier*, vol. 65, pp. 915-923.

[76] J.Q. Wang and Y.S. Yagi (2008). Integrating color and shape-texture features for adaptive real-time object tracking. *IEEE Transactions on Image Processing*, vol.17, no.2, pp.235-240.

[77] J. Ning, L. Zhang, D. Zhang and C. Wu (2009). Robust object tracking using joint color-texture histogram. *International Journal of Pattern Recognition and Artificial Intelligence*, vol.23, no.7, pp.1245-1263.

[78] F. Deboeverie, K. Teelen, P. Veelaert and W. Philips (2009). Vehicle tracking using geometric features. Advanced Concepts for Intelligent Vision Systems, LNCS, vol.5807, pp.506-515.

[79] M. Hu, W. Hu, and T. Tan (2004). Tracking people through occlusions. *Proceedings of the 17th International Conference on Pattern Recognition*, pp.724-727.

[80] C. Shen, J. Kim and H. Wang (2010). Generalized kernel-based visual tracking. *IEEE Transactions on Circuits and Systems for Video Technology*, vol.20, no.1, pp.119-130.

[81] D. Ross, J. Lim, R. Lin and M. Yang (2008) Incremental learning for robust visual tracking. *International Journal of Computer Vision, Springer*, vol.77, no.1, pp.125-141.

[82] B. Zhang, W. Tian and Z. Jin (2008). Robust appearance-guided particle filter for object tracking with occlusion analysis. *International Journal on Electronics and Communications*, Elsevier, vol.62, no.1, pp.24-32.

[83] Z. Zhao, S. Yu, X. Wu, C. Wang and Y. Xu (2009). A Multi-target Tracking Algorithm using Texture for Real-time Surveillance. *Proc. IEEE Int'l Conf. on Robotics and Biomimetics*, pp. 2150-2155.

[84] J. X. Wang, G. Bebis and M. Nicolescu (2009). Improving target detection by coupling it with tracking. *Machine Vision and Applications, Springer*, vol.20, no.4, pp.205-223.

[85] M. Khansari, H. R. Rabiee, M. Asadi and M. Ghanbari (2008). Object tracking in crowded video scenes based on the Undecimated Wavelet features and texture analysis. *EURASIP Journal on Advances in Signal Processing*, Article ID. 243534.

[86] Y. Tang (2008). Status of Pattern Recognition with Wavelet Analysis. *International Journal, Frontiers of Computer Science, Springer*, vol.2, no.3, pp.268-294.

[87] N. G. Kingsbury and J. F. A. Magarey (1997). Wavelet transforms in image processing. *Proc. First European Conference on Signal Analysis and Prediction*, pp.23-34.

[88] D. Clonda, J.M. Lina and B. Goulard (2004). Complex daubechies wavelets: properties and statistical image modeling. *Signal Processing (Elsevier)*, vol.84, pp.1-23, 2004.

[89] S. Mallat (1989). A theory of multiresolution signal decomposition: the wavelet representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol.11, pp.674-693.

[90] Y. Xu, J.B. Weaver, D.M. Healy and J. Lu (1994). Wavelet transform domain filters: a spatially selective noise filtration technique. *IEEE Transactions on Image Processing*, vol. 3, pp.747-758.

[91] J. K. Romberg, H. Choi and R. G. Baraniuk (2001). Multiscale Edge Grammars for Complex Wavelet Transforms. *Proc. IEEE Int. Conf. on Image Processing*, pp.614-617.

[92] Z. Xiong, K. Ramchandran, M. T. Orchard and Y. Q. Zhang (1999). Comparative Study of DCT and Wavelet Based Image Coding. *IEEE Transactions on Circuits and Systems for Video Technology*, vol.9, pp.692-695.

[93] Y. Wang, F. John F. Doherty, Robert E. Van Duck (2000). Moving object tracking in video. *Proceedings of 29th IEEE Int'l Conference on Applied Imagery Pattern Recognition Workshop*, pp. 95-101.

[94] J. B. Kim, C. W. Lee, K. M. Lee, T. S. Yun, and H. J. Kim (2001). Wavelet-based vehicle tracking vehicle for automatic traffic surveillance. *Proceedings of IEEE Region 10 International Conference on Electrical and Electronic Technology*, TENCON, vol.1, pp.313-316.

[95] C. He, Y. F. Zheng and S.C. Ahalt (2002). Object tracking using the gabor wavelet transform and the golden section algorithm. *IEEE Transactions on Multimedia*, vol. 4, no. 4, pp. 528–538.

[96] F.H. Cheng and Y. L. Chen (2006). Real time multiple objects tracking and identification based on discrete wavelet transform. *Pattern Recognition*, vol. 39, no. 6, pp. 1126–1139.

[97] Shyang-Li Chang, Chen-Chien Hsu, Tsung-Chi Lu, Ti-Ho Wang (2007). Human body tracking based on discrete wavelet transform. *Proceedings of the 2007 WSEAS International Conference on Circuits, Systems, Signal and Telecommunications,* pp:113–122.

[98] I.W. Selesnick, R.G. Baraniuk and N. Kingsbury (2005). The dual-tree complex wavelet transform. *IEEE Signal Processing Magazine*, pp. 123-151.

[99] M. Amiri, H. R. Rabiee, F. Behazin and M. Khansari (2003). A New Wavelet Domain Block Matching Algorithm for Real-Time Object Tracking. *Proceedings International Conference on Image Processing*, vol. 3.

[100] J. M. Lina and M. Mayrand (1995). Complex Daubechies Wavelets. *Applied and Computational Harmonic Analysis,* vol. 2, pp. 219-229.

[101] A. S. Jalal and V. Singh (2011). Robust object tracking under appearance change conditions based on Daubechies complex wavelet transform. *International Journal of Multimedia Intelligence and Security, Inderscience*, vol. 2, no. 3, pp. 252-268.

[102] P. F. Gabriel, J. G. Verly, J. H. Piater and A. Genon (2003). The state of the art in multiple object tracking under occlusion in video sequences. *Proc. of the Advanced Concepts for Intelligent Vision Systems*, pp.166-173.

[103] A. Mittal and L. Davis (2003). M2tracker: a multi-view approach to segmenting and tracking people in a cluttered scene. *International Journal of Computer Vision, Springer*, vol.51, no.3, pp.189-203.

[104] K. Smith, D. Gatica-Perez and J.-M. Odobez (2005). Using particles to track varying numbers of interacting people. *Proc. Int'l Conf. on Computer Vision and Pattern Recognition*, pp.962-969.

[105] P. Nillius, J. Sullivan and S. Carlsson (2006). Multi-target tracking - linking identities using bayesian network inference. *Proc. Int'l Conf. on Computer Vision and Pattern Recognition,* pp.2187-2194.

[106] M. Han, W. Xu, H. Tao and Y. Gong (2007). Multi-object trajectory tracking. *Machine Vision and Applications, Springer*, vol.18, no.3, pp.221-232.

[107] S.W. Joo and R. Chellappa (2007). Multiple-hypothesis approach for multi-object visual tracking. *IEEE Transactions on Image Processing*, vol.16, pp.2849-2854.

[108] K. Okuma, A. Taleghani, D. Freitas, J.J. Little, D.G. Lowe (2004). A boosted particle filter: multitarget detection and tracking. *Proc. European Conf. on Computer Vision*, pp. 28–39.

[109] Y. Cai, N. De Freitas, J.J. Little (2006). Robust visual tracking for multiple targets. *Proc. European Conf. on Computer Vision*, pp. 107–118.

[110] A. Senior, A. Hampapur, Y.-L. Tian, L. Brown, S. Pankanti and R. Bolle (2006). Appearance models for occlusion handling. *Journal of Image and Vision Computing*, Elsevier, vol.24, no.11, pp.1233-1243.

[111] R. Rad and M. Jamzad (2005). Real time classification and tracking of multiple vehicles in highways. *Pattern Recognition Letter*, Elsevier, vol.26, no.10, pp.1597-1607.

[112] A. Leykin and R. Hammoud (2010). Pedestrian tracking by fusion of thermal-visible surveillance videos. *Machine Vision and Applications, Springer*, vol.21, pp.587-595.

[113] Y. Benezeth, B. Emile, H. Laurent and C. Rosenberger (2010). Vision-based system for human detection and tracking in indoor environment. *Special Issue on People Detection and Tracking of the International Journal of Social Robotics, Springer*, vol.2, no.1, pp.41-52.

[114] A. Amer (2005). Voting-based Simultaneous Tracking of Multiple Video Objects. *IEEE Transactions on Circuits and Systems for Video Technology*, vol.15, pp.1448-1462.

[115] Y. Lao, Y. F. Zheng (2011). Tracking highly correlated targets through statistical multiplexing. *Image and Vision Computing*. vol. 29, no.12, pp. 803-817.

[116] J. Black, T. Ellis, and P. Rosin (2003). A novel method for video tracking performance evaluation. *IEEE International Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance*, pp. 125–132.

[117] A. S. Jalal, U. S. Tiwary (2009). A robust object tracking method for noisy video using rough entropy in wavelet domain. *Proceedings of the International Conference Intelligent Human Computer Interaction, Springer India*, ISBN 978818489203, pp. 113-121.

[118] S. Muller-Schneiders, T. Jager, H. S. Loos, and W. Niem (2005). Performance evaluation of a real time video surveillance systems. *IEEE International Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance*, pp. 137–143.

[119] F. Yin, D. Makris, S. A. Velastin (2007). Performance evaluation of object tracking algorithms. *Proc. of the 10th IEEE International Workshop on Performance Evaluation of Tracking and Surveillance*.

[120] Keni Bernardin and Rainer Stiefelhagen (2008). Evaluating multiple object tracking performance: the CLEAR MOT Metrics. *EURASIP Journal on Image and Video Processing*, Article ID 246309.

[121] R. Kasturi, D. Goldgof, P. Soundararajan, V. Manohar, J. Garofolo, R. Bowers, M. Boonstra, V. Korzhova, J. Zhang (2009). Framework for performance evaluation of face, test, and vehicle detection and tracking in video: data, metrics and protocol. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, no. 2, pp.319-336.

[122] M. Heikkilä, M. Pietikäinen (2006). A texture-based method for modeling the background and detecting moving objects. *IEEE Trans. Pattern Anal. Machine Intelligence*, vol. 28, no. 4, pp.657-662.

[123] L. Li, W. Huang, I. Gu, Q. Tian (2004). Statistical modeling of complex background for foreground object detection. *IEEE Transactions on Image Processing*, vol. 13, no. 11, pp. 1459-1472.

[124] S.C.S. Cheung, C. Kamath (2005). Robust techniques for background subtraction in urban traffic video. *EURASIP Journal on Applied Signal Processing*, vol.14, pp. 2330–2340.

[125] A. Prati, I. Mikic, M.M. Trivedi, and R. Cucchiara (2003). Detecting moving shadows: algorithms and evaluation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 25, no. 7, pp. 918-923.

[126] Ming Yang, Ying Wu and Gang Hua (2008). Context-aware visual tracking. *IEEE Transaction On Pattern Analysis And Machine Intelligence*, vol. 31,no.7, pp. 1195-1209.

[127] E. Maggio and A. Cavallaro (2009). Learning scene context for multiple object tracking. *IEEE Transactions on Image Processing*, vol. 18, no. 8, pp. 1873-1884.

[128] M. J. Beal, N. Jojic and H. Attias (2003). A graphical model for audiovisual object tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 25, no. 7, pp. 828-836.

Mathlab Implementation of Quantum Computation in Searching an Unstructured Database

I.O. Awoyelu and P. Okoh
Department of Computer Science & Engineering, Obafemi Awolowo University, Ile-Ife, Nigeria
E-mail: iawoyelu@oauife.edu.ng

Keywords: qubit, quantum algorithm, classical computing, quantum computing

Received: August 15, 2011

In the classical model of a computer, the most fundamental building block, the bit, can only exist in one of two distinct states, a 0 or a 1. Computations are carried out by logic gates that act on these bits to produce other bits. Unless there is duplicate (parallel) hardware, only one problem instance (i.e. input data set) can be handled at a time. In this classical computing, increasing the number of bits increases the complexity of the problem and the time necessary to arrive at a solution.
A quantum algorithm consists of a sequence of operations on a register, to transform it into a state which, when measured, yields the desired result with high probability. An n-bit quantum register can store an exponential amount of information.
This paper aims at taking advantage of the superiority of quantum computing over classical computing to solve the problem of searching unstructured databases for a particular item or more than one item in good time. The general aim of this work is to establish the correctness and optimality of Grover's quantum database search algorithm compared against classical database search methods in order to investigate the superiority or otherwise of quantum computing over classical computing. This is followed by a simulation of the algorithm using a classical computer, namely through functions that are present in MATLAB, referred to as "Quantum Computing Functions".

Povzetek: Članek predstavlja implementacijo kvantnega iskanja v nestrukturiranih bazah.

1 Introduction

The bit is the fundamental unit of storage in a classical computer; similarly, the basis of quantum computation is a qubit. The qubit is similar to a bit in that when measured, its value will be either 0 or 1. It differs primarily in what it is doing when it is not being measured. In particular, a qubit can exist in any superposition of the 0 and 1 state simultaneously. When a qubit in such a state is measured the superposition will be destroyed. It will be found to be uniquely in the 0 or 1 state with some probability for each, determined by the particulars of the superposition prior to the measurement (Williams and Clearwater, 1998).

The renowned scientist Moore's empirical law, which states that "computing power doubles approximately every 18 months" is believed by just about everyone that is into computing (Williams and Clearwater, 1998). Computer components have been steadily decreasing in size and increasing in speed, tending to follow this prediction. However, what is the fate of this trend in real life? A little extrapolation would suggest that within about 10 years the size of a transistor logic gate element will be only a few atoms. Consequently, computer power will soon reach a limit, unless another approach for computing can be developed. Quantum computing is one possible approach.

A quantum algorithm consists of a sequence of operations on a register, to transform it into a state which, when measured, yields the desired result with high probability. An n-bit quantum register can store an exponential amount of information. The register as a whole can be in an arbitrary superposition of the 2^n base states which it can be measured to be in. While in this superposition, and computation applied to the register will be applied to each component of the superposition, this behavior follows from the linearity of operators on quantum mechanical systems. This behavior, called "quantum parallelism" is the basis for most quantum algorithms. In 1982, the Nobel prize-winning physicist, Richard Feynman, thought up the idea of a 'quantum computer', a computer that uses the effects of quantum mechanics to its advantage (Deutsch and Jozsa, 1992).

In the classical model of a computer, the most fundamental building block, the bit, can only exist in one of two distinct states, a 0 or a 1. Computations are carried out by logic gates that act on these bits to produce other bits. Unless there is duplicate (parallel) hardware, only one problem instance (i.e. input data set) can be handled at a time. In this classical computing, increasing the number of bits increases the complexity of the problem and the time necessary to arrive at a solution.

In a quantum computer, the rules are changed. Not only can a 'quantum bit', usually referred to as a 'qubit', exist in the classical 0 and 1 states, it can also be in a coherent superposition and all linear combinations of both.

The number of possibilities grows exponentially with the number of qubits. For example, for a 2-qubit system, there are all possible superpositions of the states 00, 01, 10, and 11, including entangled states of the form (01 ±10). If there are N qubits, the vector space required to describe their states has dimension 2^N . Calculations are carried out on vectors by quantum gates that apply unitary transformations to these vectors to produce other vectors. Since quantum computers can process superpositions, they can (at least for some problems) be viewed as devices that can process all possible inputs simultaneously.

When a qubit is in this state, it can be thought of as existing in two universes: as a 0 in one universe and as a 1 in the other. An operation on such a qubit effectively acts on both values at the same time. The significant point being that by performing the single operation on the qubit, we have performed the operation on two different values. Likewise, a two-qubit system would perform the operation on 4 values, and a three-qubit system on eight. Increasing the number of qubits therefore exponentially increases the 'quantum parallelism' we can obtain with the system (Williams and Clearwater, 1998). This is a great leap from the classical computing we are familiar with in which increasing the number of bits increases the complexity of the problem and the time necessary to arrive at a solution.

With the correct type of algorithm, it is possible to use this parallelism to solve certain problems in a fraction of the time taken by a classical computer. A quantum computer would consist of many qubit gates with entangled states. These gates could be addressed in parallel by unitary transformations, which must be carried out reversibly, implying no loss of energy in a gate operation. Quantum computers are "wired" so that they can do many calculations at the same time. This is known as "quantum parallelism" and represents the power of a quantum computer.

Several systems have been proposed for quantum computing including photons in nonlinear optical systems, trapped ions, electron and nuclear spins, quantum dots, and Josephson junctions (Grover, 2000). There are advantages and disadvantages to all these approaches. Some, such as those employing light or trapped ions, have demonstrated that they can provide individual qubits of excellent quality. But, it is not yet known if they can be scaled up to produce systems with many qubits and many possible quantum gate operations.

There is nothing a quantum computer can do that cannot also be done by an ordinary computer. However, for some problems, a quantum computer may be many orders of magnitude faster. There are presently two important problems involving commerce and security for which a quantum computer is believed to be superior. These are finding the factors of a large number and searching an unstructured database. A quantum computer would also be superior for simulating the behavior of quantum systems, and thus have enormous implications for physics.

This paper aims at taking advantage of the superiority of quantum computing over classical computing to solve the problem of searching unstructured databases for a particular item or more than one item in good time. The general aim of this work is to establish the correctness and optimality of Grover's quantum database search algorithm compared against classical database search methods in order to investigate the superiority or otherwise of quantum computing over classical computing. This is followed by a simulation of the algorithm using a classical computer, namely through functions that are present in MATLAB, referred to as "Quantum Computing Functions".

2 Existing Works in Quantum Computing

In the early 1980's, physicist Richard Feynman observed that no classical computer could simulate quantum mechanical systems without incurring exponential slowdown (Williams and Clearwater, 1998). At the same time, it seems reasonable that a computer which behaves in a manner consistent with quantum mechanics could, in principle, simulate such systems without exponential slowdown.

For many years the study of quantum computing was primarily an academic curiosity. Shor (1994) developed a polynomial time algorithm for factoring large integers. According to Williams and Clearwater (1998), it is not known if there is a classical algorithm for factoring large integers efficiently, but the best algorithms published thus far are super-polynomial. This algorithm coupled with the prominence of cryptographic systems based on factoring large integers fueled study of quantum computation, both from an algorithmic and a manufacturing point of view. Grover (1996) provided an $O(\sqrt{n})$ time algorithm for finding a single marked element in an unsorted database of n elements. The best possible classical algorithm would run in O (n) time. This search problem was not the first problem for which a quantum computer was shown to be better than any possible classical computer, but it was the first problem of real utility found where a quantum computer outperforms a classical computer in an asymptotic sense.

While Shor's algorithm may be of more immediate utility, Grover's algorithm seems more interesting in a theoretical sense, as it highlights an area of fundamental superiority in quantum computation.

2.1 Grover's Algorithm

Assuming there is a system with $N = 2^n$ states labeled S_1, $S_2,\ldots, S_N$. These 2^n states are represented by *n* bit strings. Assuming there is a unique marked element S_m that satisfies a condition $C(S_m) = 1$, and for all other states $C(S) = 0$. Suppose that C can be evaluated in unit time. The task is to devise an algorithm which minimizes the number of evaluations of C.

The idea of Grover's algorithm is to place the register in an equal superposition of all states, and then selectively invert the phase of the marked state, and then perform an inversion about average operation a number of times. The selective inversion of the marked state followed by the inversion about average steps has the effect of increasing the amplitude of the marked state by $O(1/\sqrt{NO(1)})$. Therefore after $O(\sqrt{N})$ operations the probability of measuring the marked state approaches 1 (Grover, 1996).

Grover's algorithm is as follows:

- Prepare a quantum register to be normalized and uniquely in the first state. Then place the register in an equal superposition of all states $\left(\frac{1}{\sqrt{N}},\frac{1}{\sqrt{N}},\ldots,\frac{1}{\sqrt{N}}\right)$ by applying the Walsh-Hadamard operator W. This means simply the state vector will be in an equal superposition of each state.
- Repeat $O(\sqrt{N})$ times the following two steps (the precise number of iterations is important, and discussed below):
  - o Let the system be in any state S. If C(S) = 1, rotate the phase by π radians, else leave system unaltered. It is worth noting that this operation has no classical analog. One cannot observe the state of the quantum register, doing so would collapse the superposition. The selective phase rotation gate would be a quantum mechanical operator which would rotate only the amplitude proportional to the marked state within the superposition.
  - o Apply the inversion about average operator A, whose matrix representation is: $A_{ij} = 2/N$ if i ≠ j and $A_{ij} = -1 + 2/N$ to the quantum register.
- Measure the quantum register. The measurement will yield the n bit label of the marked state $C(S_M)$ = 1 with probability at least 1/2 (Grover, 1996).

This Grover's algorithm flow chart is as shown in Figure 1.

3 Proposed System

The proposed system is concerned with the simulation of Grover's Algorithm using MATLAB. Quantum computing uses unitary operators acting on discrete state vectors. Matlab is a well-known (classical) matrix computing environment, which makes it well suited for simulating quantum algorithms.

Appendix A contains the Matlab commands to simulate Grover's algorithm using six qubits. The number of database elements is 2^6 = 64. The desired element is randomly generated from among the 64 elements using Matlab's *rand* function and it is the 53rd element. The quantum gates are defined using Matlab's *eye* function. The optimal number of iterations is determined by $\frac{\pi}{4}\sqrt{n}$ as proposed by Grover. Figure 2 shows the probability dynamics generated by the *plot* function in Matlab while Figure 3 shows the result distribution.

Grover demonstrated that quantum computers can perform database search faster than their classical counterparts. In this simple example of Grover's algorithm, a haystack function is used to represent the database. We are searching for a needle in the haystack, i.e. there is one element of the database that we require.

Grover's algorithm works by iteratively applying the inversion about the average operator to the current state. Each iteration amplifies the probability of a measurement collapsing the state to the correct needle value. Grover showed that performing a measurement after $\frac{\pi}{4}\sqrt{n}$ iterations is highly likely to give the correct result.

Appendix B contains the Matlab commands needed to find the needle in a haystack, i.e. to find a particular element among the elements of a database. The Walsh-Hadamard transformation, operator to rotate phase and inversion about average transformation were achieved through matrices of zeros and ones available as Matlab commands. At the end of the program, the output was set to be a movie-like display of the different stages of the iteration in a 3-dimensional plane comprising the axes Amplitude, States and Time. This was achieved by using Matlab's function *movie*. The last stage of the iteration process is shown by Figure 4.

3.1 Open Questions

There are several open questions that prop up from Grover's search algorithm proposal. Foremost among these is how many times exactly we should iterate step 2 of Grover's algorithm. Grover proves the existence of some m ∈ O ($\sqrt{N}$), such that after m iterations of step 2 of the algorithm, the probability of finding the register in the marked state is greater than 1/2. Since the amplitude of the desired state, and hence the probability of measuring the desired state, is not monotonic increasing after m iterations, it is not enough to know the existence of m, its value must be determined.

3.2 Searching for More Than One Item

Grover briefly mentions that his algorithm can work in a situation where there is more than one state, such that C (Si) = 1. In fact, this poses no difficulty whatsoever, and regardless of the number of marked states, its superior performance over classical algorithms is still retained. If there are t marked states, we can find one of the marked states in O ($\sqrt{N/t}$) time. This presumes that we know the number of marked elements in advance (Boyer et al, 1996).

Another interesting special case comes when t = N/4, in this case just as in the special case where N = 4, we will find a solution with unit probability after only one iteration, which is twice as fast as the expected running time for a classical algorithm, and exponentially faster than the worst case classical running time (Boyer et al, 1996).

3.3 Optimality of Grover's Algorithm

It is stated in Grover (1996) that his result was optimal, but it is not directly proved. Bennett *et al* (1996)

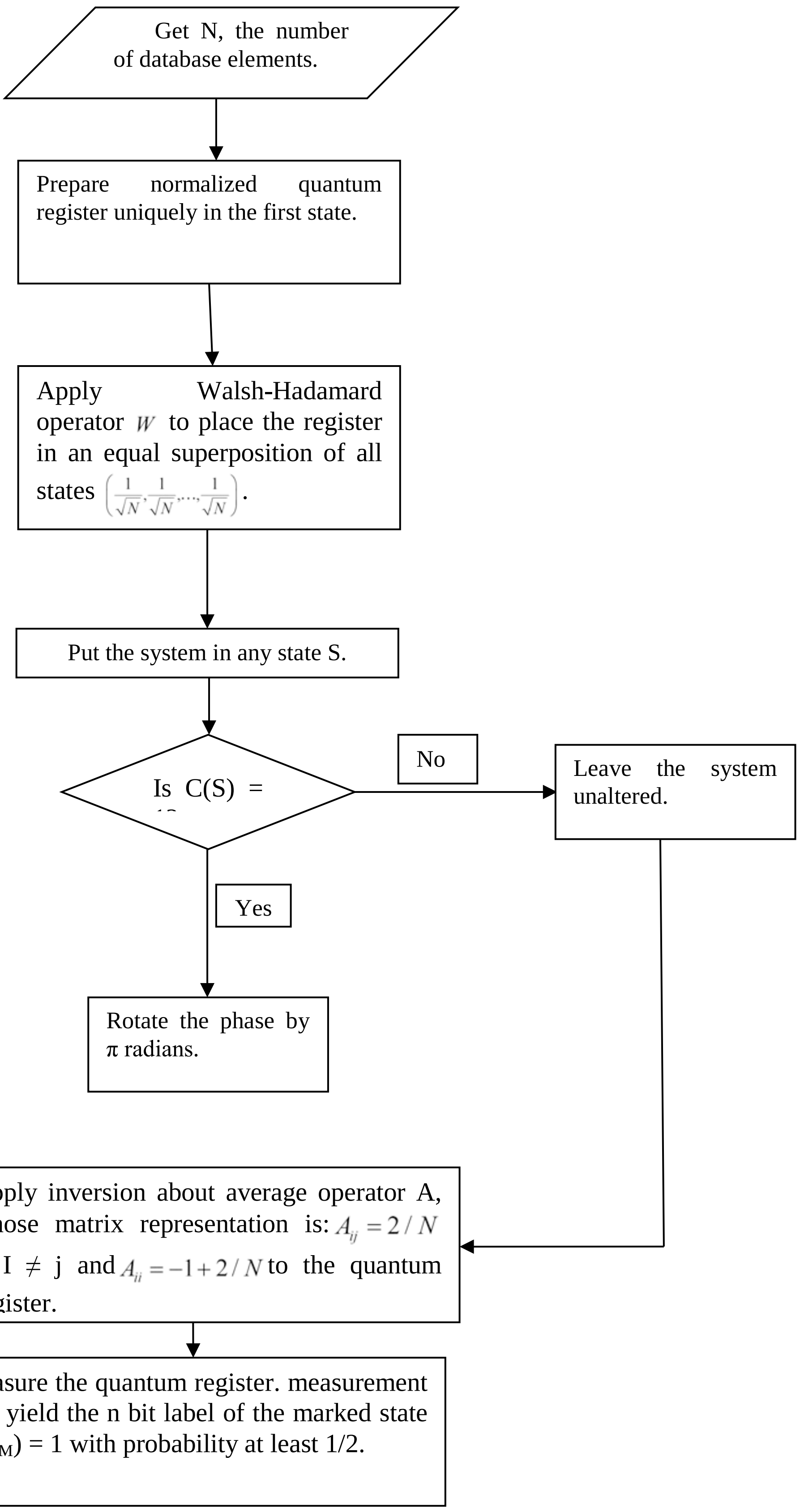

Get N, the number of database elements.
Prepare normalized quantum register uniquely in the first state.
Apply Walsh-Hadamard operator W to place the register in an equal superposition of all states $\left(\frac{1}{\sqrt{N}},\frac{1}{\sqrt{N}},\dots,\frac{1}{\sqrt{N}}\right)$.
Put the system in any state S.
Is C(S) =
No
Leave the system unaltered.
Yes
Rotate the phase by π radians.
Apply inversion about average operator A, whose matrix representation is: $A_{ij} = 2/N$ if I ≠ j and $A_{ii} = -1 + 2/N$ to the quantum register.
Measure the quantum register. measurement will yield the n bit label of the marked state $C(S_M) = 1$ with probability at least 1/2.

Figure 1: Grover’s Algorithm Flowchart.

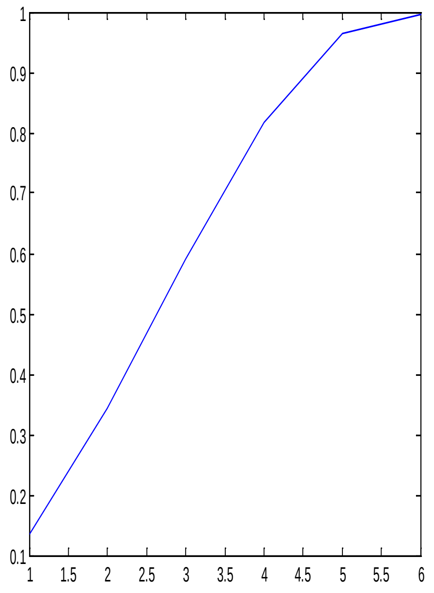


Figure 2: Probability Dynamics.

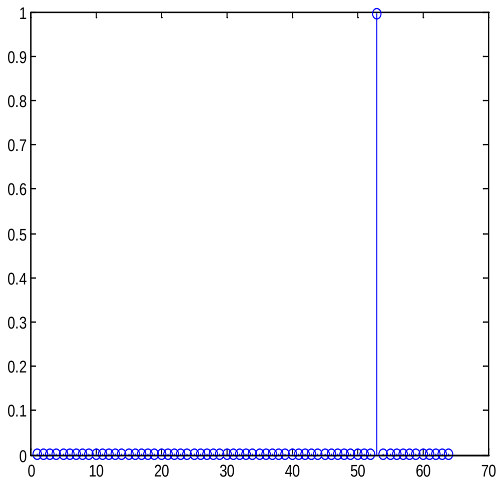


Figure 3: Result Distribution.

established that any quantum algorithm cannot identify a single marked element in fewer than Ω (√N). Grover's algorithm takes O(√n) iterations, and is thus asymptotically optimal. It has been shown since that any quantum algorithm would require at least π/4√N queries, which is precisely the number queries required by Grover's algorithm (Grover, 1999).

4 Conclusion

Intriguing breakthroughs occurred in the area of quantum computing in the late 1990s. Quantum computers under development use components of a chloroform molecule (a combination of chlorine and hydrogen atoms) and a variation of a medical procedure called magnetic resonance imaging (MRI) to compute at a molecular level. Scientists use a branch of physics called quantum mechanics, which describes the behavior of subatomic particles (particles that make up atoms), as the basis for quantum computing (Synder, 2008).

Quantum computers may one day be thousands to millions of times faster than current computers, because they take advantage of the laws that govern the behavior

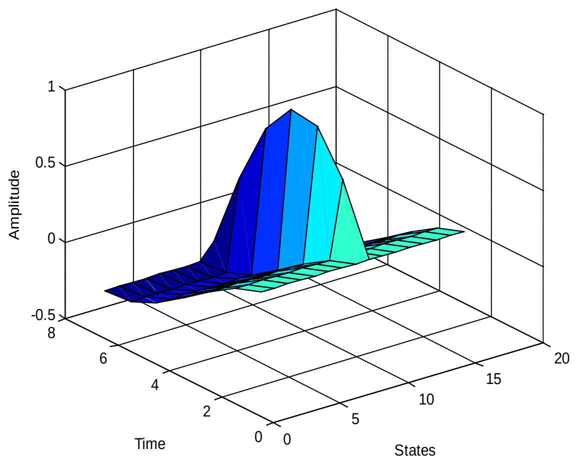


Figure 4: Amplitude-Time Graph for Simulated Grover's Algorithm.

of subatomic particles. These laws allow quantum computers to examine all possible answers to a query simultaneously. Future uses of quantum computers could include code breaking (cryptography) and large database queries. Theorists of chemistry, computer science, mathematics, and physics are now working to determine the possibilities and limitations of quantum computing.

Quantum computation allows for exponential speed up and storage in a quantum register via quantum parallelism. The more basis states represented within the register, the more speed up due to parallelism in the register, and the more improbable it is that a desired state can be measured. Grover's algorithm handles this problem by relying on transformations which cause the amplitude of the marked state to increase at the expense of the non marked states, in a number of ways this is analogous to interference of waves.

Grover's algorithm is unique among quantum algorithms in that it shows a useful calculation that a quantum computer can calculate faster than any classical computer possibly can. At the heart of Grover's algorithm are two unitary transformations, the first is a selective phase inversion, which makes the sign of the amplitude of the target negative. The second unitary transformation is an inversion about average operation. Initially we place the amplitude of all states at the same positive value, each phase switch and inversion about average increases the amplitude of the target state. The exact number of times we perform these transformations is roughly π/4√N for sufficiently large N. For a classical algorithm the best time bound is O (N).

5 Appendix A: Grover's Algorithm Simulation

This Matlab codes simulate Lov Grover's quantum database search algorithm by plotting the graph of the probability distribution of finding the marked state as the system undergoes Grover iteration.

```
%This script simulates the Quantum Mechanical Lov
    Grover's
%Search Alghorithm.
clear all;
%-----parameters-----------
nqubits=6;%number of q-bits
n=2^nqubits;%nnumber of elements in database
findmode=mod(round(n*rand+1),n);%desired element
%-----defining quantum gates
d=-eye(n)+2/n;%diffusion transform
oracle=eye(n);%oracle
oracle(findmode,findmode)=-1;
%--calculate the optimal number of iterations---
finish=round(pi/4*sqrt(n));
%--step(i)--initialization----
psistart=ones(n,1)/sqrt(n);
psi=psistart*exp(i*rand);
%step (ii)--algorithm body----
for steps=1:finish
steps
psi=d*oracle*psi;
probability(steps)=psi(findmode)*conj(psi(findmode));
end
%see the probability dynamics
plot(probability);
%see the result distribution
figure;
stem(psi.*conj(psi));
```

6 Appendix B: Searching for a Needle in a Haystack

This set of Matlab codes simulate the Lov Grover’s quantum database search algorithm by using the example of searching for a needle in a haystack.

```
s=[0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0]';
W=zeros(16,16); %Walsh-Hadamard transformation
for i=1:16
for j=1:16
W(i,j)=2^(-4/2)*(-1)^double(bitand(uint8(i-1),uint8(j-
    1)));
end
end
s=W*s;
R=zeros(16,16); %Operator to rotate phase
for i=1:16
if i==9
R(i,i)=-1;
else
R(i,i)=1
end
end
A=(2/16)*ones(16,16);   %inversion   about   average
    transformation
for i=1:16
A(i,i)=-1+2/16;
end
path=zeros(7,16);
path_i=1;
path(path_i,:)=s';
surf(path);
axis([0 20 0 8 -0.5 1]);
xlabel('States');ylabel('Time');zlabel('Amplitude');
F(path_i)=getframe;
path_i=2;
n=1;
while  n<(pi/2)*sqrt(16),  %needed  iteration  time  is
    (pi/4)*sqrt(N)
s=R*s;
s=A*s;
path(path_i,:)=s';
surf(path);
xlabel('States');ylabel('Time');zlabel('Amplitude');
F(path_i)=getframe;
path_i=path_i+1;
n=n+1;
end
movie(F,3,3);
```

7 References

[1] Bennett C.H, Bernstein E., Brassard G. and Vazirani U.

[2] (1996). Strengths and Weaknesses of Quantum Computing. *In the proceedings of SIAM Journal of Computing*.

[3] Boyer M., Brassard G., Hoyer P. and Tapp A., Tight Bounds on Quantum Searching. *In the Proceedings of PhysComp.* (lanl e-print quant-ph/9701001).

[4] Deutsch, D., and Jozsa, R.(1992). Rapid Solutions of Problems by Quantum Computation, *Proceedings of the Royal Soc. of London,* 439, 553.

[5] Eppstein D., Irani S., and Dillencthet M. (2000). ICS 260-Fall Class Notes 11: Turing Machines, Non-determinism, P and NP. Available at: http://www1.ics.uci.edu/~eppstein/260/notes/notes11.ps. Accessed on 20th April, 2010.

[6] Grover, L. K. (1996). A Fast Quantum Mechanical Algorithm for Database Search. In *Proceedings of 28th Annual ACM Symposium on the Theory of Computing,* New York, pp. 212.

[7] Grover L. (2000). Searching with Quantum Computers, lanl e-print quantph/0011118.

[8] Papadimitriou, C (1994). Computational Complexity, Addison-Onesley Publishing Company.

[9] Shor, P. W. (1994). Algorithms for Quantum Computation: Discrete Logs and Factoring. *In Proc. 35th Annual Symposium on Foundations of Computer Science.*

[10] Snyder T. (2008). Law in “Computer." Microsoft® Encarta® 2009 [DVD]. Redmond, WA: Microsoft Corporation.

[11] Williams C. and Clearwater S. (1998). Explorations in Quantum Computing, Springer-Verlag, New York, Inc.

On Similarity-based Approximate Reasoning in Interval-valued Fuzzy Environments

Zhi-Qiang Feng [a,b] and Cun-Gen Liu[a]
[a]School of Naval Architecture, Ocean and Civil Engineering, Shanghai Jiao Tong University, Shanghai 200240, China
E-mail: fzqstju@yahoo.cn (Z.Q. Feng)

[b]Jiangsu Naval Engineering Centre, Nantong 226010, China
E-mail: cgliu@sjtu.edu.cn (C.G. Liu)

Keywords: Approximate reasoning, interval-valued fuzzy sets, similarity measures, translation for rules

Received: March 30, 2011

This paper proposes a novel approach to similarity-based approximate reasoning in an interval-valued fuzzy environment. In a rule-based system, an 'if ... then ...' rule can be translated into an interval-valued fuzzy relation by suitable implication operations. The similarity grade between a case and the antecedent of a rule is computed and used to modify the relation. A consequent is derived from the well-known projection operation over the modified relation. The inference mechanism is appropriate because the techniques of the conventional Compositional Rule of Inference are incorporated into the existing similarity-based inference. Two examples of shipbuilding processing are utilized to illustrate and validate the effectiveness of the proposed schema.

Povzetek: Članek obravnava metode razmišljanje v mehki logiki, temelječe na podobnosti.

1 Introduction

As the theoretical foundation of fuzzy control, fuzzy inference has achieved successful applications in various fields. The basic Fuzzy Modus Ponens (FMP) often investigated by many researchers can be represented as:

Rule:	If X is A	then	Y is B
Case:	X is A'		
Conclusion:			Y is B'

Here X and Y are two linguistic variables, which can be also regarded as two different universes; A, A' and B, B' are fuzzy subsets of universes X and Y, respectively. Zadeh introduced the concept of Compositional Rule of Inference (CRI) [8]. By constructing a fuzzy relation R between A and B, we can derive the conclusion B' from the compositional operation of A' and R. Many fuzzy systems are based on Zadeh's compositional rule of inference [9]. In spite of their successes in various systems, researchers have pointed out certain drawbacks [4,5] in the mechanism, which motivates the introduction of Similarity-based Approximate Reasoning (SAR) mechanism as proposed in [4-7]. Compared with Zadeh's CRI, it does not require the construction of a fuzzy relation between input and output fuzzy data, and it is conceptually clearer than CRI.

According to the mechanism of SAR methodology, in rule-based system reasoning is based on the computation of similarity grade between the fact and the antecedent of a rule, and the inference result is obtained by directly modifying to the consequent part with the similarity measure. Thus, the inherent relation between the antecedent and the consequent is largely ignored. For the FMP problem, suppose that A is the antecedent part of 'If X is A then Y is B ', and A' is an input fact. In light of SAR, we first compute the similarity measure $S(A',A)$ of A' and A, then the result B' is deduced by a modification function f such that $B'(y)=f(S(A',A),B(y))$. Evidently, a same result B' will be concluded by SAR method when A and A' are interchanged. Thus, this result seems somewhat unconvincing because the inference is not always influenced by every change in the input case and the antecedent part.

Combining the conventional CRI and the existing SAR methods, in this paper we extend the works of [4,5] to develop a novel approach to approximate reasoning. First, since interval-valued fuzzy set is considered more flexible than general fuzzy set from the viewpoint of handling imprecise and fuzzy data, we deal with approximation inference within the framework of interval-valued fuzzy sets. Next, to interpret a conditional statement (rule) residing in a rule-base system, an interval-valued fuzzy relation between antecedent and consequent can be constructed by suitable implicators. Furthermore, based on a measure of similarity, the constructed relation is modified to yield a new relation called the induced relation, and the conclusion can be obtained by the well-known projection operation over the induced relation. In the end, we illustrate the effectiveness of the proposed scheme by an example of processing systems of shipbuilding.

The remainder of this article is organized as follows. Section 2 includes a brief introduction of some basic notions and state of the art on approximate reasoning techniques. Section 3 discusses the problems regarding similarity index and an approach to approximate reasoning. Two examples are provided in Section 4 to illustrate the effectiveness of the proposed methods. The final section contains the concluding remarks.

2 Preliminaries and State of the Art

Let X be a universe of discourse. In Fuzzy Sets (FSs) theory, each object $x \in X$ is assigned a single real value, called the grade of membership, between zero and one. In [1-3], Gorzalczany and Turksen proposed the notion of Interval-valued Fuzzy Sets (IVFSs), which allow using interval-based membership instead of using point-based membership as in FSs.

An interval-valued fuzzy set A on X is characterized by a pair of mappings $\underline{A}: X \to [0,1]$ and $\overline{A}: X \to [0,1]$ such that $0 \le \underline{A}(x) \le \overline{A}(x) \le 1$, where $\underline{A}(x)$ and $\overline{A}(x)$ denote a lower and an upper bounds of membership function of A, respectively. An interval-valued fuzzy set A of X can be denoted as $A = \left\{\left(x, \left[\underline{A}(x), \overline{A}(x)\right]\right) : x \in X\right\}$.

In other words, the membership degree of x with respect to A is bounded to a subinterval $\left[\underline{A}(x), \overline{A}(x)\right]$ of unit interval, which indicates the possible existence of a data value. All the interval-valued fuzzy sets of X is written as $IVFSs(X)$, and A is said to be normal if and only if there exists $x_0 \in X$ such that $\underline{A}(x_0) = \overline{A}(x_0) = 1$. For every $A \in IVFSs(X)$, the lower bound, the upper bound and the kernel function of A can be represented by

$$\underline{A} = \left\{\underline{A}(x) : x \in X\right\}$$

$$\overline{A} = \left\{\overline{A}(x) : x \in X\right\}$$

$$\kappa(A) = \left\{\underline{A}(x) + \overline{A}(x) : x \in X\right\}$$

respectively. Let $A, B \in IVFSs(X)$, the union, intersection and complement operations of interval-valued fuzzy sets are defined as follows:

$$A \cap B = \left\{\left(x, \left[\min\left\{\underline{A}(x), \underline{A}_B(x)\right\}, \min\left\{\overline{A}(x), \overline{B}(x)\right\}\right]\right) : x \in X\right\}$$

$$A \cup B = \left\{\left(x, \left[\max\left\{\underline{A}(x), \underline{A}_B(x)\right\}, \max\left\{\overline{A}(x), \overline{B}(x)\right\}\right]\right) : x \in X\right\}$$

$$A^c = \left\{\left(x, \left[1 - \overline{A}(x), 1 - \underline{A}(x)\right]\right) : x \in X\right\}$$

To improve the flexibility of fuzzy set in handling fuzzy information, Atanassov (1986) proposed the Intuitionistic Fuzzy Sets (IFSs) [14], and Gau et al. (1993) presented the Vague Sets (VSs) [15]. As the IVFSs, IFSs and VSs were proved actually isomorphic and equivalent [16-18], we will put them into a framework of IVFSs in this paper.

It is well-known that Fuzzy Set theory has been extensively applied to the field of approximate reasoning. So far there have been several approaches to approximate reasoning based on fuzzy set or interval-valued fuzzy set, in which the most influential methods are CRI and SAR algorithms. In [19], Li et al. addressed an implication operator based on IVFSs, which is suitable for CRI method. Cornelis et al. investigated a serial of implications in IVFSs and presented an extensional schema of CRI [20]. In [21], based on the CRI method, an extensional model derived from the Mizumoto's model is provided in an interval-valued fuzzy environment. Based on expansion principle, Feng et al. also presented several operators that are applicable to CRI [22].

Although the CRI algorithm had achieved notable success in various fields such as fuzzy control, expert system and decision-making support, some defects of this method were found in terms of inference mechanism, which leads to the yield of another important approach of approximate reasoning—Similarity-based Approximate Reasoning (SAR). For the FMP problem, based on the change of membership grade of the consequent part, Turksen et al. proposed two types of modification procedures—expansion type inference and reduction type inference [4], and B' may be computed by any one of the following form:

$$B'(y) = \min\left\{1, B(y)/S(A', A)\right\} \qquad \text{(Expansion form)}$$

$$B'(y) = S(A', A) \cdot B(y) \qquad \text{(Reduction form)}$$

where $S(A', A)(x)$ is the fuzzy similarity degree between a fact A' and the antecedent part A.

In [23], a new similarity measure of IVFSs was presented and inference result was obtained by Turksen's reduction form. Trough constructing a modified function based on a similarity measure of IVFSs, Tian et al. gave an approach to approximate reasoning [24]. Shi et al. addressed a bidirectional approximate reasoning scheme based on the distances of IVFSs [25], which can be actually regarded as an equivalent form of SAR method. Applying the SAR method, Guan et al. addressed a specific design scheme of fuzzy controller based on IVFSs [26].

In [10], through introducing a definition of fuzzy similarity measure, the authors provided an inference solution for FMP as follows:

$$B'(y) = \bigvee_{x \in X} \left(S(A', A)(x) \wedge B(y)\right)$$

In [27], the authors proposed the concept of similarity direction between two interval-valued fuzzy sets. Through calculating the similarity grade as well as the similarity direction of interval-valued fuzzy sets, the inference result B' is computed as

$$\underline{B}'(y) = \begin{cases} \underline{B}^s(y) & s \ge 0.5,\ d \ge 0 \\ \underline{B}^{1/s}(y) & s \ge 0.5,\ d < 0 \\ \underline{B}^s(y) & s < 0.5 \end{cases}, \quad \overline{B}'(y) = \begin{cases} \overline{B}^s(y) & s \ge 0.5,\ d \ge 0 \\ \overline{B}^{1/s}(y) & s \ge 0.5,\ d < 0 \\ \overline{B}^s(y) & s < 0.5 \end{cases}$$

where d is the evaluation function of similarity direction between A' and A, and s is the similarity grade of A' and A.

Meng presented a generalized model for fuzzy character spread reasoning [11], which was actually an approach of similarity-based weighted fuzzy inference applicable to multi-dimensional fuzzy reasoning. Through calculating the weighted similarity degree s_i between an input case and the ith rule, the result is expressed by

$$B'(y) = \mathbf{S} \circ B_i(y)$$

Here $\mathbf{S} = (s_1, s_2, \cdots, s_n)$ denotes a weighted similarity vector that holds

$s_i = \sum_{j=1}^{m} w_{ij} \cdot S\left(A'_j, A_{ij}\right)$

for $i = 1, 2, \cdots, n$. The operator '$\circ$' denotes a generalized compositional operation. For example, using $(\Sigma, \cdot)$ operation we then obtain

$B'(y) = \sum_{i=1}^{n} \left(s_i \cdot B_i(y)\right)$

In [28], a similarity-based approximate reasoning method was given based on IVFSs and fitness techniques. Using data fitness method, the similarity degree k between the interval-valued fuzzy sets is obtained by

$$k_1 = \frac{\ln \underline{A}(x) \cdot \ln \underline{A}'(x)}{\ln^2\left(\underline{A}(x)\right)},\ k_2 = \frac{\ln \overline{A}(x) \cdot \ln \overline{A}'(x)}{\ln^2\left(\overline{A}(x)\right)},\ k = \frac{k_1 + k_2}{2}.$$

And then the result is computed by

$\underline{B}'(y) = \underline{B}^k(y)$, $\overline{B}'(y) = \overline{B}^k(y)$.

3 Similarity-based Approximate Reasoning

3.1 Similarity measures of IVFSs

In [13], Zwick et al. surveyed several similarity measures of fuzzy sets and compared their performance in an experiment. In [12], Ke et al. presented a similarity function S to measure the degree of similarity based on fuzzy vectors. Let $E, F \in FSs(X)$, then the similarity grade $S(E,F)$ between E and F can be represented by

Definition1. [12]

$$S(E,F) = \frac{\sum_{x \in X} E(x) \cdot F(x)}{\max\left\{\sum_{x \in X} E^2(x), \sum_{x \in X} F^2(x)\right\}}$$

Here, the Sum-product operations represent the product of fuzzy vectors *E* and *F*, from which we may also derive another definition of similarity index, as follows.

Definition2.

$$S'(E,F) = \frac{\sup_{x \in X} \min\left\{E(x), F(x)\right\}}{\max\left\{\sup_{x \in X} E(x), \sup_{x \in X} F(x)\right\}}$$

Here, the operations Sum-product in Definition 1 are modified to Sup-min in Definition 2, respectively.

The measure proposed in Definition 2 is based on the computation of overall supremum and therefore, practically difficult to use.

Example1. Let $X = \{1,2,3,4,5\}$ be the universe of discourse. Consider the following fuzzy sets on X :

$E \triangleq \text{"small"} = 1/1 + 0.8/2 + 0.5/3 + 0.2/4$

$F \triangleq \text{"highly small"} = 1/1 + 0.41/3 + 0.06/3$

According to Definition 2, the calculation result implies that E is identical to F (i.e., $S'(E,F) = 1$), even if E is highly dissimilar to F by our intuitions. This is why we prefer the measure given by Definition 1.

To provide a definition for similarity measure $\tilde{S}(A,B)$ of two interval-valued fuzzy sets A and B, a number of factors must be considered. A primary consideration is that, whatever way we choose to define such an index, it should satisfy the following properties: for every $A, B, C \in IVFSs(X)$,

P1) $\tilde{S}(A,B) \in [0,1]$;

P2) $\tilde{S}(A,B) = \tilde{S}(B,A)$;

P3) $\tilde{S}(A,B) = 1$ if and only if $A = B$;

P4) If $\tilde{S}(A,B) = 0$, and A, B are not simultaneously empty, then $\min\left\{\underline{A}(x), \underline{B}(x)\right\} = \min\left\{\overline{A}(x), \overline{B}(x)\right\} = 0$ for all $u \in U$;

P5) If $A \subseteq B \subseteq C$ then $\tilde{S}(A,C) \le \min\left\{\tilde{S}(A,B), \tilde{S}(B,C)\right\}$.

P4) suggests that *A* and *B* are completely dissimilar only when $A \cap B = \varnothing$. If $A \cap B \neq \varnothing$, then they have some similarity when *A* and *B* have some membership degree in common.

Based on Definition 1, in the following, we develop an expression of similarity function $\tilde{S}$ to measure the degree of similarity between interval-valued fuzzy sets. Let $\underline{A}$, $\overline{A}$ and $\kappa(A)$ be subscript, superscript and kernel function of $A \in IVFSs(X)$.

Definition3. Let $A, B \in IVFSs(X)$. The degree of similarity $\tilde{S}(A,B)$ between A and B can be measured as follows:

$\alpha = S\left(\underline{A}, \underline{B}\right),\ \beta = S\left(\overline{A}, \overline{B}\right),\ \gamma = S\left(\kappa(A), \kappa(B)\right)$

$\tilde{S}(A,B) = (\alpha + \beta + 2\gamma)/4.$

It is easy to verify that the similarity measures given by Definition 3 satisfy axioms P1), P2), P3), P4) and P5). Thus, the similarity measures of the lower bound, the upper bound and the kernel values of two interval-valued fuzzy sets, are incorporated into such an index of IVFSs, where the weighted coefficients of which are given by $\omega_1 = 0.25$, $\omega_2 = 0.25$ and $\omega_3 = 0.5$, respectively.

3.2 Proposed schema for approximation inference

The conventional CRI does not consider the concept of similarity measure in deriving a consequence. The existing SAR methods modify directly the consequence part of a rule, based on a measure of similarity and therefore, the consequence becomes independent of the conditional statement. Here, we intend to integrate the above techniques for an adequate theory of similarity-based approximate reasoning.

According to CRI, a conditional statement (rule) 'If *A* then *B*' can be translated into an interval-valued fuzzy relation, denoted as $R(A,B)$. To construct the relation, some suitable operation operators are used. For example, the relation $R(A,B)$ constructed by the extensional *KD*-implicator can be represented as

$\underline{R}(A,B)(x,y) = \min\left\{1 - \overline{A}(x), \underline{B}(y)\right\}$

$\overline{R}(A,B)(x,y) = \min\left\{1 - \underline{A}(x), \overline{B}(y)\right\}$

Given a case input A', an interval-valued fuzzy relation between A' and B, denoted as $R(A',B)$, can be obtained by intersection operation of A' and $R(A,B)$. Thus, an inference result B' is computed by the well-known supremum projection operation on $R(A',B)$, i.e.,

$\underline{B}'(y) = \sup_{x \in X} \underline{R}(A',B)(x,y) = \sup_{x \in X} \min\left\{\underline{A}'(x), \underline{R}(A,B)(x,y)\right\}$

$\overline{B}'(y) = \sup_{x \in X} \overline{R}(A',B)(x,y) = \sup_{x \in X} \min\left\{\overline{A}'(x), \overline{R}(A,B)(x,y)\right\}$

Since the CRI method fails to incorporate the matching computation into the inference procedures, the accuracy of reasoning is not always satisfactory in some application occasions

The primary mechanism of SAR is to deduce result by modifying the consequent part of a rule with similarity measure [4,5]. Applying this principle, in a rule-based system we may first calculate the similarity grade $\tilde{S}(A',A)$ of the fact A' and the antecedent part A. And then, an interval-valued fuzzy relation $R(A',B)$ between A' and B, named as the induced relation, is obtained by modifying the relation $R(A,B)$ with similarity measure $\tilde{S}(A',A)$. Finally, the result B' can be deduced by the projection operation over the induced relation $R(A',B)$.

Given a conditional statement, the following cases should be taken into account to obtain an induced relation using the similarity measure.

*Case*1. If A' equals to A, then $R(A',B)$ equals to $R(A,B)$. This is to say we should not make any modification to $R(A,B)$ when $\tilde{S}(A',A)=1$.
*Case*2. If A' is completely dissimilar to A, then we can conclude nothing from the given conditional statement 'If A then B', i.e., B' is empty. Since
$\underline{B}'(y)=\sup_{x\in X}\underline{R}(A',B)(x,y)$, $\bar{B}'(y)=\sup_{x\in X}\bar{R}(A',B)(x,y)$
we then have $\underline{R}(A',B)(x,y)=\bar{R}(A',B)(x,y)=0$. i.e., $R(A',B)$ is empty when $\tilde{S}(A',A)=0$.
*Case*3. As $\tilde{S}(A',A)$ changes from 0 to 1, $R(A',B)$ changes from $\varnothing$ to $R(A,B)$. That means $R(A',B)$ is transformed from the most unknown state into a specific state.

From the cases mentioned-above, a quantitative relationship between the induced relation and similarity measure may be given as following:

Q1. If $S(A',A)=1$, then $\underline{R}(A',B)(x,y)=\underline{R}(A,B)(x,y)$, and $\bar{R}(A',B)(x,y)=\bar{R}(A,B)(x,y)$;
Q2. If $S(A',A)=0$, then $\underline{R}(A',B)(x,y)=\bar{R}(A',B)(x,y)=0$;
Q3. As $S(A',A)$ increase from 0 to 1, $\underline{R}(A',B)(x,y)$ and $\bar{R}(A',B)(x,y)$ increase from 0 to $\underline{R}(A,B)(x,y)$ and $\bar{R}(A,B)(x,y)$, respectively.

Let $\tilde{S}(A',A)=s,\underline{R}(A,B)(x,y)=\underline{r},\bar{R}(A,B)(x,y)=\bar{r},\underline{R}(A',B)(x,y)=\underline{r}',$ $\bar{R}(A',B)(x,y)=\bar{r}'$. By Q1 and Q2,

$$\underline{r}'=\begin{cases}0, & s=0\\ \underline{r}, & s=1\end{cases},\quad \bar{r}'=\begin{cases}0, & s=0\\ \bar{r}, & s=1\end{cases}$$

and by Q3, we get

$$\underline{r}'=\underline{r}'(s)=T(s,\underline{r}),\quad \bar{r}'=\bar{r}'(s)=T(s,\bar{r})$$

where T is a continuous t-norm. Thus, a modification schema for producing the induced relation $R(A',B)$, named as Q schema, may be represented by

$$\underline{R}(A',B)(x,y)=T\left(s,\underline{R}(A,B)(x,y)\right)$$
$$\bar{R}(A',B)(x,y)=T\left(s,\bar{R}(A,B)(x,y)\right)$$

Once the induced relation is derived from Q schema, the inference result B' is then obtained by the supremum projection, i.e.

$$\begin{aligned}\underline{B}'(y)&=\sup_{x\in X}\underline{R}(A',B)(x,y)=\sup_{x\in X}T\left(s,\underline{R}(A,B)(x,y)\right)\\ \bar{B}'(y)&=\sup_{x\in X}\bar{R}(A',B)(x,y)=\sup_{x\in X}T\left(s,\bar{R}(A,B)(x,y)\right)\end{aligned}\tag{1}$$

Apparently, in terms of inference mechanism, there exists a distinction between the conventional CRI and the proposed method. A logical interpretation for the CRI method is: from 'X is A' and (X,Y) is $R(A,B)$' infer 'Y is B''. For the proposed method, the inference mechanism can be interpreted as: from 'A' is similar to A and (X,Y) is $R(A,B)$' infer 'Y is B'', where the connective 'and' is associated with t-norm operation.

The proposed algorithm on performing similarity-based approximate reasoning is summarized as follows.

*Step*1. Translate a rule and compute $R(A,B)$ using some suitable operators (Translation);
*Step*2. Compute $\tilde{S}(A',A)$ using some suitable definition, possibly, Definition 3 (Matching);
*Step*3. Modify $R(A,B)$ with $\tilde{S}(A',A)$ to obtain the induce conditional relation $R(A',B)$ using a scheme Q (Modification);
*Step*4. Use supremum projection operation on $R(A',B)$ to obtain B' (Projection).

In Step1, to translate a rule 'If A and B' we should calculate an interval-valued fuzzy relation $R(A,B)$ between A and B. According to the Zadeh's CRI method, there are about eighteen operators applicable to construct $R(A,B)$, which can often be classified two main classes. The first class is called the extensional 'and' operators, such as the Mamdani operator, the Larson operators and the bounded product, etc. The second is called the extensional 'implication' operators, including the well-known S-implicator and the R-implicator. The following proposition will provide a clue on how to select the suitable operators for the construction of $R(A,B)$.

Proposition1. Suppose A is normal and does not completely cover the domain. Let $s=1$.
1) If φ is both increasing, then $B'=B$;
2) If φ is left decreasing and right increasing, then $B'=Y$.

Proof.
1) Since φ is both increasing, then

$$\underline{R}(A,B)(x,y)=\varphi\left(\underline{A}(x),\underline{B}(y)\right)$$
$$\bar{R}(A,B)(x,y)=\varphi\left(\bar{A}(x),\bar{B}(y)\right)$$

By Formula (1), we get

$$\underline{B}'(y)=\sup_{x\in X}T\left(s,\underline{R}(A,B)(x,y)\right)=\sup_{x\in X}T\left(s,\varphi\left(\underline{A}(x),\underline{B}(y)\right)\right)$$
$$=T\left(s,\sup_{x\in X}\varphi\left(\underline{A}(x),\underline{B}(y)\right)\right)=T\left(s,\varphi\left(\sup_{x\in X}\underline{A}(x),\underline{B}(y)\right)\right)$$

As A is normal, then

$$\underline{B}'(y)=T\left(s,\varphi\left(1,\underline{B}(y)\right)\right)=T\left(s,\underline{B}(y)\right).$$

Similarly, $\bar{B}'(y)=T\left(s,\bar{B}(y)\right)$, we then obtain $B'=B$ by $s=1$.

2) As φ is left decreasing and right increasing, then

$$\underline{R}(A,B)(x,y)=\varphi\left(\bar{A}(x),\underline{B}(y)\right)$$
$$\bar{R}(A,B)(x,y)=\varphi\left(\underline{A}(x),\bar{B}(y)\right)$$

By Formula (1), we get

$$\underline{B}'(y)=\sup_{x\in X}T\left(s,\underline{R}(A,B)(x,y)\right)=\sup_{x\in X}T\left(s,\varphi\left(\bar{A}(x),\underline{B}(y)\right)\right)$$
$$=T\left(s,\sup_{x\in X}\varphi\left(\bar{A}(x),\underline{B}(y)\right)\right)=T\left(s,\varphi\left(\inf_{x\in X}\bar{A}(x),\underline{B}(y)\right)\right)$$

Since A does not completely cover the domain, i.e. $\inf_{x\in X}\underline{A}(x)=\inf_{x\in X}\bar{A}(x)=0$, then

$$\underline{B}'(y)=T\left(s,\varphi\left(0,\underline{B}(y)\right)\right)=T(s,1)=s$$

Similarly, $\overline{B}'(y)=s$. By $s=1$, we then obtain $B'(y)=1$ for every $y\in Y$. Hence, $B'=Y$. □

Remark1. We can conclude from Proposition1 as following:

1) If an implication selected for constructing interval-valued fuzzy relation is both increasing, then the inference result satisfies reductive property when antecedent part is normal. Furthermore, since $\underline{B}'(y)=T(s,\underline{B}(y))$ and $\overline{B}'(y)=T(s,\overline{B}(y))$, we then obtain

$\underline{B}'(y)=s\cdot\underline{B}(y),\ \overline{B}'(y)=s\cdot\overline{B}(y)$

when t-norm takes the algebraic product. This is exactly identical to the Turksen's reduction form as mentioned in Section 2.

2) If an implication selected for constructing interval-valued fuzzy relation is hybrid monotonic, then the inference result equals to the whole set when antecedent part does not completely cover the domain. In this case, the result becomes the most unspecific case because $B'=Y$ means 'B' is anything' from the viewpoint of semantics.

Proposition2. Suppose A is normal, and φ is both increasing, then $B'\subseteq B$.

Proof. Since

$$\begin{aligned}\underline{B}'(y)&=\sup_{x\in X}T(s,\underline{R}(A,B)(x,y))=\sup_{x\in X}T\left(s,\varphi(\underline{A}(x),\underline{B}(y))\right)\\&=T\left(s,\sup_{x\in X}\varphi(\underline{A}(x),\underline{B}(y))\right)=T\left(s,\varphi\left(\sup_{x\in X}\underline{A}(x),\underline{B}(y)\right)\right)\\&\le\varphi\left(\sup_{x\in X}\underline{A}(x),\underline{B}(y)\right)=\varphi(1,\underline{B}(y))=\underline{B}(y)\end{aligned}$$

$$\begin{aligned}\overline{B}'(y)&=\sup_{x\in X}T(s,\overline{R}(A,B)(x,y))=\sup_{x\in X}T\left(s,\varphi(\overline{A}(x),\overline{B}(y))\right)\\&=T\left(s,\sup_{x\in X}\varphi(\overline{A}(x),\overline{B}(y))\right)=T\left(s,\varphi\left(\sup_{x\in X}\overline{A}(x),\overline{B}(y)\right)\right)\\&\le\varphi\left(\sup_{x\in X}\overline{A}(x),\overline{B}(y)\right)=\varphi(1,\overline{B}(y))=\overline{B}(y)\end{aligned}$$

for every $y\in Y$. Hence, we get $B'\subseteq B$. □

Example2. Suppose there is a conditional statement 'If A then B' such that

$A=\{(x_1,[0.7,0.8]),(x_2,[0.4,0.5]),(x_3,[0.2,0.3]),(x_4,0)\}$

$B=\{(y_1,1),(y_2,[0.5,0.6]),(y_3,0)\}$

Given a case input $A'=\{(x_1,1),(x_2,[0.7,0.8]),(x_3,[0.3,0.3]),(x_4,0)\}$, compute the inference result B' using the Turksen's reduction form and the proposed method, respectively.

From Definition 3, the similarity grade between the fact A' and the antecedent part A can be calculated as $\tilde{S}(A',A)=0.70$. Appling the Turksen's SAR method, we have

$B'_{\mathrm{I}}=\tilde{S}(A',A)\cdot B=\{(y_1,0.7),(y_2,[0.35,042]),(y_3,0)\}$.

According to the proposed algorithm in this paper, we first compute an interval-valued fuzzy relation of A and B using the Mamdani operator, i.e.

$$R(A,B)=\begin{bmatrix}[0.7,0.8] & [0.5,0.6] & 0\\ [0.4,0.5] & [0.4,0.5] & 0\\ [0.2,0.3] & [0.2,0.3] & 0\\ 0 & 0 & 0\end{bmatrix}$$

and then we have an induced relation $R(A',B)$ by the schema Q, i.e.

$$R(A',B)=\tilde{S}(A',A)\cdot R(A,B)=\begin{bmatrix}[0.49,0.56] & [0.35,0.42] & 0\\ [0.28,0.35] & [0.28,0.35] & 0\\ [0.14,0.21] & [0.14,0.21] & 0\\ 0 & 0 & 0\end{bmatrix}$$

where the t-norm is the algebraic product. Finally, we have the result B'_{II} by supremum projection, i.e.

$$\begin{aligned}B'_{\mathrm{II}}&=\sup\nolimits_{x\in X}R(A',B)(x,y)\\&=\{([0.49,0.56],y_1),([0.35,0.42],y_2),(0,y_3)\}\end{aligned}$$

Suppose that the fact A' and the antecedent part A are interchanged. An identical result B'_{I} can be deduced from the Turksen's SAR method, whereas a different result B'_{II} can be derived from the proposed method, i.e.

$B'_{\mathrm{II}}=\{(0.7,y_1),([0.35,042],y_2),(0,y_3)\}$.

Remark2. It can be seen from Example 2 that every change in the concept, as it appears in the conditional premise and in the fact, is incorporated into the induced interval-valued fuzzy relation. Hence, through the projection operation on the induced relation, the inference result is influenced by the changes in the fact and the antecedent of a rule.

4 Case Study

In shipbuilding technologies, some operational systems are so complex that it is very difficult for us to describe them with precise mathematical models. For these systems, the operational determinations can be acquired by means of the experiences of operators accumulated in practices. As the experiential knowledge is often fuzzy, which is suitable to describe by IVFSs, in the sequel we provide two technological cases modelled by IVFSs to illustrate applications of similarity-based method proposed in this article.

4.1 Layout of heating lines on plate

The processing practices indicate that length and density of heating lines exert a great impact on the forming of sheet metals. Let *X*, *Y* and *Z* be the linguistic variables representing curvature of a bending plate, the length of heating lines, and the space of heating lines, respectively. And the linguistic values are composed of several interval-valued fuzzy sets $A_i\in IVFSs(X)$, $B_j\in IVFSs(Y)$ and $C_k\in IVFSs(Z)$, as shown in Table 1, where $X=Y=Z=\{1,2,3,4,5\}$. And the operational rules derived from experiences of operators are summarized in Table 2

IVFSs \ Universe	1	2	……	5
A_1 (Large)	0	[0.1,0.2]	……	1
A_2 (Medium)	[0.2,0.3]	[0.7,0.8]	……	[0.1,0.2]
A_3 (Small)	1	[0.4,0.4]	……	0
B_1 (Long)	0	[0,0.1]	……	1
B_2 (Medium)	[0.2,0.4]	[0.5,0.5]	……	[0.2,0.3]
B_3 (Short)	1	[0.6,0.8]	……	0
C_1 (Large)	0	[0.1,0.1]	……	1
C_2 (Medium)	[0.1,0.3]	[0.4,0.5]	…..	[0.3,0.4]
C_3 (Small)	1	[0.7,0.8]	……	0

Table 1: Description for linguistic values based on IVFSs.

Rule No.	Antecedent part	Consequent part	
1	X is A_1	Y is B_1	Z is C_3
2	X is A_1	Y is B_2	Z is C_3
3	X is A_2	Y is B_2	Z is C_2
4	X is A_3	Y is B_3	Z is C_1
5	X is A_3	Y is B_3	Z is C_2

Table 2: Operational rules on the layout of heating lines.

Now let us conduct approximate reasoning using the scheme proposed in Section 3, and the inference procedures can be summarized as follows.

*Step*1. Translate the l^{th} rule r_l and compute $R\left(A^{(l)},B^{(l)}\times C^{(l)}\right)$ using the extensional Mamdani operator, for $l=1,2,\cdots,5$;
*Step*2. Compute the similarity grade s_l between input case A' and the antecedent $A^{(l)}$ by Definition 3;
*Step*3. Combine s_l with $R\left(A^{(l)},B^{(l)}\times C^{(l)}\right)$ to induce an interval-valued fuzzy relation $R\left(A',B^{(l)}\times C^{(l)}\right)$ using the modification schema Q;
*Step*4. Deduce a conclusion output $B'^{(l)}\times C'^{(l)}$ by the supremum projection over $R\left(A',B^{(l)}\times C^{(l)}\right)$;
*Step*5. Derive the general output $B'\times C'$ from union operation over $B'^{(l)}\times C'^{(l)}$;
*Step*6. Decouple the synthetic output $\kappa\left(B'\times C'\right)$ to obtain $\kappa\left(B'\right)$ and $\kappa\left(C'\right)$ via the projection operations on the universes Z and Y, respectively.
*Step*7. Obtain the determination values by defuzzification operations, using the maximum membership method.

In *Step*1, since $B^{(l)}\times C^{(l)}$ is a synthetic consequent part, which can be interpreted as a binary interval-valued fuzzy relation $R\left(B^{(l)},C^{(l)}\right)$ such that

$$\underline{R}\left(B^{(l)},C^{(l)}\right)(y,z)=\min\left(\underline{B}^{(l)}(y),\underline{C}^{(l)}(z)\right)$$
$$\overline{R}\left(B^{(l)},C^{(l)}\right)(y,z)=\min\left(\overline{B}^{(l)}(y),\overline{C}^{(l)}(z)\right)$$

Thus, a ternary interval-valued fuzzy relation $R\left(A^{(l)},B^{(l)}\times C^{(l)}\right)$ is constructed by

$$\underline{R}\left(A^{(l)},B^{(l)}\times C^{(l)}\right)(x,y,z)=\min\left(\underline{A}^{(l)}(x),\underline{R}\left(B^{(l)},C^{(l)}\right)(y,z)\right)$$
$$\overline{R}\left(A^{(l)},B^{(l)}\times C^{(l)}\right)(x,y,z)=\min\left(\overline{A}^{(l)}(x),\overline{R}\left(B^{(l)},C^{(l)}\right)(y,z)\right)$$

In *Step*3, according to the modification schema Q, we then obtain

$$\underline{R}\left(A',B^{(l)}\times C^{(l)}\right)(x,y,z)=s_l\cdot\underline{R}\left(A^{(l)},B^{(l)}\times C^{(l)}\right)(x,y,z)$$
$$\underline{R}\left(A',B^{(l)}\times C^{(l)}\right)(x,y,z)=s_l\cdot\underline{R}\left(A^{(l)},B^{(l)}\times C^{(l)}\right)(x,y,z)$$

where t-norm is the algebraic product. From *Step*4, we have

$$\begin{aligned}\left(\underline{B}'^{(l)}\times\underline{C}'^{(l)}\right)(y,z)&=s_l\cdot\min\left(\sup\nolimits_{x\in X}\underline{A}^{(l)}(x),\underline{R}\left(B^{(l)},C^{(l)}\right)(y,z)\right)\\\left(\overline{B}'^{(l)}\times\overline{C}'^{(l)}\right)(y,z)&=s_l\cdot\min\left(\sup\nolimits_{x\in X}\overline{A}^{(l)}(x),\overline{R}\left(B^{(l)},C^{(l)}\right)(y,z)\right)\end{aligned}\qquad(2)$$

If $A^{(l)}$ is normal, then Formula (2) can be simplified as

$$\begin{aligned}\left(\underline{B}'^{(l)}\times\underline{C}'^{(l)}\right)(y,z)&=s_l\cdot\underline{R}\left(B^{(l)},C^{(l)}\right)(y,z)\\\left(\overline{B}'^{(l)}\times\overline{C}'^{(l)}\right)(y,z)&=s_l\cdot\overline{R}\left(B^{(l)},C^{(l)}\right)(y,z)\end{aligned}\qquad(3)$$

Now let a case input

$$A'\triangleq\text{"more or less large"}$$
$$=0/1+[0.32,0.45]/2+[0.55,0.63]/3+[0.90,0.95]/4+1/5$$

For the 1st rule, according to Definition 3, the degree of similarity between A' and $A^{(1)}$ is calculated by

$$s_1=\left(S\left(\underline{A}',\underline{A}^{(1)}\right)+S\left(\overline{A}',\overline{A}^{(1)}\right)+2S\left(\kappa\left(A'\right),\kappa\left(A^{(1)}\right)\right)\right)\Big/4=0.87$$

Since $A^{(1)}$ is normal, we then obtain $B'^{(1)}\times C'^{(1)}$ by Formula (3), i.e.

$$B'^{(1)}\times C'^{(1)}=\begin{bmatrix}0&0&0&0&0\\ [0,0.09]&[0,0.08]&[0,0.09]&[0,0.09]&0\\ [0.35,0.35]&[0.35,0.35]&[0.26,0.35]&[0.09,0.17]&0\\ [0.61,0.78]&[0.61,0.70]&[0.26,0.35]&[0.09,0.17]&0\\ [0.87,0.87]&[0.61,0.70]&[0.26,0.35]&[0.09,0.17]&0\end{bmatrix}$$

Similarly, we can derive the outputs from other rules in Table 2, and the general output is calculated by

$$B'\times C'=\bigcup_{l=1}^{5}\left(B'^{(l)}\times C'^{(l)}\right)$$
$$=\begin{bmatrix}[0.17,0.35]&[0.17,0.35]&[0.17,0.35]&[0.11,0.21]&[0.13,0.21]\\ [0.44,0.44]&[0.44,0.44]&[0.26,0.35]&[0.26,0.26]&[0.15,0.21]\\ [0.87,0.87]&[0.61,0.70]&[0.51,0.51]&[0.26,0.32]&[0.15,0.21]\\ [0.61,0.78]&[0.61,0.70]&[0.32,0.36]&[0.26,0.32]&[0.15,0.21]\\ [0.87,0.87]&[0.61,0.70]&[0.26,0.35]&[0.11,0.17]&[0.15,0.21]\end{bmatrix}$$

From *Step*6, we have

$$\kappa\left(B'\right)(y)=\sup\nolimits_{z\in Z}\left(\kappa\left(B'\times C'\right)\right)(y,z)=\{0.52,0.88,1.74,1.39,1.74\}$$
$$\kappa\left(C'\right)(z)=\sup\nolimits_{y\in Y}\left(\kappa\left(B'\times C'\right)\right)(y,z)=\{1.74,1.31,0.61,0.28,0.36\}$$

According to *Step*7, since

$$\sup\nolimits_{y\in Y}\kappa\left(B'\right)(y)=\kappa\left(B'\right)(3)=\kappa\left(B'\right)(5)$$
$$\sup\nolimits_{z\in Z}\kappa\left(C'\right)(z)=\kappa\left(C'\right)(1)$$

Hence, $(y',z')=(3,1)$ or $(y',z')=(5,1)$ are selected as determination values, which can be interpreted as the conclusion is 'Y is long or medium' and 'Z is small' when the case input is 'more or less large'.

4.2 Welding deformation prediction on high-tensile steel structure

Welding experiment shows that, welding deformation of high-tensile steel structure not only relates to the leg size of weld seam, but also relates to the thickness of steel structure and welding current. Through a large amount of welding experiments, a rule-set including nine rules is summarized by the experienced welding operators, as shown in Table 3, where the linguistic values are the interval-valued fuzzy sets. For example, let Y be the thickness universe. The membership function of A_{21}, A_{22} and A_{23} is given as Table4, where linguistic values 'Thick', 'Medium' and 'Thin' are represented by A_{21}, A_{22} and A_{23}, respectively.

Rule No.	Antecedent part			Consequent part
1	*X* is A_{11}	*Y* is A_{21}	*Z* is A_{31}	*W* is D_3
2	*X* is A_{12}	*Y* is A_{21}	*Z* is A_{31}	*W* is D_3
……	… …	… …	… …	……
9	*X* is A_{13}	*Y* is A_{23}	*Z* is A_{33}	*W* is D_2

Table 3: Decision rule-set of welding deformation.

Linguistic value \ *Y*	1	2	3	4	5
Thick	[0,0]	[0,0]	[0.2,0.2]	[0.6,0.7]	[1,1]
Medium	[0,0]	[0.1,0.3]	[0.9,0.9]	[0.3,0.3]	[0,0]
Thin	[1,1]	[0.8,0.8]	[0.2,0.3]	[0,0]	[0,0]

Table 4: Membership function of linguistic value of thickness universe.

Once the knowledge model based on decision rules of welding deformation is obtained, we can set up a model of approximate reasoning, using the proposed method in this article. Through fuzzification of input data, similarity-based reasoning as well as defuzzification of fuzzy data, we then obtain the inference result of welding deformation. The running interface on welding deformation prediction system is shown as Figure 1. To examine the effectiveness of inference model, we arrange a welding experiment including ten test samples, as shown in Table 5.

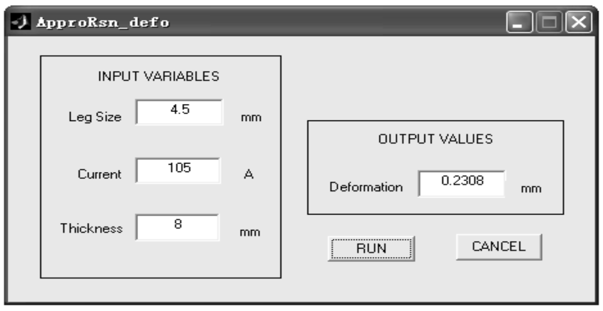


Figure 1: Running interface on welding deformation prediction system.

Test sample	Leg size /mm	Thickness /mm	Current /A	Deformation value /mm
p_1	5.5	6	130	0.53
p_2	4.5	8	95	0.26
……	……	……	……	……
p_{10}	4.5	5	115	0.46

Table 5: Test data on welding experiment.

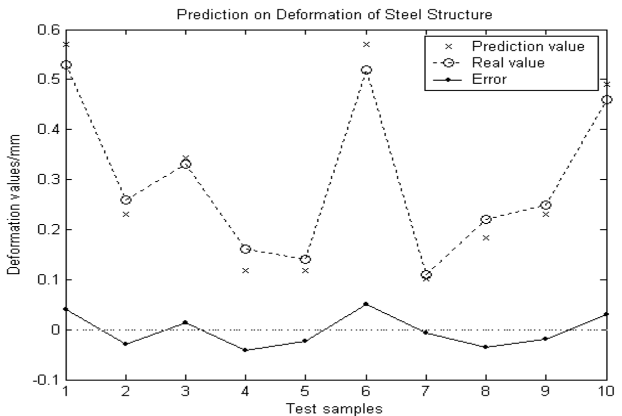


Figure 2: Comparison of prediction values and real values

From error curve of prediction values and real values shown in Figure 2, we can calculate the maximum error $E_{\max} = 0.050$, the mean error $E_m = 0.029$ and the standard error $E_{\text{std}} = 0.0317$, respectively. In terms of prediction accuracy of welding deformation, the result justifies the effectiveness of the proposed method.

5 Conclusion

In this paper, we investigate the similarity measures of interval-valued fuzzy sets. Based on the Turksen's reasoning model, we develop an approach to inference by combining the conventional CRI with similarity-based approximate reasoning. It is shown that a general representation for inference conclusion can be yielded by the procedures including translation, matching, modification, and projection. Besides, as the approximation inference is performed under the framework of IVFSs, the proposed method seems more flexible than is done with the general FSs. In the end, we utilize two examples concerning shipbuilding techniques to illustrate and validate the proposed schema.

For the nonlinear and coupling shipbuilding technology, the conventional modelling schema mainly contains physical simulation and Finite Element Analysis (FEA). As to the former, it not only costs a large amount of lab funds but also limits to experimental conditions. As for FEA, although the precise of this method is relatively high, the program is so time-consuming that it can hardly be applied to manufacture practices. Compared with the traditional methods, modelling based on fuzzy data can fully take advantage of the experiences of experts in their field, and accuracy of inference result is also adequate to meet the needs of technological practices. Therefore, we have lots of research opportunities for future applications of similarity-based inference to complex shipbuilding systems, such as layout of heating lines, welding parameters design and welding deformation prediction, etc.

Acknowledgement

This research was partially supported by the national key research project 'Knowledge Based Ship Design and Hyper Integrated Platform'.

References

[1] M.B. Gorzalczany, 'A method of inference in approximate reasoning based on interval-valued

fuzzy sets', *Fuzzy Sets and Systems*, vol. 21, pp. 1-17, 1987.

[2] M.B. Gorzalczany, An interval-valued fuzzy inference method－some basic properties, *Fuzzy Sets and Systems*, vol. 31, pp. 243-251, 1989.

[3] I.B. Turksen, 'Interval valued fuzzy sets based on normal forms', *Fuzzy Sets and Systems*, vol. 20, pp. 191-210, 1986.

[4] I.B. Tursken, Z. Zhao, 'An approximate analogical reasoning based on similarity measures', *IEEE Transactions on Systems Man, and Cybernetics*, vol.18, pp. 1049-1056, 1988.

[5] I.B. Tursken, Z. Zhao, 'An approximate analogical reasoning scheme based on similarity measures and interval valued fuzzy sets', *Fuzzy Sets and Systems*, vol. 34, pp. 323-346, 1990.

[6] S.M. Chen, 'A new approach to handling fuzzy decision making problems', *IEEE Transaction SMC*, vol. 18, pp. 1012-1016, 1988.

[7] D.S. Yeung, E.C.C. Tsang, 'A comparative study on similarity based fuzzy reasoning methods', *IEEE Transaction SMC Part B: Cybernetic*, vol. 27, pp. 216-227, 1997.

[8] L.A. Zadeh, 'Fuzzy logic and approximate reasoning', *Syntheses*, vol. 30, pp. 407-428, 1975.

[9] L.A. Zadeh, 'A theory of approximate reasoning', *Machine Intelligence*, vol. 9, pp. 149-194, 1979.

[10] D.G. Wang et al., 'A fuzzy similarity inference method for fuzzy reasoning', *Computers and Mathematics with Applications*, vol. 56, pp. 2445-2454, 2008.

[11] Y.P. Meng, 'Similarity-based information weighted fuzzy reasoning', *Journal of Beijing Normal University*, vol. 44, pp. 21-24, 2008.

[12] J.S. Ke, G.T. Her, 'A fuzzy information retrieval system model', in: Proc. *1983 National Computer Symposium, Taiwan, Republic of China*, pp. 147-155, 1983.

[13] R. Zwick, E. Carlstein, and D.R. Budescu, 'Measures of similarity among fuzzy concepts: a comparative analysis', *International Journal of Approximate Reasoning*, vol. 1, pp. 221-242, 1987.

[14] K.T. Atanassov, 'Intuitionistic fuzzy sets', *Fuzzy Sets and Systems*, vol. 20, pp. 87-96, 1986.

[15] W.L. Gau, D.J. Buehrer, 'Vague sets', *IEEE Transactions on Systems, Man and Cybernetics*, vol. 23, pp. 610-614, 1993.

[16] Atnassov K, Gargov G., 'Interval valued intuitionistic fuzzy sets', *Fuzzy Sets and Systems*, vol.31, pp.343-349, 1989.

[17] Bustince H, Burillo P, 'Vague sets are intuitionistic fuzzy sets', *Fuzzy Sets and Systems*, vol. 79, pp. 403-405, 1996.

[18] Deschrijver G, Kerre E E., 'On the relationship between some extensions of fuzzy set theory', *Fuzzy Sets and Systems*, vol. 133, pp. 227-235, 2003.

[19] F. Li et al., 'Implication operator based on vague sets', *J. Huazhong Univ. Sci. & Tech.* (Nature Science Edition), vol. 29, pp. 53-55, 2001.

[20] C. Cornelis, G. Deschrijver et al., 'Implication in intuitionistic fuzzy and interval-valued fuzzy set theory: construction, classification, application', *International Journal of Approximate Reasoning*, vol.35, pp. 55-95, 2004.

[21] X.L. X, Y.J. Lei, 'Approximate reasoning method based on some intuitionistic fuzzy relations', *Computer Engineering*, vol. 34, pp. 4-6, 2008.

[22] Z.Q Feng et al., 'Vague propositional logic and approximation inference based on linear transformation', *Journal of Harbin Engineering Universe*, vol. 9, pp. 1222-1227, 2010.

[23] F. Li et al., 'An approximate reasoning approach based on the measures of similarity between vague sets', *J. Huazhong Univ. Sci. & Tech.* (Nature Science Edition), vol. 32, pp. 44-46, 2004.

[24] Y. Tian et al., 'An approximate reasoning approach based on the measures of similarity between intuitionistic fuzzy sets', *Journal of Air Force Engineering University* (Nature Science Edition), vol. 8, pp. 81-83, 2007.

[25] Y.Q. Shi et al., 'Bidirectional approximate reasoning based on distances of vague set', *Journal of Chinese Computer System*, vol. 28, pp. 661-665, 2007.

[26] X.Z. Guan et al., 'Controller design based on the measures of similarity reasoning using vague sets', *Control Engineering of China*, vol. 13, pp. 15-24, 2006.

[27] T.J. Wang et al., 'Bidirectional approximate reasoning based on weighted similarity measures of vague set', *Mini-Micro Systems*, vol. 25, pp. 211-215, 2004.

[28] X.M. Li et al., 'Approximate reasoning method based on intuitionistic fuzzy', *Journal of Air Force Engineering University* (Nature Science Edition), vol. 9, pp. 82-85, 2008.

Autonomous Push-down Automaton Built on DNA[*]

Tadeusz Krasiński, Sebastian Sakowski
Faculty of Mathematics and Computer Science, University of Łódź
Banacha 22, 90-238 Łódź, Poland
E-mail: krasinsk@uni.lodz.pl, sakowski@math.uni.lodz.pl

Tomasz Popławski
Department of Molecular Genetics, University of Łódź
Pomorska 141/143, 90-236 Łódź, Poland
E-mail: tplas@biol.uni.lodz.pl

Keywords: push-down automaton, DNA computing

Received: July 19, 2012

In the paper we introduce a biomolecular implementation of the push-down automaton (one of the theoretical models of computing devices with unbounded memory) using DNA molecules. The idea of this improved implementation was inspired by Cavaliere et al. (2005).

Povzetek: Predstavljen je avtonomni avtomat na osnovi DNK po vzoru Cavaliereja.

1 Introduction

In the paper written by Cavaliere, Janoska, Yogev, Piran, Keinan, Seeman [4] the authors propose a theoretical model (*i.e.* not tested in laboratory) of implementation of the push-down automaton built on DNA. The idea was inspired by two papers: the first one by Rothemund [7] who proposed a simulation of the Turing machine - the basic theoretical model of computation - and the second one by Benenson, Paz-Elizur, Adar, Keinan, Livneh, Shapiro [1] who proposed a simulation of the finite automaton – the simplest model of computation. The above three implementations represent all the basic theoretical models of computers in the Chomsky hierarchy. But all these simulations have weak points in different places.

The Rothemund model is not autonomous. A person must interfere in the process to obtain the required sequences of actions through many restriction enzymes. This is likely a reason why nobody tested it experimentally.

Next, Benenson *et al*. [1] model is autonomous, programmable and was tested in laboratory but it represents the simplest computational model - a finite automaton (in fact it was only 2-states 2-symbols finite automata). The next propositions along the same idea (Soreni *et al.* [10], Unold *et al.* [11], Krasiński and Sakowski [6]) essentially did not improve the situation.

The last model, Cavaliere *et al.* [4] is more powerful (a push-down automaton), autonomous, programmable (although the action of it was illustrated only on one simple example) but the problem lies in obtaining the right sequence of ligations of transition molecules to the input and to the stack (represented by the same circular DNA). The authors themselves indicate this problem “It is first important to know which side is ligated first, since there is degeneracy in the stack side … and therefore different transition molecules may be ligated at that end at any stage” and propose two ways to reduce (not eliminate) the problem. Moreover, another problem in their model is that it is not clear biochemically whether the used enzyme *Psr*I could not accidentally cut transition molecules of the first kind (which add the symbol Z to the stack) before ligating it to the input and to the stack.

In this paper we suggest an improvement of the last model of push-down automata to avoid these problems. However, it is a theoretical model not tested yet in laboratory. We propose a new shape of transition molecules and another kind of restriction enzymes, which cut only when the ligation of a transition molecule to the circular molecule of the input will be accomplished on both sides.

2 Push-down Automaton

In this section we recall shortly the definition of the push-down automaton (PDA). More information can be found in any textbook (Hopcroft and Ullman [5]; Sipser [8]).

A push-down automaton is a finite automaton (nondeterministic) which has a stack, a kind of simple memory in which it can store information in a last-in-first-out fashion.

[*] This project is supported by the National Science Centre of Poland (NCN). Grant number: DEC-2011/01/B/NZ2/03022.

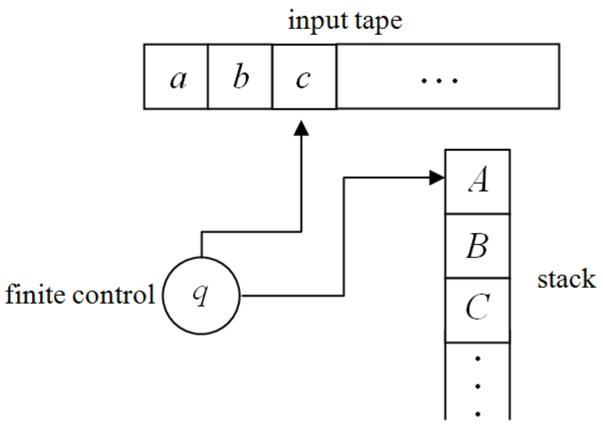


Figure 1: A scheme of the PDA.

In each step the machine, based on its current state (q), the input symbol which is being currently read (c) and the top symbol on the stack (A) performs a move according to a transition rule (from a list of transition rules associated to a given PDA): pops the top symbol from the stack, pushes a symbol (or a sequence of symbols) onto the stack, moves its read head one cell to the right and enters a new state. We also allow ε - transitions in which a PDA can pop and push without reading the next input symbol. The PDA is nondeterministic, so there may be several transitions that are possible in a given configuration. We will denote transition rules in the following way

$$(q, c, A) \to (q', A')$$

where: q' - a new state, A' - a new symbol or a sequence of symbols (may be an empty sequence) which replaces A on the top of the stack.

There are two (equivalent) alternative definitions of acceptance of an input word w: by empty stack and by final state. Since in the presented implementation we use the second one we will recall only that one. A PDA accepts an input word w if it enters a final state (from a distinguished subset of all states) after scanning the entire word w, starting from the initial configuration with w on the input tape and with the special initial symbol $\perp$ on the stack.

The class of languages accepted by PDA is the class of context-free languages which strictly includes the class of regular languages (accepted by finite state automata) and is strictly contained in the class of recursive enumerable languages (accepted by Turing machines).

We will illustrate the above definition by giving an example of PDA which adds integers. It will be our main example in the implementation.

Example 1. A PDA accepting the language

$$ADD = \{a^n b^m c^{n+m} : n, m \in N\}$$

has four states: q_0 - initial state, q_1, q_2, q_3 - final state. The PDA has the following transitions:

1. $(q_0, a, \perp) \to (q_0, A\perp)$
2. $(q_0, a, A) \to (q_0, AA)$
3. $(q_0, b, A) \to (q_1, AA)$
4. $(q_1, b, A) \to (q_1, AA)$
5. $(q_1, c, A) \to (q_2, \varepsilon)$
6. $(q_2, c, A) \to (q_2, \varepsilon)$
7. $(q_2, \varepsilon, \perp) \to (q_3, \varepsilon)$

A sequence of configurations (state, remaining input word, stack) of this PDA on the input word $aabccc \in ADD$ is as follows.

$$(q_0, aabccc, \perp) \overset{1}{\to} (q_0, abccc, A\perp) \overset{2}{\to} (q_0, bccc, AA\perp) \overset{3}{\to} (q_1, ccc, AAA\perp) \overset{5}{\to} (q_2, cc, AA\perp) \overset{6}{\to} (q_2, c, A\perp) \overset{6}{\to} (q_2, \varepsilon, \perp) \overset{7}{\to} (q_3, \varepsilon, \varepsilon)$$ - acceptation,

and on the input word $abc \notin ADD$ is as follows.

$$(q_0, abc, \perp) \overset{1}{\to} (q_0, bc, A\perp) \overset{3}{\to} (q_1, c, AA\perp) \overset{5}{\to} (q_2, \varepsilon, A\perp)$$ - stop the action.

3 The Structure of DNA

DNA (deoxyribonucleic acid) is the storage medium for genetic information in all living things. It is a single-stranded (ss) or a double-stranded (ds) chain made of four nucleotides A, C, T, G. In a dsDNA two ssDNA (with the inverse orientations) are linked by hydrogen bonds in such a way that A can only pair together with T and C with G. To manipulate DNA we take various enzymes from a variety of organisms for catenating, splitting, cutting and copying DNA. In our consideration we will use restriction enzymes (restrictases) which recognize fixed sites in a DNA and cut it, leaving sticky ends on both sides of the cutting place. For instance the restrictase *Fok*I cuts in the following way (Fig. 2).

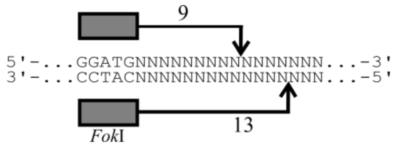

9
5'-...GGATGNNNNNNNNNNNNNNNNN...-3'
3'-...CCTACNNNNNNNNNNNNNNNNN...-5'
13
FokI

Figure 2: The action of the enzymes *Fok*I.

4 The implementation of PDA

The implementation of a PDA is similar to that of Cavaliere *et al.* [4] with changes which eliminate their obstacles. The main idea of the implementation is as follows.

The basic elements of a PDA i.e. the input tape and the stack are represented in the same circular dsDNA molecule of which one end represents the stack and the second one the input word (Fig. 3).

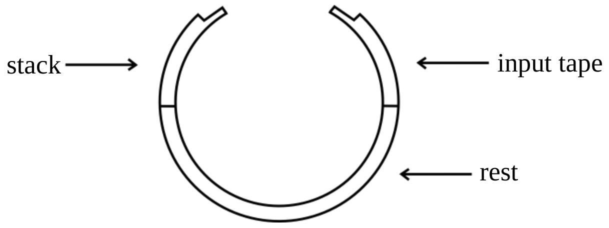


Figure 3: The basic elements of implementation of a PDA.

The sticky end of the stack represents the top symbol on the stack and the sticky end of the input tape represents the first symbol of the input word (to be read) and simultaneously the state of the PDA.

The transition rules of a PDA are suitable DNA molecules which hybridize to both ends of the circular DNA representing this PDA (Fig. 4).

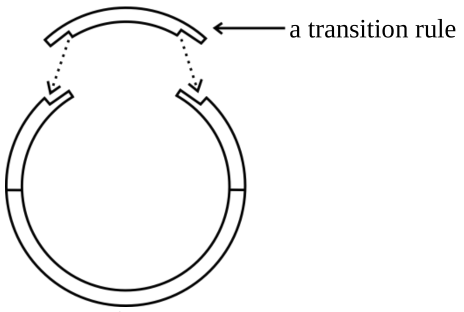


Figure 4: Process of hybridizing a transition rule to both ends of DNA.

After ligation, appropriate restriction enzymes cut this circular molecule. Their actions cause changes in the stack and in the input word according to the move which is represented by this transition molecule. A new idea is that the action of restriction enzymes will take place only when the transition molecule ligate to both ends of the circular molecule. It happens because the chosen restriction enzyme (*Bgl*I) has two separated recognition sites (Fig. 5), which appear both only when a transition molecules ligates to both ends of the circular molecule. After the cut additional molecules and restriction enzymes make adequate changes in the stack and in the input word. Then the next transition rule may act. When a sequence of such transitions leads to reading out the input word and the last sticky end would represent the final state of the PDA, then a long additional DNA molecule ligates to the molecule. It can be detected in the solution by gel electrophoresis. The word is accepted.

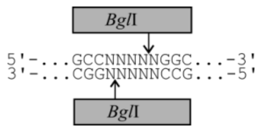


Figure 5: The action of the enzyme *Bgl*I.

5 The Practical Implementation

The idea of the practical implementation will be illustrated on the PDA given in Example 1 i.e. on a PDA performing the addition of integers. The graph of it is represented in Fig. 6.

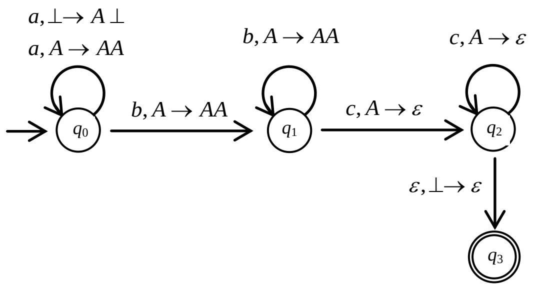


Figure 6: The graph of a PDA which adds integers.

It has seven moves. Each of them is represented by a transition molecule, additional molecules and suitable restriction enzymes (see Appendix 1).

The action of the enzyme *Bgl*I is presented in Fig. 5. The remaining enzymes act as follows (Fig. 7).

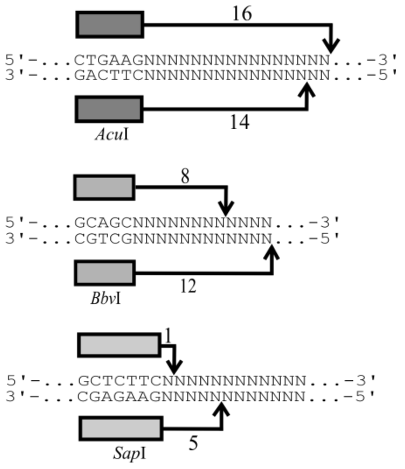

16
5'-...CTGAAGNNNNNNNNNNNNNNNN...-3'
3'-...GACTTCNNNNNNNNNNNNNNNN...-5'
14
AcuI
8
5'-...GCAGCNNNNNNNNNNNN...-3'
3'-...CGTCGNNNNNNNNNNNN...-5'
12
BbvI
1
5'-...GCTCTTCNNNNNNNNNNNN...-3'
3'-...CGAGAAGNNNNNNNNNNNN...-5'
5
SapI

Figure 7: The action of the enzymes *Acu*I, *Bbv*I, *Sap*I.

The sticky end of an input word represents both a symbol and a state of the PDA according to the rules (Fig. 8).

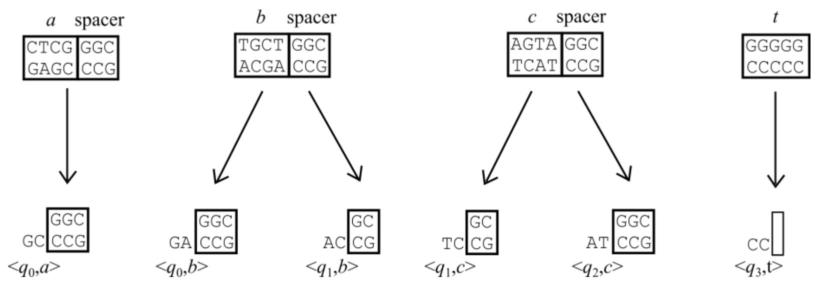


Figure 8: DNA codes of the symbols and pairs <state, symbol>.

The symbols $\{A, \perp\}$ on the stack and their representations on the top of the stack are presented in Fig. 9.

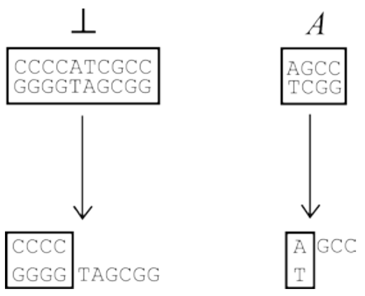


Figure 9: The representations of the stack symbols.

The representation of the considered PDA with the input word *aabccc* in the initial state q_0 and the symbol $\perp$ on the stack is shown in Fig. 10.

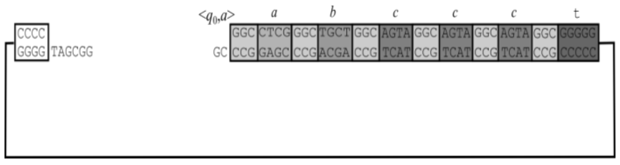

<q0,a>
a
b
c
c
c
t
CCCC
GGGG TAGCGG
GGC CTCG GGC TGCT GGC AGTA GGC AGTA GGC AGTA GGC GGGGG
GC CCG GAGC CCG ACGA CCG TCAT CCG TCAT CCG TCAT CCG CCCCC

Figure 10: The PDA with the input word *aabccc*.

The action of the PDA will be illustrated on two moves, the first of which pushes a symbol on the stack (Appendix 2) and the second one of which pops the symbol from the stack (Appendix 3).

The main idea of the first move $(q_0, a, \perp) \rightarrow (q_0, A\perp)$ which pushes the symbol A on the stack is to use the restriction enzyme *Bgl*I, which cuts the DNA strand only when the transition molecule merges the stack and the input tape. It is caused by the fact that the enzyme *Bgl*I has two separated recognition sites 5'...GCC(5nt)GGC...3' which appear when the transition molecule ligates to the stack and to the input word. An important fact is to use spacers GGC between symbols of the input word. After the cut the second restriction enzyme *Acu*I together with an additional molecule make a change in the input word.

A second move $(q_1, c, A) \rightarrow (q_2, \varepsilon)$ which pops the symbol from the stack acts by using also the restriction enzyme *Bgl*I (Appendix 3). After cutting with the enzyme *Bgl*I we have to remove actual symbols from the input word and from the stack. The operation of removing from the input word is the same as in the first move (using the restriction enzyme *Acu*I).

Since we could not find a commercial enzyme which cuts a DNA molecule in a long distance from the recognition site and leaves a 3-nt sticky end we have to apply two restriction enzymes (*Bbv*I and *Sap*I)

The remaining moves act similarly. The whole process on the word *aabccc* is presented in Appendix 4.

6 Conclusions

We have presented a new method to implement a push-down automaton based on DNA molecules and restriction enzymes. It is an improved version of the idea presented in [4]. Other attempts (not fully matured and functioning) are in [9], [13], [14]. A new idea is to use a restriction enzyme which has two separate recognition sites. It allows to cut DNA molecules representing elements of a PDA after ligating of transition molecules to both sides of circular DNA. It avoids problems that appeared in Cavaliere *et al.* [4]. This will enable us in the future to construct more powerful automata than PDA, which provides the possibility to solve more complicated problems. Actually we implemented our theoretical model of finite automata (more powerful than the one presented in Benenson *et al.* [1] in a laboratory in the cooperation with a research group from the Department of Molecular Genetics of the Łódź University. This attempt of a laboratory implementation of our research groups is described by Błasiak, Krasiński, Sakowski, Popławski [3]. We tested in the laboratory simultaneous action of two restriction enzymes *Acu*I and *Bbv*I which is a crucial step in the experiment presented in this paper. The next step could be laboratory implementation of the PDA presented in this article.

The circular molecule dsDNA used in our model opens a new possibility to insert and apply our automaton to the bacteria cell. Such a type of DNA molecules are plasmids - heritable DNA molecules that are transmissible between bacterial cells and bacterial genomes. Bacteria controls DNA replication process via origin replication elements. These genetic elements are built with blocks of repeated sequences and replication is initiated when special proteins (e. g. DnaA in *E. coli*) binds to series of repeats. Regulations of bacterial genome and plasmid propagation is possible with use of our automaton by controlling the number of repeat motifs presented in origin (by inserting to the stack or removing from the stack). In a similar way it is possible to control in bacteria not only DNA replication but also transcription of some bacterial genes. Transcription starts when RNA polymerase binds to special genetic elements called promoter. The bacterial promoter is built with some genetic elements essential for efficient initiation of transcription (e.g. -10 and -30 blocks), thus we can switch on and off gene transcription by inserting or deleting some sequence blocks within promoter or even changing the distance between them. This method of DNA replication or transcription control with the use of an automaton has one major advantage in comparison of natural scheme of control – it allows to make some logical calculations before cell take the final decision.

Acknowledgement

We thank prof. Jacek Hejduk for an improvement of the text in the paper.

References

[1] Benenson, Y., Paz-Elizur, T., Adar, R., Keinan, E., Livneh, Z., Shapiro, E. (2001). Programmable and autonomous computing machine made of biomolecules. Nature 414, 430-434.

[2] Benenson, Y., Adar, R., Paz-Elizur, T., Livneh, Z., Shapiro, E. (2003). DNA molecule provides a computing machine with both data and fuel. PNAS 100, 2191-2196.

[3] Błasiak, J., Krasiński, T., Popławski, T., Sakowski S. (2011). More powerful biomolecular automaton. ArXiv:109.5893v1 [Cs.ET]. Cornel University Library.

[4] Cavaliere, M., Jonoska, N., Yogev, S., Piran, R., Keinan, E., Seeman, N. (2005). Biomolecular implementation of computing devices with unbounded memory. Lecture Notes in Computer Science 3384, 35-49.

[5] Hopcroft, J., Ullman, J. (1979). Introduction to Automata Theory, Languages and Computation. Addison-Wesley.

[6] Krasiński, T., Sakowski S. (2008). Extended Shapiro Finite State Automaton. Foundations of Computing and Decision Science 33, 241-256.

[7] Rothemund, P. (1995). A DNA and restriction enzyme implementation of Turing machines. In DNA Based Computers, American Mathematical Society, Providence, RI , 75-119.

[8] Sipser., M. (2006). Introduction to the Theory of Computation. Thomson Course Technology.

[9] Shi, X., Xin, L., Zhang, Z., Xu, J. (2005). Improve Capability of DNA Automaton: DNA Automaton with Three Internal States and Tape Head Move in Two Directions. Lecture Notes in Computer Science 3645, 71-79.

[10] Soreni, M., Yogev, S., Kossoy, E., Shoham Y., Keinan, E. (2005). Parallel biomolecular computation on surfaces with advanced finite automata. Journal of the American Chemical Society 127, 3935-3943.

[11] Unold O., Troć M., Dobosz T., Trusiewicz A. (2004): Extended molecular computing model. WSEAS Transactions on Biology and Biomedicine 1, 15-19.

[12] Yin, P., Turberfield, A., Reif, J. (2004). Designs of Autonomous Unidirectional Walking DNA Devices. Tenth International Meeting on DNA Computing (DNA10), Milano, Italy. Lecture Notes in Computer Science 3384, 7-10.

[13] Zhang, Z., Xu, J., Liu, J., Pan, L. (2006). Programmable pushdown store base on DNA computing. Lecture Notes in Computer Science 4115, 286-293.

[14] Zhang, Z., Jie, L., Xiao-Long, S. (2008). Biomolecular Pushdown Automaton Based on the DNA Computing. Chinese Journal of Computers 31, 2168-2172.

Appendix 1

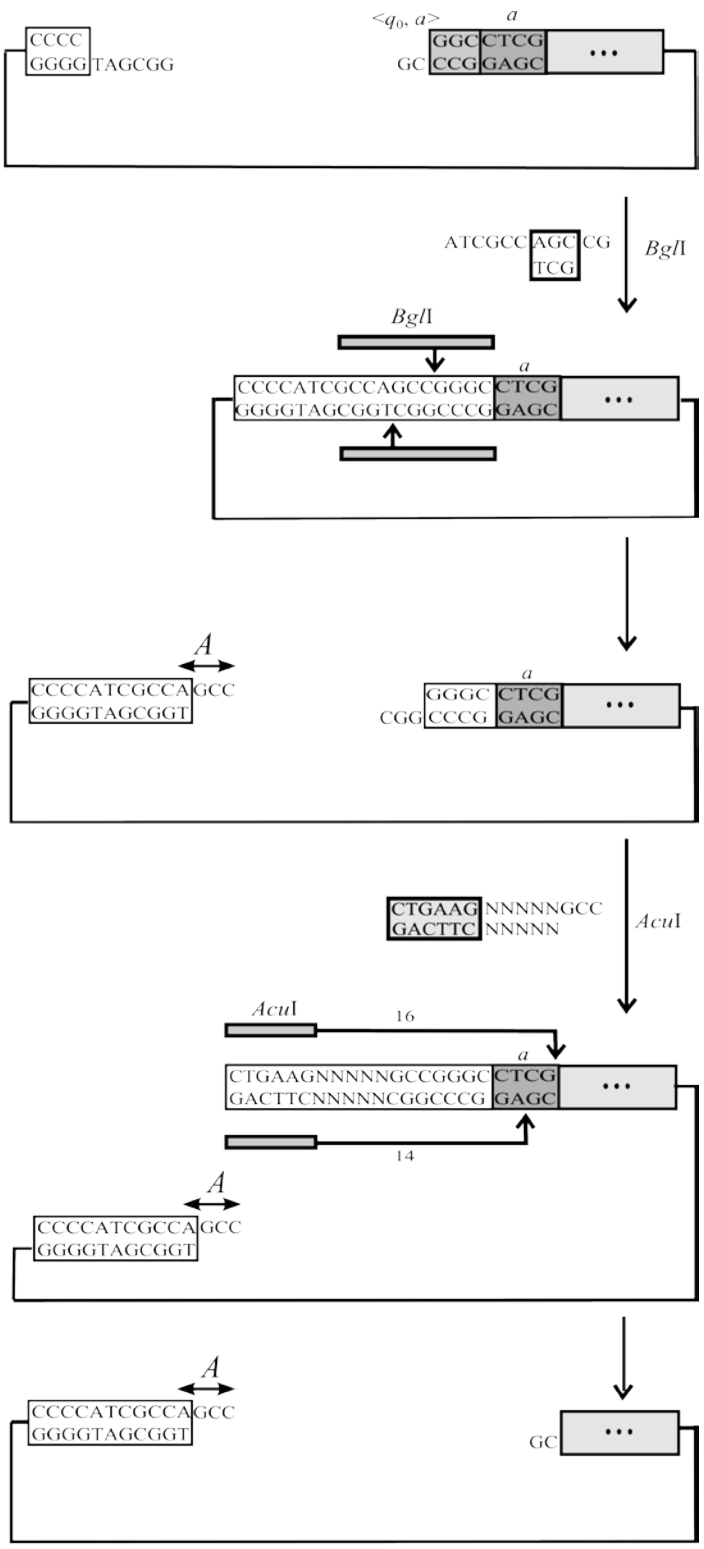
The transition rules and their molecular representations.

Table 1

Transition rule	Transition molecule	Additional molecule	Restriction enzymes
$(q_0, a, \perp) \to (q_0, A\perp)$	ATCGCC AGC CG TCG	CTGAAG NNNNN GCC GACTTC NNNNN	*Bgl*I *Acu*I
$(q_0, a, A) \to (q_0, AA)$	AGC CG CGG TCG	CTGAAG NNNNN GCC GACTTC NNNNN	*Bgl*I *Acu*I
$(q_0, b, A) \to (q_1, AA)$	AGC CT CGG TCG	CTGAAG NNNN CGG GACTTC NNNN	*Bgl*I *Acu*I
$(q_1, b, A) \to (q_1, AA)$	AGCC TG CGG TCGG	CTGAAG NNNN GCC GACTTC NNNN	*Bgl*I *Acu*I
$(q_1, c, A) \to (q_2, \varepsilon)$	AAAG AG CGG TTTC	CTGAAG NNNNNAAG GACTTC NNNNN CGTCG N GCAGC N CTT GCTCTTC ACCG CGAGAAG	*Bgl*I *Acu*I *Bbv*I *Sap*I
$(q_2, c, A) \to (q_2, \varepsilon)$	GGG TA CGG CCC	CTGAAG NNNNNGGT GACTTC NNNNN CGTCG N GCAGC NACC GCTCTTC ACCG CGAGAAG	*Bgl*I *Acu*I *Bbv*I *Sap*I
$(q_2, \varepsilon, \perp) \to (q_3, \varepsilon)$	GGG TA CGG CCC	CTGAAG NNNNGGT GACTTC NNNN CGTCG N GCAGC NACC 300 bp GCCA 500 bp GG	*Bgl*I *Acu*I *Bbv*I

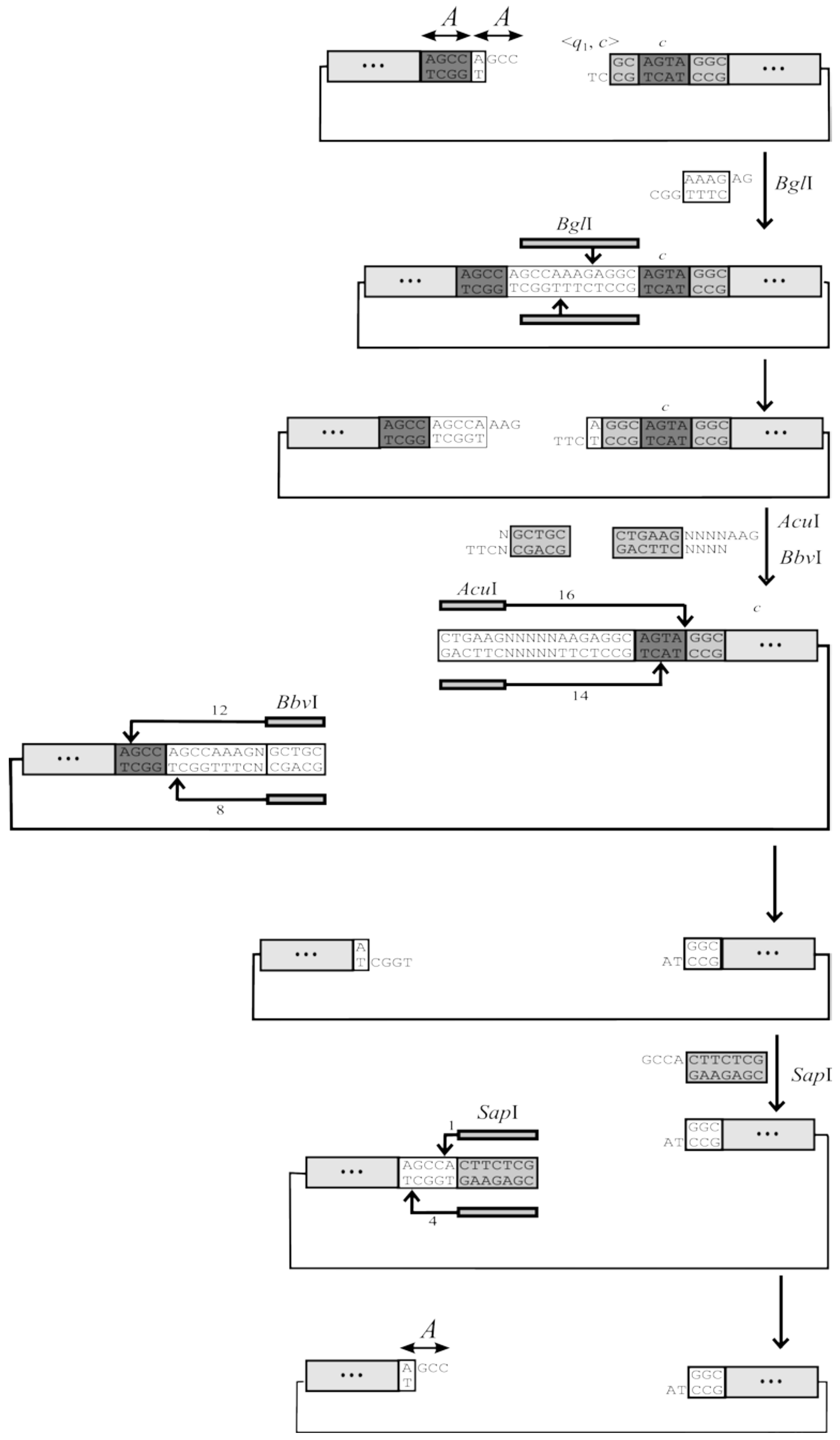
Appendix 2

The push a symbol on the stack $(q_0, a, \perp) \rightarrow (q_0, A\perp)$.



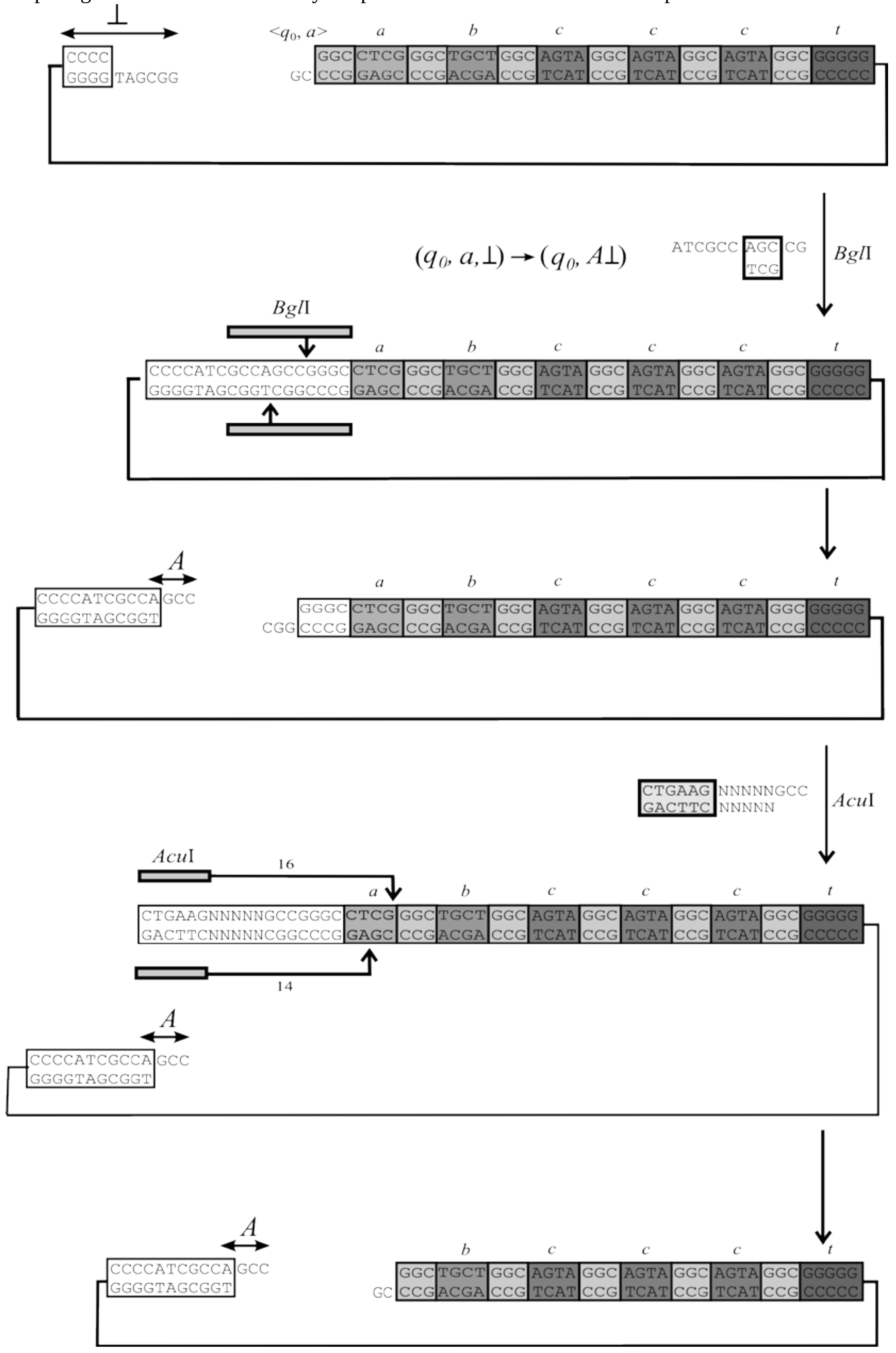
Appendix 3

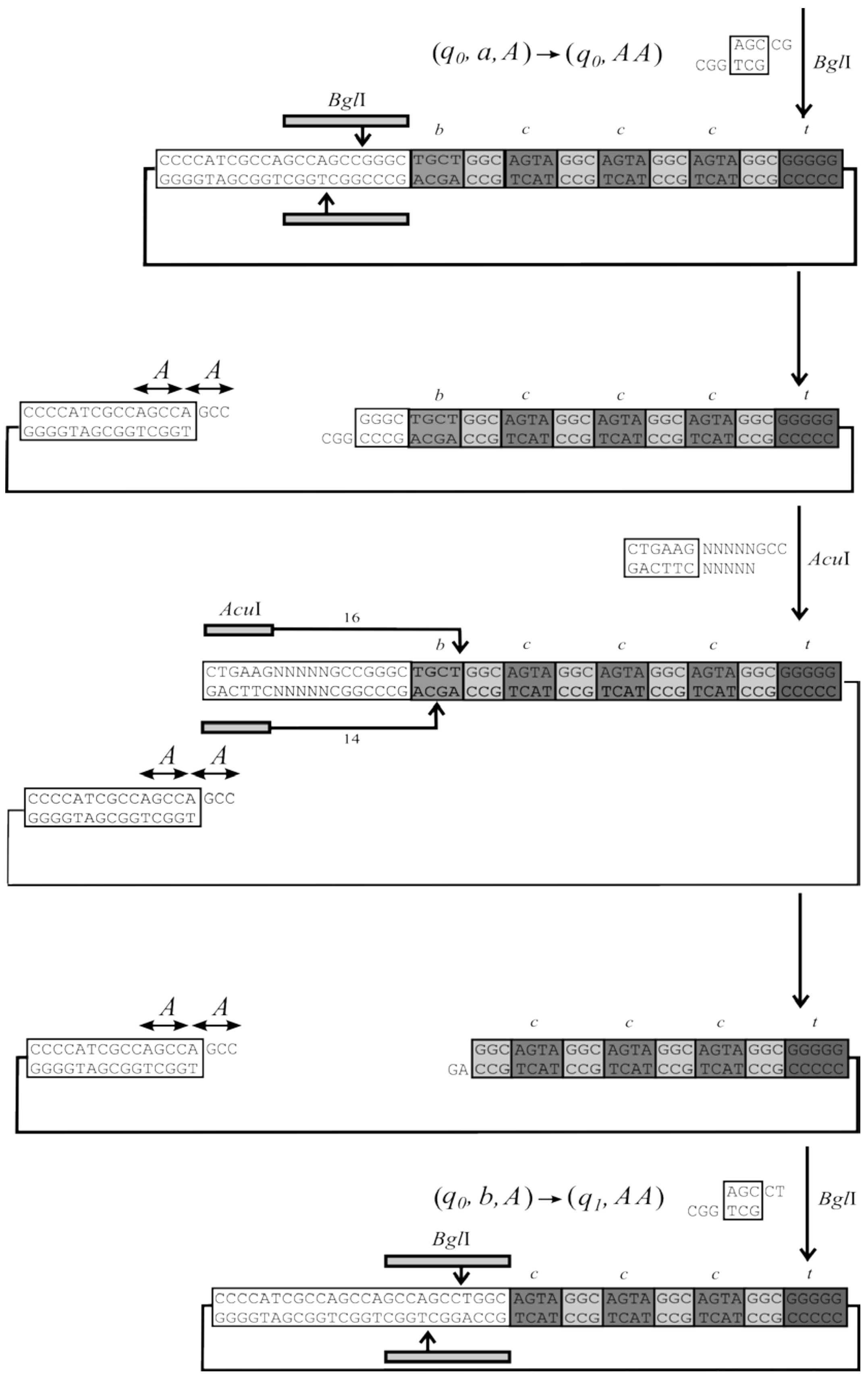
The pop a symbol from the stack $(q_1, c, A) \rightarrow (q_2, \varepsilon)$.

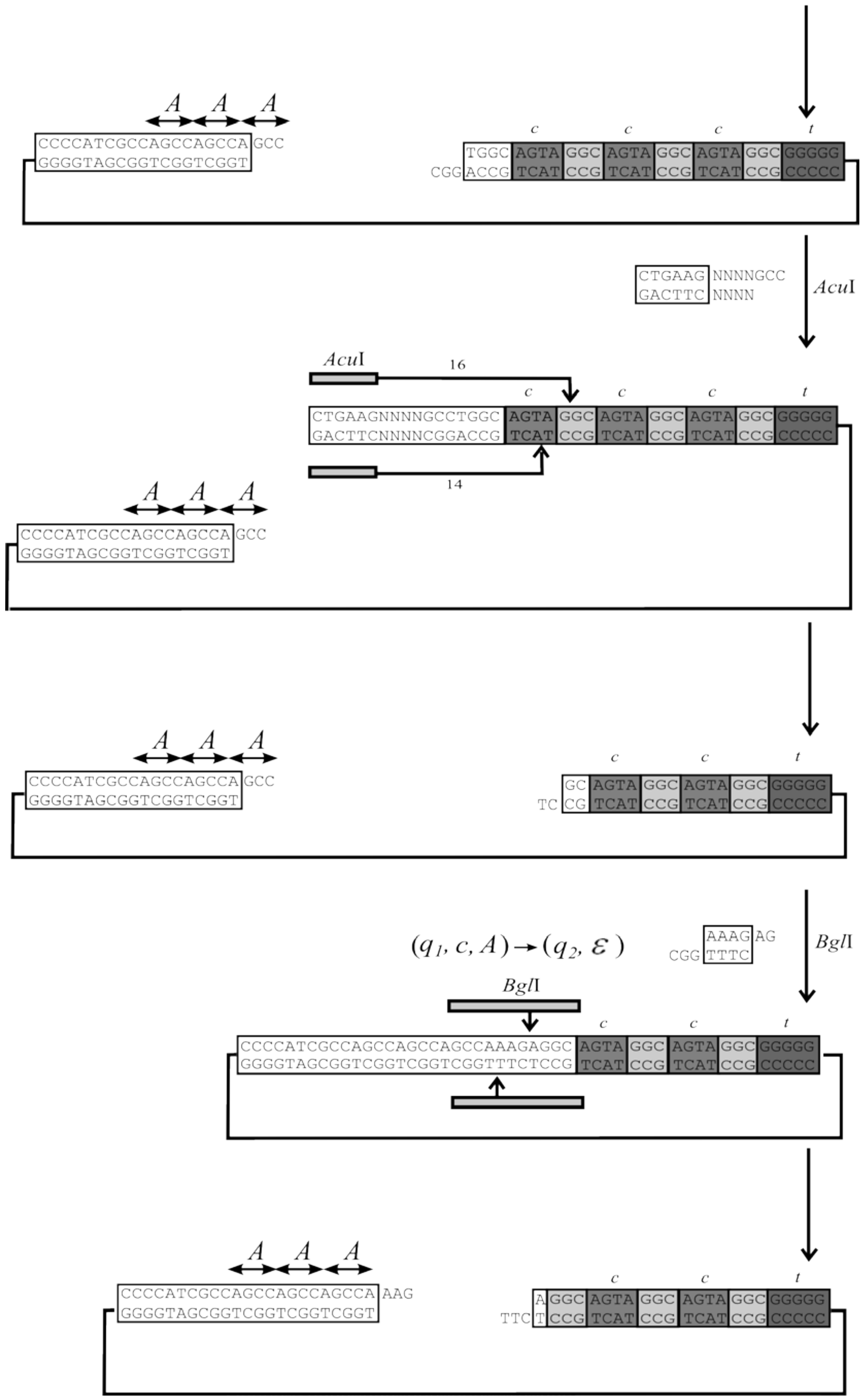


Appendix 4

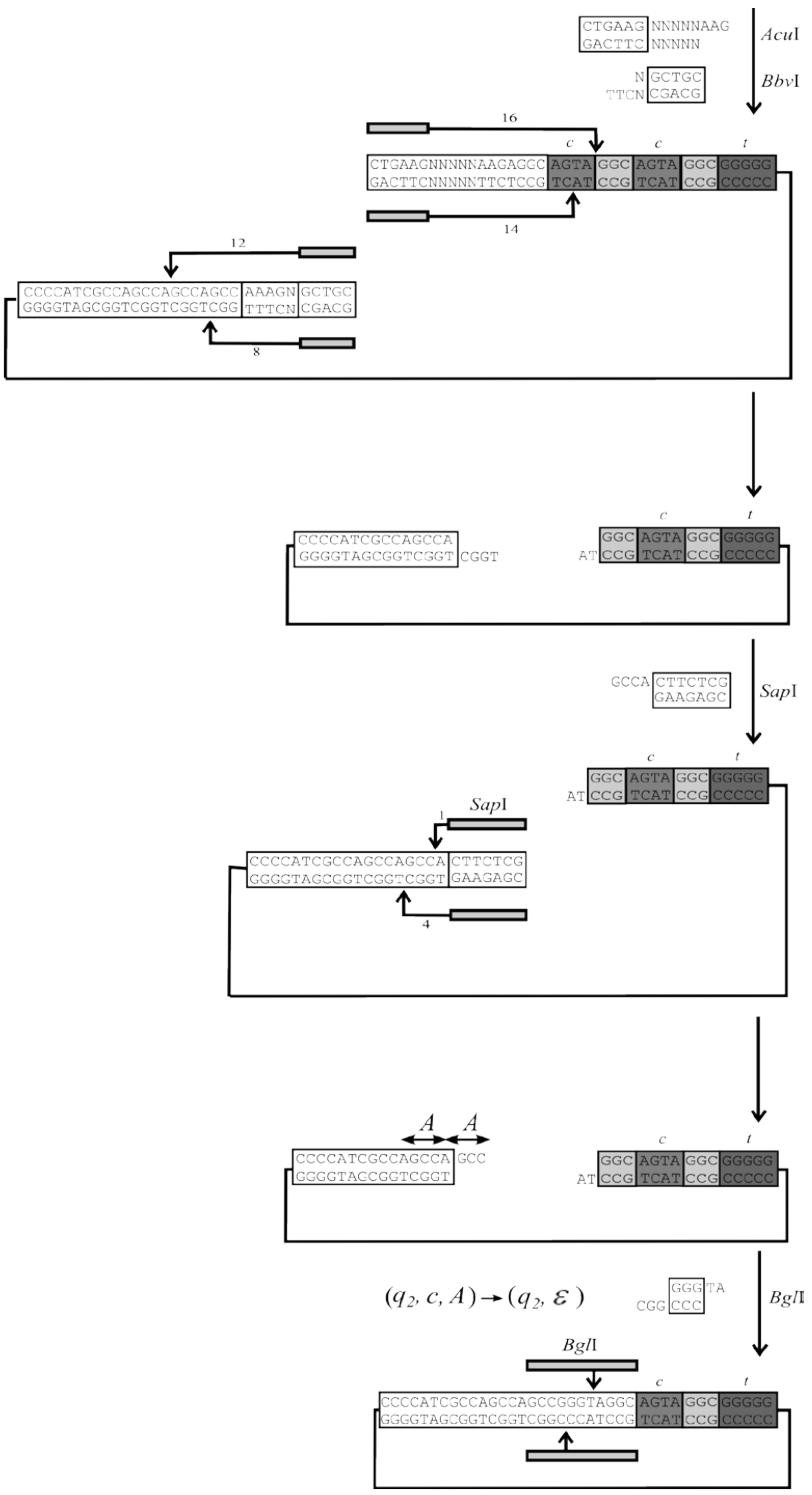
Process of computing of the word *w*=*aabccc* by the push-down automaton from Example 1.

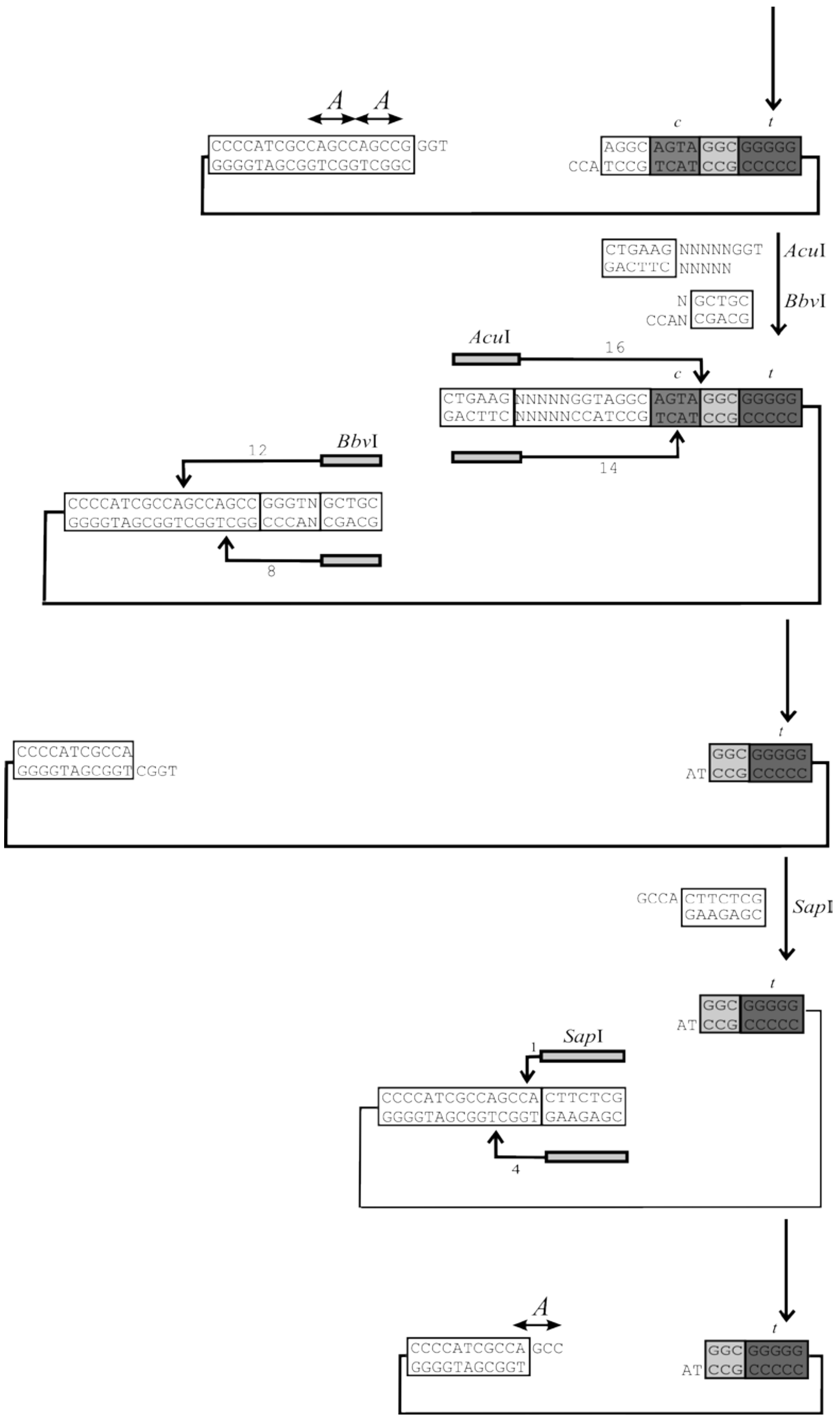


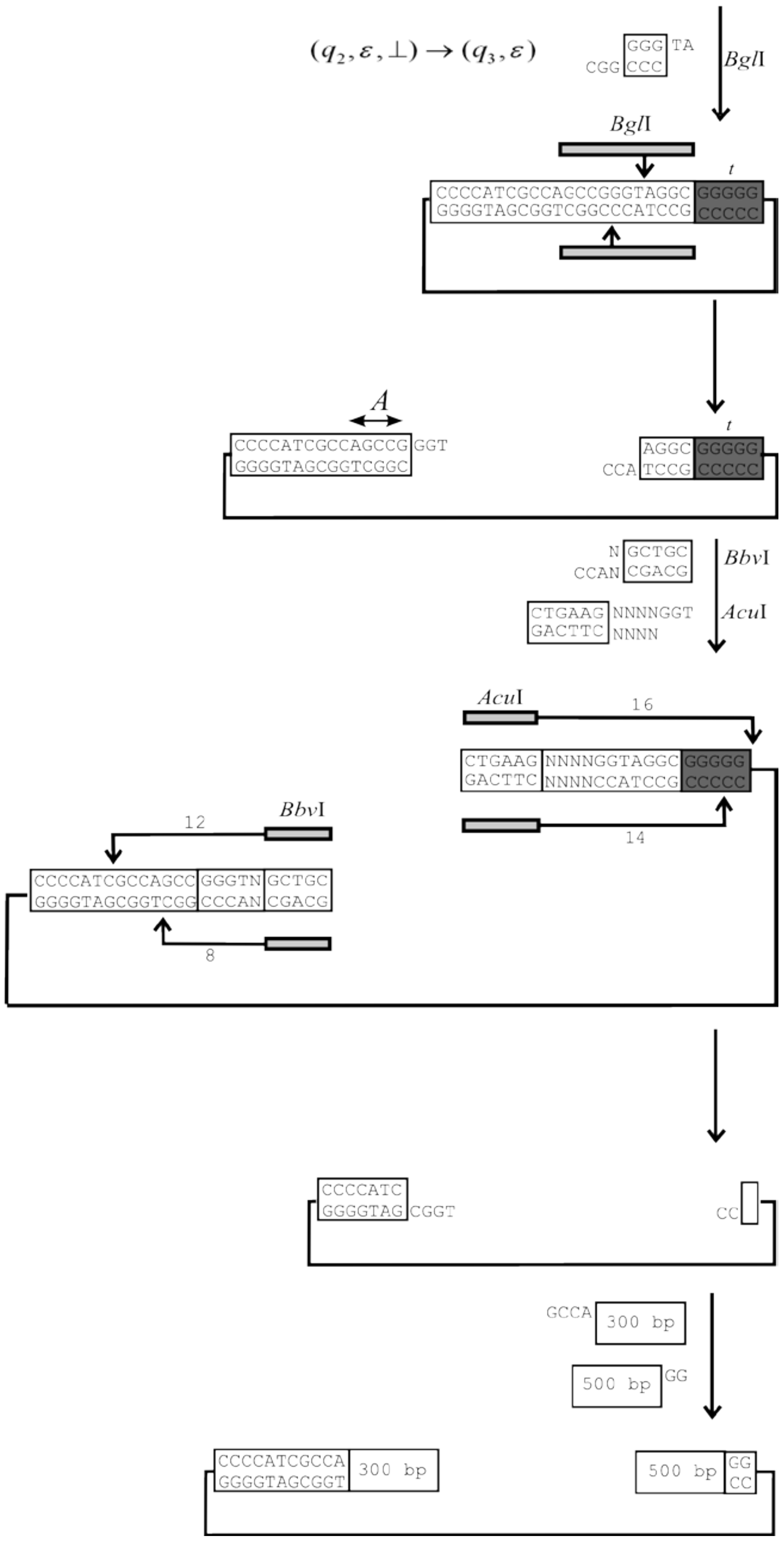

(q0, a, A) → (q0, AA)
AGC CG
CGG TCG
BglI
b c c c t
CCCCATCGCCAGCCAGCCGGGC TGCT GGC AGTA GGC AGTA GGC AGTA GGC GGGGG
GGGGTAGCGGTCGGTCGGCCCG ACGA CCG TCAT CCG TCAT CCG TCAT CCG CCCCC
A A
CCCCATCGCCAGCCA GCC
GGGGTAGCGGTCGGT
GGGC TGCT GGC AGTA GGC AGTA GGC AGTA GGC GGGGG
CGG CCCG ACGA CCG TCAT CCG TCAT CCG TCAT CCG CCCCC
CTGAAG NNNNNGCC
GACTTC NNNNN
AcuI
16
14
CTGAAGNNNNNGCCGGGC TGCT GGC AGTA GGC AGTA GGC AGTA GGC GGGGG
GACTTCNNNNNCGGCCCG ACGA CCG TCAT CCG TCAT CCG TCAT CCG CCCCC
GGC AGTA GGC AGTA GGC AGTA GGC GGGGG
GA CCG TCAT CCG TCAT CCG TCAT CCG CCCCC
(q0, b, A) → (q1, AA)
AGC CT
CGG TCG
CCCCATCGCCAGCCAGCCAGCCTGGC AGTA GGC AGTA GGC AGTA GGC GGGGG
GGGGTAGCGGTCGGTCGGTCGGACCG TCAT CCG TCAT CCG TCAT CCG CCCCC



A A A
CCCCATCGCCAGCCAGCCA GCC
GGGGTAGCGGTCGGTCGGT
c c c t
TGGC AGTA GGC AGTA GGC AGTA GGC GGGGG
CGG ACCG TCAT CCG TCAT CCG TCAT CCG CCCCC
CTGAAG NNNNGCC
GACTTC NNNN
*Acu*I
16
14
CTGAAGNNNNGCCTGGC
GACTTCNNNNCGGACCG
GC
TC CG
$(q_1, c, A) \rightarrow (q_2, \varepsilon)$
AAAG AG
CGG TTTC
*Bgl*I
CCCCATCGCCAGCCAGCCAGCCAAAGAGGC
GGGGTAGCGGTCGGTCGGTCGGTTTCTCCG
CCCCATCGCCAGCCAGCCAGCCA AAG
GGGGTAGCGGTCGGTCGGTCGGT
A
TTC T


CTGAAG NNNNNNAAG
GACTTC NNNNN
AcuI
N GCTGC
TTCN CGACG
BbvI
16
c c t
CTGAAGNNNNNNAAGAGCC AGTA GGC AGTA GGC GGGGG
GACTTCNNNNNNTTCTCCG TCAT CCG TCAT CCG CCCCC
14
12
CCCCATCGCCAGCCAGCCAGCC AAAGN GCTGC
GGGGTAGCGGTCGGTCGGTCGG TTTCN CGACG
8
c t
CCCCATCGCCAGCCA
GGGGTAGCGGTCGGT CGGT
GGC AGTA GGC GGGGG
AT CCG TCAT CCG CCCCC
GCCA CTTCTCG
GAAGAGC
SapI
c t
GGC AGTA GGC GGGGG
AT CCG TCAT CCG CCCCC
SapI
1
CCCCATCGCCAGCCAGCCA CTTCTCG
GGGGTAGCCGTCGGTCGGT GAAGAGC
4
A A
CCCCATCGCCAGCCA GCC
GGGGTACCGGTCGGT
c t
GGC AGTA GGC GGGGG
AT CCG TCAT CCG CCCCC
$(q_2, c, A) \rightarrow (q_2, \varepsilon)$
GGG TA
CGG CCC
BglI
BglI
c t
CCCCATCGCCAGCCAGCCGGGTAGGC AGTA GGC GGGGG
GGGGTAGCGGTCGGTCGGCCCATCCG TCAT CCG CCCCC


A
A
CCCCATCGCCAGCCAGCCG GGT
GGGGTAGCGGTCGGTCGGC
c
t
AGGC AGTA GGC GGGGG
CCA TCCG TCAT CCG CCCCC
CTGAAG NNNNNGGT
GACTTC NNNNN
AcuI
N GCTGC
CCAN CGACG
BbvI
AcuI
16
CTGAAG NNNNNGGTAGGC AGTA GGC GGGGG
GACTTC NNNNNCCATCCG TCAT CCG CCCCC
14
12
BbvI
CCCCATCGCCAGCCAGCC GGGTN GCTGC
GGGGTAGCGGTCGGTCGG CCCAN CGACG
8
CCCCATCGCCA
GGGGTAGCGGT CGGT
GGC GGGGG
AT CCG CCCCC
GCCA CTTCTCG
GAAGAGC
SapI
SapI
1
CCCCATCGCCAGCCA CTTCTCG
GGGGTAGCGGTCGGT GAAGAGC
4
A
CCCCATCGCCA GCC
GGGGTAGCGGT
GGC GGGGG
AT CCG CCCCC


$(q_2, \varepsilon, \perp) \rightarrow (q_3, \varepsilon)$
GGG TA
CGG CCC
*Bgl*I
*Bgl*I
t
CCCCATCGCCAGCCGGGTAGGC GGGGG
GGGGTAGCGGTCGGCCCATCCG CCCCC
A
CCCCATCGCCAGCCG GGT
GGGGTAGCGGTCGGC
t
AGGC GGGGG
CCA TCCG CCCCC
N GCTGC
CCAN CGACG
*Bbv*I
CTGAAG NNNNGGT
GACTTC NNNN
*Acu*I
*Acu*I
16
CTGAAG NNNNGGTAGGC GGGGG
GACTTC NNNNCCATCCG CCCCC
14
*Bbv*I
12
CCCCATCGCCAGCC GGGTN GCTGC
GGGGTAGCGGTCGG CCCAN CGACG
8
CCCCATC
GGGGTAG CGGT
CC
GCCA 300 bp
500 bp GG
CCCCATCGCCA 300 bp
GGGGTAGCGGT
500 bp GG
CC

Privacy-preserving Two-party Rational Set Intersection Protocol

Atsuko Miyaji and Mohammad Shahriar Rahman
Japan Advanced Institttue of Science and Technology,
1-1 Asahidai, Nomi, Ishikawa, Japan 923-1292.
E-mail: miyaji@jaist.ac.jp, md.shahriarr@gmail.com

Keywords: privacy-preserving set-intersection, game theory, computational strict Nash equilibrium, stability with respect to trembles

Received:

Privacy-preserving data mining has been an active research area in recent years due to privacy concerns in many distributed data mining settings. Protocols for privacy-preserving data mining have considered semi-honest, malicious, and covert adversarial models in cryptographic settings, whereby an adversary is assumed to follow, arbitrarily deviate from the protocol, or behaving somewhere in between these two, respectively. Semi-honest model provides weak security requiring small amount of computation, on the other hand, malicious and covert models provide strong security requiring expensive computations like homomorphic encryptions. However, game theory allows us to design protocols where parties are neither honest nor malicious but are instead viewed as rational and are assumed (only) to act in their self-interest. In this paper, we build efficient and secure two-party set-intersection protocol in game-theoretic setting using cryptographic primitives. Our construction allow to avoid the use of expensive tools like homomorphic encryption and zero knowledge proof. We also show that our protocol satisfies computational versions of strict Nash equilibrium and stability with respect to trembles.

Povzetek: Predstavljen je protokol med dvema stranema na osnovi Nashevega ravnotežja.

1 Introduction

A key utility of large databases today is scientific or economic research. Despite the potential gain, this is often not possible due to the confidentiality issues which arise, leading to concerns over privacy infringement while performing the data mining operations. The need for privacy is sometimes due to law (e.g., for medical databases) or can be motivated by business interests. To address the privacy problem, several privacy-preserving data mining protocols using cryptographic techniques have been suggested.

Depending on the adversarial behavior assumptions, those protocols use different models. Classically, two main categories of adversaries have been considered, called Semi-honest and malicious adversaries. Following Goldreich's definition [9], protocols secure in the presence of semi-honest adversaries (or honest-but-curious) assume that parties faithfully follow all protocol specifications and do not misrepresent any information related to their inputs, e.g., set size and content. However, during or after protocol execution, any party might (passively) attempt to infer additional information about the other party's input. On the other hand, security in the presence of malicious parties allows arbitrary deviations from the protocol. It is well known that the protocols secure in the malicious model offer more security. However, these are not efficient enough to be used in practice. Most of these constructions use general zero-knowledge proofs for fully malicious multi-party computation (MPC) protocols. These zero-knowledge compilers lead to rather inefficient constructions [28]. Recently, a new type of adversarial model, named covert adversary, has been proposed by Aumann et al. [3]. These adversaries are somewhere in between the semi-honest and malicious models.

Since the work of Halpern and Teague [12], protocols for some cryptographic tasks (e.g., secret sharing, multi-party computation) have begun to be re-evaluated in a game-theoretic light (see [6, 18] for an overview of work in this direction). In this setting, parties are neither honest nor corrupt but are instead viewed as rational and are assumed (only) to act in their self-interest. This feature is particularly interesting for data mining operations where huge collection of data is used, since parties will not deviate (i.e., abort) as there is no incentive to do so. In many real-world settings, parties are willing to actively deviate/cheat, but only if they are not caught. This is the case in many business, financial, political and diplomatic settings, where honest behavior cannot be assumed, but where the companies, institutions and individuals involved cannot afford the embarrassment, loss of reputation, and negative press associated with being caught cheating, hence having smaller incentive.

In data mining area, private set-intersection and set-union protocols allow two parties interact on their respective input sets. These protocols address several realistic privacy issues. Typical application examples include:

1. Business Interest: Companies may want to decide

whether to make a business alliance by the percentage of customers shared among them, without publishing their customer databases including the shared customers among them. This can be treated as an intersection cardinality problem. As another example, to determine which customers appear on a "do-not-receive-advertisements" list, a store must perform a set-intersection operation between its private customer list and the produce's list.
2. Aviation Security: The Department of Homeland Security (DHS) of the U.S. needs to check whether any passenger on each flight from/to the United States must be denied boarding, based on some passenger watch list. For this purpose, airlines submit their entire list of passengers to DHS, together with other sensitive information, such as credit card numbers. This poses liability issues with regard to innocent passengers' data and concerns about potential data losses. In practice, information only related to the passengers on the list should obtained by DHS without disclosing any information to the airlines.
3. Healthcare: Insurance companies often need to obtain information about their insured patients from other parties, such as other insurance carriers or hospitals. The insurance carriers cannot disclose the identity of inquired patients, whereas, the hospitals cannot provide any information on other patients.

1.1 Related Work

Cryptographic techniques have been used to design many different distributed privacy-preserving data mining algorithms. Secure distributed protocols have been developed for horizontally partitioned data for mining decision trees [24], k-means clustering [22], k-nn classifiers [16]. In the case of vertically partitioned data, it is assumed that different sites collect information about the same set of entities but they collect different feature sets. For example, both a university and a hospital may collect information about a student. Again, secure protocols for the vertically partitioned case have been developed for mining association rules [33], and k-means clusters [14, 32]. All of those previous protocols claimed to be secure only in the semi-honest model. In [7, 17], authors present two-party secure protocols in the malicious model for data mining. They follow the generic malicious model definitions from the cryptographic literature, and also focus on the security issues in the malicious model, and provide the malicious versions of the subprotocols commonly used in previous privacy-preserving data mining algorithms. Assuming that at least one party behaves in semi-honest model, they use threshold homomorphic encryption for malicious adversaries presented by Cramer et al. [4]. Recently, Miyaji et al. presented a new adversarial model named covert adversaries [28] for performing data mining algorithms. They show that protocols under covert adversarial model behave in between semi-honest and malicious models. Oblivious transfer (OT) and homomorphic encryption have been used as the building blocks in [28]. Since homomorphic encryption and zero-knowledge proof are considered too expensive [25], the protocols proposed in malicious and covert adversarial models are not very practical for operations on large data items. Game theory and data mining, in general, have been combined in [15, 30] for constructing various data mining algorithms. Rational adversaries have also been considered in privacy-preserving set operations [2, 34]. These protocols consider Nash equilibrium to analyze the rational behavior of the participating entities. As discussed by Kol and Naor in [21], using Nash equilibrium is not suitable in many cases, since many bad strategies are not ruled out by it. Instead, they suggest the stronger notion of strict Nash equilibrium in the information-theoretic setting, in which every player's strategy is a strict best response. Due to the restrictive nature of this notion, it is regarded as a sufficient condition and not as a necessary one. As in all of cryptography, computational relaxations are meaningful and should be considered; doing so allows us to get around the limitations of the information-theoretic setting. So, analyzing set operations from the viewpoint of computational strict Nash equilibrium is interesting, since it gives more realistic results. There have been several works on game theory based MPC/secret sharing schemes [1, 8, 12, 13, 20, 31]. But [12, 31] require the continual involvement of the dealer even after the initial shares have been distributed or assume that sufficiently many parties behave honestly during the computation phase. Some schemes [1, 20] rely on multiple invocations of protocols. Other work [13] relies on physical assumptions such as secure envelopes and ballot boxes. [8] proposed efficient protocols for rational secret sharing. But secret sharing schemes cannot be directly used for our purpose since they require the existence of TTP and their set up is different.

1.2 Our Contribution

In this work[1], we build two-party secure set-intersection protocol in game-theoretic setting using cryptographic primitives. It is assumed that parties are neither honest nor corrupt but are instead rational. Our construction does not use the expensive tools like homomorphic encryption and zero-knowledge proof. We have used verifiable random functions (VRF) as the underlying cryptographic primitive. We also discuss about replacing VRF with a cheaper tool like message authentication code (MAC) or trapdoor permutation (TDP). We show that our protocol satisfies computational versions of strict Nash equilibrium and stability with respect to trembles, defined by Fuchsbauer et al. [8].

Organization of the paper: The remainder of the paper is organized as follows: Section 2 presents the background and preliminaries. Section 3 describes the protocol model. Section 4 includes protocol construction. In Section 5, we analyze the protocol formally. Section 6 includes performance comparison. We give some concluding remarks in Section 7.

[1]A preliminary version [29] of this paper appears at DBSec2011.

2 Background and Preliminary

2.1 Definitions

In this section, we will state the definitions of computational strict Nash equilibrium and computational strict Nash equilibrium w.r.t. trembles introduced in [8]. A protocol is in Nash equilibrium if no deviations are advantageous; it is in strict Nash equilibrium if all deviations are disadvantageous. In other words, there is no incentive to deviate in the case of a Nash equilibrium whereas there is an incentive not to deviate for a strict Nash equilibrium. Another advantage of strict Nash is that protocols satisfying this notion inhibit subliminal communication. A party who tries to use protocol messages as a covert channel has the risks to lose utility if there is any reasonable probability that the other player is following the protocol, since any detectable deviation by a party from the protocol results in lower utility while the other party follows the protocol. The computational version of strict Nash equilibrium is intuitively close to strict Nash considering the computational limitations. Moreover, our protocol satisfies a strong condition that each party can send a unique legal message that at every point in the protocol. Our protocol thus rules out subliminal communication in a strong sense. We denote the security parameter by n. A function ϵ is negligible if for all $c > 0$ there is a $n_c > 0$ such that $\epsilon(n) < 1/n^c$ for all $n > n_c$; let $negl$ denote a generic negligible function. We say ϵ is noticeable if there exist c, n_c such that $\epsilon(n) > 1/n^c$ for all $n > n_c$.
We consider the strategies in our work as the PPT interactive Turing machines. Given a vector of strategies $\vec{\sigma}$ for two parties in the computation phase, let $u_j(\vec{\sigma})$ denote the expected utility of P_j, where the expected utility is a function of the security parameter n. This expectation is taken over the randomness of the players' strategies. Following the standard game-theoretic notation, $(\sigma'_j, \vec{\sigma}_{-j})$ denotes the strategy vector $\vec{\sigma}$ with P_j's strategy changed to σ'_j.

Definition 1: Π *induces a computational Nash equilibrium if for any PPT strategy* σ'_1 *of* P_1 *we have* $u_1(\sigma'_1, \sigma_2) \leq u_1(\sigma_1, \sigma_2) + negl(n)$*, and similarly for* P_2.

The computational notion of stability with respect to trembles models players' uncertainty about other parties' behavior, and guarantees that even if a party P_i believes that other parties might play some arbitrary strategy with small probability δ (but follow the protocol with probability $1 - \delta$), there is still no better strategy for P_i than to follow the protocol. The following definition is stated for the case of a deviating P_1 (definition for a deviating P_2 is analogous). Let P_1 and P_2 interact, following σ_1 and σ_2, respectively. Let mes denote the messages sent by P_1, but not including any messages sent by P_1 after it writes to its (write-once) output tape. Then $view_2^{\Pi}$ includes the information given by the trusted party to P_2, the random coins of P_2, and the (partial) transcript mes. We fix a strategy γ_1 and an algorithm A. Now, let P_1 and P_2 interact, following γ_1 and σ_2, respectively. Given the entire view of P_1, algorithm A outputs an arbitrary part mes' of mes. Then $view_2^{A,\gamma_1}$ includes the information given by the trusted party to P_2, the random coins of P_2, and the (partial) transcript mes'.

Definition 2: *Strategy* γ_1 *yields equivalent play with respect to* Π*, denoted* $\gamma_1 \approx \Pi$*, if there exists a PPT algorithm* A *such that for all PPT distinguishers* D*, the following holds:* $\mid Pr[D(1^n, view_2^{A,\gamma_1}) = 1] - Pr[D(1^n, view_2^{\Pi}) = 1] \mid \leq negl(n)$

Definition 3: Π *induces a computational strict Nash equilibrium if: 1.* Π *induces a computational Nash equilibrium; 2. For any PPT strategy* $\sigma'_1 \not\approx \Pi$ *, there is a* $c > 0$ *such that* $u_1(\sigma_1, \sigma_2) \leq u_1(\sigma'_1, \sigma_2) + 1/n^c$ *for infinitely many values of* n *.*

In stability with respect to trembles, we say that γ_i is δ-close to σ_j if with probability $1 - \delta$ party P_j plays σ_j, while with probability δ it follows an arbitrary PPT strategy σ'_j. In fact, a pair of strategies (σ_1, σ_2) is stable with respect to trembles if σ_1 (resp., σ_2) remains the best response even if the other party plays a strategy other than σ_2 (resp., σ_1) with some small (but noticeable) probability δ. The fact that the prescribed strategies are in Nash equilibrium ensures that any (polynomial-time) local computation performed by either party is of no benefit as long as the other party follows the protocol. Stated differently, even if a party P_j believes that the other party might play a different strategy with some small probability δ, there is still no better strategy for P_j than to outwardly follow the protocol.

Definition 4: Π *induces a computational strict Nash equilibrium that is stable with respect to trembles if: 1.* Π *induces a computational Nash equilibrium; 2. There is a noticeable function* δ *such that for any PPT strategy* γ_2 *that is* δ*-close to* σ_2*, and any PPT strategy* γ_1*, there exists a PPT strategy* $\sigma'_1 \approx \Pi$ *such that* $u_1(\gamma_1, \gamma_2) \leq u_1(\sigma'_1, \gamma_2) + negl(n)$

Verifiable Random Functions (VRFs): A VRF is a keyed function whose output is random-looking but can still be verified as correct, given an associated proof. The notion was introduced by Micali et al. [27], and various efficient constructions in the standard model are known [5, 26]. It has been shown in [26] that efficient VRFs can be constructed without relying on zero-knowledge proofs[2]. A VRF with range $R = \{R_n\}$ is a tuple of PPT algorithms (Gen, $Eval$, $Prove$, $Verify$) such that: $G(1^n)$ generates the key pair (pk, sk). $Eval_{sk}(x)$ computes the value $y = F_{pk}(x)$; $Prove_{sk}(x)$ computes the proof z that $y = F_{pk}(x)$; and $Verify_{pk}(x, y, z)$ verifies that $y = F_{pk}(x)$ using the proof z. For such a VRF, the properties like correctness, verifiability and pseudorandomness hold.

[2]The VRF gives us computational security. However, it is also possible to design our protocol with information-theoretic security using information-theoretically secure MACs.

3 Model

In a typical protocol, parties are viewed as either honest or semi-honest/malicious. To model rationality, we consider players' utilities. Here we assume that $\mathcal{F} = \{f : X \times Y \to Z\}$ is a functionality where $\mid X \mid = \mid Y \mid$ and their domain is polynomial in size ($poly(n)$). Let $\mathcal{D}$ be the domain of output which is polynomial in size. The function returns a vector I that represents the set-intersection where I_t is set to one if item t is in the set-intersection. In other words, for all the data items of the parties (i.e., X and Y), we will compute $X \cap Y$, and we get I as the output of the function. Clearly for calculating set-intersection, we need to calculate $x_e \wedge y_e$ for each e where $x_e \in X$ and $y_e \in Y$. Similarly, for set-union, we need to calculate $x_e \vee y_e$ for all e. This can be rewritten as $\neg(\neg x_e \wedge \neg y_e)$. Computing the set-union is thus straight forward.

Given that j parties are active during the computation phase, let the outcome o of the computation phase be a vector of length j with $o_j = 1$ iff the output of P_j is equal to the exact intersection (i.e., P_j learns the correct output). Let $\nu_j(o)$ be the utility of player P_j for the outcome o. Following [12], we make the following assumptions about the utility functions of the players:
- If $o_j > o'_j$, then $\nu(o_j) > \nu(o'_j)$
- If $o_j = o'_j$ and $\sum_j o_j < \sum_j o'_j$, then $\nu(o_j) > \nu(o'_j)$

In other words, player P_j first prefers outcomes in which he learns the output; otherwise, P_j prefers strategies in which the fewest number of other players learn the result (in our two-party case, the other player learns). From the point of view of P_j, we consider the following three cases of utilities for the outcome o where $U^* > U > U'$:
- If only P_j learns the output, then $\nu_j(o) = U^*$.
- If P_j learns the output and the other player does also, then $\nu_j(o) = U$.
- If P_j does not learn the output, then $\nu_j(o) = U'$.

So, we have the expected utility of a party who outputs a random guess for the output[3] (assuming other party aborts without any output, or with the wrong output) as follows: $U_{rand} = \frac{1}{|\mathcal{D}|} \cdot U^* + (1 - \frac{1}{|\mathcal{D}|}) \cdot U'$.
Also, we assume that $U > U_{rand}$; else players have almost no incentive to run the computation phase at all. We make no distinction between outputting the wrong secret and outputting a special 'don't know' symbol- both are considered as a failure to output the correct output.

To complete the protocol, we need to provide a way for parties to identify the real iteration. Some work [1, 10, 20] allows parties to identify the real iteration as soon as it occurs. This approach could be used in our protocol if we assume simultaneous channels. But, this approach is vulnerable to an obvious rushing strategy when simultaneous channels are not available. To avoid this, delaying the signal indicating whether a given iteration is real or fake until the following iteration has been used. In this case, until being sure of the occurence of real iteration, a party cannot risk aborting. Moreover, once a party learns that the real iteration occurred, the real iteration is over and all parties can compute the real output. Simultaneous channels are thus not needed in this process at the price of adding only a single round.

[3]We do not consider U''- the utility when neither party learns the output, since 'not learning the output' is not the target of a rational adversary in practice.

4 Rational Set-Intersection Protocol

4.1 An Overview of the Protocol

Let x denote the input of P_1, let y denote the input of P_2, and let f denote the set-intersection function they are trying to compute. We follow the same high-level approach as in [1, 10, 12, 20, 21]. Our intersection computation protocol proceeds in a sequence of 'fake' iterations followed by a single 'real' iteration. As in [8, 11, 19], our protocol is composed of two stages, where the first stage can be viewed as a pre-processing stage and the second stage that computes the intersection takes place in a sequence of $r = r(n)$ iterations. More specifically, in the pre-processing phase the trusted third party chooses $i^* \in \{1, \ldots, r\}$ uniformly at random and defines $\{a_i\} = \{a_1, \ldots, a_r\}$ and $\{b_i\} = \{b_1, \ldots, b_r\}$ as follows: First, it choose $a_1, \ldots, a_{i^*-1} \in \{0, 1\}$ and $b_1, \ldots, b_{i^*-1} \in \{0, 1\}$ independently and uniformly at random. Then, it chooses $c \in \{0, 1\}$ uniformly at random and lets $a_{i^*} = \cdots = a_r = b_{i^*} = \cdots = b_r = c$. The trusted third party creates secret shares of the values $\{a_1, \ldots, a_r\}$ and $\{b_1, \ldots, b_r\}$ using a secure 2-out- of-2 secret sharing scheme, and these shares are given to the parties. For concreteness, we use the specific secret-sharing scheme that splits a bit x into $(x^{(1)}; x^{(2)})$ by choosing $x^{(1)} in \{0, 1\}$ uniformly at random and letting $x^{(2)} = x \oplus x^{(1)}$. In every round $i \in \{1, \ldots, r\}$ the parties exchange their shares for the current round, which enables P_1 to reconstruct a_i, and P_2 to reconstruct b_i as discussed in the Intersection Computation Phase below. Clearly, when both parties are honest, the parties produce the same output bit which is uniformly distributed.

Now, we talk about how to remove the trusted party. We eliminate the need for the trusted third party by relying on a potentially unfair sub-protocol that securely computes with abort the functionality $ShareGen_r$, formally described in Figure 1. Such a protocol with a constant number of rounds can be constructed assuming the existence of oblivious transfer as in [23]. Briefly speaking, the stages have the following form:

Pre-processing stage:

– A value $i^* \in \{1, \ldots, r\}$ is chosen according to some geometric distribution $0 < \alpha < 1$ where α depends on the players' utilities (discussed later in Section 5). This represents the iteration, in which parties will learn the 'true output'.

– For $i < i^*$, $\{a_i\} = \{a_1, \ldots, a_r\}$ (resp.,$\{b_i\} = \{b_1, \ldots, b_r\}$) are chosen according to some distribution that is independent of y (resp., x). For $i \geq i^*$, $a_i = b_i = f(x, y)$.

- Each a_i is randomly divided into shares $a_i^{(1)}$, $a_i^{(2)}$ with $a_i^{(1)} \oplus a_i^{(2)} = a_i$ (and similarly for each b_i). The stage concludes with P_1 being given $a_1^{(1)}, b_1^{(1)}, \ldots, a_r^{(1)}, b_r^{(1)}$, and P_2 being given $a_1^{(2)}, b_1^{(2)}, \ldots, a_r^{(2)}, b_r^{(2)}$ alongside the VRFs [4] ($ShareGen_r$ provides the parties with VRFs so that if a malicious party modifies the share it sends to the other party, then the other party will almost certainly detect this due to the property of VRFs. It will be treated as an abort if such manipulation is detected.).

After this stage, each party has a set of random shares that reveal nothing about the other party's input.

Intersection Computation Phase:

In each iteration i, for $i = 1, \ldots, r$, the parties do the following: First, P_2 sends $a_i^{(2)}$ to P_1 who reconstructs a_i; then P_1 sends $b_i^{(1)}$ to P_2 who reconstructs b_i. (Parties also check the VRF but we omit this here.) If a party aborts in some iteration i, then the other party outputs the value reconstructed in the previous iteration. Otherwise, after reaching iteration r the parties output a_r and b_r, respectively. To compute the correct intersection, parties run a sequence of iterations until the real iteration is identified, and both parties output the result at that point. If some party fails to follow the protocol, the other party aborts. In fact, it is rational for P_j to follow the protocol as long as the expected gain of deviating is positive only if P_j aborts exactly in iteration i^*; and is outweighed by the expected loss if P_j aborts before iteration i^*. The intersection computation phase proceeds in a series of iterations, where each iteration consists of one message sent by each party. Since we want to avoid simultaneous communication, we simply require P_2 to communicate first in each iteration.

When X and Y (the domains of f) are polynomial size, we follow [11, 19] and set $a_i = f(x, \hat{y})$ for $\hat{y}$ chosen uniformly from Y , and set $b_i = f(\hat{x}, y)$ for $\hat{x}$ chosen uniformly (and independently) from X. Note that a_i (resp., b_i) is independent of y (resp., x), as desired.

4.2 Protocol Construction

As described above, our protocol Π consists of two stages. Let p be an arbitrary polynomial, and set $r = p\cdot \mid Y \mid$. We implement the first stage of Π using a sub-protocol π for computing a randomized functionality $ShareGen_r$ (parameterized by a polynomial r) defined in Figure 1. This functionality returns shares to each party, alongside r-time VRF ($Gen, Eval, Prove, Verify$). In the second stage of Π, the parties exchange these shares in a sequence of r iterations as described in Figure 2. The protocol returns I at the end of the operations on all the data items.

[4]It is the parties' own interest that they input the correct values for $ShareGen_r$. Otherwise, they will receive incorrect shares that will give them no chance to compute the correct intersection result, which will only enable them of having smaller incentives.

5 Protocol Analysis

Here we will give some intuition as to why the reconstruction phase of Π is a computational Nash equilibrium for an appropriate choice of α. Let us assume that P_2 follows the protocol, and P_1 deviates from the protocol. (It is easier to analyze the deviations by P_2 since P_2 starts in every iteration.) As soon as it receives $z_2^{(i)} = signal1$, P_1 can abort in iteration $i = i^* + 1$, or it can abort in some iteration $i < i^* + 1$. While aborting in $i = i^* + 1$, P_1 'knows' that it learned the correct output in the preceding iteration (iteration i^*) and can thus output the correct result; however, P_2 will output the correct result as well since it sent the $z_2^{(i)} = signal1$ value to P_1. So P_1 does not increase its utility beyond what it would achieve by following the protocol. In the second case, when P_1 aborts in some iteration $i < i^* + 1$, the best strategy P_1 can adopt is to output $a_1^{(i)}$ hoping that $i = i^*$. Thus, following this strategy, the expected utility that P_1 obtains can be calculated as follows:

- P_1 aborts exactly in iteration $i = i^*$. In this case, the utility that P_1 gets is at most U^*.
- When $i < i^*$, P_1 has 'no information' about correct a_r and so the best it can do is guess. In this case, the expected utility of P_1 is at most U_{rand}.

Considering the above, P_1's expected utility of following this strategy is at most:

$$\alpha \times U^* + (1 - \alpha) \times U_{rand}$$

Now, it is possible to set the value of α such that the expected utility of this strategy is strictly less than U, since $U_{rand} < U$ by assumption. In such a case, P_1 has no incentive to deviate. Since there is always a unique valid message a party can send and anything else is treated as an abort, it follows that the protocol Π induces a strict computational Nash equilibrium which is stable with respect to trembles. The proofs of the propositions below mostly follow those in [8].

Proposition 1: *The protocol* Π *induces a computational Nash equilibrium given that* $0 < \alpha < 1$, $U > \alpha \times U^* + (1 - \alpha) \times U_{rand}$, *and the pseudorandomness of VRFs.*
Proof: We first show that Π is a valid set-intersection protocol. Computational secrecy follows from the proof, below, that the intersection computation is a computational Nash equilibrium. Because if secrecy did not hold then computing the output locally and not participating in the intersection computation phase at all would be a profitable deviation. We next focus on correctness. Assuming both parties run the protocol honestly, the correct output is computed unless:
- $i^* \geq 2^n - 1$

Input: Let the inputs to $ShareGen_r$ be $x \in X_r$ and $y \in Y_r$. (If one of the received inputs is not in the correct domain, a default input is substituted.)

Computation:

- Define values $a_1, \ldots, a_r$ and $b_1, \ldots, b_r$ in the following way:
  - Choose i^* according to some geometric distribution α
  - For $i < i^*$ do,
    - Choose $\hat{y} \leftarrow Y_r$ and set $a_i = f_r(x, \hat{y})$
    - Choose $\hat{x} \leftarrow X_r$ and set $b_i = f_r(\hat{x}, y)$
  - For $i = i^*$, set $a_i = b_i = q = f_r(x, y)$.
  - For $i > i^*$, set $a_i = b_i = NULL$
- For all iteration i, choose $(a_i^{(1)}, a_i^{(2)})$ and $(b_i^{(1)}, b_i^{(2)})$ as random secret shares of a_i and b_i, respectively. (I.e., $a_i^{(1)} \oplus a_i^{(2)} = a_i$, $b_i^{(1)} \oplus b_i^{(2)} = b_i$)
- Let $\mathcal{D} = \{0,1\}^l$ be the domain of the output. Let $(Gen, Eval, Prove, Verify)$ and $(Gen', Eval', Prove', Verify')$ be VRFs with range $\{0,1\}^l$ and $\{0,1\}^n$, respectively. Compute (pk_1, sk_1), $(pk_2, sk_2)$$\leftarrow Gen(1^n)$ and (pk'_1, sk'_1), $(pk'_2, sk'_2)$$\leftarrow Gen'(1^n)$. For all i, compute $share1_i = Eval_{sk_2}(i \| b_i^{(1)})$ and $share2_i = Eval_{sk_1}(i \| a_i^{(1)})$. Also compute $signal1 = Eval'_{sk'_2}(i^* + 1)$ and $signal2 = Eval'_{sk'_1}(i^* + 1)$

Output:

- Send to P_1 the values $(sk_1, sk'_1, pk_2, pk'_2, a_1^{(1)}, \ldots, a_r^{(1)}, (b_1^{(1)}, share1_1), \ldots, (b_r^{(1)}, share1_r), signal1)$.
- Send to P_2 the values $(sk_2, sk'_2, pk_1, pk'_1, b_1^{(1)}, \ldots, b_r^{(1)}, (a_1^{(1)}, share2_1), \ldots, (a_r^{(1)}, share2_r), signal2)$.

Figure 1: Functionality $ShareGen_r$

- For some $i < i^* + 1$, either $signal1 = Eval'_{sk'_2}(i)$ or $signal2 = Eval'_{sk'_1}(i)$

The first event occurs with negligible probability. Pseudorandomness of the VRF, along with the fact that $i^* \leq n$ with all but negligible probability, easily imply that the latter two events happen with only negligible probability as well. We next show that Π induces a computational Nash equilibrium. Assume P_2 follows the strategy σ_2 prescribed by the protocol, and let σ'_1 denote any PPT strategy followed by P_1. (The other case, where P_1 follows the protocol and we look at deviations by P_2, follows similarly with an even simpler approach.) In a given execution of the reconstruction phase, let i denote the iteration in which P_1 aborts (where an incorrect message is viewed as an abort); if P_1 never aborts then set $i = 1$. Let $early$ be the event that $i < i^*$; let $exact$ be the event that $i = i^*$; and let $late$ be the event that $i > i^*$. Let $correct$ be the event that P_1 outputs the correct output. We will consider the probabilities of these events in two experiments: the experiment defined by running the actual intersection computation scheme, and a second experiment where P_1 is given $share1$, $signal1$ chosen uniformly at random from the appropriate ranges. We denote the probabilities in the first experiment by $Pr_{real}[\cdot]$, and the probabilities in the second experiment by $Pr_{ideal}[\cdot]$. We have the following equation using the fact (as discussed above) that whenever late occurs P_2 outputs the correct result. Since when both parties follow the protocol P_1 gets utility U, we need to show that there exists a negligible function ϵ such that $u_1(\sigma'_1, \sigma_2) \leq U + \epsilon(n)$:
$u_1(\sigma'_1, \sigma_2) \leq U^* \times Pr_{real}[exact] + U^* \times Pr_{real}[correct \wedge early] + U' \times Pr_{real}[\overline{correct} \wedge early] + U \times Pr_{real}[late]$
Now we have the following claim that follows from the pseudorandomness of the VRFs:
Claim 1: There exists a negligible function ϵ such that

$$\begin{aligned} | Pr_{real}[exact] &- Pr_{ideal}[exact] | \leq \epsilon(n) \\ | Pr_{real}[late] &- Pr_{ideal}[late] | \leq \epsilon(n) \\ | Pr_{real}[correct \wedge early] &- Pr_{ideal}[correct \wedge early] | \leq \epsilon(n) \\ | Pr_{real}[\overline{correct} \wedge early] &- Pr_{ideal}[\overline{correct} \wedge early] | \leq \epsilon(n) \end{aligned}$$

Now, we have $U_{ideal} = U^* \cdot Pr_{ideal}[exact] + U^* \cdot Pr_{ideal}[correct \wedge early] + U' \cdot Pr_{ideal}[\overline{correct} \wedge early] + U \cdot Pr_{ideal}[late]$

From Claim 1 we get that $| u_1(\sigma'_1, \sigma_2) - U_{ideal} | \leq \epsilon(n)$ for some negligible ϵ. We bound U_{ideal} as follows: Let $abort = exact \vee early$, so that $abort$ is the event that P_1 aborts before iteration $i^* + 1$. We have $Pr_{ideal}[exact \mid abort] = \alpha$ and $Pr_{ideal}[correct \mid early] = 1/\mathcal{D}$. It is

Input: Party P_1 has input x and party P_2 has input y.

Computation:

– Preliminary phase:
1. P_1 chooses $\hat{y} \in Y_r$ uniformly at random, and sets $a_0 = f_r(x, \hat{y})$. Similarly, P_2 chooses $\hat{x} \in X_r$ uniformly at random, and sets $b_0 = f_r(\hat{x}, y)$.
2. Parties P_1 and P_2 run a protocol π to compute $ShareGen_r$, using their inputs x and y.
3. If P_2 receives $\perp$ from the above computation, it outputs b_0 and halts. Otherwise, the parties proceed to the next step.
4. Denote the output of P_1 from π by $(sk_1, sk_1', pk_2, pk_2', a_1^{(1)}, \dots, a_r^{(1)}, (b_1^{(1)}, share1_1), \dots, (b_r^{(1)}, share1_r), signal1)$.
5. Denote the output of P_2 from π by $(sk_2, sk_2', pk_1, pk_1', b_1^{(1)}, \dots, b_r^{(1)}, (a_1^{(1)}, share2_1), \dots, (a_r^{(1)}, share2_r), signal2)$.

– Intersection Computation Phase
For all i do:
P_2 **sends message to** P_1:
1. P_2 computes $y_2^{(i)} = Prove_{sk_2}(i \| a_i^{(2)}), z_2^{(i)} = Eval'_{sk_2'}(i), \bar{z}_2^{(i)} = Prove'_{sk_2'}(i)$. It sends $(a_i^{(2)}, share2_i, y_2^{(i)}, z_2^{(i)}, \bar{z}_2^{(i)})$ to P_1.
2. If P_2 does not send anything to P_1, then P_1 outputs a_{i-1} and halts. P_2 sends $(a_i^{(2)}, share2_i, y_2^{(i)}, z_2^{(i)}, \bar{z}_2^{(i)})$ to P_1. If $Verify_{pk_2}(i \| a_i^{(2)}, share2_i, y_2^{(i)}) = 0$ or $Verify'_{pk_2'}(i, z_2^{(i)}, \bar{z}_2^{(i)}) = 0$, then P_1 outputs a_{i-1} and halts. If $signal1 \neq z_2^{(i)}$ then P_1 outputs a_{i-1}, sends its iteration-i message to P_2, and halts.
3. If $Verify_{pk_2}(i \| a_i^{(2)}, share2_i, y_2^{(i)}) = 1$ and $a_i^{(1)} \oplus a_i^{(2)} \neq NULL$ (i.e., $x = x_i$), then P_1 sets $a_i = a_i^{(1)} \oplus a_i^{(2)}$, and continues running the protocol.
P_1 **sends message to** P_2:
1. P_1 computes $y_1^{(i)} = Prove_{sk_1}(i \| b_i^{(1)}), z_1^{(i)} = Eval'_{sk_1'}(i), \bar{z}_1^{(i)} = Prove'_{sk_1'}(i)$. It sends $(b_i^{(1)}, share1_i, y_1^{(i)}, z_1^{(i)}, \bar{z}_1^{(i)})$ to P_2.
2. If P_1 does not send anything, then P_2 outputs b_{i-1} and halts. P_1 sends $(b_i^{(1)}, share1_i, y_1^{(i)}, z_1^{(i)}, \bar{z}_1^{(i)})$ to P_2. If $Verify_{pk_1}(i \| b_i^{(1)}, share1_i, y_1^{(i)}) = 0$ or $Verify'_{pk_1'}(i, z_1^{(i)}, \bar{z}_1^{(i)}) = 0$, then P_2 outputs b_{i-1} and halts. If $signal2 \neq z_1^{(i)}$ then P_2 outputs b_{i-1}, sends its iteration-i message to P_1, and halts.
3. If $Verify_{pk_1}(i \| b_i^{(1)}, share1_i, y_1^{(i)}) = 1$ and $b_i^{(1)} \oplus b_i^{(2)} \neq NULL$ (i.e., $y = y_i$), then P_2 sets $b_i = b_i^{(1)} \oplus b_i^{(2)}$, and continues running the protocol.

Output: If all r iterations have been run, party P_1 outputs a_r and party P_2 outputs b_r.

Figure 2: Protocol for computing the functionality for set-intersection

easy to find that $U_{ideal} = U + (\alpha \cdot U^* + (1-\alpha) \cdot U_{rand} - U) \cdot Pr_{ideal}[abort] \leq U$ given that $\alpha \cdot U^* + (1-\alpha) \cdot U_{rand} - U < 0$. This shows that Π induces a computational Nash equilibrium.

Proposition 2: *If* $0 < \alpha < 1$, $U > \alpha \times U^* + (1-\alpha) \times U_{rand}$, *VRFs are pseudorandom, and there is always a unique valid message each party can send, then the protocol* Π *induces a computational strict Nash equilibrium.*
Proof: The analysis of Proposition 1 and the fact that there is always a unique valid message each party can send show us that Π induces a computational strict Nash equilibrium. In other words, say P_1 plays a strategy σ_1' with $\sigma_1' \not\approx \Pi$. This implies that $Pr_{real}[abort] \geq 1/poly(n)$ for infinitely many values of n. Claim 1 then shows that $Pr_{ideal}[abort] \geq 1/poly(n)$ for infinitely many values of n, and so $U - U_{ideal} \geq 1/poly(n)$. Since $\mid u_1(\sigma_1', \sigma_2) - U_{ideal} \mid$ is negligible, we conclude that $U - u_1(\sigma_1', \sigma_2) \geq 1/poly(n)$ for infinitely many values of n.

Proposition 3: *The protocol* Π *is stable with respect to trembles given that* $0 < \alpha < 1$ *and* $U > \alpha \times U^* + (1-\alpha) \times U_{rand}$.
Proof: Let δ be a parameter. Let ρ_2 be any PPT strategy that is δ-close to σ_2, and let ρ_1 be an arbitrary PPT strategy for P_1. There exists a PPT strategy σ_1' satisfying Definition 3. Let strategy σ_1' be defined as follows:

1. Run ρ_1 on the output of $ShareGen_r$. Set $aborted = 0$.
2. In each iteration i:

– Receive the iteration-i message m_i from P_2. If P_2 aborts, then set $aborted = 1$.

– Give m_i to ρ_1 and get message m'_i as response.

– If $aborted = 1$ then forward m'_i to P_2; otherwise, compute the response (e.g., the protocol transcripts) as prescribed by Π and send that to P_2 instead.

3. If $aborted = 0$ determine the output according to Π; otherwise, output whatever ρ_1 outputs.

When σ'_1 interacts with σ_2, then $aborted$ is never set to 1; thus, σ'_1 yields equivalent play w.r.t Π, and $u_1(\sigma'_1, \sigma_2) = u_1(\rho_1, \rho) = U$. It remains to show $u_1(\rho_1, \rho_2) \leq u_1(\sigma'_1, \rho_2) + negl(n)$. Let $\hat{\rho}_2$ is run only with probability δ by ρ_2. During a session where P_1 follows strategy ρ_1, let $abort$ denote the event that ρ_1 aborts before P_2 aborts, and let $pr_{abort}(a)$ be the probability of $abort$ when P_2 follows strategy a. We now state two claims. The first one says that the only advantage to P_1 of playing ρ_1 rather than σ'_1 because of σ_1 aborting first.

Claim 2: $u_1(\rho_1, \hat{\rho}_2) - u_1(\sigma'_1, \hat{\rho}_2) \leq pr_{abort}(\hat{\rho}_2) \cdot (U^* - U')$

The following claim shows that $abort$ occurs at least as often when ρ_1 interacts with σ_2 as when ρ_1 interacts with $\hat{\rho}_2$.

Claim 3: $pr_{abort}(\sigma_2) \geq pr_{abort}(\hat{\rho}_2)$

We omit the proofs of the above since they are analogous to those in [8].

Now, let $\tilde{U} = \alpha \times U^* + (1 - \alpha) \times U_{rand}$, and we have $\tilde{U} < U$ by assumption. Using $U_{ideal} \leq U$, *Claim 1*, *Claim 2*,and *Claim 3* we get that $u_1(\rho_1, \rho_2) - u_1(\sigma'_1, \rho_2) \leq (1 - \delta) \times (\tilde{U} - U) \times pr_{abort}(\hat{\rho}_2) + \delta \times (U^* - U') \times pr_{abort}(\hat{\rho}_2) + negl(n)$. Since $\tilde{U} - U$ is strictly negative, there exists $\delta > 0$ for which the above expression is negligible for n large enough. This completes the proof sketch.

According to the above propositions and their proofs, we give the theorem as follows:

Theorem 1: *If* $0 < \alpha < 1$, $U > \alpha \times U^* + (1 - \alpha) \times U_{rand}$, *and VRFs are pseudorandom, then* Π *induces a computational strict Nash equilibrium that is stable with respect to trembles.*

6 Performance Comparison

For a single data item, the protocol in covert model [28] requires only a constant number of rounds, single oblivious transfer to the number of input items, and requires $n|C|$ number of communication bits where n is the security parameter and $|C|$ is the size of the circuit being computed. Whereas the protocol in malicious model [17] requires d number of rounds (d is the depth of C), more communication bits (dependent on the number of parties), and expensive computation like ZK proof which is linear to the number of data items. Both the covert and malicious models rely on homomorphic operations. On the other hand, in our rational model, we do not need any ZK proof or homomorphic encryption computation. As discussed earlier, use of ZK proof and homomorphic encryption leads to inefficiency in practical world and we want to avoid using the expensive tool like ZK proofs. As for the other parameters in rational model, the share size is $|t| + O(n)$, where t is the size of data items and n is the security parameter. The round complexity of the protocol for each item is $O(\alpha^{-1})$, where α is the geometric distribution used to pick up the value of i^* (typically, we will need only two rounds for each items in our protocol). In our construction, we have showed the use of VRFs, which is also an expensive tool. However, it is possible to design our protocol with information-theoretic security using information-theoretically secure MACs. It is also possible to replace VRFs with TDPs, since the properties of VRF that we require for our constructions are also available with TDPs. Using TDPs would give us much more efficient protocol as compared to using VRFs. Construction using TDP is straightforward and we omit the details here. Clearly, the rational model requires much lighter computation than the protocol designed in malicious model and performs even better than the covert model in terms of computational overhead given that MAC or TDP is used instead of VRF.

7 Conclusion

In this paper, we have proposed a privacy-preserving set-intersection protocol in two-party settings from the game-theoretic perspective. We have used VRFs as the underlying cryptographic primitive. We also suggest replacing VRFs with information-theoretic secure MACs or TDPs, which are simple and efficient. Our protocol satisfies strong equilibrium notions like computational versions of strict Nash equilibrium and stability with respect to trembles.

References

[1] Abraham, I., Dolev, D., Gonen, R., and Halpern, J.: Distributed Computing Meets Game Theory: Robust Mechanisms for Rational Secret Sharing and Multiparty Computation. In 25th ACM Symposium Annual on Principles of Distributed Computing, pp. 53-62, 2006.

[2] Agrawal, R. and Terzi, E.: On Honesty in Sovereign Information Sharing. In the 10th International Conference on Extending Database Technology-EDBT'06, pp. 240-256 2006.

[3] Aumann, Y. and Lindell, Y.: Security Against Covert Adversaries: Efficient Protocols for Realistic Adversaries, In Theory of Cryptography- TCC'07, pp. 137-156, 2007.

[4] Cramer, R., Damgard, I., and Nielsen, J.B.: Multiparty Computation from Threshold Homomorphic Encryption. In Advances in Cryptology- EUROCRYPT'01, pp. 280-299, 2001.

[5] Dodis, Y.: Efficient Construction of (distributed) Verifiable Random Functions. In 6th International

Workshop on Theory and Practice in Public Key Cryptography- PKC'03, pp. 1-17, 2003.

[6] Dodis, Y. and Rabin, T.: Cryptography and Game Theory. In N. Nisan, T. Roughgarden, E. Tardos, and V. Vazirani, editors, Algorithmic Game Theory, pp. 181-207, Cambridge University Press, 2007.

[7] Emura, K., Miyaji, A., and Rahman, M.S.: Efficient Privacy-Preserving Data Mining in Malicious Model. In The 6th International Conference on Advanced Data Mining and Applications, ADMA'10. pp. 370-382, 2010.

[8] Fuchsbauer, G., Katz, J., and Naccache, D.: Efficient Rational Secret Sharing in Standard Communication Networks. In Theory of Cryptography- TCC'10, pp. 419-436, 2010.

[9] Goldreich, O.: Foundations of cryptography: Basic applications. Cambridge Univ. Press, Cambridge, 2004.

[10] Gordon, S.D., Katz, J.: Rational Secret Sharing, Revisited. In 5th International Conference on Security and Cryptography for Networks- SCN'06, pp. 229-241, 2006.

[11] Gordon, S.D., Hazay, C., Katz, J., Lindell, Y.: Complete Fairness in Secure Two-party Computation. In 40th Annual ACM Symposium on Theory of Computing- STOC'08, pp. 413-422, 2008.

[12] Halpern, J. and Teague, V.: Rational Secret Sharing and Multi-party Computation: Extended abstract. In 36th Annual ACM Symposium on Theory of Computing- STOC'04, pp. 623-632, 2004.

[13] Izmalkov, S., Micali, S., and Lepinski, M.: Rational Secure Computation and Ideal Mechanism Design. In 46th Annual Symposium on Foundations of Computer Science- FOCS'05, pp. 585-595, 2005.

[14] Jagannathan, G. and Wright, R.N.: Privacy-preserving Distributed k-means Clustering over Arbitrarily Partitioned Data. In ACM SIGKDD International Conference on Knowledge Discovery and Data Mining- KDD'05, pp. 593-599, 2005.

[15] Jiang, W., Clifton, C. and Kantarcioglu, M.: Transforming Semi-Honest Protocols to Ensure Accountability. In Data and Knowledge Engineering (DKE), 65(1), pp. 57-74, 2008.

[16] Kantarcioglu, M. and Clifton, C.: Privately Computing a Distributed k-nn Classifier. In 7th European Conference on Principles and Practice of Knowledge Discovery in Databases- PKDD'04, pp. 279-290, 2004.

[17] Kantarcioglu, M., and Kardes, O.: Privacy-preserving Data Mining in the Malicious model. In International Journal of Information and Computer Security, Vol. 2, No. 4, pp. 353-375, 2008.

[18] Katz, J.: Bridging Game Theory and Cryptography: Recent Results and Future Directions. In Theory of Cryptography- TCC'08. pp. 251-272, 2008.

[19] Katz, J.: On Achieving the Best of Both Worlds in Secure Multi-party Computation. In 39th Annual ACM Symposium on Theory of Computing- STOC'07, pp. 11-20, 2007.

[20] Kol, G. and Naor, M.: Cryptography and Game Theory: Designing Protocols for Exchanging Information. In Theory of Cryptography- TCC'08, pp. 320-339, 2008.

[21] Kol, G. and Naor, M.: Games for Exchanging Information. In 40th Annual ACM Symposium on Theory of Computing- STOC'08, pp. 423-432, 2008.

[22] Lin, X., Clifton, C. and Zhu, M.: Privacy-preserving Clustering with Distributed EM Mixture Modeling. In Knowledge and Information Systems, July, Vol. 8, No. 1, pp. 68-81, 2005.

[23] Lindell,Y: Parallel coin-tossing and constant-round secure two-party computation. Journal of Cryptology, 16(3):143-184, 2003.

[24] Lindell, Y. and Pinkas, B.: Privacy-preserving Data Mining. In Advances in Cryptology- CRYPTO'00, pp. 36-54, 2000.

[25] Liu, J., Lu, Y.H., and Koh, C.K.: Performance Analysis of Arithmetic Operations in Homomorphic Encryption. In ECE Technical Reports, Purdue University, 2010.

[26] Lysyanskaya, A.: Unique Signatures and Verifiable Random Functions from the DH-DDH Separation. In Advances in Cryptology- CRYPTO'02, pp. 597-612, 2002.

[27] Micali, S., Rabin, M. O., and Vadhan, S. P.: Verifiable Random Functions. In 40th Annual Symposium on Foundations of Computer Science- FOCS'99, pp. 120-130, 1999.

[28] Miyaji, A., and Rahman, M.S.: Privacy-preserving Data Mining in Presence of Covert Adversaries. In The 6th International Conference on Advanced Data Mining and Applications, ADMA'10. pp. 429-440, 2010.

[29] Miyaji, A., and Rahman, M.S.: Privacy-Preserving Data Mining: A Game-Theoretic Approach. In The 25th Annual WG 11.3 Conference on Data and Applications Security and Privacy, DBSec'11. pp. 186-200, 2011.

[30] Nix, R. and Kantarcioglu, M.: Incentive Compatible Distributed Data Mining. In IEEE International Conference on Privacy, Security, Risk and Trust, pp. 735-742, 2010.

[31] Ong, S. J., Parkes, D., Rosen, A., and Vadhan, S.: Fairness with an Honest Minority and a Rational Majority. In Theory of Cryptography- TCC'09, pp. 36-53, 2009.

[32] Su, C., Bao, F., Zhou, J., Takagi, T., Sakurai, K.: Security and Correctness Analysis on Privacy-Preserving k-Means Clustering Schemes. In IEICE Trans. Fundamentals, Vol.E92-A, No.4, pp. 1246-1250, 2009.

[33] Vaidya, J. and Clifton, C.: Privacy Preserving Association Rule Mining in Vertically Partitioned Data. ACM SIGKDD International Conference on Knowledge Discovery and Data Mining- KDD'02, pp. 639-644, 2002.

[34] Zhang, N. and Zhao, W.: Distributed Privacy-preserving Information Sharing. In the 31st International Conference on Very large data bases-VLDB'05, pp. 889-900, 2005.

Hardware-Software Co-design for Reconfigurable Field Programmable Gate Arrays Using Mixed-Integer Programming

Faridah M. Ali
Department of Computer Engineering,
Kuwait University, P.O. Box 5969, Safat 13060, Kuwait
E-mail: faridah@puc.edu.kw

Helal Al-Hamadi
Department of Information Science,
Kuwait University, P.O. Box 5969, Safat 13060, Kuwait
E-mail: helal@cfw.kuniv.edu

Ahmed Ghoniem
Department of Finance and Operations Management,
Isenberg School of Management, University of Massachusetts Amherst,
Amherst, MA 01002, U.S.A.
E-mail: aghoniem@isenberg.umass.edu

Hanif D. Sherali
Grado Department of Industrial and Systems Engineering (0118),
Virginia Tech,
Blacksburg, VA 24061, U.S.A.
E-mail: hanifs@vt.edu

Keywords: hardware-software co-design, FPGAs, task partitioning, scheduling, mixed-integer 0-1 programming

Received: September 19, 2011

This paper presents a novel mixed-integer programming formulation for scheduling non-preemptive, aperiodic, hard real-time tasks with precedence constraints. It provides an integrated partitioning and scheduling co-synthesis approach. The problem formulation maps some n precedence-related, indivisible jobs having specified processing requirements, release times, and due-dates to a system involving a single Central Processing Unit (CPU) and up to m potential reconfigurable Field Programmable Gate Arrays (FPGAs). We provide a time-indexed mixed-integer 0-1 programming formulation that jointly assigns tasks to either the CPU or to one of the FPGAs, and determines the task sequence for each software or hardware component that is utilized, with the objective of minimizing a composite cost of task partitioning and scheduling. Computational experience is provided using randomly generated instances to demonstrate the applicability of the proposed methodology.

Povzetek: Predstavljen je algoritem za porazdeljevanje opravil pri snovanju programske in strojne opreme.

1 Introduction and Motivation

The task partitioning and scheduling problem bears practical significance in software/hardware co-design of hard real-time applications that arise in a host of applications such as flight and defense control, telecommunication, or nuclear power plants, to name a few. Specifically, we consider the problem of partitioning and scheduling n indivisible (no preemption), aperiodic (which could be considered as the body of a looped system), precedence-related jobs that are characterized by specific processing requirements, release times, and due-dates (which are deadlines that cannot be violated). The system architecture is depicted in Figure 1, and involves a single Central Processing Unit (CPU) and a maximum of some m potential reconfigurable Field Programmable Gate Arrays (FPGAs) controlled by a single controller unit. The system components are connected with two explicit communication buses (channels). The first bus is the system bus, which is used for input/output (I/O) data transfers, whereas the second is used for FPGA reconfiguration transfers. The task processing effort is primarily carried out by the CPU, in general, and the FPGAs are incrementally utilized if the CPU alone cannot conform to all due-date restrictions. In contrast with scheduling problems

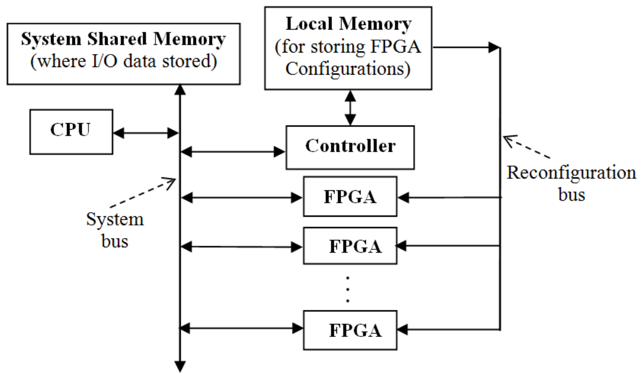


Figure 1: System Architecture

that arise in production and logistical systems where it is often desirable to meet imposed due-dates, it is *imperative* to comply with the specified due-dates in the problem under investigation.

Another reason for the use of such dual systems in practice resides in the benefits accruing from the cooperation between software and hardware components. As a consequence, the co-design problem has gained increasing attention over the last decade, (see [6], [7], [10], [12], [16], [19], and [20]). This process involves three main operations, namely, resource (software, hardware, and auxiliary components) *allocation* to the system, task *partitioning* among software and hardware processing components, and *scheduling* of the tasks over their assigned processing components (CPU or FPGA). Most works in the literature have addressed this joint problem via two-phase methods [14], where a task partitioning is achieved first, and then tasks are subsequently scheduled over the relevant processing components. It is important, however, to note that the task partitioning and scheduling operations are intertwined [9], and need to be dealt with concurrently in order to determine optimal operational solutions.

An algorithm that aims at partitioning and scheduling the tasks on two CPUs and several hardware components with the objective of minimizing the total execution time and the hardware cost was presented by Liu and Wong [13]. Arato et al. [2] designed a procedure for scheduling tasks on a software component, where tasks that violate their deadlines are partially assigned to an auxiliary hardware component. A mixed-integer 0-1 formulation for partitioning precedence-related tasks on a single-CPU-single-FPGA system, as well as a genetic algorithm to address larger problem instances was presented in [2]. Ali and Das [1] presented a heuristic algorithm that progressively assigns indivisible tasks to dynamically reconfigurable FPGAs when certain tasks cannot meet their deadlines on the CPU. In contrast, this challenging scheduling problem is tackled in the present paper using a mixed-integer 0-1 programming model with the objective of minimizing a composite cost of task partitioning and scheduling on a single CPU along with several potentially reconfigurable FPGAs.

Jeong et al. [11] proposed a mixed-integer 0-1 formulation and a heuristic for hardware-software partitioning in systems consisting of a single CPU and a single FPGA, with the objective of minimizing the reconfiguration overhead. A mixed-integer programming model was developed by Niemann and Marwedel [14] that employs a two-phase method for hardware/software partitioning, where a tentative schedule is first proposed and is subsequently verified; if the timing constraints are violated, the partitioning step is repeated with timing constraints that are tighter than the estimated scheduling horizon length. Bender [3] discussed an alternative mixed-integer programming approach for mapping real-time precedence graphs into a system of Application Specific Integrated Circuits (ASICs) that are used as hardware components, and pipelined microprocessors that are used as software components. Initially, only a limited number of hardware components are used, and these are incrementally increased until a feasible solution is obtained.

Recently, hardware/software partitioning for multimedia and wireless mobile applications has gained great importance. Brogioli et al. [4] proposed a set of criteria for partitioning real-time embedded multimedia applications between software programmable Digital Signal Processors (DSPs) and hardware-based FPGA coprocessors. Dasu and Panchanathan [5] investigated the design and development of a dynamically reconfigurable multimedia processor that involves an optimal hardware/software codesign methodology. Furthermore, a hardware/software partitioning for multimedia application that utilizes process-level pipelining and a heuristic technique based on simulating annealing was presented by Juan et al. [12].

A design space exploration tool that supports both explicit communication and reconfigurable hardware was addressed by Haubelt et al. [10]. The developed algorithm strictly separates functionality from the architecture, and maps a process graph onto components such that data dependencies given by the process graph can be handled in the resulting implementation. The work presented in the present paper bears some similarity to this approach in that we also use explicit communication channels and reconfigurable hardware in our system architecture.

Observe that the problem at hand is also related to the challenging class of unrelated parallel machine scheduling problems (see [15]). It is characterized by the presence of release dates, imperative due-dates, precedence constraints, inter-task data communication, reconfiguration of hardware resources (FPGAs), and a composite objective function that minimizes the processing and resource utilization costs. Also, it is specially structured due to the fact that all processing components are identical, except for the CPU.

The main contribution of the present paper is a novel formulation of a time-indexed mixed-integer 0-1 programming model for the co-synthesis hardware/software integrated partitioning and scheduling problem. The motivation for the proposed work is two-fold: first, it provides a tool for simultaneously partitioning (or mapping) and scheduling tasks onto hardware/software units, and second, it offers a design space exploration tool to ascertain the minimum number of FPGAs required for a particular application before a system is actually built.

The remainder of this paper is organized as follows. In Section 2, we formally describe the problem under investigation along with our notation. Thereafter, we introduce in Section 3 a mixed-integer 0-1 programming formulation that simultaneously captures the requirements pertaining to the partitioning and scheduling operations. Section 4 delineates our data generation scheme, and reports our computational experience using a set of random test instances to demonstrate the effectiveness of the proposed solution approach. We close the paper in Section 5 with a summary of our findings.

2 Problem Description and Notation

We address the problem of scheduling some n precedence-related jobs having specified processing requirements, release times, and inviolable due-dates in a system involving a single *Central Processing Unit* (CPU) and a maximum of some m potential reconfigurable *Field Programmable Gate Arrays* (FPGAs). Each job can be processed either by the CPU itself, or it can be scheduled for processing on one of the m available FPGAs. In either case, no preemption is permitted, and each resource (CPU or FPGA) can process at most one job at any point in time. However, whenever an FPGA begins processing a job, it must be reconfigured to perform the required operation by a single available reconfiguration *controller*. This reconfiguration process consumes a specified duration that is part of the total processing time required for performing the job on the associated FPGA. Note that while the controller is reconfiguring any FPGA to begin processing a job, it is occupied and cannot simultaneously reconfigure another FPGA. Again, no preemption is permitted in the reconfiguration process. Also, not all the m available FPGAs need be used; in fact, there is a fixed cost for using an FPGA that competes with the cost related to achieving scheduling efficiency. When an FPGA finishes executing any task, it becomes available for the next task if needed. This is described more in detail in the model formulation given in Section 3.

In our analysis, the FPGAs cannot be preconfigured since it is not known in advance which task will be executed on the FPGA rather than on the CPU. Moreover, if the system has more than one FPGA, it is not known which FPGA will be used until scheduling is complete. The proposed system (Figure 1) is generic and can be used for executing any precedence-related jobs. However, for a given real-time set of jobs, once an optimal system configuration and scheduling decisions are determined, then a partial run-time reconfiguration can be performed during implementation where only the configuration bits of the particular task are transferred to the FPGA in order to reduce the configuration time.

Notation:

- $j = 1, ..., n$: Index for jobs.
- For establishing precedences, we define $P = \{(j_1, j_2) : j_1 \rightarrow j_2$, i.e., the processing of job j_1 must precede that of job $j_2\}$.
- m = maximum potential number of FPGAs available for use.
- Index for *time-slots*: Let the time be discretized so that the scheduling time-line for the CPU and each FPGA contains s time-slots, where the end of time-slot s is estimated to be the maximum allowable makespan duration, based on due-dates. We index the time-slots over this maximum makespan duration as: $t = 1, ..., s$. (Note that the actual duration of the time-slots is arbitrary and rescalable, and typically ranges from 1 to 300 seconds in practice. Also, the time measurement (in discretized units) begins at time $t = 0$ at the beginning of slot 1.)
- Index for *slots*: These correspond to a sequential indexing of the foregoing time-slots over the resources, where the slots for the CPU are indexed as $k = 1, ..., s$, the slots for FPGA 1 are indexed as $k = s + 1, ..., 2s$, the slots for FPGA 2 are indexed as $k = 2s + 1, ..., 3s$, and so on, up to slots $k = ms + 1, ..., (m + 1)s$ for FPGA m. (Note that the *time-slots* are numbered $1, ..., s$, whereas the *slots* are indexed contiguously over the CPU and the m FPGAs as $k = 1, ..., (m + 1)s$.)
- Index for resources: $r = 0, 1, ..., m$, where $r = 0$ is the CPU and $r = 1, ..., m$ index the FPGAs.
- $r(k)$ = resource corresponding to slot k. (So $r(k) = 0$ for $k = 1, ..., s$, $r(k) = 1$ for $k = s + 1, ..., 2s$, and so on.)
- δ_{jr} = processing time of job j on resource r (in integral time units that conform with the time-slot duration).
- π_{jr} = reconfiguration time on resource FPGA r to process job j (in integral time units that conform with the time-slot duration). We assume that π_{jr} is included within $\delta_{jr}, \forall j, \forall r \geq 1$.
- α_j = release time of job j. That is, if a job j has no predecessors, then the earliest time-slot to start its processing is $\alpha_j + 1$.
- d_j = due-date of job j.
- lb_j = lower bound on the starting time-slot for job j. If a job has no predecessor, then $lb_j = \alpha_j + 1$; otherwise we may simply take $lb_j = \max\{\alpha_j + 1, \max_{j_1:(j_1,j)\in P}\{lb_{j_1} + \min_r\{\delta_{j_1 r}\}\}\}$.

- $k \bmod^{+} (s)$ = remainder for the division k/s, except that this is taken as s if the remainder is zero.

Principal Decision Variables:
The principal decision variables are defined below:

- $x_{jk} = \begin{cases} 1 & \text{if job } j \text{ is assigned to start at slot } k \\ 0 & \text{otherwise}, \quad \forall j, k. \end{cases}$
- $y_r = \begin{cases} 1 & \text{if FPGA } r \text{ is utilized} \\ 0 & \text{otherwise}, \quad r = 1, ..., m. \end{cases}$

Auxiliary Decision Variables:
The following auxiliary variables are defined based on the x_{jk}-variables:

- s_j = time-slot at which the processing of job j starts.
- f_j = time-slot at which the processing of job j ends.

Key Sets:
We define certain key sets based on individual job processing times (which could, in general, be CPU- and FPGA-dependent), reconfiguration times (which could again be FPGA-dependent), job release/availability times, and job due-dates:

- $S_j \equiv$ {slots k: $x_{jk} = 1$ is a possible decision based on release, due-date, processing, and reconfiguration times}, $\forall j = 1, ..., n$.

  Observe that we may express S_j as

  $S_j = \{k : k \in \{1, ..., (m+1)s\}, lb_j \leq k \bmod^{+}(s) \leq d_j - \delta_{jr(k)} + 1\}, \forall j = 1, ..., n.$
- $S_{jr} = \{k \in S_j :$ slot k is associated with resource $r\}, \forall j = 1, ..., n, r = 0, ..., m.$
- $J_k = \{(j, \ell) : j \in \{1, ..., n\}, \ell \in S_j$, and $x_{j\ell} = 1$ would imply that slot k would be occupied by the reconfiguration/processing of job $j\}$, $\forall k = 1, ..., (m+1)s$.

  Note that J_k can be expressed as

  $J_k = \{(j, \ell) : j \in \{1, ..., n\}, \ell \in S_j, r(\ell) = r(k), \ell \bmod^{+}(s) \leq k \bmod^{+}(s) \leq \ell \bmod^{+}(s) + \delta_{jr(\ell)}\}.$
- $R_t = \{(j, \ell) : j \in \{1, ..., n\}, \ell \in S_j$, and $x_{j\ell} = 1$ would imply that during the time-slot t, the controller is busy performing a reconfiguration$\}, \forall t = 1, ..., s - \Delta$, where $\Delta = \min_{j,r}\{\delta_{jr} - \pi_{jr}\}$.

  We can formally state R_t as

  $R_t = \{(j, \ell) : j \in \{1, ..., n\}, \ell \in S_j, \ell \geq s + 1, \ell \bmod^{+}(s) \leq t \leq \ell \bmod^{+}(s) + \pi_{jr(\ell)}\}.$

  Note that $R_t \equiv \emptyset$ for $t = s - \Delta + 1, ..., s$.

Cost Parameters:

- c_{jk} = cost of commencing the operation of job j at the duration corresponding to slot k.
- λ = cost per FPGA used.

Remark 1. The cost of resources (number of FPGAs used) and efficiency (as predicated by the term $\sum_{j=1}^{n} \sum_{k \in S_j} c_{jk} x_{jk}$) compete in the objective function of the mathematical program formulated in Section 3. In addition, observe that it might be desirable to preclude alternative optimal solutions that allow idleness on the available resources. To this end, we may require the hierarchy of cost parameters c_{jk} associated with any job j to be strictly increasing with respect to the time-slot $k \bmod^{+} (s)$. □

3 Mathematical Programming Formulation

We present below our proposed mixed-integer 0-1 programming formulation, denoted by **HWSW**, which ascertains the task partitioning and scheduling decisions in order to minimize the total processing and resource costs.

$$\textbf{HWSW}: \text{Minimize} \quad \sum_{j=1}^{n} \sum_{k \in S_j} c_{jk} x_{jk} + \lambda \sum_{r=1}^{m} y_r \tag{1a}$$

$$\text{subject to} \quad \sum_{k \in S_j} x_{jk} = 1, \quad \forall j = 1, ..., n \tag{1b}$$

$$\sum_{(j,\ell) \in J_k} x_{j\ell} \leq 1, \quad \forall k = 1, ..., (m+1)s \tag{1c}$$

$$\sum_{(j,\ell) \in R_t} x_{j\ell} \leq 1, \quad \forall t = 1, ..., s - \Delta \tag{1d}$$

$$s_j = \sum_{k \in S_j} [k \bmod^{+}(s)] x_{jk}, \quad \forall j = 1, ..., n \tag{1e}$$

$$f_j = \sum_{k \in S_j} [k \bmod^{+}(s) + \delta_{jr(k)} - 1] x_{jk}, \quad \forall j = 1, ..., n \tag{1f}$$

$$f_{j_1} + 1 \leq s_{j_2}, \quad \forall (j_1, j_2) \in P \tag{1g}$$

$$y_r \geq \sum_{k \in S_{jr}} x_{jk}, \quad \forall j = 1, ..., n, \forall r = 1, ..., m \tag{1h}$$

$$1 \geq y_1 \geq y_2 \geq ... \geq y_m \geq 0 \tag{1i}$$

$$\sum_{j=1}^{n} \sum_{k \in S_{jr}} x_{jk} \geq \sum_{j=1}^{n} \sum_{k \in S_{j,r+1}} x_{jk}, \quad \forall r = 1, ..., m-1 \tag{1j}$$

$$x \text{ binary}, y \text{ continuous.} \tag{1k}$$

The objective function (1a) seeks to minimize the total processing and resource costs. Constraint (1b) requires each job to be feasibly scheduled on either the CPU or on an FPGA. Constraint (1c) asserts that no resource can be processing more than one job simultaneously during any associated slot. Likewise, Constraint (1d) enforces the restriction that the controller can be reconfiguring at most one job at any point in time. Note that whenever $x_{jk} = 1$

for some job $j \in \{1, ..., n\}$, and $k \in S_j$, where slot k corresponds to FPGA r, say, then it is assumed that job j starts its reconfiguration by the controller on FPGA r at the time corresponding to the beginning of slot k, after which it immediately proceeds to be processed by FPGA r. Constraints (1e) and (1f) state the definitional identities for the start and finish time-slots for each job j in terms of the x-variables, and Constraint (1g) represents the precedence relationships. Constraint (1h), along with the second objective term, invokes that $y_r = 1$ if and only if some job is processed on FPGA r, and is zero otherwise, even when restricted to be a continuous variable on [0, 1]. Constraints (1i) and (1j) attempt to defeat the inherent symmetry in the problem with respect to the FPGAs, assuming that the FPGAs are identical with respect to processing times. (Note that if there are subgroups of identical FPGAs, then these types of constraints can be incorporated within each such subgroup.) Specifically, Constraint (1i) requires that the lower-indexed FPGAs be utilized first, and more importantly, Constraint (1j) attempts to impart an identity to the utilized FPGAs by imposing the hierarchy that FPGA r should process at least as many jobs as FPGA $r + 1$. Without such hierarchical constraints, the inherent symmetry in the problem can hopelessly mire the solution process by requiring it to search among symmetric reflections of essentially the same sets of solutions (see Sherali and Smith [17]). Finally, (1k) represents the logical restrictions on the variables, where the y-variables would automatically turn out to be binary-valued at optimality, even when permitted to be continuous variables on the interval [0, 1].

The model is a linear mixed-integer 0-1 program (MIP), which can be solved using a commercial solver to any desired percentage of optimality.

Remark 2. It is possible to accommodate different alternative objective functions of practical interest within this modeling framework involving the makespan, resource usage (number of FPGAs, durations of usage of FPGAs, etc.), and the completion times of jobs, as desired. □

4 Computational Experience

In this section, we begin by delineating the data generation scheme for constructing random, small- to moderately-sized test instances. Next, we present our computational experience to test the efficiency of the proposed mathematical programming formulation. Our proposed formulation was coded in AMPL and solved using CPLEX 10.1 on a Dell Precision 650 workstation having a Xeon(TM) CPU 2.40 GHz processor and 1.50 GB of RAM.

4.1 Data generation

To demonstrate the usefulness of the optimization scheduling model, we have used randomly simulated, realistic graphs along with the associated data. Although the resulting test cases do not pertain to actual hardware data, they simulate what one might expect in practice.

In our test-bed, the number of jobs n was selected to be 10, 20, or 30, and the number of potential FPGAs, m, was specified to ensure the feasibility of the resulting instance upon generating the different processing times and key sets.

Random Parameters:

- The precedence relationships between the tasks were randomly generated according to the following scheme. Given $j_2 \in \{2, ..., n\}$, and for all $j_1 \in \{1, ..., j_2 - 1\}$, we generated $\varphi_{j_1 j_2}$ using a uniform distribution over the range [0, 1]. For some threshold ρ, if $\varphi_{j_1 j_2} > \rho$, then the arc (j_1, j_2) was added to P, that is, task j_1 was required to be a predecessor of task j_2. In our scheme, we took $\rho = 0.75$, which induced a desired density of the precedence arcs in the task graph. Also, redundant arcs were suppressed from the set P by invoking transitivity in the precedence relationships. That is, if the arc (j_β, j_γ) was generated while there also exists an alternative path from j_β to j_γ in the precedence graph, then the direct arc (j_β, j_γ) is redundant, and was consequently deleted from the set P.

- The δ_{j0}-parameters and the release dates, α_j, were generated using a uniform discrete distribution over the sets {1,...,10} and {0,...,15}, respectively.

- Following a scheme similar to that employed by Ali and Das [1], we took the reconfiguration times π_{jr} to be given by $\lfloor 0.015\zeta_j \rfloor$, where ζ_j was randomly generated using a uniform distribution over the range [0, 200], and where $\lfloor \cdot \rfloor$ denotes the rounding-down operation.

- We set $d_j = \lfloor 1.3lb_j + \theta_j \rfloor$, where θ_j is a randomly generated value using a uniform distribution over the interval $[\bar{\delta} - \tau - \frac{\Lambda}{2}, \bar{\delta} - \tau + \frac{\Lambda}{2}]$, and where $\bar{\delta}$ is the average processing time over the CPU, and τ and Λ are parameters that influence the tightness of the due-dates. Here, the term $[\bar{\delta} - \tau - \frac{\Lambda}{2}, \bar{\delta} - \tau + \frac{\Lambda}{2}]$ is based on Fisher's method [8]; we took $\tau = 0.2$ and $\Lambda = 1$ in our experiments.

- The processing costs were computed as $c_{jk} = \lfloor \aleph_j \rfloor + k \bmod^{+}(s)$, where $\aleph_j$ was generated using a uniform distribution over the interval [0, 5], and where s was computed as noted below.

Additional Deduced Parameters:

- $\delta_{jr} = \lceil 0.3\delta_{j0} \rceil + \pi_{jr}, \forall j, \forall r \geq 1.$

- $s = \max\limits_{j=1,...,n} \{d_j\}.$

- $\lambda = 4 \max\limits_{j=1,...,n} \{d_j\}.$

Remark 3. If the reconfiguration times, as well as certain processing times on FPGAs, are fractional, then all time-related parameters may be suitably rescaled to achieve data integrality. This process, however, could entail a significant growth in the size of the problem. An alternative approach would be to round up all fractional processing times (and to round down due-dates, as appropriate). The marginal time amounts that are introduced by this rounding process may be viewed as idleness-buffers on the relevant hardware or software components. By solving the problem instance with such integerized data, we would obtain a heuristic solution to the original problem. Further improvements can be achieved via a routine that shifts operations to the left to eliminate the marginal idleness that has been introduced by this rounding process. $\square$

4.2 Illustrative example

As a prelude, we present below an example to illustrate the problem under investigation and to gain insights into the proposed formulation. Consider the problem instance where the jobs to be processed are related via the precedence graph depicted in Figure 2 and where the associated parameters are provided in Table 1 (with the remaining data being generated as prescribed in Section 4.1). The available resources include a single CPU and one FPGA for potential use. The discrete, time-indexed scheduling horizon has a projected length of $s = 39$ time-slots.

The solution produced by Model HWSW is summarized in Table 1, and is depicted in the Gantt-chart in Figure 3. This small-sized problem instance was solved to optimality in 0.04 seconds. Observe that the slot values k for which $x_{jk} = 1$, as specified in Table 1, indicate indirectly when operations start their processing, as well as the processing components on which these operations are scheduled (by observing the time-slot ranges attributed to each software/hardware component). For instance, both jobs 9 and 10 start at the beginning of time-slot 28 and are completed at the end of time-slot 31. However, since $x_{9,28} = x_{10,67} = 1$, job 9 is scheduled on the CPU, whereas job 10 is processed on FPGA 1.

Job j	α_j	δ_{j0}	π_{j1}	d_j	$k : x_{jk} = 1$	s_j	f_j
1	4	1	1	11	5	5	5
2	1	9	1	9	45	6	9
3	5	4	3	14	10	10	13
4	0	6	1	23	14	14	19
5	3	9	2	24	53	14	18
6	5	2	3	29	20	20	21
7	14	10	3	29	61	22	27
8	1	7	3	39	71	32	37
9	2	4	2	38	28	28	31
10	4	7	1	39	67	28	31

Table 1: Data and results for an illustrative example

4.3 Computational results

Table 2 summarizes the results obtained for instances having $n = 10$ and 20. The first column of this table specifies the instance number as well as the number of jobs and the maximum number of FPGAs involved. The second column provides the length of the scheduling horizon (number of time-slots). In the next three columns, we present the results obtained for solving the continuous or linear programming (LP) relaxation of Model HWSW, denoted $\overline{\textbf{HWSW}}$, by allowing the x-variables to assume continuous values between 0 and 1. The optimal objective value of the continuous relaxation ($\nu(\overline{\text{HWSW}})$), the ensuing computational time in seconds, and the % optimality gap, are reported for each instance. We define the % Gap as $= 100\frac{\nu(\text{HWSW})-\nu(\overline{\text{HWSW}})}{\nu(\text{HWSW})}$. The final two columns relate to solving Problem HWSW to optimality. Table 2 reveals that for these instances having up to 20 jobs, optimal solutions were obtained within manageable times.

Table 3 provides the results for the more challenging 30-job problem instances. Here, in addition to solving the problem to optimality, we demonstrate the effectiveness of employing two heuristics to derive good quality feasible solutions in a relatively timely fashion. In Table 3, in addition to the LP solution, we report the first MIP solution produced by CPLEX during its branch-and-bound (B&B) exploration, as well the best available MIP solution that the solver could obtain within a specified computational limit of 300 CPU seconds. We compare these two heuristic approaches to solving the problem to optimality by reporting the percentage deviation (% Dev.) between the heuristic solution value from the optimal solution value.

Whereas the LP relaxation objective values for $n = 10$ were particularly tight (within an average optimality gap of 0.7% as seen in Table 2), the LP relaxation for larger problem instances ($n \in \{20, 30\}$ over Tables 2 and 3) exhibited an average optimality gap of 19.5%. As a consequence, these larger problem instances required a significant amount of branching operations within the B&B algorithm employed by the solver. However, it is worthwhile mentioning that by considering the first MIP solution obtained by the solver, or by imposing a time-bound (of 300 CPU seconds) on the computational effort, the resulting heuristic solutions respectively sacrificed only 2% and 0.18% of optimality on average for $n = 30$, while achieving an average computational savings of 98.6% and 93.8%, respectively, as compared with determining optimal solutions. This indicates that near-optimal solutions are typically identified at early stages of the B&B exploration and, hence, motivates the use of such B&B-based heuristic approaches for larger problem instances. It is also important to highlight that the efficiency of such B&B-based heuristic approaches is predicated on the integration of a rounding heuristic scheme and a relaxation-induced neighborhood search (RINS) heuristic that is implemented within CPLEX. At a frequency that the user may control (based on the difficulty and the size of the test instance under investigation), the solver triggers the rounding scheme and/or the RINS heuristic at any node of the B&B tree in an attempt to identify a good quality MIP solution. In our quest for the first MIP solution and the best MIP solution within 300 CPU seconds, the rounding scheme and the RINS heuristic were triggered by the solver at every node explored in the B&B tree.

5 Conclusions

We have proposed a novel formulation for partitioning and scheduling precedence-related jobs on both hardware and software components over a time-indexed horizon. Our model effectively captures the nonpreemption assumption, the precedence constraints, the inviolable due-date restrictions, and the processing times required over hardware and software components. Computational experience reported using randomly generated test instances reveals that optimal solutions can be computed (for $n \leq 20$) within about 20 seconds. For $n = 30$, we demonstrated the effectiveness of two branch-and-bound-based heuristic approaches in producing near-optimal solutions. In particular, using the branch-and-bound algorithm of CPLEX 10.1 to output the first MIP solution and the best MIP solution within a timelimit of 300 CPU seconds, the resulting heuristic solutions respectively sacrificed only 2% and 0.18% of optimality on average, while achieving an average computational savings of 98.6% and 93.8%, respectively, as compared with determining optimal solutions. Thus, we recommend the use of such heuristic approaches for large instances in order to obtain near-optimal solutions with manageable effort. Also, we have focused our attention on minimizing the total processing and resource costs. However, the proposed model is flexible enough to accommodate different alternative objective functions of practical interest within this same modeling framework, which involve the makespan, resource usage (number of FPGAs, durations of usage of FPGAs, etc.), and the completion times of jobs, as desired.

Acknowledgement

This work has been partially supported by the *National Science Foundation* under Grant CMMI-0552676.

References

[1] Ali, F. M. and Das A. S. (2004), Hardware-software co-synthesis of hard real-time systems with reconfigurable FPGAs, *Computers and Electrical Engineering*, 471-489.

[2] Arato P., Juhasz S., Mann Z. A., Orban A. and Papp, D. (2003), Hardware-software partitioning in embedded system design, *IEEE International Symposium on Intelligent Signal Processing*, 197-202.

[3] Bender, A. (1996), Design of an optimal loosely coupled heterogeneous multiprocessor system, *Proceedings of European Design and Test Conference*, 275-281.

[4] Brogioli, M., Radosavljevic, P. and Cavallaro, J. R. (2006), A general hardware/software co-design methodology for embedded signal processing and multimedia workloads, *Fortieth Asilomar Conference on Signals, Systems and Computers*, 1486-1490.

[5] Dasu, A. and Panchanathan, S. (2001), Reconfigurable media processing, *Proceedings of International Conference on Information Technology: Coding and Computing*. 2-4 April 2001, 300 - 304.

[6] Dittmann, F., Gotz, M. and Rettberg, A. (2007), Model and methodology for the synthesis of heterogeneous and partially reconfigurable systems, *Parallel and Distributed Processing Symposium*, 2007, IPDPS 2007, IEEE International, 26-30 March 2007, 1-8.

[7] Edwards, M.D. and Forrest, J. (1995), Hardware/software partitioning for performance enhancement, *IEEE Colloquium on Partitioning in Hardware-Software Codesigns*, 2, 1-5.

[8] Fisher, M. L. (1976), A dual algorithm for the one machine scheduling problem, *Mathematical Programming*, 11, 229-251.

[9] Giovanni, D.M. and Rajesh, K. G. (1997), Hardware/software co-design, *Proceedings of the IEEE*, 85(3), 349-365.

[10] Haubelt, C., Otto, S., Grabbe C. and Teich, J. (2005), A system-level approach to hardware reconfigurable systems, *Proceedings of the Design Automation Conference ASP-DAC 2005*, 298-301.

[11] Jeong, B., Yoo, S., Lee, S. and Choi, K. (2000), Hardware-software cosynthesis for run-time incrementally reconfigurable FPGAs, *Proceedings of the Design Automation Conference ASP-DAC 2000*, 169-174.

[12] Juan P. C., David, S., Onassis, C. and Alvaro, S. (2000), Pipelining-based tradeoffs for hardware/software codesign of multimedia systems, *8th Euromicro Workshop on Parallel and Distributed Processing*, 383-390.

[13] Liu, H. and Wong, D. F. (1998), Integrated partitioning and scheduling for hardware/software co-design, *Proceedings of the International Conference on Computer Design*, 609-614.

[14] Niemann, R. and Marwedel, P. (1997), An algorithm for hardware/software partitioning using mixed integer linear programming, *Design Automation for Embedded Systems*, 2(2), 165-193.

[15] Pinedo, M. (1995), *Scheduling: Theory, Algorithms and Systems*, Prentice-Hall, NJ.

[16] Saul, J. M. (1999), Hardware/software codesign for FPGA-based systems, *Proceedings of the 32nd Annual Hawaii International Conference on System Sciences*.

[17] Sherali, H. D. and Smith, J. C. (2001), Improving discrete model representations via symmetry considerations, *Management Science*, 47, 1396-1407.

[18] Shin, Y. and Choi, K. (1997), Enforcing schedulability of multi-task systems by hardware-software codesign, *International Workshop on Hardware/Software Co-Design*, 3-7.

[19] Sipper, M. and Sanchez, E. (2000), Configurable chips meld software and hardware, *IEEE Computer*, 33(1), 120-121.

[20] Takeuchi, Y., Shibata, K. and Kunieda, H. (1994), Codesign methodology on programmable hardware and software system, *IEEE Asia-Pacific Conference on Circuits and Systems*, 182-187.

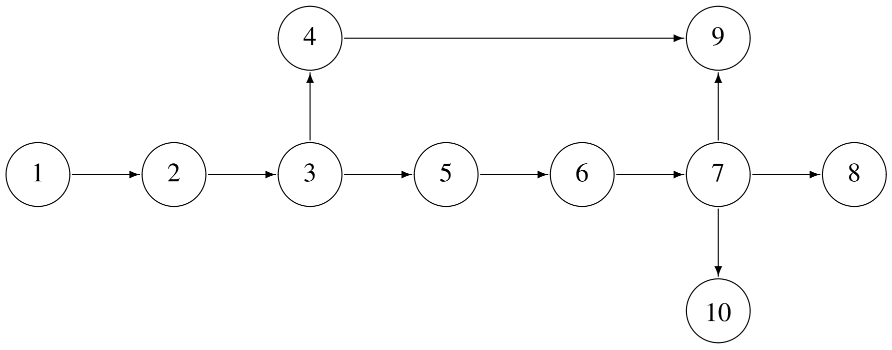


Figure 2: Task precedence graph for the illustrative example involving 10 tasks

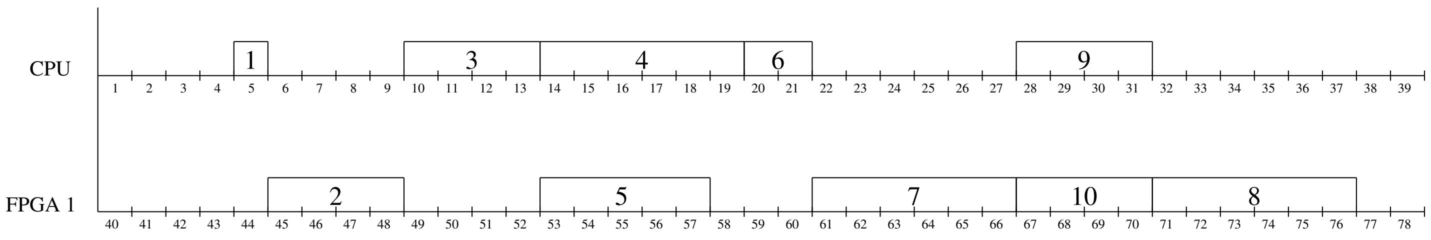


Figure 3: Gantt chart for an illustrative example involving 10 tasks and one FPGA

Instance	s	**LP Solution**			**MIP solution**	
#, (n, m)		ν(**HWSW**)	**Time (s)**	**% Gap**	ν(**HWSW**)	**Time (s)**
1, (10,3)	42	387	0.05	1.5	393	0.35
2, (10,3)	69	413	0.09	0	413	0.09
3, (10,3)	59	363.8	0.11	1.6	370	0.20
4, (10,3)	71	431	0.12	0.2	432	0.14
5, (10,3)	62	396	0.09	0.2	397	0.12
6, (20,2)	40	639.66	0.12	20.2	803	0.39
7, (20,2)	44	649.30	0.11	22.4	838	1.45
8, (20,2)	58	829.14	0.14	23.0	1079	17.12
9, (20,2)	72	1008	0.10	22.2	1296	0.54
10, (20,2)	63	841	0.17	23.5	1100	20.93

Table 2: Performance of Model HWSW for instances having $n = 10$ and 20

Instance	s	**LP Solution**			**First MIP**			**MIP within 300 s**		**Optimal MIP**	
#, (n, m)		ν(**HWSW**)	**Time (s)**	**% Gap**	ν(**HWSW**)	**Time (s)**	**% Dev.**	ν(**HWSW**)	**% Dev.**	ν(**HWSW**)	**Time (s)**
11, (30,2)	77	1444.7	0.18	17.9	1769	6.3	0.4	1761	0	1761	10.8
12, (30,2)	74	1228.11	0.21	20.0	1571	17.4	2.2	1537	0	1537	103.93
13, (30,2)	57	1166.49	0.23	17.7	1484	21.6	4.5	1433	0.9	1419	6647.9
14, (30,2)	92	1175.98	0.39	19.2	1485	165.6	1.9	1466	≈ 0	1456	2819.3
15, (30,2)	81	1639.24	0.39	9.6	1836	17.6	1.1	1815	0	1815	6943.7

Table 3: Performance of Model HWSW for instances having $n = 30$

Adaptive Random Testing Based on Two-Point Partitioning

Chengying Mao
School of Software and Communication Engineering,
Jiangxi University of Finance and Economics, 330013 Nanchang, China
Email: maochy@yeah.net

Keywords: adaptive random testing, test data generation, partition, failure pattern, candidate set

Received: October 24, 2011

Test data generation is a key issue in the field of software testing. Adaptive random testing (ART) method has been proposed by Chen et al. to improve the fault-revealing ability of random testing. In the paper, we are mainly concerned with the partitioning-based adaptive random testing and present a new ART based on two-point partitioning. In the new algorithm, the current max-area region is partitioned by the midpoint of two points instead of a single point. The first point is randomly generated, and the second point is picked out from the candidate set according to the farthest distance criterion. In order to compare our algorithm with other two well-known algorithms, the experiments for the case of two-dimension are performed. The results show that our ART-TPP algorithm has a positive improvement for the other two, i.e. ART-RP and ART-BP. Moreover, the appropriate size of candidate set is determined as 2 or 3 based on our sensitivity analysis.

Povzetek: Predstavljena je nova metoda za generiranje podatkov na osnovi dvo-točkovne porazdelitve

1 Introduction

In the past decades, software testing has proved to be an effective way to ensure software quality. As stated by NIST, software errors cost the U.S. economy about \$59.5 billion each year, but the testing infrastructure could save 1/3 cost [1]. However, software testing is also a time-consuming and high cost activity in the whole life-cycle of software development. Consequently, it is is necessary to realize the automation of testing activity so as to improve the efficiency. At present, test data generation is recognized as the most difficult for automated software testing.

In fact, test data generation is a search process of selecting the representative data from the input domain of the program under test. In recent years, the most popular way is to use meta-heuristic search (MHS) techniques such as simulated annealing (SA) [2], ant colony optimization (ACO) [3], and generic algorithm (GA) [4], to produce test inputs which can find faults with high probability. But this kind of test data generation method has two limitations: (1) It needs the guideline information about program's internal constructs, and (2) the search process consumes a lot of time due to slow convergence speed. On the other hand, in general, black-box (functional) testing methods, such as random testing and boundary value analysis, can produce test data with high speed and low cost. However, these methods fail to show strong fault revealing capability. Therefore, a possible solution is to rebuild the low-cost functional testing method to generate more effective test inputs.

Random testing (RT) is a naïve method for generating test data, and has been widely adopted by most popular testing tools. However, the size of test data set is very limited in comparison with the whole input space of program under test, so the test inputs generated by RT are not really even distribution yet. In order to overcome this problem, Chen *et al.* proposed an improved method, called *adaptive random testing* (ART) [5, 6], to produce more decentralized test inputs. Their experimental results show that ART can find potential faults faster than the traditional RT. At present, two kinds of ART have been confirmed effective. One is partitioning-based method [6], and the other is distance-based method [7]. In the former method, random partitioning and bisection are two well-known strategies. In the paper, a new partitioning strategy named two-point partitioning is proposed. We believe that it is a useful supplement for the existing ART methods.

The paper is structured as follows. In the next section, it reviews the background of adaptive random testing. It mainly includes two parts: software failure pattern and the basic idea of ART. Then, the two-point partitioning-based ART is addressed in Section 3. In Section 4, some experiments are studied to validate the effectiveness of our method. Finally, the concluding remarks are given in Section 5.

2 Background

2.1 Software failure pattern

In the field of software testing and debugging, the empirical knowledge may play an important role in finding failure-causing input or locating faults. Therefore, it is necessary to summarize the failure pattern [8] or bug pattern [9] so as to generate good fault-revealing test inputs. Generally speaking, the knowledge about location and shape of failure patterns can facilitate black-box testing methods to select test data.

The failure pattern of program under test, in fact, it is the rule of failure-causing inputs of the program. According to Chen and Schneckenburger's analysis [6, 7, 8], the patterns of failure-causing inputs can be classified into three types: block pattern, strip pattern and point pattern. As illustrated in Figure 1, for the block failure pattern, the inputs causing program failure are within a specific area. From the perspective of program code, this kind of fault may lie in the statement block under a compound predicate. For example, if a fault exists in the branch such as `if(a<=x && x<=b && c<=y && y<=d)`, the failure-causing input area can be denoted as $\{(a,b),(c,d)\}$. In the second pattern, i.e. strip failure pattern, the failure may be attributed to predicate fault in a branch. For example, if an expected form of predicate `if(x+y>=k1)` is wrongly written as `if(x+y>=k2)` by a programmer, the failure-causing inputs will lie in a strip, whose width is determined by the value of $|k1-k2|$. In the last failure pattern, the failure-causing inputs will scatter into some points or small areas in the whole input domain. The corresponding faults may occur in a branch with modulo operation or bitwise operation etc. For instance, if some statements in a branch `if(x%10==0 && y%10==0)` contain faults, the corresponding failure pattern belongs to the point case.

It should be noted that, we only discuss the two-dimensional case in the above analysis. But these three failure patterns are also applicable to other cases, such as one dimension or high dimension.

2.2 Adaptive random testing

As mentioned above, ART attempts to generate test data which can evenly scatter in the input space with the greatest possible. Hence, this method can enhance the fault-revealing capability of test cases. Based on this idea, Chen *et al.* developed a series of ART methods for generating test data set [5, 6, 7]. Furthermore, Ciupa *et al.* have successfully used ART to test object-oriented software, and their ARTOO method [10] can reduce the number of tests generated to reveal the first fault.

All existing ART methods can be classified into two types: distance-based strategy and partitioning-based strategy. In the paper, we only pay attention to the second strategy. Here, we primarily introduce two well-known partitioning algorithms proposed by Chen *et al* [6].

(1) *Random Partitioning Algorithm* (ART-RP). This kind of partitioning algorithm samples test data according to the proportion of region area to whole input space. The basic process of producing test cases can be described as below.

Without loss of generality, here we assume the input domain as a rectangle for two-dimension case. As illustrated in Figure 2(a), the initial test input is randomly selected from the whole input domain. Then, the space is divided into four sub-rectangles according to X and Y coordinates of the initial input point. Next, the max-area region is selected out from them, and the second test input is randomly generated from this area (as shown in Figure 2(b)). At this moment, the whole input domain has been divided into seven regions. Hereafter, the random partitioning is iteratively performed on the current max-area region until the termination condition is satisfied.

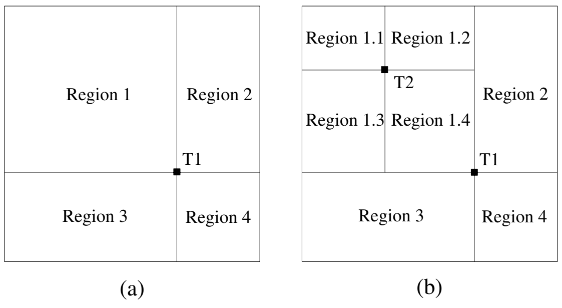


Figure 2: The illustration for random partitioning algorithm (2-*D* case).

(2) *Bisection Algorithm* (ART-BP). The second kind of partitioning strategy proposed by Chen *et al.* is bisection. In this strategy, the partitioning is not based on the coordinates of test input points but the width and height of a region.

Initially, as shown in Figure 3(a), a test input T1 is randomly generated from the whole input domain. Then, the whole region is divided into two parts through performing a partition on the bisector of region height. Meanwhile, another test input T2 is randomly generated in the sub-region, which previously does not contain any test input points (refer to Figure 3(b)). Similarly, the region can also be divided on the bisector of width. As shown in Figure 3(c), test input T3 and T4 can be generated in the next step. Subsequently, the partition process can be continued by alternately bisecting the height and width of each sub-region until the termination condition is satisfied.

2.3 Basic terms

In order to facilitate the expression below, we also follow and define some basic terms about ART. For an input domain D, the corresponding domain size is denoted as d. Meanwhile, we use m and n to denote the number of failure-causing inputs and number of test inputs, respectively. Then, the sampling rate σ and failure rate θ can be defined as n/d and m/d, respectively.

It should be noted that, the case of two-dimension input domain is utilized to describe our partitioning algorithm. For the two-dimension case, a region can be expressed via point P_{ll} and P_{ur}, where P_{ll} represents the lower-left point of the region and P_{ur} is the upper-right point. For each point P, it can be denoted by X and Y coordinates, i.e.

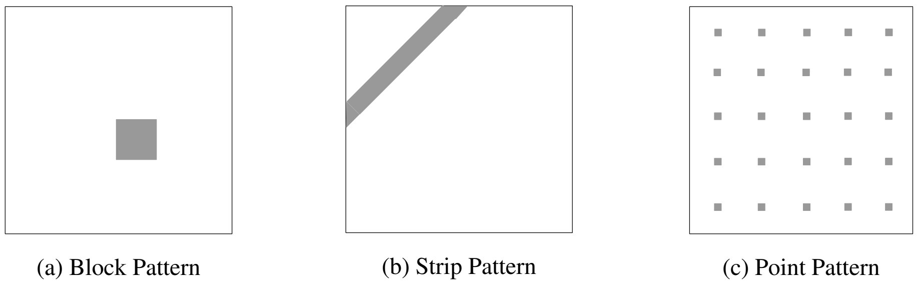


Figure 1: Block, strip, and point failure patterns in a two-dimensional input domain. Here, the shaded areas represent failure-causing inputs.

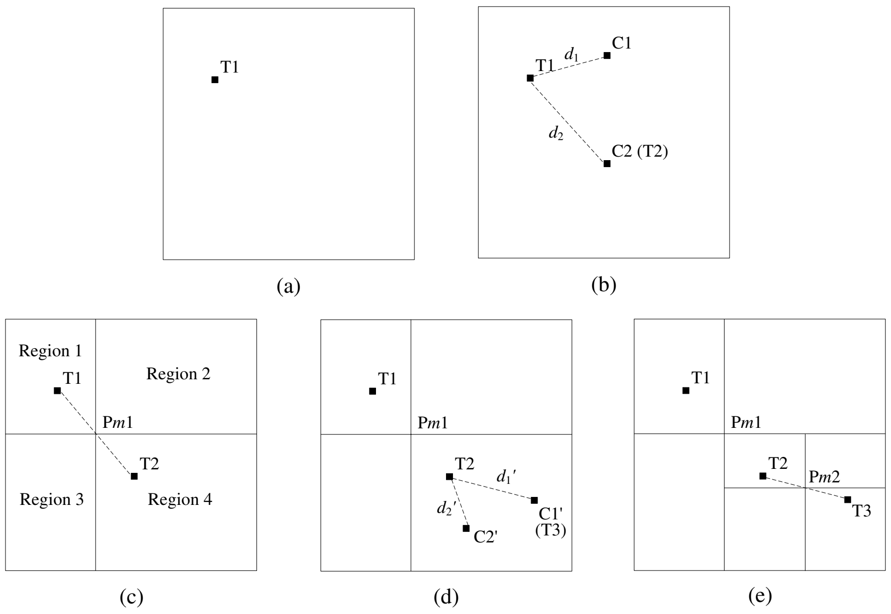


Figure 4: The illustration for the algorithm of two-point partitioning strategy.

P=(x, y). During the process for generating test inputs, the set of test data can be denoted as S_T=$\{T_i|1 \leq i \leq n\}$, and the candidate set of random points is denoted as S_C=$\{C_r|1 \leq r \leq k\}$, where k is the pre-specified size of test input candidate set.

Here, we also use the number of test cases required to detect the first failure (referred to as the F-measure) as the effectiveness metric. For a test data set S_T=$\{T1, T2, \cdots, Tn\}$, if the corresponding F-measure is equal to u $(1 \leq u \leq n)$, which means that the test input between 1 to $u-1$ can't reveal faults but the u-th can find them. Formally, it can be denoted as F=u. With regarding to the traditional random testing, in theory, its F-measure is equal to d/m. When comparing two test data generation methods, the lower value of F-measure means the higher effectiveness, because the low F-measure means few test inputs which are required to reveal the first fault.

3 Two-point Partitioning Algorithm

In the traditional partitioning strategies, it is possible that the two sampled test inputs are very close with each other. Here, we propose a new test data generation strategy based on two points partitioning. The basic process is illustrated in Figure 4, and the steps can be stated as below.

(1) Add the whole input domain into the region list L, and set S_T=Ø.

(2) Select the max-area region from L, denote it as *curReg*, and remove it from L.

(3) If there are no previous test inputs in *curReg*, a new input point should be randomly generated in this region, then add it into S_T. Otherwise, go to step (4).

(4) Suppose the existing test input in *curReg* denoted as T_i, randomly generate k candidate points in *curReg*. Calculate the distance from T_i to each candidate

Algorithm ART-TPP

```
Input: The boundary point of input domain bndP[2], where bndP[0] represents the lower-left point of region and bndP[1] is the
    upper-right point.
Output: The set of test data $S_T$={$T1, T2, \cdots, Tn$}.
    Stage 1: Initialization
 1: regionList = Ø; //regionList is a list of regions
 2: set the region (bndP[0], bndP[1]) as curReg; //curReg represents the current region needed to be partitioned
 3: tempP = generateRandPoint(curReg.ll, curReg.ur);
 4: $S_T = S_T \bigcup \{tempP\}$;
 5: add curReg into regionList;
    Stage 2: Test Data Generation
 6: while true do
 7:     pIndex = findMaxRegion(regionList); //find the max-area region in regionList, and pIndex is the index of region needed to
        be partitioned
 8:     curReg = regionList.get(pIndex);
 9:     regionList.remove(pIndex); //remove the max-area region from regionList
10:     if curReg doesn't contain an existing test input then
11:         generate a new test input using function generateRandPoint(curReg.ll, curReg.ur), denote it as P1;
12:         $S_T = S_T \bigcup \{P1\}$;
13:         if P1 hits the failure-causing region then
14:             break;
15:         end if
16:         randomly generate $k$ candidate points in curReg, and store them in $S_C$; //$S_C$ is the test input candidate set
17:         select the point from $S_C$ which is the farthest from P1 as the second test input P2;
18:     else
19:         P1 = the existing test input in curReg;
20:         randomly generate $k$ candidate points in curReg, and store them in $S_C$;
21:         select the point from $S_C$ which is the farthest from P1 as the second test input P2;
22:     end if
23:     $S_T = S_T \bigcup \{P2\}$;
24:     if P2 hits the failure-causing region then
25:         break;
26:     end if
27:     calculate the midpoint of P1 and P2, divide curReg into four new sub-regions via this midpoint, and then add them into region-
        List;
28:     locate P1 and P2 to their corresponding sub-regions;
29: end while
30: return $S_T$;
```

point, and select the point which is farthest from T_i as the second test input (denoted as T_{i+1}) in *curReg*, S_T=$S_T \bigcup \{T_{i+1}\}$.

(5) In the step (3) and (4), once a new test input is generated and added into S_T, we should validate whether it can hit the failure-causing region. If true, terminate the process and output S_T. Otherwise, continue the process to append other test inputs.

(6) Compute the midpoint of the corresponding points of T_i and T_{i+1}, denoted as P_{mi}. Then, partition the current max-area region into four sub-regions via P_{mi}, add these sub-regions into L, go to step (2).

Formally, the adaptive random test data generation algorithm based on two-point partitioning can be described in the form of pseudo-code (cf. algorithm **ART-TPP**). It should be noted that, we are mainly concerned with the case of two dimension here. Accordingly, the high dimension case can be treated in the similar way. In the line 3 and 11 of algorithm pseudo-code, function `generateRandPoint`(*curReg*.ll, *curReg*.ur) can randomly generate a test input (or point) in the current region, where *curReg*.ll refers to the lower-left point of region, and *curReg*.ur is the upper-right point. *curReg* is a rectangle region object which contains two main member variables: lower-left point (ll) and upper-right point (ur). The function `findMaxRegion`(*regionList*) in line 7 returns the index of the max-area region in *regionList*. There are several kinds of operations can be invoked by object *regionList*, such as `get`(), `remove`() etc.

Here, we suppose the size of test data set is n. Obviously, the time complexity of function `findMaxRegion`() is $O(l)$, where l is the length of region list, and it varies from 0 to n. Therefore, the time complexity of whole algorithm is $O(n^2/2)$. In general, n is in the same magnitude of θ^{-1} and $n < \theta^{-1}$. Accordingly, the complexity can be expressed as $O(\theta^{-2}/2)$.

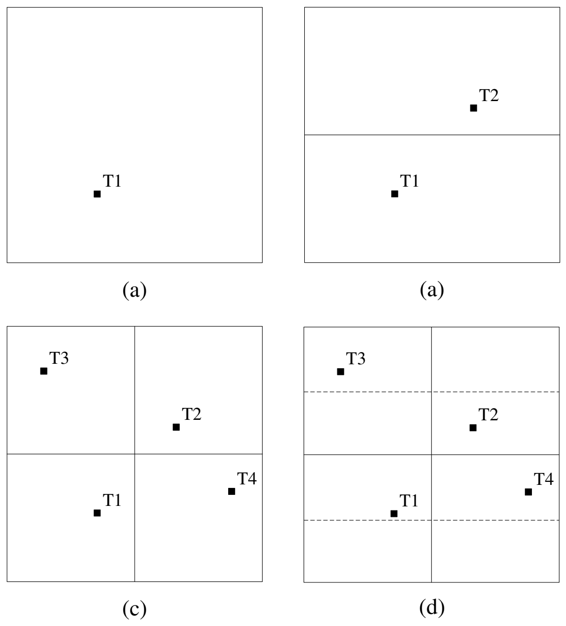


Figure 3: The illustration for bisection algorithm (2-*D* case).

4 Experimental Analysis

4.1 Analysis on failure patterns

In this section, we perform the simulation analysis on the case of two-dimension. The experiment is employed in the environment of Eclipse 3.6 and JRE 1.6.0_05. The program runs on a computer with Pentium IV 1.8 GHz CPU, 1 GB RAM and Windows XP SP2. In this sub-section, we want to investigate the following two questions.

RQ1: How effective is the ART-TPP algorithm for three types of failure patterns?

RQ2: Does the failure rate affect the F-measure ratio (F-ratio for short)?

F-measure ratio = $\frac{F_{ART}}{F_{RT}}$, where F_{ART} represents the F-measure value of ART method, and F_{RT} is the F-measure of the general RT method. According to the description in section 2.3, F_{RT} is equal to $1/\theta$.

As shown in Figure 5, we firstly analyze the F-measure ratios for 20 different random failure regions. For each region, we repeat 5000 runs to get the average value of F-ratio. In the figure, the square-marked curve represents the theoretical value of RT's F-ratio, the other three curves are ART's F-ratios for three kinds of failure patterns. It is not hard to find that, the location of failure region does not impose a great impact on F-ratio. The F-ratio of ART-TPP for block failure pattern is about 76.5%, but its F-ratios of strip and point failure patterns are very close to the theoretical value of RT. This phenomenon is in accordance with Chen's research results [6]. It means that the ART is more effective than the general random testing method, and especially for the block failure pattern.

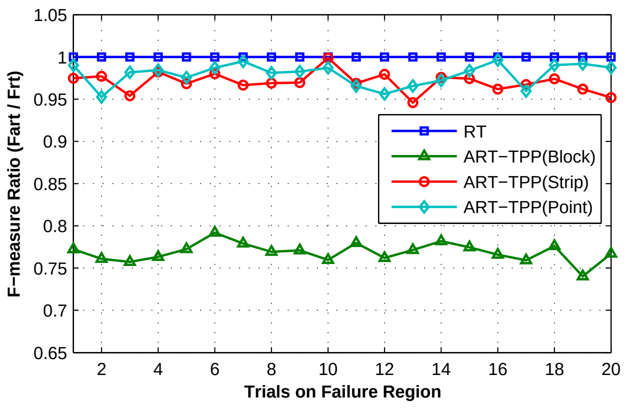


Figure 5: The F-measure ratios for different failure region locations.

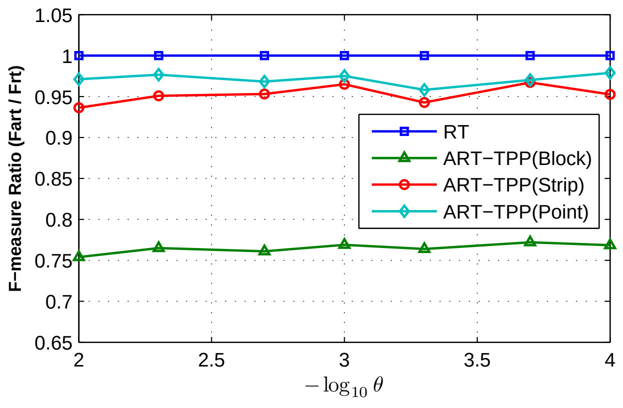


Figure 6: The F-measure ratio vs. failure rate θ.

Meanwhile, we also analyze the F-ratio through varying the failure rate θ from 0.01 to 0.0001 (see Figure 6). For the reason of computing time, when θ is equal to 0.01, 0.005, 0.002 and 0.001, we randomly select 100 locations of failure regions, and repeat 5000 runs for each location to get the average F-ratio. When θ is equal to 0.0005 and 0.0002, 20 random locations and 1000 runs are used for experiments. Moreover, we set 20 random locations and 200 runs when θ is 0.0001. As illustrated in Figure 6, the failure rate does not cause obvious fluctuation on F-ratio. When θ varies from 0.01 to 0.0001, F-ratio approximately keeps the value of 0.765 for block failure pattern, 0.95 for strip pattern and 0.97 for point pattern.

4.2 Comparison analysis

In this sub-section, we want to perform a comparison analysis and answer the question as below.

RQ3: Is our algorithm more effective than the existing two partition-based ART algorithms?

Failure rate	Block pattern			Strip pattern			Point pattern		
	ART-RP	ART-BP	ART-TPP	ART-RP	ART-BP	ART-TPP	ART-RP	ART-BP	ART-TPP
0.01	77.4%	72.1%	75.3%	92.8%	91.9%	93.6%	99.5%	97.7%	97.1%
0.005	77.4%	74.3%	76.5%	97.1%	95.5%	95.0%	99.6%	98.1%	97.9%
0.002	78.0%	73.6%	76.0%	95.3%	95.1%	95.3%	99.7%	98.7%	96.8%
0.001	79.6%	74.1%	76.9%	96.5%	96.6%	96.5%	98.0%	96.8%	97.5%

Table 1: Comparison analysis between ART-TPP and other two ART algorithms

In the experiment, our algorithm is run for different failure rates and different failure region types, the results are shown in Table 1. In the same way, we randomly select 100 locations of failure regions, and repeat 5000 runs for each location to get the average F-ratio. The results of other two algorithms are referred from [6]. Here, we compare the F-ratios of three algorithms as follows.

For the block failure pattern, the effect of ART-TPP algorithm is better than ART-RP but worse than ART-BP for all the values of θ. On average, the F-ratio of our ART-TPP algorithm is lower than that of ART-RP about 2 percent, but greater than that of ART-BP about 2.6 percent.

For the strip failure pattern, the results of ART-TPP are equal to those of other two algorithms on the whole. When θ=0.01, the performance of ART-BP is the best, while ART-TPP is on the worst performance. When θ=0.005, the F-ratio of ART-TPP is the lowest one, and ART-RP acts the worst performance. For the rest values of failure rate (θ), the difference of three algorithms is basically within 0.2 percentage points. The results indicate that, for the strip failure pattern, ART-TPP is quite good in the case of low fault density, but the performance is not good in the case of high fault density.

For the point failure pattern, the F-ratio of ART-TPP is lower and more stable than other two ART methods. When θ is from 0.01 to 0.002, the F-ratio of ART-TPP is always the lowest one in the three algorithms. When θ is equal to 0.001, ART-TPP's F-ratio is higher than that of ART-BP, but lower than that of ART-RP. It is worth noting that the fluctuations of the F-ratios of ART-TPP is the least of three algorithms

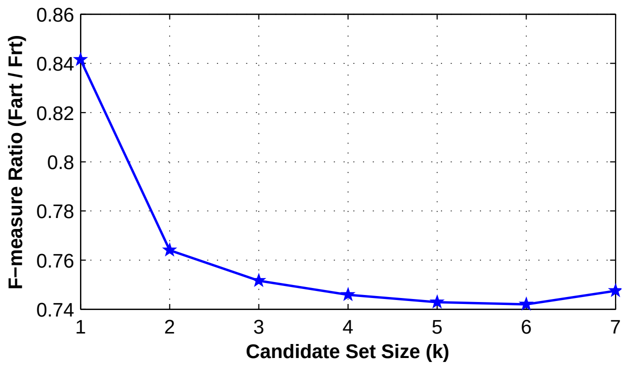
Based on the above analysis, we can argue that our ART-TPP algorithm has a positive improvement over the existing two well-known ART algorithms, especially for the random partitioning algorithm (ART-RP).

4.3 Sensitivity analysis

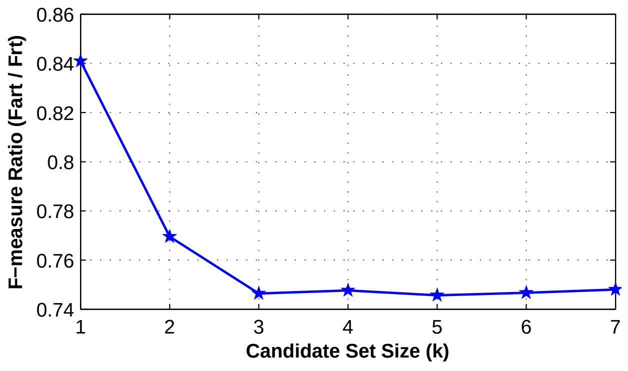
In the above experiments, we fix the candidate set size to 2. In fact, k acts as a parameter of our algorithm, so we have to answer the following question.

RQ4: Does k play an important role in algorithm effectiveness? And which k value is the appropriate choice?

In order to answer this question, we take the block failure pattern as an example to analyze the impact of candidate set size. The results of θ=0.001 and θ=0.0005 are shown in Figure 7(a) and 7(b), respectively. We can find that, the F-ratio value greatly decreases when k is from 1 to 2, i.e. from 0.84 to 0.767. The F-ratio value decreases 0.017 when k is from 2 to 3. When k is greater than 3, the F-ratio has no obvious change and keeps the value on 0.745. Therefore, k=3 is the best value for the size of candidate set, and k=2 is also a suitable choice while considering the computing cost. More importantly, the F-ratio will sharply increase if we choose the second point without candidate selection (i.e. k=1).



(a) θ=0.001



(b) θ=0.0005

Figure 7: The F-measure ratio vs. candidate set size k.

5 Conclusion

How to generate the test data with high fault-revealing capability is a critical problem in the field of software testing. Random testing has been widely adopted in automated testing tools due to its advantages such as simpleness, easy realization and low cost. Unfortunately, this method usually reveals the potential faults with the large amount of test inputs, so its cost-benefit is not very good. Chen *et al.* proposed an improved strategy named *adaptive random testing* to overcome this shortage. In the paper, we are mainly concerned with the partitioning-based ART. A new algorithm based on two-point partitioning (i.e. ART-TPP) is presented here. In our algorithm, we also select the current max-area region as partition object, but the region is partitioned at the midpoint of two points. At first, we randomly generate a test point in the region. Then, the second point is picked out from a candidate set according to the farthest distance criterion. The partition can be iteratively performed until the potential faults are found or the size of test data set reaches the pre-set limit. In order to validate the effectiveness of ART-TPP algorithm, we compare it with the other two well-known algorithms: random partitioning (ART-RP) and bisection partitioning (ART-BP). The experimental results show that ART-TPP is better than ART-RP but worse than ART-BP in the case of block failure pattern. For the strip failure pattern, the three algorithms have no obvious difference. While considering the point failure pattern, ART-TPP algorithm is better and more stable than other two algorithms.

Of course, there are still some open research issues that need to explored in the next step. For example, we can continue to conduct comparative analysis of the three algorithms in high-dimensional case. Meanwhile, we are planning to use some real-world programs to analyze the effect of our ART-TPP algorithm.

Acknowledgement

This work was supported in part by the National Natural Science Foundation of China (NSFC) under Grant No. 60803046 and 61063013, the Natural Science Foundation of Jiangxi Province under Grant No. 2010GZS0044, the Science Foundation of Jiangxi Educational Committee under Grant No. GJJ10433, the Open Foundation of State Key Laboratory of Software Engineering under Grant No. SKLSE2010-08-23, and the Program for Outstanding Young Academic Talent in Jiangxi University of Finance and Economics.

References

[1] National Institute of Standards and Technology, (2002). *The Economic Impacts of Inadequate Infrastructure for Software Testing*, Planning Report 02-3.

[2] N. Tracey, J. Clark, K. Mander, and J. McDermid, (1998). An Automated Framework for Structural Test-Data Generation, *Proc. of the 13th Int'l Conference on Automated Software Engineering (ASE'98)*, IEEE CS Press, Honolulu, Hawaii, USA, pp. 285–288.

[3] K. Ayari, S. Bouktif, and G. Antoniol, (2007). Automatic Mutation Test Input Data Generation via Ant Colony, *Proc. of the 9th Annual Conference on Genetic and Evolutionary Computation (GECCO'07)*, ACM Press, London, England, UK, pp. 1074–1081.

[4] R. P. Pargas, M. J. Harrold, and R. Peck, (1999). Test-Data Generation Using Genetic Algorithms, *Software Testing, Verification and Reliability*, vol. 9, no. 4, pp. 263–282.

[5] T. Y. Chen, F.-C. Kuo, R. G. Merkel, and T. H. Tse, (2010). Adaptive Random Testing: The ART of Test Case Diversity, *The Journal of Systems and Software*, vol. 83, pp. 60–66.

[6] T. Y. Chen, R. G. Merkel, G. Eddy, and P. K. Wong, (2004). Adaptive Random Testing Through Dynamic Partitioning, *Proc. of the 4th Int'l Conference on Quality Software (QSIC'04)*, IEEE CS Press, Braunschweig, Germany, pp. 79–86.

[7] T. Y. Chen, F.-C. Kuo, and H. Liu, (2007). Distribution Metric Driven Adaptive Random Testing, *Proc. of the 7th Int'l Conference on Quality Software (QSIC'07)*, IEEE CS Press, Portland, Oregon, USA, pp. 274–279.

[8] C. Schneckenburger and J. Mayer, (2007). Towards the Determination of Typical Failure Patterns, *Proc. of the 4th Int'l Workshop on Software Quality Assurance in conjunction with ESEC/FSE'07*, ACM Press, Dubrovnik, Croatia, pp. 90–93.

[9] E. Allen, (2002). *Bug Patterns in Java (2nd Edition)*, Apress.

[10] I. Ciupa, A. Leitner, M. Oriol, and B. Meyer, (2008). ARTOO: Adaptive Random Testing for Object-Oriented Software, *Proc. of the 30th Int'l Conference on Software Engineering (ICSE'08)*, ACM Press, Leipzig, Germany, pp. 71–80.

Optimal Motion Paths in Ambient Fields

Thomas Kämpke
InMach Intelligente Maschinen GmbH
Kässbohrerstr. 19, 89077 Ulm, Germany
E-mail: kaempke@inmach.de, www.inmach.de

Keywords: autonomous mobile platforms, graph algorithms, energy minimization

Received: July 3, 2012

The motion of water or the atmosphere that surrounds or supports a mobile platform affects the motion of the platform in a favorable or unfavorable way. Exploiting and compensating the ambient field is investigated for different motion objectives like the classical point to point motion, observation motion, exploration motion and so-called survival motion. Time and energy optimal trajectories in the presence of fields are formalized and shown to be computable by means of discrete optimization.

Povzetek: Analizirano je gibanje mobilne platforme v okolju.

1 Introduction

Mobile platforms that float, dive or fly are affected by the motion of their ambient fields. These fields may be wind, oceanic streams like the Gulf and the Humboldt stream, tidal or diurnal motions, currents and arbitrary combinations thereof. The fields may be stationary or nonstationary.

Platforms considered here include autonomous sailing ships, commercial vessels that receive additional propulsion from airfoils such as skysails [14], (gliding) planes and aerobots designed for the Earth or for a foreign planet with an atmosphere such Venus and Mars as well as Titan (moon of Saturn) and Europe (moon of Jupiter). The motion of a platform is abstracted here from almost all kinematic and dynamic constraints, so that neither inertia nor restrictions on the control variables like bounds on turning angles apply. This allows, in particular, to ignore the motion history for computing any motion continuation. Uncontrolled motion (drift) is considered separately from motion with controls.

Key issues for trajectories without control are reachability of a destination point by pure drift and the computability of the distance of pure drift towards a destination point. Key issues for trajectories with controls are computations of minimum time and minimum energy trajectories towards destination points, optimal trajectories according to the two foregoing issues towards a destination region, minimum energy trajectories towards a destination point to meet a given due date, optimal orbits to continuously observe a stationary point of interest, optimal trajectories to explore a given region of interest and longest survival orbits allowing a flying platform to stay airborne with a fixed energy budget for a maximum time.

Horizontal motion will be considered first, followed by straightforward as well as by a particular extension to three dimensions. The field is assumed to be known throughout. Thus, no uncertainty of the field is admitted, neither in strength nor in direction. This amounts to deterministic path planning rather than handling the unexpected like motion adaptation for collision avoidance [15] or even managing completely unforeseeable environmental changes. Also, selflocalization, which is the determination of the position platform in some reference frame, is not part of this work.

Time and energy minimization of trajectories will be obtained by discrete optimization methods, in particular by graph algorithms. The majority of the motion objectives which are introduced below, go beyond point to point motion and use this well-known problem as starter. Optimal trajectories for two-dimensional motion and some extensions to the 3D case of homogenous, vertically stacked wind layers were investigated in [5]. Optimal control of hot-air balloons is considered for so-called linear wind fields in [3]. A 3D wind field is linear if it changes linearly with the position. For large scales, this model does not apply for two reasons. First, the wind speed does typically not grow linearly in the vertical component. Second, the Coriolis force, terrain effects and else cause a wind field to be curved and even be locally turbulent.

A sophisticated analysis of continuous 2D flows and potentially emerging cyclic structures is given in [13]. The cyclic structures allow orbits, but no platform motion is considered. Platforms may execute controls that exploit the fields but they may even oppose the field, at least along part of the trajectory. The case of exploiting the wind field for sailing ships has been covered by several references including [1].

The remainder of this paper is organized as follows. Sections 2, 3 and 4 cover problems in two dimensions at increasing complexity levels. The main distinction is that between drift and deliberate motion. Section 5 deals with point to point motion under different and joint objectives. Section 6 considers motion for continuous observation of a point location. Section 7 deals with exploration of an area. Section 8 extends approaches to three dimensions and considers a so-called survival problem in systems of updraft and downdraft areas.

2 Basic Model

The platform position at any moment is a 2D point $P(t) = (x(t), y(t))^T$ with the superscript T denoting transposition. Start and destination points of a motion are abbreviated by $P_S = (x_S, y_S)^T$ and $P_D = (x_D, y_D)^T$. The scale of considerations is chosen so (large) that platform orientation does not matter. Positions at future moments resulting only from drift by the ambient fields are given by the forward equation $P(t + \Delta t) = P(t) + \Delta t \cdot w(P(t), t) + \Delta t \cdot c(P(t), t)$. The effects of all ambient fields are linearized over time.

Time increments Δt range from a fraction of a second to many minutes and even to six hours for vessel routing [12]. The wind field w, the current field c and possible further fields are dealt with in the same way. Fields are vector fields $Dom \rightarrow I\!R^2$ in the stationary case and vector fields $Dom \times T \rightarrow I\!R^2$ in the nonstationary case with $Dom \subseteq I\!R^2$ in both cases. T is the index set for time with typical setting $T \subseteq [0, \infty)$. The drift caused by a vector field may eventually lead to positions where the field is not defined. This view circumvents the specification of domain bounds. Fields are assumed to generously cover considered regions so that boundary problems are practically irrelevant.

Fields can be specified discretely in space and, if applicable, in time with hyperbolic spatial interpolation. This means that the field at some point P in the convex hull of, say, four support points $P_1, \ldots, P_4$ is

$$f(P) = \sum_{i=1}^{4} \frac{\frac{1}{\|P-P_i\|}}{\frac{1}{\|P-P_1\|} + \ldots + \frac{1}{\|P-P_4\|}} f(P_i)$$

in the stationary case and

$$f(P,t) = \sum_{i=1}^{4} \frac{\frac{1}{\|P-P_i\|}}{\frac{1}{\|P-P_1\|} + \ldots + \frac{1}{\|P-P_4\|}} f(P_i, t)$$

in the nonstationary case. The effect of one support point on a point of interest is inversely proportional to its relative distance from that support point. A feature of this interpolation is that it is also operational if the point of interest lies to the outside of the convex hull of the support points. Discontinuities may occur when the set of considered support points changes along a trajectory but this applies to other interpolation schemes as well. Interpolation in time is always linear between two adjacent support moments $t_1 < t_2$. For any moment t between the two support moments, time interpolation is

$$\begin{aligned} f(P,t) &= \frac{t_2 - t}{t_2 - t_1} f(P, t_1) + \frac{t - t_1}{t_2 - t_1} f(P, t_2) \\ &= \frac{\frac{1}{t-t_1}}{\frac{1}{t-t_1} + \frac{1}{t_2-t}} f(P, t_1) + \frac{\frac{1}{t_2-t}}{\frac{1}{t-t_1} + \frac{1}{t_2-t}} f(P, t_2). \end{aligned}$$

3 Motion Without Control

Natural fields may be curved so that a drifting platform may come closer to a destination point, then increase its distance and repeat this behavior. A field with two distance minima is sketched in figure 1. A refined analysis reveals that a second approach towards the destination point can occur even if the cumulative angular change along all drift trajectories is less than 90^o. The existence of more complicated behaviors cannot be excluded completely but is ignored here. Yet fields in which optimal trajectories may spiral have been investigated [9]. The closest approximation to a destination point is computable by tracing the platform drift until distances begin to increase for the second time or until a boundary of the field is reached; whatever is first. Passing through the destination point amounts to shortest distance of value zero. The shortest distance between the destination and a drift trajectory can be denoted as drift distance. This is not a regular distance, because it is not even symmetric. The drift distance may serve as guideline for trajectory computations since it is a shortest segment along which the platform requires to invoke propulsion.

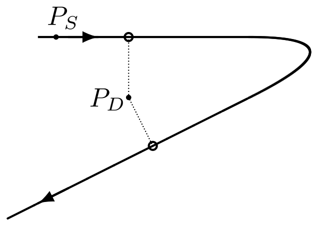


Figure 1: Drift trajectory with two local distance minima (white points) to the destination point.

4 Control Model

4.1 Structure

The position update for motion with control is $P(t+\Delta t) = P(t)+\Delta t \cdot f(P(t))+\Delta t \cdot u(t)$ for a stationary field $f(\cdot)$ and $P(t + \Delta t) = P(t) + \Delta t \cdot f(P(t), t) + \Delta t \cdot u(t)$ for a nonstationary field $f(\cdot, \cdot)$. $u(t)$ is the control vector executed at time t and held fixed over the time increment. Control vectors specify heading and speed relative to the field and, depending on the type of platform, may head into the opposite direction of the field. The effect of the field along the transition during the time increment is, so far, attributed only to the emanation point of the transition. Since the field typically varies along the transition, its effect should be inferred from the field along the transition or, at least, from the first and the last point of the transition. Thus, for $P = P(t)$ and $Q = P(t + \Delta t)$ the effect of the field is

$$\frac{f(P) + f(Q)}{2} \text{ or } \frac{f(P,t) + f(Q, t + \Delta t)}{2}.$$

In the stationary case, the control vector for reaching Q after the time increment is

$$u(t) = \frac{P(t+\Delta t) - P(t)}{\Delta t} - \frac{f(P) + f(Q)}{2};$$

the formula for nonstationary fields is analogous. Controls and fields need not superimpose linearly. The execution of control vectors entails cost that are integrated over time until the destination point is reached. Following standard control approaches with some performance function $cost : I\!R^2 \to I\!R_{\geq}$, the total cost is

$$C = \int_{t_S}^{t_D} cost(u(t))\, dt.$$

The starting time is assumed to be known while the time of arrival in the destination point depends on the motion which depends on the sequence of controls. Prominent examples of cost functions are $cost(u(t)) = 1$ with total cost accounting for the time and $cost(u(t)) = power(u(t))$ with total cost accounting for the energy spent until the destination point is reached. The function $power(\cdot)$ encodes the physical energy spent per time to execute the control. Executing no control incurs no cost. The aim of finding an optimal motion is a minimization problem over the set of feasible controls $\mathcal{U}$

$$\min_{u(t)\in\mathcal{U}} \int_{t_S}^{t_D} cost(u(t))\, dt.$$

The minimization may be endowed with a further constraint so that trajectories meet a due date

$$\begin{array}{ll} \min_{u(t)\in\mathcal{U}} & \displaystyle\int_{t_S}^{t_D} cost(u(t))\, dt \\ \text{such that} & t_D \leq t_{due}. \end{array}$$

Trajectory optimization in its most general form, thus, is a variational problem so that the ultimate method is solving the Euler-Lagrange equation (computing the "zero" of the derivative). But the difficulties of finding an exact solution are enormous as illustrated, for example, by fields that force a mobile platform to make abrupt turns [10]. Even the most simple problem version is not trivial when the cost function is not trivial. This is illustrated for the field being zero everywhere. The cost minimal trajectory then connects the starting point to the destination point by a straight line leaving the optimal transition time to be computed as a univariate minimization problem. The controls are set to

$$u(t_S) = \frac{P_D - P_S}{t_D - t_S}$$

with the arrival time in the destination point being in variation. The energy spent along the trajectory is

$$\begin{aligned} C &= \int_{t_S}^{t_D} cost(\frac{P_D - P_S}{t_D - t_S})\, dt \\ &= (t_D - t_S) \cdot cost(\frac{P_D - P_S}{t_D - t_S}). \end{aligned}$$

The univariate minimization problem with variable t_D thus tentatively becomes

$$\min_{t_D:\, t_D > t_S} (t_D - t_S) \cdot cost(\frac{P_D - P_S}{t_D - t_S}).$$

Transition times between distinct positions cannot be arbitrarily small. If the minimum transition time from P_S to P_D admitted by the controls is $t_{min} > 0$, then the final optimization problem becomes

$$\min_{t_D:\, t_D \geq t_S + t_{min}} (t_D - t_S) \cdot cost(\frac{P_D - P_S}{t_D - t_S}).$$

The solution strongly depends on the cost function. Minimum energy solutions may have the "single crossing property" which describes a certain uniformity of the field in relation to a given trajectory and which is stated here without formal proof. This property claims that an energy minimal trajectory crosses every drift trajectory at most once, if all vectors of the ambient field on every normal of the energy minimal trajectory point into the halfspace which contains the destination point. This property is illustrated by figure 2.

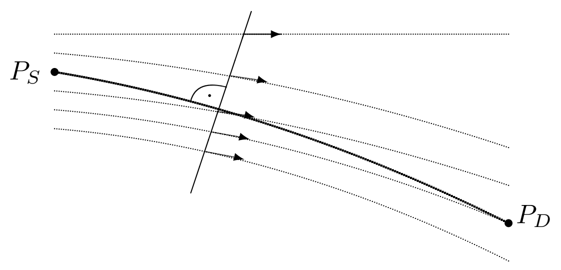


Figure 2: Energy optimal trajectory with one normal and five drift trajectories shown. The field at the intersections of the normal with all five trajectories points into the same ("lower right") halfspace of the normal as is supposed for all non-visualized drift trajectories. The destination point lies in that halfspace.

The single crossing property is not a necessary optimality criterion as it may or may not hold in heavily turning fields such as in figure 1. Also, the property does not hold if the destination point lies too far upstream. The energy minimal trajectory may then have to slalom around regions of strongly opposing drift thus crossing at least one drift trajectory more than once, see figure 3.

4.2 Computational preliminaries

Computations of optimal trajectories follow two principal approaches. The first is put a grid into the region of interest and compute motion for grid points only. The spatial resolution of the grid can be selected independently from the spatial resolution of the fields. An interesting, non-uniform grid has been derived for long distance vessel routing [7].

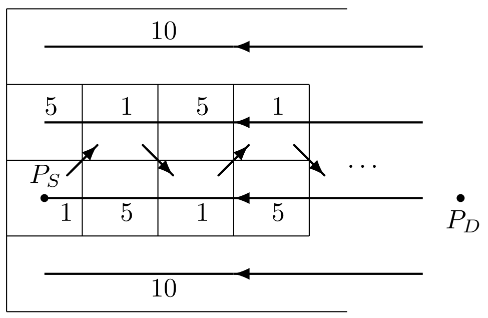


Figure 3: Numbers and arrows indicate field velocity and direction in the rectangles. An energy minimal trajectory of a platform with suitable characteristics will multiply cross drift trajectories (slanted arrows).

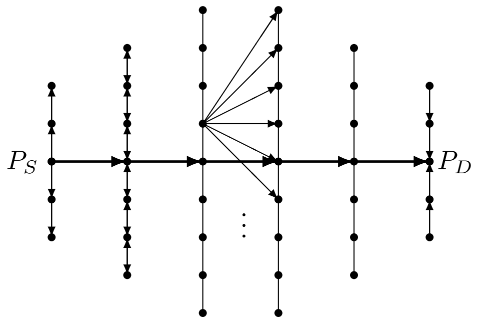


Figure 4: Static waypoint graph with herringbone pattern. Arcs reach from all points of a bone to all points of the next bone until the bone with destination point is reached. Only five of the nine arcs reaching from one node to the subsequent bone are indicated. Also, arcs within a bone are only indicated for three bones.

The nodes are placed along the great circle between start and the destination point and perpendicular to the great circle up to a specified width. Alternatively, the motion is free meaning that waypoints are not set before computations. This is the dynamic graph approach and dynamically placed nodes have been proposed for motion planning by splines [4].

Static and dynamic graphs are directed and denoted as waypoint graphs. Static waypoint graphs allow computations by standard graph algorithms like the Dijkstra algorithm and dynamic programming. The disadvantage is that an optimal path must, typically, be smoothed to become a real-world path. Dynamic waypoint graphs allow to place nodes in control-compliant positions. But then, computations may suffer from the number of nodes growing at an unmanageable rate which puts heavy burden on elimination techniques.

The static graph is the concept of choice here. A static waypoint graph may be defined universally for a region or it may be defined specifically for the starting and destination points as in figures 4 and 11. Several points may lie at different distances in the same direction from some waypoint P_0 which is not intuitive for neighbors. This phenomenon is excluded by requiring that only the closest point in a given direction belongs to any neighbor set [17]. Moreover, far points are excluded from that set if the direction of the field deviates beyond a threshold angle from the direction in P_0. The set of all waypoints is denoted by V.

Arc labels for waypoint graphs depends on characteristics of the platform. When the objective is time, velocity predictions and interpolations from so-called polar plots may apply [1]. Computations of energy values are illustrated by concrete data and figure 5. The distance from A to B is 10 nautical miles with current setting in orthogonal direction at 3 knots (nautical miles per hour). When traveling at an apparent speed of 5 (8) knots, fuel consumption is 10 (21) liters per hour. Only the two speeds are allowed. In the first case, ground speed towards B is 4 knots, so that the transition time from A to B is $10nm/(4nm/h) = 2.5h$ and fuel consumption is $10l/h \cdot 2.5h = 25l$. In the second case, ground speed towards B is $\sqrt{55} = 7.416$ knots, so that the transition time from A to B is $10nm/(7.416nm/h) = 1.348h$ and fuel consumption is $21l/h \cdot 1.348h = 28.31l$. The better of the two options results in the energy label 25 for the arc from A to B.

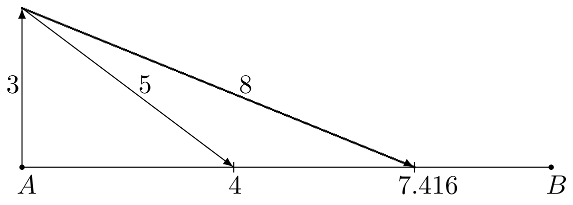


Figure 5: Vector additions for sample energy computations.

Transition costs along an arc (A, B) are specified by a label $c(A, B)$ for stationary fields and by parameterized labels $c(A, B; t)$ for nonstationary fields with t denoting the moment when the platform leaves A and heads for B. Two waypoints may or may not be connected by opposing arcs. If so, their labels may be different.

5 Point Motion with Controls

5.1 Single objective for point to point motion

Paths will preferably be computed by the Dijkstra algorithm and modifications thereof instead by dynamic programming. Though optimal paths satisfy the dynamic principle [2], the Dijkstra algorithm proceeds along forward computations while standard dynamic programming applies backward computations. Backward computations are not intuitive for nonstationary fields; the field is more likely to be known at starting conditions than at terminating conditions since the latter, among others, depend on the computed trajectory.

The Dijkstra algorithm in standard form [6] suffices for shortest path computations in stationary fields. For nonstationary fields the algorithm is formulated for transitions along arcs without idleness. This means that the platform will never wait for more opportunistic conditions of the field; the arrival time in an intermediate waypoint is identical to the departure time in that waypoint. The departure time from the starting node is $t_{P_S} \geq 0$ and arrival times at nodes are specified by the labels $m(\cdot)$.

Dt (Dijkstra algorithm for nonstationary fields)

```
1. (Initialization). Set $L = V$, $m(l) = \infty\ \forall l \in V - \{P_S\}$, and $m(P_S) = t_{P_S}$.
2. (Iteration). While $P_D \in L$ do:
    (a) If $min_{l \in L}\, m(l) < \infty$ then selection of $i = argmin_{l \in L}\, m(l)$, else output "Destination not reachable" and stop.
    (b) $L = L - \{i\}$.
    (c) $\forall j \in L$ with $(i, j) \in A$ do:
        if $m(i) + c(i, j; m(i)) < m(j)$, then $m(j) = m(i) + c(i, j; m(i))$ and $pred(j) = i$.
3. (Termination). Output $m(P_D)$ and $P_D, pred(P_D), pred(pred(P_D)), \ldots, pred(\ldots(P_D)\ldots) = P_S$.
```

An optimal path from start to destination results from tracing the waypoints which attain minima of the node labels. These labels denote the cost of the best path found so far from the starting point. The list L consists of the waypoints to which an optimal path has not yet been found or has not yet been confirmed to be found. These waypoints are tentatively labeled, while all others are permanently labeled. Preceding vertices are stored in the $pred(\cdot)$ function so that the waypoints of an optimal motion are specified reversely.

Replacing the time dependent labels $c(i, j; t)$ by time independent labels $c(i, j)$ results in the ordinary Dijkstra algorithm. The Dijkstra algorithm for both label kinds is a one-to-many algorithm. This means that the algorithm may find optimal paths from the initial node to several nodes as discussed next.

5.2 Single objective for point to region motion

Computations of optimal paths from the starting point to all other waypoints can be facilitated by a single run of algorithm **Dt** by modifying the stopping criterion as to the list L being empty. Optimal paths to each node of a selected region of waypoints are computed when the list is cleared of all target nodes. A variation of the last problem is to reach only one waypoint in a given region, namely the one which is reachable at minimal cost. This point need not be known prior to the computations.

An interesting case is that of the region consisting of all waypoints from which the destination point is observable. The computed motion is then one from which the destination point comes earliest "in sight", see figure 6. If observability solely depends on the Euclidean distance to the destination point, the optimization problem is formalized as

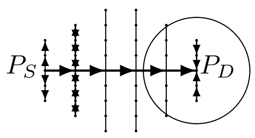


Figure 6: Sketch of waypoint graph from figure 4 and a circle around the destination point from which the destination point is observable.

$$\min_{i \in V:\, \|i - P_D\| \leq d} m(i);$$

$m(i)$ is the permanent label assigned to waypoint i by algorithm **Dt** and the maximum observation distance $d > 0$ must be specified as input. The region from which to reach a waypoint is the observability region $O_D(d) = \{i \in V : \|i - P_D\| \leq d\}$. The cost function for reaching the region of observability may differ from that for motion within that region. For example, it may be appropriate to reach the region of observability in minimal time but then to finally approach the destination with minimum energy.

5.3 Multiple objectives for point to point motion

Energy minimization subject to a due date can be handled as a two-criteria minimization problem. Such problems are solvable by a variation of the Dijkstra algorithm which operates on sets of labels for arcs and nodes instead of single labels. For the sake of simplicity, we consider only finite sets of arc labels. The "classical" multicriteria shortest path problem with one vector-valued label per arc already is NP-hard. But, occasionally, it is considered as one of the least intractable problems [16].

The arc (i, j) receives paired labels of the form $(d(i, j; t), e(i, j; t))$. The value $d(i, j; t)$ is a feasible duration of the transition along the arc while the transition begins at time t. The complete energy required therefore is $e(i, j; t)$. Longer durations correspond to smaller energy values. Practically, very long durations may again lead to larger energy values as engines consume fuel even if the platform does not move or move at extremely low speed. Such combinations of time and energy are not considered for the purpose of optimization. Thus, the labels of each arc form a Pareto optimal set which means that an improvement in one coordinate can only be achieved by a deterioration of the other. All feasible labels are arranged in the list $\Lambda(i,j;t)$ with time dependency disappearing in the stationary case. An example is $\Lambda(2,7) = \{(10,95),(11,91),(13,80)\}$. The transition from waypoint 2 to waypoint 7 may take 10 time units requiring 95 energy units or 11 time units requiring 91 energy units etc. For convenience all these lists are sorted by increasing time.

A simple approximation of an energy minimal path with due date in a stationary field is to compute a shortest path and then to maintain the sequence of waypoints while successively allowing more time for arc transitions. These relaxations, which conserve energy, terminate if no further prolongation is admitted.

Relaxations are selected along maximal energy savings per additional transition time. Therefore, $\Delta t(t(i,j)) = next(t(i,j)) - t(i,j)$ denotes the time increment for the duration $t(i,j)$ being replaced by the next value in $\Lambda(i,j)$. Similarly, the increment of saved energy is denoted as $\Delta e(e(i,j)) = e(i,j) - next(e(i,j))$. Both increments are positive due to sorting. Each relaxation then picks an arc attaining the maximum of the savings ratio provided that the due date is still met. Formally, a relaxation amounts to

$$\max_{(i,j)\in\mathcal{P}}\{\frac{\Delta e(e(i,j))}{\Delta t(t(i,j))} \mid t_D + \Delta t(t(i,j)) \le t_{due}\},$$

where $\mathcal{P}$ is an arc-wise specified path from P_S to P_D. The arrival time at the destination point is updated after each relaxation by $t_D^{new} = t_D + \Delta t(t(i,j))$ and $t_D \leftarrow t_D^{new}$. The foregoing procedure is of a greedy type and, therefore, need not find the energy minimum even along the given path.

An exact algorithm for energy minimization under due dates requires a quite extensive modification of the Dijkstra algorithm. First, nodes will receive sets of labels in the same way as arcs have received sets of labels prior to the start of the algorithm. Second, sets of labels will be propagated. Yet the strongest modification is concerned with permanence declarations of node labels. It is not correct – in a straightforward adaptation of the original Dijkstra method – to declare that tentatively labeled node as the next one permanently labeled which carries the minimum energy value. This wrongful propagation is illustrated in figure 7. It calls for decoupling of label propagation and permanence declarations. For stationary fields, label propagation along one arc (i,j) proceeds along the following three steps which may have to be executed multiply.

PL (Propagation of Labels)

1. $L'(j) = L(i) \oplus \Lambda(i,j) = \{((d_1,e_1) + (d_2,e_2), i) | (d_1,e_1) \in L(i)$ and $(d_2,e_2) \in \Lambda(i,j)\}$.
2. $L''(j) = L(j) \cup L'(j)$.
3. $L(j) =$ Pareto boundary$(L''(j))$.

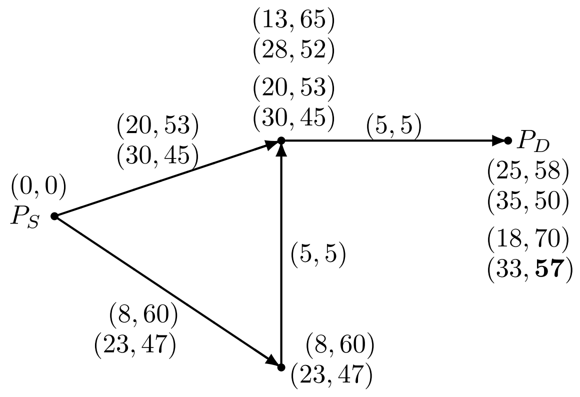


Figure 7: Initially, any reasonable computing scheme will label both successors of the starting point with the label sets of the two emanating arcs. The minimum energy value over the four pairs is 45. So, selecting the upper intermediate waypoint as the next one permanently labeled – by an oversimplified propagation scheme – results in only two label pairs for the destination point; one with arrival time 25 and another with 35. For due date $t_{due} = 34$ only label $(25,58)$ remains. But propagation from the 'lowest' node gives the energy minimal solution taking time 33 and requiring only 57 energy units.

Each label has an additional coordinate which will eventually allow to identify a sequence of predecessors. This information cannot be attached – in contrast to the single criterion case – to the nodes themselves because different pairs of objective values may require different paths to attain them. The additional coordinate is occasionally omitted for easy of notation and the starting node does not have an additional coordinate. The first step of **PL** is the propagation step in which all arc labels are added to all node labels. The second steps unites old and new node labels and third step cleans them up. It deletes all labels for which there is another label which has a smaller duration as well as a smaller energy value.

The overall algorithm works with a list T of tentative node labels instead of a list of tentative nodes. Labels from this list do not loose their node assignments. This allows multiple propagations from a node during one run of the subsequent labeling algorithm. When the list of tentative labels becomes void, the best energy value is selected from all labels of the destination node whose time coordinate meets the due date.

E-due (Energy with due dates)

1. (Initialization). Set $T = (0,0)$ with $(0,0)$ belonging to $P_S = 1$.
2. (Iteration). While $T \neq \emptyset$ do:
   (a) Selection of lexicographically smallest label (d_0,e_0) from T belonging to node $i \in V$.

(b) $T = T - \{(d_0, e_0)\}$.

(c) $\forall j \in V$ with $(i, j) \in A$ do:

   i. update node label set $L(j)$ by **PL**.

   ii. $T = T \cup L(j)$.

   iii. Deletion of all labels of T belonging to j that are dominated by other labels of T that also belong to j.

3. (Termination). Output $e(P_D) = \min\{e| (d, e) \in L(P_D)$ and $d \leq t_{due}\}$ and optimal path traced by labels.

Though the multicriteria shortest path problem is symmetric as it considers both objectives as equally important, even in the present case of constrained energy minimization, it introduces a lexicographic order of the objectives. This break of symmetry affects the operations but not on the result. Considering time as the more important criterion renders a depth-first search behavior to the algorithm. Considering energy as the more important criterion renders a breadth-first search behavior to the algorithm.

Several technical improvements are possible. When a label generated in step 2(c)i. has an arrival time that exceeds the due date, this label can trivially be omitted. Also, a label of some node that is dominated by a label of the destination node can be omitted. The method is illustrated in figure 8.

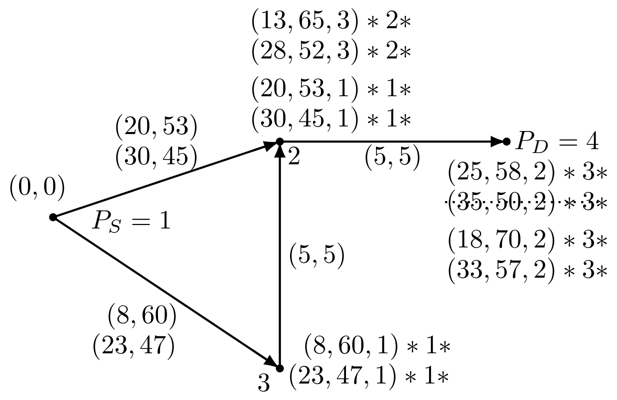


Figure 8: Labels set by the multicriteria Dijkstra algorithm for the problem from figure 7. In addition, nodes are numbered and predecessors are specified by these numbers as third coordinates of the node labels. Starred numbers indicate the iteration of step 2 in which the node labels are generated. The second label of the destination node is deleted because the due date $t_{due} = 34$ is violated. At most nine iterations of step 2 empty the tentative label set T. The node label $(8, 60)$ is the first to be deleted from T, $(13, 65)$ is the second etc. The minimum energy value belonging to feasible arrival times is $57 = \min\{58, 70, 57\}$.

6 Observability Motion

Instead of reaching a destination point, coming (and staying) close may suffice for observation. Staying close is specified by a region of observability RoO around the destination point so that the objective becomes to uninterruptedly stay within the RoO at minimum energy use. If a time bound for the observation is lacking, the formal objective becomes to indefinitely stay in the RoO at minimum energy use per time. Once the RoO has been reached, the compensation of drift caused by the ambient field is considered to be the only reason for energy consumption. The RoO may be circular as in figure 6, but other shapes are feasible. The essential trade-off for observability motions is that between staying in one position while continuously consuming little energy vs. drifting for some time at no energy consumption, moving upstream at some energy consumption and drifting again etc. Depending on the characteristics of the platform and the ambient field, the second strategy, called orbiting or oscillating, may be superior.

As an example, a ship may remain still somewhere in a RoO see figure 9. This requires to continuously compensate the drift which is assumed to be constant at two knots throughout the field. Fuel consumption for drift compensation is assumed to equal 5l/h. When the distance between A and B is 10 nautical miles and the platform initially is located at A, it may drift five hours at zero consumption and then travel back to A at an assumed speed of six knots through the water at an assumed fuel consumption of 10l/h. So, it takes 10nm/(6 knots - 2 knots) = 2.5h to travel from B to A thereby consuming 25l fuel. Thus, the average consumption over one orbit is $25l/7.5h = 3.333l/h$ which is less than the consumption for immobility.

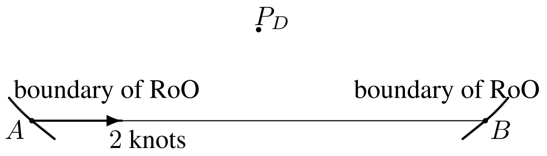


Figure 9: Region of observability for P_D and drift in a constant field from boundary point A to boundary point B. Upon arrival in B, the platform travels upstream to A where it resumes drifting.

Boundary points of a RoO at which the field directs into the interior of the RoO are denoted as entry points (of the RoO) and boundary points at which the field directs to the outside are denoted as exit points (of the RoO). An orbit is a closed trajectory from an entry point to an exit point and back. A drift orbit is an orbit of which the partial trajectory from the entry to the exit point is a mere drift trajectory. The trajectory from the entry point to the exit point may be different from the inverse trajectory back to the entry point. The situation from figure 9 is special in this respect.

When the observation time is unbounded, an optimal orbit or, more precisely, a minimum average power (MAP) orbit adheres to the fractional program

$$\min_{O \in \mathcal{O}} \frac{energy(O)}{time(O)}.$$

The orbits O range through some set $\mathcal{O}$ of orbits that may be finite or infinite. In the finite case, a MAP drift orbit can be approximated by the following two steps:

1. (Drift trajectory) For each of the entry points compute the drift trajectory according to the forward equation $P(t + \Delta t) = P(t) + \Delta t \cdot f(P(t))$ until an exit point is reached. A drift trajectory with maximum duration is selected.

2. (Partial trajectory back to entry point) An energy minimal path is computed by algorithm **Dt** from the exit point of the selected drift trajectory to its entry point.

Minimum average power orbits need not be of a drift type. "Jockeying" between drift trajectories, which consumes energy, can be favorable if the field changes differently along different drift trajectories, figure 10.

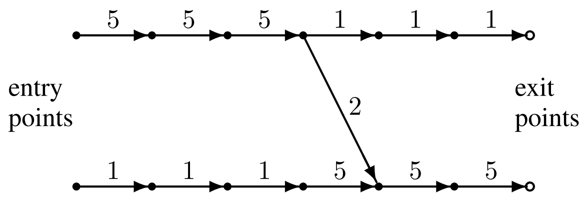


Figure 10: Two drift trajectories with labels indicating time. Maximizing drift time requires to switch from the upper to the lower trajectory. The return trajectory to the upper entry point is omitted.

A MAP orbit which is not necessarily of a drift type can be approximated by concatenations of minimum energy paths from entry points to exit points and back. As orbits are cyclic, this order can be reversed so that an exit point becomes the starting point of an orbit. When a platform arrives at a RoO it may do so either in an entry point or in an exit point and the beginning of an orbit is chosen accordingly. If the best computed orbit does not pass through the arrival point, the arrival point will connected to the best orbit. One way to do so is to compute a minimum energy path from the arrival point to the orbit by the Dijkstra algorithm with all orbit points forming a destination region. The orbit computation itself is as follows.

MAP-orbit (Minimum Average Power-orbit approximation)

1. Input RoO and finite set En of entry points. (Initialization). For each $a \in En$ compute its exit point $ex(a)$ reached by mere drift. All these exit points are summarized as exit set $Ex = \{ex(a) | a \in En\}$.

2. (Computations).
   (a) For all $a \in En$ and all $b \in Ex$ computation of a minimum energy path $P(a, b)$ from a to b.
   (b) For all $b \in Ex$ and all $a \in En$ computation of a minimum energy path $Q(b, a)$ from b to a.
   (c) For all $a \in En$ and all $b \in Ex$ concatenate minimum energy paths to form orbits through a and b as $O(a, b) = P(a, b) \circ Q(b, a)$.
   (d) $O_{MAP} = argmin_{O(a,b),\, a \in En,\, b \in Ex} \frac{energy(O(a,b))}{time(O(a,b))}$.

3. (Termination). Output O_{MAP}.

Algorithm **Dt** can be used for the path computations in steps 2 (a) and (b) and durations are recorded for all paths so that the cycle times of all orbits are known in step 2 (d). The computations of algorithm **MAP-orbit** for all paths emanating from the same entry and same exit point can be interleaved. The formation of orbits from paths is obvious by aligning the waypoints of one path behind the other while avoiding immediate repetitions of the exit and entry points. Path alignment is denoted by the concatenation operation $\circ$.

An approximation of a MAP orbit without concatenation can be facilitated by integrating the two computing stages of **MAP-orbit**. Essentially, this requires to provide a double waypoint graph as sketched in figure 11. The first part of the waypoint graph allows approximations of energy minimal paths from entry to exit points and the second graph allows for the inverse. The structure of the second graph can be obtained by reflecting the first graph, but arc labels may be completely different. In order to admit a minimum path computation, a hypothetical source node s is connected at zero cost to all entry nodes and the copy of the entry nodes is connected at zero cost to a hypothetical sink node t.

An energy minimal path approximates a MAP orbit. If such a path uses the same entry point in both copies of the entry set, then the path actually amounts to an orbit. If not, a sequence of orbit-like paths is better than a single orbit traced repeatedly. Instances of the waypoint graph can then be added to result in a fourfold waypoint graph, a sixfold waypoint graph etc. A shortest path from source to sink then yields an optimal motion that is more complex than an orbit. This, particularly, applies to nonstationary fields.

When a best orbit is searched for, an energy minimal orbit is computed for each entry point by setting to infinity the arc labels between source and entry set as well as between the copy of the entry set and the sink except for one entry point. Then, the minimum energy orbit through this entry point is computed and the procedure is repeated for all other entry points. The best of all these orbits is eventually chosen. Noteworthy, the arc labels in the grid sections of the double waypoint graph are not affected. Computations are summarized in the next algorithm and a computation sample with Scilab [11] is illustrated in figure 12.

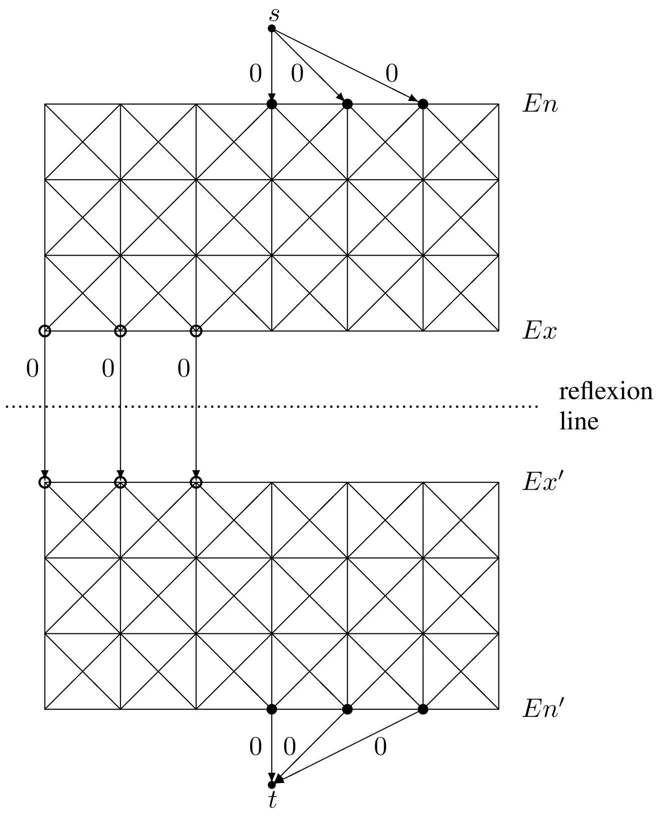


Figure 11: Double waypoint graph with arc directions and labels omitted in the RoO. The RoO is here assumed to be rectangular with a discretization of 28 nodes. Also, every node of the boundary either is an entry point or an exit point (unless the field is zero in a boundary point or heads along a straight segment of the boundary), but the entry set and the exit set are only shown partially.

MAP-orbit-double (Minimum Average Power-orbit approximation in double waypoint graph)

1. Input RoO and finite set En of entry points.
   (Initialization). For each $a \in En$ compute the exit point $ex(a)$, summarize all these as exit set $Ex = \{ex(a) | a \in En\}$ and create the double waypoint graph.

2. (Computations).

   (a) For all $a \in En$ do:

      i. $c(s, a) = 0$.
      ii. $c(s, b) = \infty$ for all $b \in En - \{a\}$.
      iii. $c(a', t) = 0$.
      iv. $c(b', t) = \infty$ for all $b' \in En' - \{a'\}$.
      v. Computation of a minimum energy path $P(s, t)$ from s to t by **Dt**.
      vi. Deletion of s and t from $P(s, t)$ to result in orbit $O(a)$ through a.

   (b) $O_{MAP} = argmin_{O(a),\, a \in En} \frac{energy(O(a))}{time(O(a))}$.

3. (Termination). Output O_{MAP}.

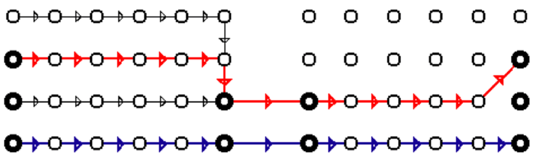


Figure 12: Double waypoint graph with original waypoint graph (left) with three entry nodes and two exit nodes. Only arcs with zero energy cost are shown in that part. The reflected graph section (right) has different energy labels: horizontal and vertical transitions incur 10 units and diagonal transitions incur 15 units. Transition times are identically one for all horizontal and vertical transitions and 1.5 for all diagonal transitions. The two prolonged arcs between the exit points and their reflections have no physical meaning and incur zero time and energy cost. The upper path requires 0 + 55 = 55 energy units while using 6+5.5 = 11.5 time units requiring average power 55/11.5 = 4.783 which is optimal. Another path is the lower path which requires 0 + 50 = 50 energy units and 5 + 5 = 10 time units thus requiring average power 50/10 = 5.

The average energy spent by orbiting is compared to the power spent by remaining still at the most favorable point inside RoO and the better of the two options is selected. A most favorable point is one with minimum field strength: $P_{min} = argmin_{P \in RoO} \|f(P)\|$. To remain still there requires to continuously execute the control vector $u(t) = -f(P_{min})$ which incurs the power $cost(u(t)) = power(-f(P_{min}))$, see section 4.1.

The objective of minimizing energy for uninterrupted observation can be overlayed with a variety of other objectives. These include the requirement for observation from sufficiently many positions and angles, intended unobservability or positional irregularity of the observing platform itself and interleaving observations of different objects that cannot be observed from any one position.

7 Exploration

Exploration of a region is similar to observation with the difference being that a set of locations must be traversed so that each point of the exploration region is observable at least once. If the whole exploration region were observable from a single location, say by a circular scan, then exploration were trivial. It is thus assumed that no single observation region – be it circular, rectangular or else – covers the exploration region so that the platform must move, see figure 13. It is further assumed that the cost of making observations is negligible compared to the cost of moving the platform with the exception of drift which incurs zero cost.

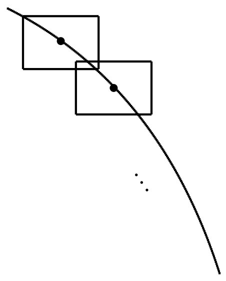


Figure 13: Trajectory with observation regions from two locations from which the observations are made.

The aim is to find minimum time and minimum energy trajectories which to cover an exploration region by finite many observation regions. A minimum energy exploration trajectory need not trace "adjacent" drift trajectories as indicated by figure 14. For computing exploration paths, the exploration region is endowed with an exploration graph. This is a directed graph whose nodes indicate all locations that must be visited for making observations. The positions in figure 13 may serve as nodes of an exploration graph. Its arcs describe possible motions between nodes and each arc receives a label with the same meaning as in waypoint graphs. Physical resolutions of exploration graphs and waypoint graphs may differ.

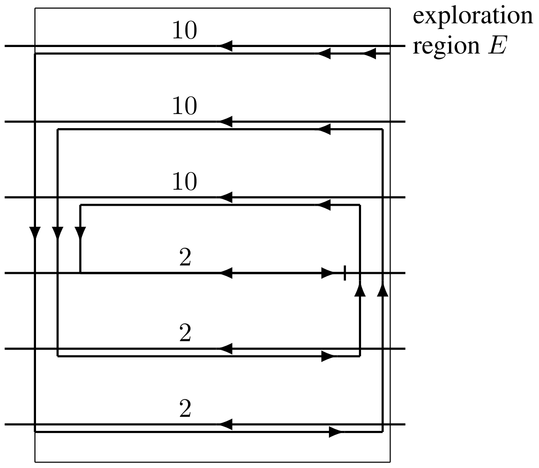


Figure 14: Drift trajectory with high velocity field regions being traveled downstream while low velocity field regions are traveled upstream for exploring region E. Observation regions are not shown.

An exploration path may repeatedly visit nodes of the exploration graph before all other nodes have been visited. Thus, exploration paths relax Hamilton paths which visit each node of a graph exactly once; a Hamilton path is a Hamilton cycle in which "the last" arc is missing. A standard transformation [8, p. 23] allows to reduce the computation of an exploration path to a Hamilton path. The exploration graph is therefore endowed with all missing arcs. The resulting complete directed graph receives arc labels that denote the shortest path from head to tail for each node in terms of the original arc labels. For stationary fields the new labels are formally denoted as

$$C(i,j) \quad = \quad \text{length of shortest path from } i \text{ to } j \text{ according to the arc labels } c(\cdot,\cdot).$$

The new labels can be computed by several applications of algorithm **Dt** or by a single application of the Floyd-Warshall (triple) algorithm [6]. A Hamilton path is then computed for the "completed" exploration graph. When such a path uses an arc which does not exist in the original exploration graph, the platform travels along the corresponding shortest path as indicated in figure 15. The start-

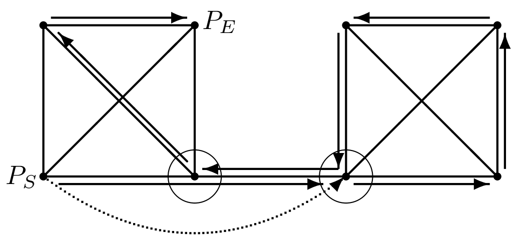


Figure 15: Original exploration graph (shown as undirected graph for notational ease) with one additional arc (dashed arrow) shown for the completed exploration graph and used by an optimal Hamilton path from a starting node P_S to an end node P_E. The resulting exploration trajectory in the original exploration graph is indicated by solid arrows. Two of the nodes (circled) are visited twice.

ing node of an exploration path, typically, is that node of the exploration graph which is reachable at minimum cost from the present platform location – either inside or outside the exploration region. While the starting point P_S is predetermined for computations of exploration paths, the end point P_E is determined by the optimization.

The shortest Hamilton path problem is known to allow a great variety of initialization and improvement heuristics. An initial heuristic which works under all circumstances is the nearest neighbor heuristic which is now adopted to yield exploration paths. The approach, simply, is to choose the nearest unvisited node for the next visit until all nodes have been visited. This greedy procedure works on the completed exploration graph is that node repetitions may occur.

NN (Nearest Neighbor heuristic)

1. Input exploration graph with node set V_E, original arc labels $c(\cdot,\cdot)$ and starting point P_S.
(Initialization). Set $L = V_E - \{P_S\}, path = (P_S)$ and

```
    $P_{current} = P_S$, computation of shortest path labels $C(\cdot,\cdot)$.
2. (Iteration). While $L \neq \emptyset$ do:
    (a) Computation of $j_0 = argmin_{j \in V_E - L}\, C(P_{current}, j)$.
    (b) $L = L - \{j_0\}$.
    (c) $path = path \circ j_0$.
    (d) $P_{current} = j_0$.
3. (Termination). Output exploration trajectory $path = (P_S, \ldots, P_E)$.
```

The exploration path is incrementally built by node concatenation in step 2(c). An alternative to greedy procedures is to solve an assignment problem which maps each node to a successor node such that each successor appears only once overall and the sum of transition costs from all nodes to their successors becomes minimal. Assignment problems can be solved efficiently. Subcycles which potentially occur in an optimal assignment need to be connected to form a Hamilton cycle. Eventually, this Hamilton cycle is broken up "right before" the starting point is revisited.

8 Motion with Controls in 3D

The foregoing approaches generalize to three dimensions in an evident manner with controls being 3D vectors. Static waypoint graphs typically are regular 3D grids with each interior node having at least six neighbors, depending on the platform characteristics; see figure 16. Regularity of the 3D grid may be weakened by the horizontal grid spacing being different from the vertical spacing. Also, depending on the characteristics of the platform, pure vertical motion may be feasible (hot air balloons) or impossible (planes). The last case is expressed by the time and energy labels for those arcs being artificially high or infinity.

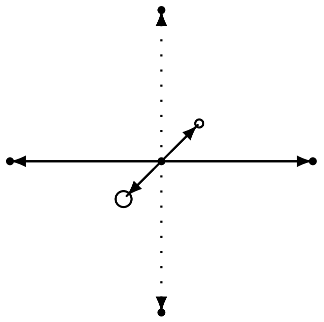


Figure 16: Grid point with four neighbors at the same altitude and two at different altitudes. If vertical transitions are impossible, these arcs are assigned high transition costs and slanted arcs are added.

The 3D case allows to pose original motion problems like finding a maximum survival trajectory. For an aerial platform with fixed energy budget this means that the platform should stay airborne with for a maximum time or travel a maximum distance. Both versions of the problem differ when energy for lift can be traded for propulsion. Survival problems make sense only for platforms that are not lighter than the atmosphere.

A survival trajectory will seek updraft areas and avoid downdraft areas, see figure 17. These may be part of circular atmospheric patterns like Hadley cells. Vertical and horizontal motion of the atmosphere are coupled inside one Hadley cell. At high altitude, the atmosphere horizontally moves from an updraft area towards a downdraft area while at low altitude motion heads in the opposite direction. It can hence be reasonable to even come close to downdraft areas. Boundaries of updraft and downdraft areas are not crisp.

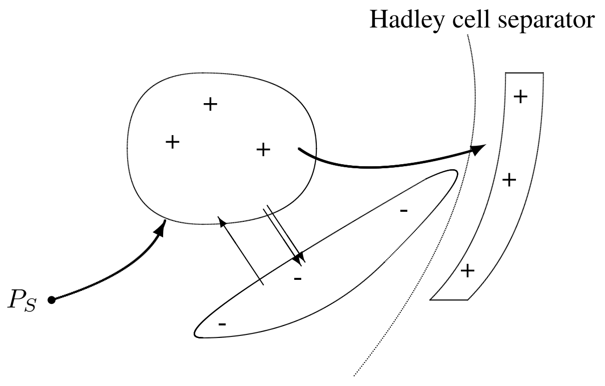


Figure 17: A survival trajectory connects the updraft areas while it avoids the downdraft area. Horizontal field motion is indicated for high altitudes (double arrow) and low altitudes (single arrow).

A trajectory that intends to visit all updraft areas with minimum energy – be it a survival trajectory or not – requires to determine the order of the visits and the entry and exit points. It is reasonable to allow revisits of updraft areas before all other updraft areas have been visited. This is in analogy to exploration paths, so that survival trajectories need not form Hamilton paths. A survival trajectory can be approximated by the nearest insertion method.

NIns (Nearest Insertion heuristic)

```
1. Input collection of updraft areas $L = (U_1, \dots, U_m)$, starting position $P_S$.
   (Initialization). Set $P_{in} = P_S$, $\mathcal{P} = \emptyset$.
2. (Iteration). While $L \neq \emptyset$ do:
   (a) Computation of energy minimum path $\mathcal{Q}$ by algorithm Dt from $P_{in}$ to an intermediate destination point = first point $P_l$ reached in an updraft area $U_i \in L$.
   (b) $L = L - \{U_i\}$.
   (c) $\mathcal{P} = \mathcal{P} \circ \mathcal{Q}$.
   (d) $P_{in} = P_l$.
3. (Termination). Output survival trajectory $\mathcal{P}$.
```

Though all intermediate destination points computed in step 2(a) lie in different updraft areas, the incremental paths $\mathcal{Q}$ are allowed to revisit updraft areas as well as waypoints inside and outside updraft areas.

When an updraft area is not exited at maximum altitude, an additional ascent adds a safety margin. When vertical motion is expensive, it is not obvious how far to climb so that the next updraft area is entered higher than originally planned. Modifying a path to enter the next updraft area at a higher altitude can be facilitated by the following incremental lift procedure. To simplify the notation, it is assumed that sufficient (feasible) altitude can be gained in the current updraft area.

IncLift (Incremental Lift heuristic)

```
1. Input current entry point $P_1$ and exit point $P_2 = (x_2, y_2, z_2)^T$ of updraft area $U_i$ and entry point $P_3 = (x_3, y_3, z_3)^T$ of the next updraft area $U_j$.
   (Initialization). Set $z_3' = z_3$, $k = 1$.
2. (Iteration). While $z_3' \leq z_3$ do:
   (a) Computation of energy minimal path $\mathcal{Q}_1$ by algorithm Dt from $P_1$ to the first point $P_2' \in U_i' = \{P = (x, y, z)^T \in U_i | \text{ with } z = z_2 + k \cdot d\}$.
   (b) Computation of energy minimal path $\mathcal{Q}_2$ by algorithm Dt from $P_2'$ to the entry point $P_3' = (x_3', y_3', z_3')^T$ of $U_j$.
   (c) $k = k + 1$.
3. (Termination). Output new partial survival trajectory $\mathcal{Q} = \mathcal{Q}_1 \circ \mathcal{Q}_2$.
```

The remainder of the survival trajectory must be adapted to the next updraft area U_j and incremental liftings can be applied to other transitions as well. A greedy behavior suggests to continue climbing in each updraft area until the average cumulative climbing energy in the present updraft area increases.

9 Conclusion

It has been shown how to model and perform trajectory computations in ambient fields beyond mere point to point motion. Trajectories are specified in space and time resolutions that typically range above the control level, though the computations can be embedded into standard control frameworks. Advantageous field effects are exploited where possible and disadvantageous effects are mitigated where needed. All approaches lend to identical or similar spatial discretizations as well as to related graph algorithms.

Future work is to include uncertainty of the ambient field such as given for simple point to point motion in the framework of Markov decision processes [18]. Of particular interest are deviations from the assumed field which can be sensed by the mobile platform itself. The case of no a-priori information about the actual field is an even more challenging task.

References

[1] Allsopp, T., Mason, A., Philpott, A., ”Optimizing yacht routes under uncertainty”, Proceedings of the National Conference of the Operations Research Society of Japan, Tokyo, 2000, p. 176-183.

[2] Bertsekas, D., ”Dynamic programming”, Prentice Hall, Englewod Cliffs, 1987.

[3] Das, T., Mukherjee, R., Cameron, J., ”Optimal trajectory planning for hot-air balloons in linear wind fields”, Journal of Guidance, Control and Dynamics 26, 2003, p. 416-424.

[4] Harries, S., Hinnenthal, J., ”A systematic study on posing and solving the problem of pareto optimal ship routing”, 3rd International Conference on Computer Applications and InformationTechnology in the Maritime Industries COMPIT 2004, Sigu’nza, Spain, May 2004.

[5] Kämpke, T., Elfes, A., ”Optimal aerobot trajectory planning for wind-based opportunistic flight control”, Proceedings of the International Conference of Intelligent Robots and Systems IROS, Las Vegas, 2003, # 862.

[6] Lawler, E.S., ”Combinatorial optimization: networks and matroids”, 2nd ed., Dover, Mineola, 2001.

[7] Lee, H., et al., ”Optimum ship routing and it’s implementation on the web”, Springer Lecture Notes in Computer Science 2402, Berlin, 2002, p. 125-136.

[8] Lawler, E.L., Lenstra, J.K., Rinnoy Kan, A.H.G., Shmoys, D.B., ”The traveling salesman problem”, Wiley, Chichester, 1985.

[9] Reif, J., Sun, Z., ”Movement planning in the presence of flows”, Springer Lecture Notes in Computer Science 2125, Berlin, 2000, p. 450ff.

[10] Rowe, N.C., ”Obtaining optimal mobile-robot paths with non-smoth anisotropic cost functions using qualitative state reasoning”, International Journal of Robotics Research 16, 1997, p. 375-399.

[11] Scilab, The free platform for numerical computation, www.scilab.org.

[12] Searoutes, ”The searoute system for vessel optimal routing planning”, www.searoutes.sg.

[13] Shadden, S.C., Lekien, F., Marsden, J.E., ”Definition and properties of Lagrangian coherent structures from finite-time Lyapunov exponents in two-dimensional aperiodic flows”, Physica D 212, 2005, p. 271-304.

[14] Skysails, ”Skysails technology information”, www.skysails.info.

[15] Statheros, T., Howells, G., McDonald-Maier, K., ”Autonomous ship collision avoidance navigation concepts, technologies and techniques”, The Journal of Navigation 61, 2008, p. 129-142.

[16] Tarapate, Z., ”Selected multicriteria shortest path problems: an analysis of complexity, models and adaptation of standard algorithms”, Int. Journal of Mathematical Computing Science 17, 2007, p. 269-287.

[17] Veldhuizen, T., ”Sailing around the world in minimal time”, preprint, University of Waterloo, Canada, 2000.

[18] Wolf, M. et al., ”Probabilistic motion planning of balloons in strong, uncertain wind fields”, Proc. International Conference on Robotics and Automation ICRA2010, Anchorage, 2010, p. 1123-1129.

Discovering Comfortable Driving Strategies Using Simulation-Based Multiobjective Optimization

Erik Dovgan and Tea Tušar
Jožef Stefan Institute, Jamova cesta 39, Ljubljana, Slovenia
Jožef Stefan International Postgraduate School, Jamova cesta 39, Ljubljana, Slovenia
E-mail: {erik.dovgan, tea.tusar}@ijs.si, http://dis.ijs.si

Matija Javorski
Faculty of Mechanical Engineering, University of Ljubljana, Aškerčeva cesta 6, Ljubljana, Slovenia
E-mail: matija.javorski@fs.uni-lj.si, http://www.fs.uni-lj.si

Bogdan Filipič
Jožef Stefan Institute, Jamova cesta 39, Ljubljana, Slovenia
Jožef Stefan International Postgraduate School, Jamova cesta 39, Ljubljana, Slovenia
E-mail: bogdan.filipic@ijs.si, http://dis.ijs.si

Keywords: driving strategies, multiobjective optimization, traveling time, fuel consumption, driving comfort

Received: May 28, 2012

Driving a vehicle along a route consists of control actions applied to the vehicle by taking into account the vehicle and route states. Control actions are usually selected by optimizing the traveling time and the fuel consumption. However, the resulting vehicle behavior can be uncomfortable for the driver/passengers. The comfort is measured as the change of acceleration, i.e., jerk. To obtain more comfortable driving strategies, we introduce comfort as an objective to the Multiobjective Optimization algorithm for discovering Driving Strategies (MODS), thus obtaining the Multiobjective Optimization algorithm for discovering Comfortable Driving Strategies (MOCDS). The two algorithms are compared on a real-world route. The results show that MOCDS finds more comfortable driving strategies than MODS, while not significantly deteriorating their traveling time and fuel consumption. The most significant improvement in comfort is achieved on driving strategies with low fuel consumption, which are highly uncomfortable and therefore have the most room for improvement. On the other hand, the driving strategies found by MODS with short traveling time are already comfortable and therefore cannot be additionally improved.

Povzetek: Prispevek predstavlja algoritem za iskanje strategij vožnje, ki ne optimira le časa vožnje in porabe goriva, temveč tudi udobje za voznika/potnike.

1 Introduction

When driving a vehicle along a route, two objectives are usually optimized: the traveling time and the fuel consumption. However, the algorithms optimizing only these objectives usually find pulse-and-glide driving strategies [10, 12]. Such driving strategies repeatedly exchange high throttle percentage and zero throttle percentage. Therefore, the acceleration continuously changes which significantly reduces the driving comfort [14]. Low driving comfort is unacceptable from the user point of view, even though such driving strategies efficiently reduce the traveling time and the consumed fuel. Consequently, the driving comfort has to be taken into account when discovering driving strategies.

In our previous work we designed and implemented the Multiobjective Optimization algorithm for discovering Driving Strategies (MODS) [2, 3], which searches for driving strategies by modeling a real vehicle driving on a real route as a black box, and optimizing the traveling time and the fuel consumption. The obtained driving strategies are better than the driving strategies found by optimization algorithms used so far [4], i.e., predictive control [16] and dynamic programming [7, 8]. However, MODS fails to find comfortable driving strategies, especially with low fuel consumption. In order to obtain comfortable driving strategies, we introduce the third objective, i.e., the comfort that has to be maximized, or equivalently, the discomfort that has to be minimized. To quantify the discomfort, Nilsson [14] suggested to measure the magnitude of the jerk. This measure was added to the dynamic programming algorithm presented in [7] and the obtained algorithm found more comfortable driving strategies. Wu et al. [19] presented a car-following model focused on passenger comfort. A comfortable vehicle driving was achieved by limiting the jerk. The same measure was also used by Haj-Fraj et al. [6] who searched for the optimal control of gear shift operations. In order to obtain a comfortable shifting, a dynamic programming algorithm was implemented which minimizes the jerk.

Comfort can be defined in various other ways, too. For example, a comfortable driving strategy may be a strategy that does not change the control actions frequently. In addition, it is not compulsory to consider the comfort as an objective in the algorithm to obtain comfortable driving strategies. For example, Gerdts [5] and Kirches et al. [9] developed single objective algorithms that search for the optimal double-lane-change manoeuver on a short (140 m) horizontal route and minimize the traveling time. The same problem was tackled by Logist et al. [13], who developed a multiobjetive algorithm that minimizes the traveling time and the fuel consumption. Although none of them optimizes the comfort, they obtained comfortable driving strategies that do not change the control actions frequently. However, such driving strategies were obtained by using model-based approaches, which cannot be applied when a black-box simulator is used. In addition, the algorithms were tested only on a short horizontal artificially generated route. Therefore, it is not clear if these algorithms would produce comfortable driving strategies also on data from (longer) real-world routes with inclined route segments and velocity limits.

In this paper we present the two-level Multiobjective Optimization algorithm for discovering Comfortable Driving Strategies (MOCDS) that minimizes the traveling time, fuel consumption and discomfort, i.e., jerk. The lower-level algorithm is based on breadth-first search [17] and Nondominated Sorting Genetic Algorithm (NSGA-II) [1]. The best input-parameter values for the lower-level algorithm are found by the upper-level evolutionary algorithm. MOCDS returns a set of nondominated [1] driving strategies and leaves the selection of the preferred driving strategy to the user.

The paper is further organized as follows. The MOCDS algorithm is described in Section 2. Section 3 presents the experiments and the obtained results. Finally, Section 5 concludes the paper with ideas for future work.

2 The Algorithm for Discovering Comfortable Driving Strategies

This section presents the two-level algorithm for discovering comfortable driving strategies (MOCDS) that minimizes the traveling time t, the fuel consumption c, and the driving discomfort d.

2.1 Strategy representation and evaluation

A driving strategy is a set of connections between the vehicle and route states, i.e., the state space, on the one hand, and the weights used to select the control action that is applied to the vehicle during the driving simulation on the other hand. The vehicle state is defined with the vehicle velocity, while the route state is defined with the inclinations and the velocity limits of the current and the next segments, and the route to the next segment. The control action is defined with the throttle and braking percentage ε_{V} and the gear g_{V}, while the weights are the consumption weight ω_{c} and time weight ω_{t}. The state space, control actions and weights are discretized in advance. The subspaces obtained by the state space discretization are called hypercubes [18]. Each hypercube stores a consumption weight and a time weight. These data are used to select the appropriate control action when the vehicle and route states correspond to the hypercube.

The driving strategy is evaluated with a black-box vehicle driving simulator that was implemented based on the vehicle description from [11, 15] and is described in [4]. The simulator receives the control action for the vehicle, simulates the vehicle driving for one route step, where the length of a step is Δs, and returns the spent time, the consumed fuel, the driving discomfort, and the new vehicle and route states. The new vehicle and route states are then used to select the current hypercube, and consequently to find the new control action that is used for the simulation of the next route step. This process continues until the traveling along the entire route has been simulated, i.e., until $\sum_1^x \Delta s = s$, where s is the length of the route and x is the number of already simulated route steps.

2.2 Lower-level algorithm

The lower-level algorithm is a deterministic multiobjective algorithm for discovering comfortable driving strategies that minimizes the traveling time, the fuel consumption and the driving discomfort (see Figure 1). It starts with a single driving strategy with empty hypercubes. Then it simulates the vehicle driving for several route steps with several driving strategies until the driving along the entire route has been simulated (Main procedure in Figure 1). If the current hypercube of a driving strategy at a route step is empty, the driving strategy is cloned for each discrete set of weights $\{\omega_{\mathrm{c}}, \omega_{\mathrm{t}}\}$ and this data is stored in the hypercube. More precisely, the driving strategy is cloned when the vehicle and route states correspond to the current hypercube for the first time during the driving simulation. When the current hypercube stores the weights, these data are used to select the most preferred control action as shown in Figure 1. The control action is selected by predicting the vehicle driving for N_{P} prediction steps ahead for each possible discrete control action $\{\varepsilon_{\mathrm{V}}, g_{\mathrm{V}}\}$. Afterwards, the spent time t, the consumed fuel c and the driving discomfort d are com-

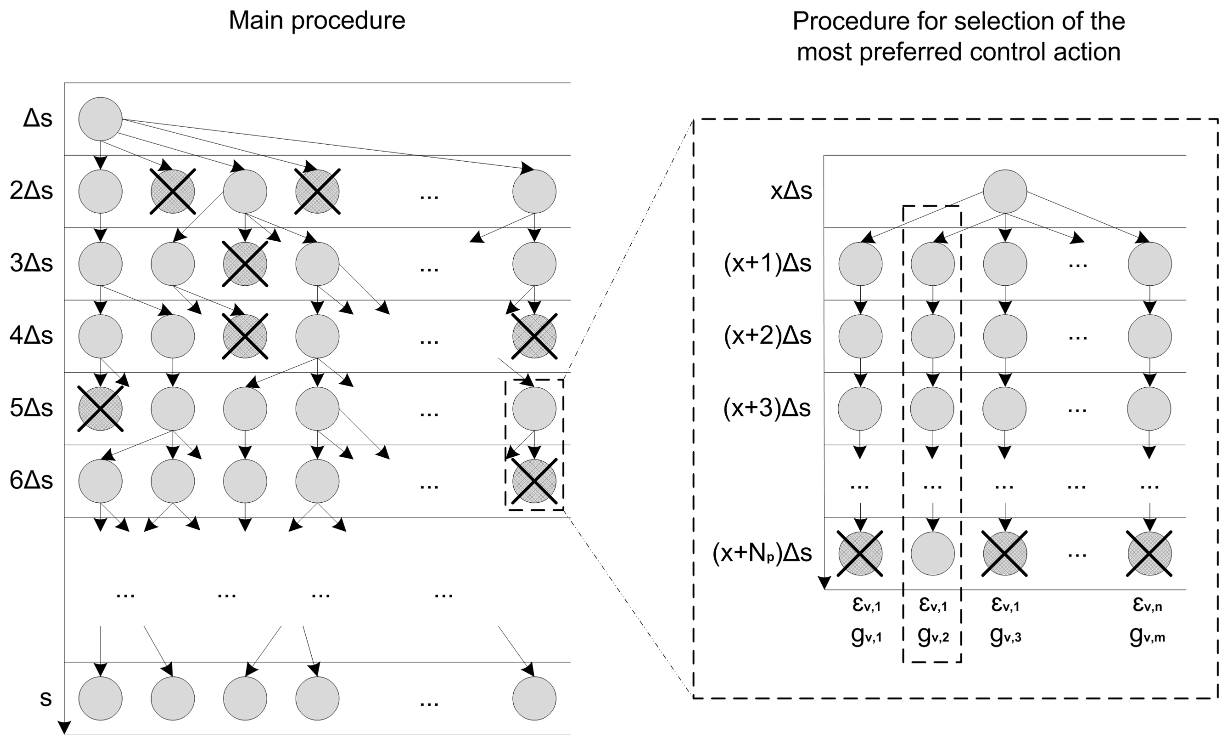


Figure 1: The lower-level algorithm for discovering driving strategies.

bined into the cost function f:

$$f = \omega_c c + \omega_t t + (1 - \omega_c - \omega_t)d, \quad (1)$$

and the control action that minimizes f is selected for one step simulation (in the Main procedure in Figure 1). The driving discomfort is calculated by summing up the magnitudes of the jerk, i.e., the differences in acceleration a denoted as Δa, during the driving simulation as follows:

$$d = \sum_{x\Delta s}^{(x+N_P)\Delta s} |\Delta a| \quad (2)$$

Since the driving strategies are cloned, the number of driving strategies grows exponentially. To reduce their number, fast nondominated sort and crowding distance mechanisms from the Nondominated Sorting Genetic Algorithm (NSGA-II) [1] are used at each route step to select the most promising driving strategies with respect to the objectives and maintain a constant number of driving strategies. The non-promising driving strategies are deleted and are marked with "$\times$" in the Main procedure in Figure 1. When the vehicle driving along the entire route has been simulated, the algorithm returns a set of nondominated driving strategies.

The lower-level algorithm requires the following input parameters:

– discretization of vehicle and route state space,

– discretization of control actions,

– discretization of weights, and

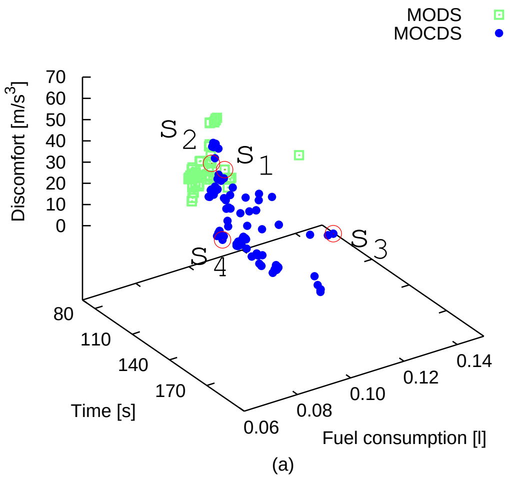
– number of prediction steps N_P.

2.3 Upper-level algorithm

The upper-level evolutionary algorithm searches for the best sets of input-parameter values for the lower-level algorithm and maximizes the hypervolume [20]. A set of input-parameter values is an upper-level solution. The upper-level algorithm applies evolutionary principles, i.e., selection, crossover and mutation, to the set of upper-level solutions through several generations [1]. The evaluation of an upper-level solution is carried out as follows. Firstly, the lower-level algorithm finds the nondominated driving strategies using the input-parameter values stored in the upper-level solution. Finally, the hypervolume covered by the driving strategies is calculated. For more details see [3, 4].

3 Experiments and Results

MOCDS was tested on data describing a real-world route and the obtained driving strategies were compared to the driving strategies obtained by MODS in order to determine the influence of the comfort as an objective. The selected route was an urban road of around 1100 m that includes a few uphills and downhills. Its characteristics are summarized in Figure 2.



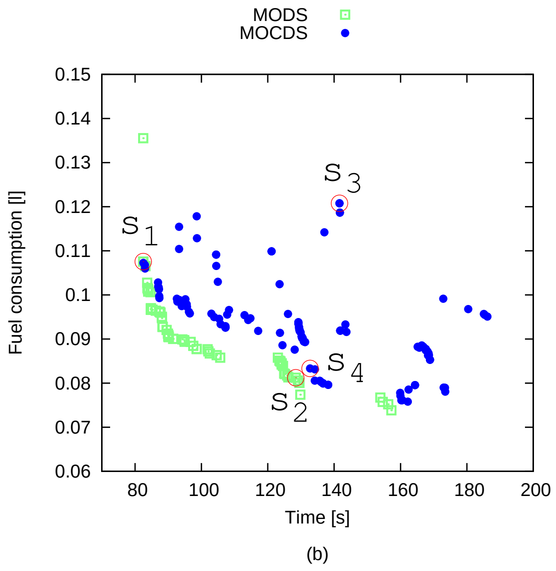


Figure 3: Driving strategies found by MODS and MOCDS.

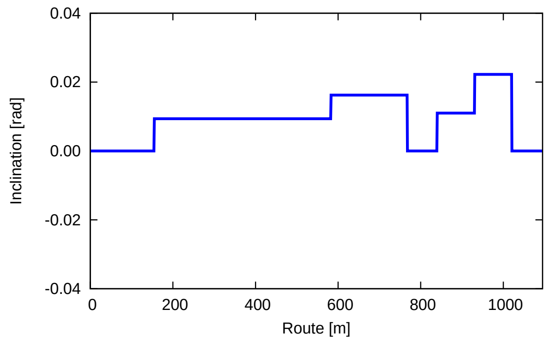


Figure 2: Inclinations of the testing route; the velocity limit is 50 km/h along the entire route.

Figure 3 shows the nondominated driving strategies found by MOCDS and MODS. Specifically, Figure 3(a) shows the driving strategies in the objective space of all three objectives, while Figure 3(b) shows a projection of the driving strategies to the objective space with only the objectives t and c. Ideally, MOCDS should find all the driving strategies obtained by MODS. However, Figure 3 shows that MOCDS does not find (all) these driving strategies. This is due to the fact that the number of driving strategies grows exponentially at each route step and, therefore, several driving strategies have to be deleted as described in Subsection 2.2. More precisely, MOCDS and MODS maintain the same number of driving strategies but MOCDS has a larger search space due to the additional objective. Consequently, several driving strategies that are promising in the values of t and c are deleted by MOCDS since a larger search space has to be covered by the same number of driving strategies. Those driving strategies are not deleted by MODS and therefore MODS better optimizes the traveling time and fuel consumption. Nevertheless, the results show that MOCDS is able to find, in addition to the comfortable driving strategies, driving strategies similar to the ones found by MODS in terms of traveling time and fuel consumption.

Four interesting driving strategies were further analyzed. They are marked in Figure 3 as follows:

– s_1 is a driving strategy with short traveling time found by MODS;

– s_2 is a driving strategy with low fuel consumption found by MODS;

– s_3 is the driving strategy with the highest comfort but also long traveling time and high fuel consumption found by MOCDS; and

– s_4 is a driving strategy found by MOCDS, which has similar traveling time and fuel consumption but significantly higher comfort than s_2.

The objective values of these driving strategies are shown in Table 1. Moreover, the vehicle behavior obtained by applying these driving strategies can be seen in Figures 4 and 5. These figures show the control actions, i.e., the throttle and braking percentage and the gear, the vehicle velocity and the jerk along the entire route. The results show that in order to obtain highly comfortable driving strategies (e.g., s_3), the control actions must rarely change. Consequently, the vehicle velocity slowly changes and the jerk is low along the entire route. On the other hand, when the comfort is not taken into account (e.g., s_1 and s_2), the control actions change frequently and consequently the jerk is higher. Finally, Figure 5 shows the vehicle behavior obtained by applying the driving strategies that are similar

Driving strategy	t [s]	c [l]	d [m/s^3]
s_1	82.46	0.1076	12.784
s_2	128.41	0.0812	44.573
s_3	141.60	0.1208	1.298
s_4	132.75	0.0833	9.518

Table 1: The objective values of the driving strategies marked in Figure 3.

in terms of traveling time and fuel consumption, but significantly differ in comfort (see also Table 1). More precisely, it shows that a driving strategy of the same quality in terms of traveling time and fuel consumption but significantly more comfortable can be obtained by reducing the changes of control actions. Such driving strategy can be obtained by MOCDS but not by MODS.

Although the MOCDS driving strategies change the control actions, such as the gear, less frequently than the MODS driving strategies, the number and frequency of changes remains high when nondominated driving strategies in terms of traveling time and fuel consumption are taken into account, e.g., s_4 (see Figure 5). Nevertheless, MOCDS also finds driving strategies with a significantly lower number and frequency of changes, see the driving strategy with the highest comfort, s_3 (see Figure 4). To even further reduce the number and frequency of changes of control actions, the comfort should be redefined, e.g., by penalizing the changes in control actions, or the search space should be limited, for example, by restricting the changes of control actions.

Figure 6 shows the driving strategies found by MODS and MOCDS which are nondominated with regard to objectives t and c. These driving strategies found by MOCDS are the most interesting ones since they are similar to the driving strategies obtained by MODS in terms of traveling time and fuel consumption. The figure shows that MOCDS does not find more comfortable driving strategies than MODS when traveling time is short (the driving strategies outside the dashed rectangle in Figure 6), since MODS finds driving strategies with short traveling time that are already comfortable and cannot be improved in comfort anymore. However, MOCDS finds significantly more comfortable driving strategies than MODS when fuel consumption is low (the driving strategies inside the dashed rectangle in Figure 6). This is due to the fact that MODS finds driving strategies with low fuel consumption that are highly uncomfortable and, therefore, have the most room for improvement. In summary, the results show that MOCDS finds more comfortable driving strategies than MODS, while not significantly deteriorating the other objectives, especially when the fuel consumption is reduced.

Finally, the computation and simulated times are shown in Table 2. It shows that the average computation time per driving strategy is longer than the simulated traveling times of driving strategies but still in the order of minutes.

4 Conclusion

We presented a two-level multiobjective optimization algorithm for discovering comfortable driving strategies (MOCDS). The lower-level algorithm is a deterministic multiobjecitve algorithm that searches for comfortable driving strategies, while the upper-level algorithm is an evolutionary algorithm that searches for the best input-parameter values for the lower-level algorithm. The obtained driving strategies were compared to the driving strategies found by the algorithm that does not optimize the comfort, i.e., MODS. The results show that comfortable driving strategies either rarely change the control actions or reduce the changes of the control actions. Moreover, when comparing the driving strategies with low fuel consumption, those found by MOCDS are significantly more comfortable than those found by MODS. However, when comparing the driving strategies with short traveling time, there is no significant difference in comfort between those found by MOCDS and those found by MODS, since both are already comfortable and cannot be improved anymore.

In the future work, we will test other approaches for increasing the driving comfort. These approaches will include an objective other than jerk. However, the comfortable driving strategies may be obtained by not including the third objective but limiting the search space, e.g., restricting the changes of control actions. It would be also interesting to include the third objective in the algorithms used so far, i.e., predictive control and dynamic programming, and/or limit the search space of these algorithms to compare the obtained driving strategies with those found by MOCDS.

Acknowledgement

We would like to thank Matjaž Gams for his contribution to the research presented in this paper.

References

[1] K. Deb, A. Pratap, S. Agrawal and T. Meyarivan (2002) A fast and elitist multiobjective genetic algorithm: NSGA-II, *IEEE Transactions on Evolutionary Computation*, IEEE Computational Intelligence Society, vol. 6, no. 2, pp. 182–197.

[2] E. Dovgan, M. Gams and B. Filipič (2011) A multiobjective optimization algorithm for discovering driving strategies, *GECCO'11 proceedings*, ACM, New York, pp. 751–754.

[3] E. Dovgan, M. Javorski, M. Gams and B. Filipič (2011) A two-level approach for discovering driving strategies according to conflicting objectives, *Proceedings of the 14th International Multiconference Information Society*, Jožef Stefan Institute, Ljubljana, pp. 41–44.

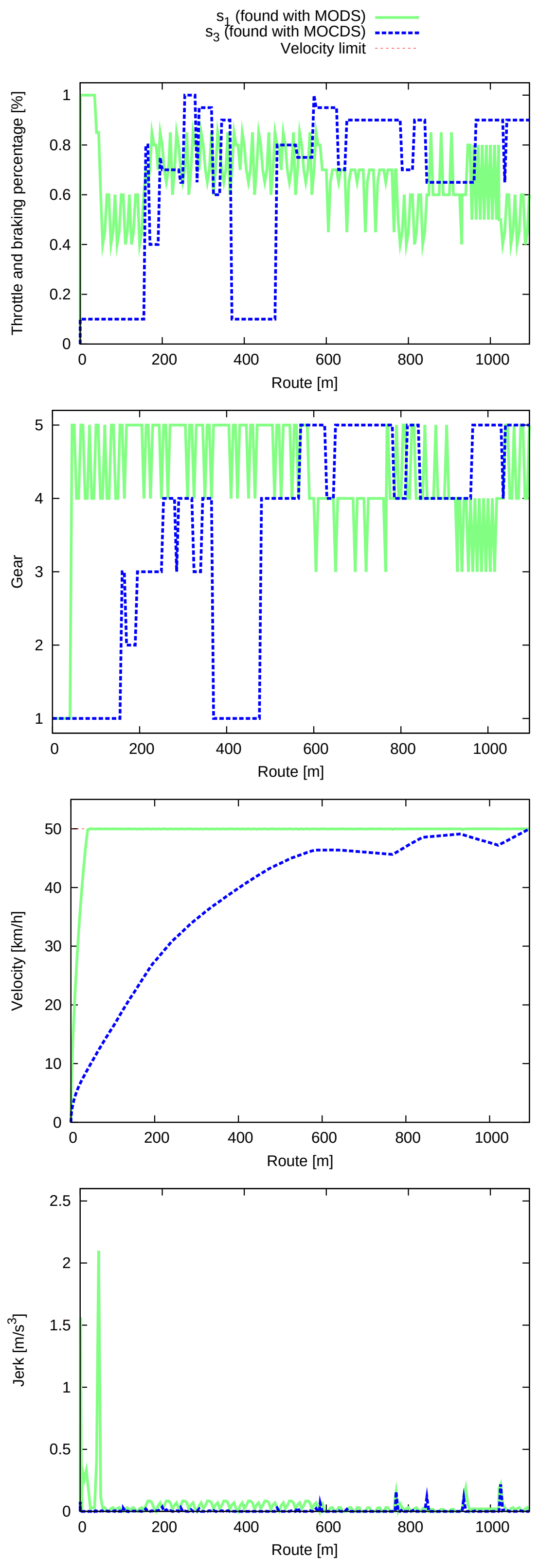


Figure 4: Examples of vehicle behavior obtained by applying the driving strategies with high fuel consumption (s_1 and s_3 from Figure 3).

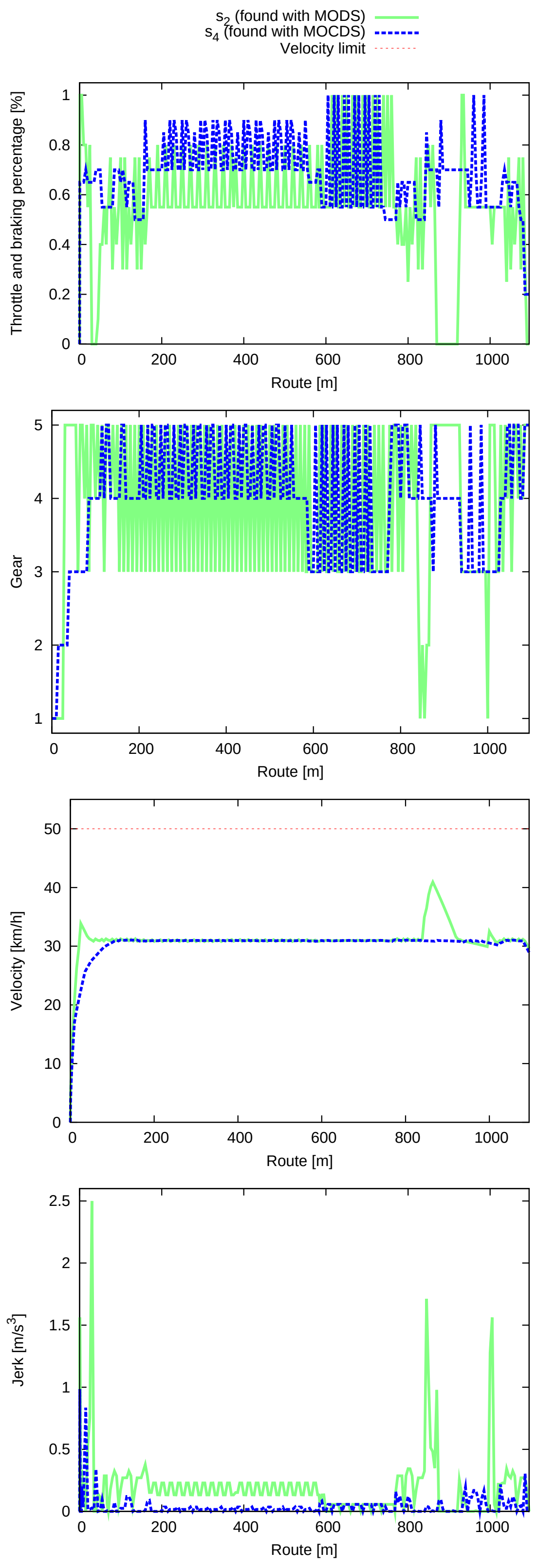
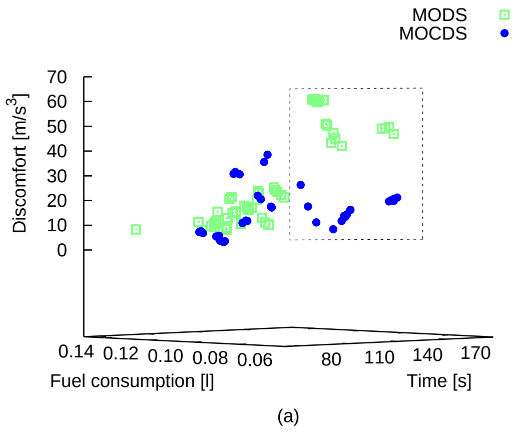


Figure 5: Examples of vehicle behavior obtained by applying the driving strategies with low fuel consumption (s_2 and s_4 from Figure 3).



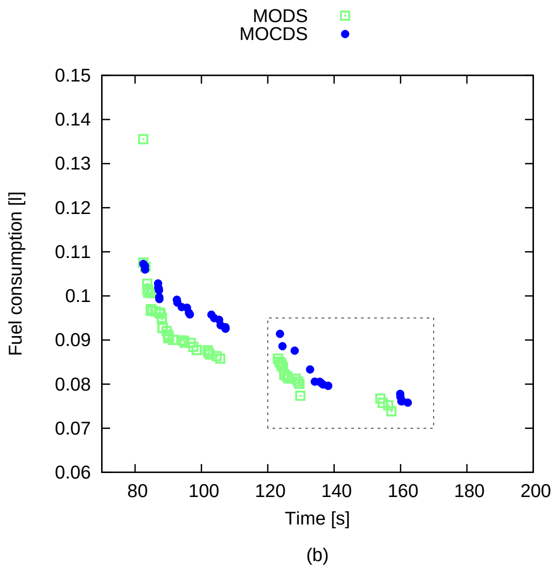


Figure 6: Nondominated driving strategies in the objective space with only the objectives t and c found by MOCDS and MODS. The dashed rectangle denotes the driving strategies with low fuel consumption.

Algorithm	MODS	MOCDS
Total computation time (h:m:s)	6:41:49	23:18:48
Number of found nondominated driving strategies	90	119
Average computation time per driving strategy (m:s)	4:28	11:45
Minimal simulated traveling time of driving strategies (m:s)	1:22	1:22
Maximal simulated traveling time of driving strategies (m:s)	2:37	3:06

Table 2: Comparison of computation and simulated times of MODS and MOCDS.

[4] E. Dovgan, M. Javorski, T. Tušar, M. Gams and B. Filipič (2012) Discovering Driving Strategies with a Multiobjective Optimization Algorithm, submitted for publication.

[5] M. Gerdts (2005) Solving mixed-integer optimal control problems by branch&bound: a case study from automobile test-driving with gear shift, *Optimal Control Applications and Methods*, John Wiley & Sons, vol. 26, no. 1, pp. 1–18.

[6] A. Haj-Fraj and F. Pfeiffer (2001) Optimal control of gear shift operations in automatic transmissions, *Journal of the Franklin Institute*, Elsevier, vol. 338, pp. 371–390.

[7] E. Hellstrom, J. Aslund and L. Nielsen (2010) Design of an efficient algorithm for fuel-optimal look-ahead control, *Control Engineering Practice*, Elsevier, vol. 18, no. 11, pp. 1318–1327.

[8] E. Hellstrom, M. Ivarsson, J. Aslund and L. Nielsen (2009) Look-ahead control for heavy trucks to minimize trip time and fuel consumption, *Control Engineering Practice*, Elsevier, vol. 17, no. 2, pp. 245–254.

[9] C. Kirches, S. Sager, H. G. Bock and J. P. Schloder (2010) Time-optimal control of automobile test drives with gear shifts, *Optimal Control Applications and Methods*, John Wiley & Sons, vol. 31, no. 2, pp. 137–153.

[10] D. Kroushl (2005) *Prius marathoners top 100 mpg*, available online: `http://www.toyota.com/html/hybridsynergyview/2005/fall/marathon.html`.

[11] G. Lechner and H. Naunheimer (1999) *Automotive Transmissions: Fundamentals, Selection, Design and Application*, Springer.

[12] J. Lee (2009) *Vehicle inertia impact on fuel consumption of conventional and hybrid electric vehicles using acceleration and coast driving strategy*, Virginia Polytechnic Institute and State University.

[13] F. Logist, S. Sager, C. Kirches and J. F. Van Impe (2010) Efficient multiple objective optimal control of dynamic systems with integer controls, *Journal of Process Control*, Elsevier, vol. 20, no. 7, pp. 810–822.

[14] A. Nilsson (2009) *Fuel and ride comfort optimization in heavy vehicles*, Linkoping University.

[15] T. Randolph (2007) *Waste heat regeneration systems for internal combustion engines*, available online: `http://www.heat2power.net/downloads/GPC2007/20070618_heat2power_GPC_WHR_presentation.pdf`.

[16] L. Del Re, F. Allgower, L. Glielmo, C. Guardiola and I. Kolmanovsky (2010), *Automotive Model Predictive Control: Models, Methods and Applications*, Springer.

[17] S. J. Russell and P. Norvig (2010) *Artificial Intelligence: A Modern Approach*, Prentice Hall.

[18] E. Vareilles, M. Aldanondo and P. Gaborit (2009) How to take into account piecewise constraints in constraint satisfaction problems, *Engineering Applications of Artificial Intelligence*, Elsevier, vol. 22, no. 4–5, pp. 778–785.

[19] Z. Wu, Y. Liu and G. Pan (2009) A smart car control model for brake comfort based on car following, *IEEE Transactions on Intelligent Transportation Systems*, IEEE Intelligent Transportation Systems Society, vol. 10, no. 1, pp. 42–46.

[20] E. Zitzler and L. Thiele (1999) Multiobjective evolutionary algorithms: A comparative case study and the strength Pareto approach, *IEEE Transactions on Evolutionary Computation*, IEEE Computational Intelligence Society, vol. 3, no. 4, pp. 257–271.

Optimized Rostering of Workforce Subject to Cyclic Requirements

François Ramond and David De Almeida
SNCF - Direction de l'Innovation et de la Recherche,
40 avenue des Terroirs de France, 75611 Paris Cedex 12, France
E-mail: francois.ramond@sncf.fr, david.de_almeida@sncf.fr

Stéphane Dauzère-Pérès
Ecole des Mines de Saint-Etienne
Centre Microélectronique de Provence
880 avenue de Mimet, 13541 Gardanne, France
E-mail: dauzere-peres@emse.fr

Hanif D. Sherali
Virginia Tech, Grado Department of Industrial and Systems Engineering,
250 Durham Hall, Blacksburg, VA 24061, USA
E-mail: hanifs@vt.edu

Keywords: optimization, rostering, workforce

Received: June 23, 2012

SNCF is a large railway transportation company that operates 365 days a year and 24 hours a day. In order to schedule a certain category of workers at train stations and ticket selling points, rosters are designed to cover a cyclical demand. However, the highly combinatorial nature of the rostering problem makes it very difficult to solve it manually, and experts spend a huge amount of time to make them legally feasible and to improve a certain number of preference criteria. This paper presents a mixed-integer programming model to address the cyclical rostering problem using patterns corresponding to feasible blocks of seven days and assigning them to each week of the roster. Some valid inequalities are presented to improve the linear relaxation of the model and thereby enhance computational performance. Implementation results are presented, including comparisons with an alternative daily-variables model

Povzetek: Opisano je optimirano cikli?no razporejanje delavcev z aplikacijo za podjetje SNCF.

1 Introduction

Like many public transportation companies, the French national railways (SNCF, for Société Nationale des Chemins de fer Franąis) require some work to be performed 365 days a year and 24 hours a day. Because of complex legal issues, planning of human resources is very difficult to implement. People responsible for human resources in different operational units spend a considerable amount of time preparing timetables. Yet the plan they finally obtain is rarely optimal with respect to preferences of workers and unions or with respect to costs. Indeed, this kind of problem is highly combinatorial, and so far, no software-based solution approach has been implemented at SNCF. The lack of automation in producing timetables was put under the spotlight when a new regulation of work schedules in 1997 reduced the overall time of work over a year for all employees and required most timetables to be redesigned.

Nevertheless, many papers dealing with personnel scheduling have been published in the open literature, focusing on different problems specific to several fields [20]. Hospitals, public services (firefighting and police units), and airline and railway companies are among the most studied domains. These organizations share the characteristic of being operated 365 days a year and 24 hours a day, which makes workforce scheduling particularly fastidious and justifies the effort to design effective decision-support systems.

Two approaches have been considered for workforce scheduling: the first approach aims at minimizing the costs of production through the number of employees required to perform a certain amount of work [1, 2, 3, 4], whereas the goal of the second approach is to actually schedule the work performed by a certain number of employees with respect to a set of operational constraints, while minimizing costs. Some research is also focused on integrating these two phases into a single stage to produce either cyclic or non-cyclic rosters [14].

In the case of the second approach, Beaumont [5] develops a mixed-integer programming (MIP) model to design cyclic rosters of length one year, including four or five holiday weeks and a certain number of rostered off-days. Constraints on minimal and maximal lengths of work stretches and rest periods are explicitly expressed, and the objective function aims at minimizing costs related to workload coverage and acceptability of the roster. Freling et al. [17]

present a similar problem, which is divided into four parts. A first module enables the feasibility checking of given rosters, a second one is responsible for the generation of feasible rosters, and a third one evaluates each roster with respect to its cost and preferential criteria. Finally, the fourth module selects the best quality rosters through mathematical programming based methods (set partitioning problem).

Problems of workforce scheduling relative to nurses focus more specifically on the satisfaction of employee preferences. Thus, Miller et al. [26] consider two sets of constraints in their integer program, namely hard constraints defining the feasibility set, and soft constraints whose violation is permitted but penalized by an associated cost in the objective function. Sherali et al. [30] develop a mixed-integer program for the resident scheduling problem (RSP) at the *St John Hospital and Medical Center* and exploit the inherent network structure of the problem to design a solution procedure. The advantage of this methodology lies in its capacity to propose compromise solutions when the MIP model turns out to be infeasible. For surveys on nurse rostering problems, we refer the interested reader to [10] and [7]. In a recent paper, Glass and Knight [19] study the nurse rostering problem structure, and propose an mixed-integer programming approach validated on four benchmark problem instances. Another contribution is a methodology for handling continuity between rostering periods.

The problems encountered in the airline and railway industries are, in general, divided into two sub-problems: the Crew Scheduling Problem (CSP) [35] and the Crew Rostering Problem (CRP) [11, 16, 22, 23]. The CSP deals with the design of *pairings*, which are sequences of tasks and rest periods lasting typically 24 to 72 hours, and starting and ending at the same domicile location. The CRP can be seen as the natural consequence of the CSP because its aim is to assign pairings to specific employees and to sequence them over a longer term planning horizon, typically one to four months. Caprara et al. [8, 9] address the Crew Scheduling Problem and the Crew Rostering Problem for the railway industry. In [9], a procedure is proposed to combine the CSP and the CRP in an iterative fashion, through the computation of the Lagrangian cost of potential pairings. De Pont [12] discusses the construction of rosters for Dutch railway operators. Other research in the context of the airline industry is concerned with the integration of aircraft routing and crew scheduling (see for instance [24] and [25]). This is not relevant in our case since we are interested in designing rosters for so-called sedentary workers.

Similar problems to those cited above can be found in urban transportation companies. The problem studied by Townsend [34] for the bus drivers of "London Regional Transport" is of certain interest in the sense that it has similarities with the design of rosters at SNCF. A solution procedure based on the utilization of pre-built patterns of one, four, or five weeks is proposed. This procedure is problem-specific and cannot be used here. However, the use of pre-built patterns, such as the use of pairings in the CRP, is worthwhile since it enables a higher level of abstraction in the formulation of the model. We refer the reader to [15] for an annotated bibliography of personnel scheduling and rostering.

Regarding commercial software, a few rostering packages are available on the market. The software modules developed by Quintiq are among the most popular ones used in the industry and rely heavily on Operations Research techniques (see [28]). The rostering problem described in this paper is, however, too specific to be solved using a generic software tool.

The present paper makes the following specific contributions:

1. We describe the employee rostering problem faced at SNCF for a class of workers, and discuss related specific labor rules and work restrictions along with employee and management performance criteria for assuring high quality rosters.

2. We design an MIP model using special weekly pattern blocks composed of feasible compositions of work stretches and rest periods, and we further enhance the solvability of the model by incorporating two classes of valid inequalities. The proposed model is structured to facilitate a direct implementation using a commercial MIP software package (we used CPLEX [21] for this purpose).

3. We present computational results based on real data at SNCF and provide comparisons against an alternative daily-variable model. Some practical implementation guidelines are also discussed.

The remainder of this paper is organized as follows. Section 2 explains the rosters to be generated. Section 3 develops the formulation designed to solve the problem. Some experimental results are presented in Section 4. Section 5 concludes the paper with some perspectives for future research.

2 Problem Definition

A roster is basically a table (see Figure 1) whose rows correspond to work cycles; there are as many rows in the roster as employees in the team. Each row or work cycle is a sequence of work stretches (sequence of consecutive working days associated with different shifts - for example, morning, evening, or night) and rest days. Once designed and validated, the roster is used until a major change arises such as, for example, an evolution of the requirements in terms of personnel or a modification of labor policies or union rules.

The idea behind a roster is that every worker in the team begins on a different row and then progresses cyclically through the rows of the roster. Hence, the first worker begins on the first row, and then continues for the next cycle according to the second row, and so on. Likewise, the second worker begins with the second row and proceeds cyclically down the roster, returning back to the first row. As a

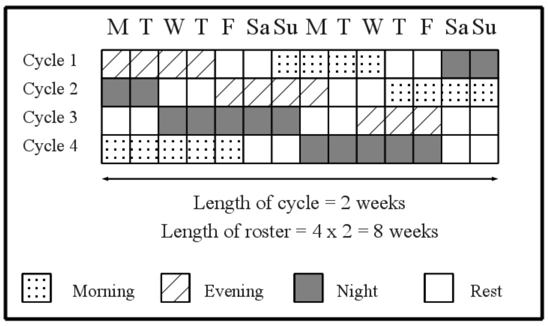

M T W T F Sa Su M T W T F Sa Su
Cycle 1
Cycle 2
Cycle 3
Cycle 4
Length of cycle = 2 weeks
Length of roster = 4 x 2 = 8 weeks
Morning
Evening
Night
Rest

Figure 1: Example of a roster with three daily shifts and cycle length of two weeks

result, the work covered by the team of workers is cyclical, with a period equal to the length of the cycle of the roster. The work covered by a given employee is also cyclical, with the associated period being the total length of the roster (product of the length of the cycle with the number of rows).

In practice, for economic reasons, the workload is not fully covered by the employees in the roster. Certain *reserve employees* are typically called to cover the shifts that are not covered by the team scheduled in the roster. Although such reserve employees are called on a regular basis by a certain number of rosters corresponding to their skills and experience, their work is not scheduled through the use of rosters.

Once the phase of roster design is complete, the process of scheduling is executed using this roster. This has one major advantage: it does not require any modification from the human resources managers, except for slight adjustments to take production disturbances into account and to plan the holidays of the employees. It also guarantees a certain equity between employees in a given roster, since they all share the same plan (although with different starting points). Finally, workers enjoy transparency with respect to their work schedules. However, major changes cannot be made to the schedules without having to build a new roster; hence leading to a lack of flexibility.

2.1 Labor rules

Rosters have to comply with various rules described in [32, 33]. Recall that, in this paper, rosters are constructed for sedentary workers that operate, for instance, at train stations and ticket selling points. To clarify the following, a distinction is made between *periodic* and *daily* rests: a periodic rest designates the off-day(s) between work stretches, whereas a daily rest is the rest period between two consecutive workdays within a work stretch.

A work stretch cannot last less than three days or longer than five days. Periodic rests must last one, two, or three consecutive off-days.

Furthermore, there are two categories of employees: one is given 114 off-days per year, and the other one is given 118 off-days. The category to which an employee belongs depends on the difficulty of working conditions (and in particular, on the length of night work); all employees that share a given roster, however, must belong to the same category, and so, the data for any instance specifies the particular targeted number of off-days per year for each employee.

Each employee must have at least 12 consecutive pairs of Saturday-Sundays off, and at least a total of 22 Sundays off combined with an adjacent Saturday off or an adjacent Monday off (called *weekends*). For instance, a roster providing yearly 12 Saturday-Sundays off and 10 Sunday-Mondays off (hence, 12+10=22 weekends) satisfies these rules.

Finally, the daily rest between two consecutive workdays must last 12 hours at a minimum. Also, starting with the first off-day of periodic rest there should be a break for at least 36 hours. If this rule cannot be respected, the rest period must last at least 24 hours and the reduction below these 36 hours must be replaced at the latest during the second periodic rest that follows. In any case, the length of the second and third off-days cannot last less than 24 hours.

2.2 Preference criteria

Aside from the constraints cited above, the quality of a roster is evaluated by employees and managers according to several criteria such as:

1. The number of single off-days (i.e., non-consecutive to other off-days) : to be minimized.
2. The range-width of consecutive Saturday-Sundays off over the different cycles of the roster : to be minimized.
3. The range-width of weekends over the different cycles of the roster : to be minimized.
4. The peaks in the use of reserve teams, i.e., the maximal number of calls to reserve employees over the days of the roster : to be minimized.

3 MIP Formulation of the Problem

The formulation of the roster design problem we propose uses a set of pre-built patterns of one week and generates rosters within which each day is either a work day or an off-day (daily shifts are not considered). Section 3.1 discusses four important assumptions, and the MIP model itself is presented in Section 3.2.

3.1 Model assumptions

1. Approximations: Labor rules set the annual number of days of periodic rest to 114 or 118 days, which corresponds to a proportion of $\frac{114}{365}$ or $\frac{118}{365}$ of off-days. However, this proportion can rarely lead to an integer in the case of real-life rosters. For instance, a roster of length 8 weeks (56 days) for employees of Category 1 (114 off-days) must include at least $\frac{114 \times 56}{365} \approx 17.49$ off-days. This number is rounded down (17 in the example) because it is always preferable for a manager to assign an off-day that was rostered as a workday than to assign a workday that was rostered as an off-day. Indeed, managers can perform slight changes on the rosters throughout the year to meet labor rules by calling reserve employees. On the other hand, numbers concerning the minimum proportions of Saturday-Sundays off and weekends will be rounded up.

2. Workload: We assume that the required workload is constant over each week, which therefore facilitates the use of cyclical schedules.

3. Personnel capacities: We assume that personnel capacities are not limited, meaning that whatever the work requirements, there are enough employees to cover the workload. This remark concerns both rostered employees and reserve employees.

4. Shifts: Typically, each workday is partitioned into three shifts : Morning, Evening, and Night. In practice, it is desirable (but not necessary) to assign employees to particular shifts over each work stretch by varying the type of shift in order; for example, Morning, then Night, and then the Evening shift. This facilitates satisfying daily rest constraints and is also desirable from the viewpoint of work rhythm and health perspectives. Note that the proposed model built on one-week patterns ignores such shift considerations, and focuses mainly on determining work stretches comprising workdays and off-days. We assume that managers would determine shift assignments along with any subsequent tweaking of the generated schedules as necessary at a later stage. (Section 4.2 provides additional discussion, including possible enhancements in the proposed model.)

3.2 Formulation using one-week patterns

The proposed Mixed-Integer Programming (MIP) model is inspired by research performed in the airline industry and uses one-week patterns that can be considered as "blocks" composed of work stretches and rest periods. They are built such that, within a pattern, the rules relative to the length of work periods (3 to 5 days) and rest periods (1 to 3 days) are respected. Some constraints on the succession of patterns ensure that these rules are also respected when coupling two adjacent patterns. A limited number of patterns is used, which represents the exhaustive set of all different "types" of weeks respecting labor rules from the basic work-or-rest point of view. Thus, these patterns only include two types of days, which are work days and rest days. Such patterns are assumed to be generated *a priori* using historical experience and managerial insights. Column generation approaches, as for example reviewed in [18] and [13] could be used alternatively – we recommend such an investigation for future research. The set of patterns used in our model is denoted by P.

3.2.1 Model parameters

An instance of the problem is fully defined by the following parameters:
- The number of daily shifts (nds) used in the roster. This is a positive integer.
- The work requirements in number of employees ($req_{s,d}$) for each shift $s \in S = \{0, 1, ..., nds - 1\}$ and each day of the week $d \in DW = \{1, 2, ..., 7\}$ (1 for Monday to 7 for Sunday). Note that for the proposed one-week patterns model, only the aggregate requirement $\sum_{s \in S} req_{s,d}$ for each day $d \in DW$ is of relevance.
- The number of cycles in the roster, corresponding to the number of employees in the team: ncr ($CR = \{1, 2, ..., ncr\}$ denotes the set of cycles).
- The number of weeks in a cycle: nwc ($WC = \{1, 2, ..., nwc\}$ denotes the set of weeks of a cycle).
- The number of weeks in the roster: $nwr = nwc \times ncr$ ($WR = \{1, 2, ..., nwr\}$ denotes the set of weeks of the roster, indexed consecutively in order of occurence over the cycles of the roster).
- The number of days in a cycle of the roster: $ndc = 7 \times nwc$ ($DC = \{1, 2, ..., ndc\}$ denotes the set of days of a cycle, indexed consecutively in order of occurence over the cycle).
- The number of days in the roster: $ndr = 7 \times nwr$ ($DR = \{1, 2, ..., ndr\}$ represents the set of days in the roster).
- The minimum numbers of off-days, Saturday-Sundays off, and weekends off to be included in the roster, respectively: $minNbOd$, $minNbSatSun$, and $minNbWkendOff$.
- The nature of day d of pattern p, $pattern_{p,d}$, is equal to 0 if d is a rest day and equal to 1 if d is a work day. The index d varies between 1 (Monday) and 7 (Sunday).
- The binary parameter $potMonSingle_p$ is equal to 1 if Monday of pattern p is a rest day and Tuesday of the same pattern p is a work day, and 0 otherwise. In this case, Monday of pattern p is a *potential* single off-day.
- The parameter $potSunSingle_p$ is equal to 1 if Sunday of pattern p is a rest day and Saturday of the same pattern is a work day.
- The number of single off-days within pattern p (i.e., from Tuesday to Saturday) is represented by $withinSingle_p$.
- The parameter $pattern_{p,67}$ is equal to 1 if both Saturday and Sunday of pattern p are rest days, 0 otherwise. Note that these last four parameters are computed while constructing the actual MIP model.
- $cannotFollow_p$ are sets that are used to determine successive pairs of patterns that would violate the rules stating that work stretches vary "between three and five days" and

rest periods “between one and three days”:

$$cannotFollow_p = \{p' \in P : p' \text{cannot be selected just after } p \text{ in a roster}\}.$$

3.2.2 Principal decision variables

The principal decision variables of this model decide whether pattern p is associated with week w of the roster:

$$x_{p,w} = 1 \text{ if pattern } p \text{ is associated with week } w, \text{ and 0 otherwise.}$$

3.2.3 Auxiliary decision variables

- $satSun_w$ is equal to 1 if Saturday and Sunday of week w of the roster are both rest days, and 0 otherwise.
- $wKend_w$ is equal to 1 if $satSun_w$ is equal to 1 *or* if Sunday of week w and Monday of week $w + 1$ are both off-days, and 0 otherwise.
- likewise, $sunMon_w$ is equal to 1 if Sunday of week w and Monday of week $w + 1$ are both off-days, and 0 otherwise.
- $monSingle_w$ (respectively $sunSingle_w$) is equal to 1 if Monday (respectively Sunday) of week w is a single off-day, and 0 otherwise.
- $nSingle_w$ is an integer variable equal to the number of single off-days of week w of the roster.
- res_d is an integer variable equal to the number of reserve employees used to fully cover the workload on day d of a cycle.
- $nSingleOffDays$ is an integer variable associated with the number of single off-days.
- $minSatSun$ and $maxSatSun$ are integer variables associated respectively with the minimal and maximal numbers of consecutive Saturday-Sundays off over all cycles, and $diffMinMaxSatSun$ denotes their difference.
- $minWkend$ and $maxWkend$ are, likewise, integer variables associated respectively with the minimal and maximal numbers of weekends off over all cycles of the roster, and $diffMinMaxWkend$ denotes their difference.
- Other dependent variables are evident through the constraint definitions below.

3.2.4 Constraints of the formulation

The unique choice of pattern for each week of the roster is imposed by:

$$\sum_{p\in P} x_{p,w} = 1, \quad \forall w \in WR. \tag{1}$$

The constraint on the admissible succession of patterns is enforced by (in light of (1)):

$$x_{p,w} + \sum_{p' \in cannotFollow_p} x_{p',w+1} \le 1, \quad \forall w \in WR, \quad \forall p \in P. \tag{2}$$

The number of Mondays that are single off-days is defined by the following constraint, which imposes that $monSingle_w$ is equal to 1 if $potMonSingle_p = 1$ (for pattern p associated with week w) and if the adjacent Sunday is a work day:

$$\begin{aligned} monSingle_w &\ge \sum_{p\in P}(x_{p,w} \cdot potMonSingle_p) & (3)\\ &+ \sum_{p\in P}(x_{p,w-1} \cdot pattern_{p,7}) & \\ &-1, \forall w \in WR. & (4)\\ monSingle_w &\ge 0, \qquad \forall w \in WR. & (5)\end{aligned}$$

The same types of constraints are used to determine the number of Sundays that are single off-days, $sunSingle_w$. Then, the number of single off-days in week w is given by the sum of single off-days within pattern p associated with week w, plus Monday or Sunday if these are single off-days:

$$nSingle_w \ge \sum_{p\in P}(x_{p,w} \cdot withinSingle_p) + monSingle_w + sunSingle_w, \quad \forall w \in WR. \tag{6}$$

Accordingly, the total number of single off-days in the roster is given by:

$$nSingleOffDays = \sum_{w\in WR} nSingle_w. \tag{7}$$

Constraint (8) determines if Saturday or Sunday of week w are off-days, and, if both are off-days, sets $satSun_w$ to 1.

$$satSun_w = \sum_{p\in P} x_{p,w} \cdot pattern_{p,67}, \forall w \in WR. \tag{8}$$

Constraints (9) – (14), expressed for Sunday and Monday, enable to define if Sunday of week w and the consecutive Monday of week $w + 1$ are off-days, in which case $sunMon_w$ is set equal to 1. It is more complex to express here, compared with the above Saturday-Sunday constraint, because the value of the variable depends on the choice of patterns for both weeks w and $w + 1$.

$$sun_w = \sum_{p\in P} x_{p,w} \cdot pattern_{p,7}, \forall w \in WR. \tag{9}$$

$$mon_w = \sum_{p\in P} x_{p,w} \cdot pattern_{p,1}, \forall w \in WR. \tag{10}$$

$$sunMon_w \ge 1 - sun_w - mon_{w+1}, \forall w \in WR. \tag{11}$$

$$\begin{aligned} sunMon_w &\le 1 - sun_w, \forall w \in WR, & (12)\\ sunMon_w &\le 1 - mon_w, \forall w \in WR, & (13)\\ sunMon_w &\ge 0, \forall w \in WR. & (14)\end{aligned}$$

The definition of weekends is then obtained by (15) to (18):

$$\begin{aligned} wKend_w &\geq satSun_w, \forall w \in WR. && (15)\\ wKend_w &\geq sunMon_w, \forall w \in WR. && (16)\\ wKend_w &\leq satSun_w + sunMon_w, && (17)\\ & \quad \forall w \in WR. && (18)\end{aligned}$$

Note that Equation (19) helps further tighten the LP relaxation, besides enforcing $wKend_w \leq 1$.

$$wKend_w \leq 1 - sun_w, \forall w \in WR. \qquad (19)$$

The number of consecutive Saturday-Sundays off in each cycle, as well as the difference between the maximum and the minimum values over all cycles, are determined through Constraints (20) to (23).

$$satSunCycle_c = \sum_{w \in WC} satSun_{nwc \cdot (c-1)+w}, \forall c \in CR. \qquad (20)$$

$$maxSatSun \geq satSunCycle_c, \forall c \in CR. \qquad (21)$$

$$minSatSun \leq satSunCycle_c, \forall c \in CR. \qquad (22)$$

$$diffMinMaxSatSun = maxSatSun - minSatSun. \qquad (23)$$

Similar constraints are used to compute $wKendCycle_c$, $minWkend$, $maxWkend$, and $diffMinMaxWkend$ for weekends.

The respecting of labor rules on the number of consecutive Saturday-Sundays off, on the number of weekends, and on the total number of off-days over the roster is ensured via Constraints (24), (25), and (26), respectively.

$$\sum_{w \in WR} satSun_w \geq minNbSatSun. \qquad (24)$$

$$\sum_{w \in WR} wKend_w \geq minNbWkendOff. \qquad (25)$$

$$\sum_{\substack{w \in WR\\ p \in P}} \sum_{d=1}^{7} x_{p,w} \cdot pattern_{p,d} = ndr - minNbOd. \qquad (26)$$

Reserve calls (res_d) on each day d of a cycle are equal to the total requirements minus the shifts assigned to employees in the roster, as expressed by the following constraint (in which $\bmod^+ 7$ designates an integer between 1 and 7, computed via modulo 7, except that a modulo value of 0 is replaced by 7):

$$\begin{aligned} res_d = & \sum_{s \in S} req_{s,(d \bmod^+ 7)} \\ & - \sum_{\substack{c \in CR\\ p \in P}} x_{p,(c-1)\cdot nwc + \lceil d/7 \rceil} \cdot pattern_{p,(d \bmod^+ 7)}, \\ & \forall d \in DC. \end{aligned} \qquad (27)$$

The maximum number of reserve calls over all days d of a cycle is then bounded by res_d.

$$maxRes \geq res_d, \forall d \in DC. \qquad (28)$$

Finally, all auxiliary variables are automatically explicitly restricted to be binary or integer variables, where the principal decision variables $x_{p,w}$ are required to be binary-valued.

$$x_{p,w} \in \{0,1\}, \qquad \forall p \in P, \forall w \in WR. \qquad (29)$$

3.2.5 Objective function

The objective function is a weighted sum (with suitable positive weights A, B, C, and D prioritizing the different terms) of the criteria described in Section 2.2, expressing the desirability of the roster from the point of view of both employees and managing staff:

$$\begin{aligned} & Minimize: \\ A \cdot & \ nSingleOffDays \text{ (see Constraint (7))} \\ +B \cdot & \ diffMinMaxSatSun \\ & \text{(see Constraint (23))} \\ +C \cdot & \ diffMinMaxWkend \\ & \text{(see statement after Constraint (23))} \\ +D \cdot & \ maxRes \text{ (see Constraint (28)).} \end{aligned}$$

3.2.6 Model symmetry

Note that the model possesses inherent symmetries due to its cyclical nature that could be inhibited to achieve greater computational efficiency (see [31], for example). In particular, to address this issue, we tried (see [29]) to add some constraints expressing the fact that the first pattern of the first cycle should be the one having the lowest index among all patterns assigned to the beginning of cycles. Some preliminary tests revealed that these symmetry-defeating restrictions did not help much in reducing the computational times, so we did not go further in this direction. However, we advocate a further investigation of this issue for future research.

3.3 Formulation using daily variables

Another formulation to address the cyclical rostering problem at SNCF was also developed based on daily variables that determine, for each day d of the roster, whether a rest period or a work period of any permitted duration, and any required shift among Morning, Night, and Evening, begins on day d (see [6] and [29]). This formulation turned out to be computationally prohibitive due to the large number of integer variables, and we refer the interested reader to [29] for details.

4 Model Refinements and Experimental Results

All computations were performed on a PC equipped with a 2.4 GHz processor, 512 MB memory, and using the ILOG CPLEX 8.0 optimization software with default settings (although more efficient versions of CPLEX presently exist [21], this was the software available to us at the time of the study). The method used to solve the MIP model is the Branch & Cut algorithm [27] implemented within CPLEX and the runs were executed until an optimal solution was found.

Some additional refinements were made to the model for improving its computational performance. First, we relaxed all auxiliary varibles, earlier defined as integer variables. Integer variables are, in general, known to be difficult to deal with. However, this extensive use of integer variables is not necessary since many of them are simply defined as intermediate variables that depend only on the principal variables of the formulation. Indeed, these variables automatically take on integer values at optimality.

Furthermore, we introduced some valid inequalities (or cuts) in the model before calling CPLEX to solve the problem. These cuts simply deal with the values that can be taken by some variables under given conditions:

1. We explicitly imposed $maxRes \geq 1$ if the number of employees in the roster is *a priori* known to be insufficient to cover the entire workload.

2. $diffMinMaxSatSun$ and $diffMinMaxWkend$ cannot be equal to 0 if the number of consecutive Saturday-Sundays off (respectively, weekends off) that is specified to be included in the roster is not a multiple of the number of cycles. For example, if one wants to include five Saturday-Sundays off and nine weekends off in a four-cycle roster, it is not possible to obtain an equal number of Saturday-Sundays and weekends off in each cycle. Hence, we specifically impose $diffMinMaxSatSun \geq 1$ and $diffMinMaxWkend \geq 1$ in such cases.

Our test set was comprised of $432 = 16 \times 27$ realistic instances constructed by composing :

– 16 combinations of employee requirements and numbers of weeks per cycle corresponding to real-world instances that vary between 2 and 4 employees per day, and between 2 and 5 weeks per cycle; along with:

– 27 $(= 3 \times 3 \times 3)$ realistic sets of values for the minimum annual number of: (a) off-days (114, 118, or 122, where this third case was introduced for experimental purposes); (b) consecutive Saturday-Sunday off-days (10, 12, or 14); and (c) number of weekends(19, 22, or 25).

Table 1 displays results on the computational times (in seconds) required to obtain optimal rosters with the different cuts. We tested the aforementioned 432 instances for each configuration : without the two cuts mentioned above, with cut (1), with cuts (2), and with cuts (1) and (2). The columns of this table provide the average and maximum CPU times over all the test problems, and the number of problems whose computation requires more than 1 s., 10 s., and 100 s., and the mean CPU times in these three categories.

The results presented in Table 1 show that cut (1), by itself or in combination with cut (2), does not help the resolution process as it increases the mean CPU time. More recent versions of CPLEX might be able to handle such lower-bounding restrictions more efficiently. However, adding cut (2) to the model leads to major improvements in the computational effort: the mean CPU time with (2) is smaller by a factor of more than five, and there is only one instance requiring more than 100 s. In general, cuts alter LP relaxation solutions of different nodes of the branch-and-bound tree, thereby affecting the choice of branching variables, and consequently result in varying effects on overall performance. In our runs, for three particular instances, where the root node relaxation was unaffected (although other node relaxations can still be affected after branching), cut (1) increased the CPU times for two instances from 7.7 to 9.1 seconds and from 55.8 to 255.0 seconds, respectively, but decreased it for a third instance from 195 to 109.5 seconds. On average, cut (1) increased the overall effort as indicated in Table 1.

Since the best results were obtained by adding cuts (2) to the formulation, this refinement was included in the model for further experiments.

We also observe from Table 1 that the computational times to solve the roster generation problem vary widely. Many parameters of the formulation impact the CPU times but the computational times are mainly dependent on two of them that determine the number of integer variables: the number of weeks in the roster (nwr) and the number of cycles in the roster (ncr). Tables 2 and 3 present mean CPU times obtained on sets of instances of the problem having varying values of nwr and ncr. The general trend is an exponential increase of the computational times with respect to these two parameters.

	Mean	Max	Nb of problems			Mean CPU		
	CPU	CPU	>1s.	>10s.	>100s.	>1s.	>10s.	>100s.
No cuts	11.3	498.5	167	66	9	28.7	66.7	278.9
Cut (1)	12.7	423.9	143	73	14	37.9	70.5	240.1
Cuts (2)	2.1	121.2	75	12	1	11.4	55.4	121.2
Cuts (1) & (2)	12.8	385.7	162	84	21	33.5	61.9	184.7

Table 1: Number of problems solved and CPU times (in s.) by introducing cuts within the formulation using one-week patterns

nwr	CPU
4	0.09
6	0.12
8	0.72
9	0.13
10	0.34
12	11.48
16	1,022.64
20	1,640.15

Table 2: Mean CPU times (in s.) with respect to nwr for the formulation using one-week patterns

ncr	CPU
2	0.34
3	0.15
4	304.83
8	5,108.80

Table 3: Mean CPU times (in s.) with respect to ncr for the formulation using one-week patterns

4.1 Computational comparison with the formulation using daily variables

For the formulation using daily variables mentioned in Section 3.3, over the 432 tested instances, 360 were infeasible and, among the remaining 72 feasible instances, there was no case where this model required less time than the formulation using one-week patterns. For some particularly hard to solve instances (where ncr equals 4 and nwc equals 3 – hence, the rosters comprise 12 weeks overall), the required CPU time ranged from 1,498 s. to 7,972 s. with the daily variables model, whereas the CPU times ranged from 1 s. to 98 s. with the formulation using one-week patterns. We refer the interested reader to [29] for further details on this daily-variables model.

4.2 Practical use of the model

The formulation using one-week patterns reaches optimal solutions much faster than the formulation using daily variables - 10 to 20 times faster on average. Therefore, if multiple solutions are to be produced quickly, for example for the sake of comparison of different feasible rosters, the formulation using one-week patterns is well adapted.

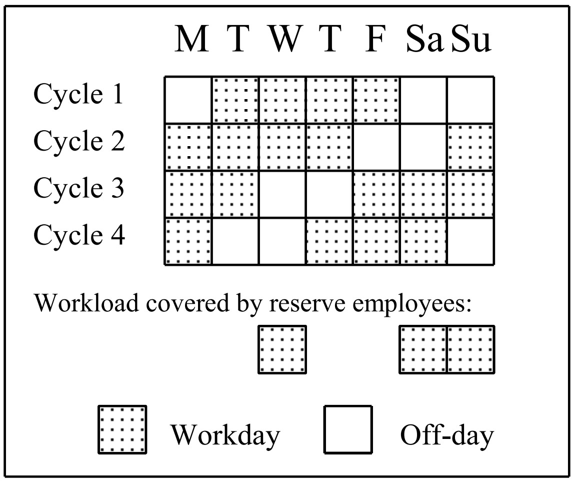


Figure 2: A valid roster for the formulation using one-week patterns

One should keep in mind that the formulation using one-week patterns specifies only the nature of each day of the roster as either a workday or an off-day. This has a significant impact on computational times but also has consequences on the usability of the rosters. For example, consider the roster presented in Figure 2. This roster has 10 off-days, one Saturday-Sunday off-day, and two weekend off-days. Its work stretches are three to five days long, and the rest periods are between one and three days. Hence, it is valid with respect to work regulations. However, the sequence of assigned shifts Morning $\rightarrow$ Night $\rightarrow$ Evening $\rightarrow$ Morning, and so on, one type for each work stretch, which is commonly implemented for three-shift rosters, cannot be respected here because the roster has five work stretches, which is not a multiple of three. On the other hand, the formulation using daily variables directly handles such restrictions on sequences of shifts.

The pattern-based formulation recognizes a large number of rosters as valid, but these rosters do not necessarily correspond to certain expectations of workforce managers. A way to resolve this issue would be to use one-week patterns that specify for each workday the type of shift that is worked. If we consider three shifts that alternate with each other according to the sequence Morning, then Night, and then Evening, and so on, the number of such patterns is exactly three times the number of "basic" patterns (indeed, for each basic pattern, there are three possible choices for the first workday, and the nature of the following workdays derives from this initial choice). This new set of patterns is still of manageable size for the MIP formulation using patterns, and could be used instead of the original set of patterns to produce more precise rosters. Moreover, the pattern-based formulation can be useful to design other types of rosters, for example, rosters having only one or two different shifts.

Finally, we comment on handling infeasibilities in the generated solutions in practice. In such cases, we advise the user to try another computational run with a different number of weeks per cycle. This is acceptable from a practitioner's point of view since roster specialists know that some numbers of weeks per cycle fit better certain requirements that need to be covered (although there is no formal proof of this), and thus accordingly, they manually adapt the roster width to meet these requirements.

5 Summary and Conclusions

The research described in this paper on the rostering problem at SNCF has enabled the design of a model using one-week patterns, producing rosters where each day is defined as being either a workday or an off-day. As discussed in Section 4, it can be solved very fast and allows users to generate several rosters for the sake of comparison, or to obtain "long" rosters within reasonable computational times.

5.1 From theoretical models to an industrial tool

This study was initiated by SNCF to develop a decision-support system that would be available to all workforce managers of the railway company. A prototype of this system was designed based on the proposed formulation, and has been used online by several SNCF workforce managers via the company intranet. The system is basically composed of a set of dynamical Web pages. Once problem instances are submitted and optimal solutions are found, all relevant pieces of information are written in a database and displayed in a graphical manner on a Web page.

The feedback obtained from the users is very encouraging. The consensus is that the prototype is very helpful and saves a great deal of time. Also, the solutions that are proposed are sometimes quite different from the ones that workforce managers would have found manually, thus enlarging the field of possibilities. Of course, efforts are needed to improve the user interface and the reliability of the system, but the prototype proved the need and interest for such a decision-support system. The next step is the industrialization of the tool to make it available to all workforce managers of the company.

5.2 Recommendations for future research

Although the formulations designed in this study are capable of designing feasible rosters in acceptable times, the following features can be investigated to further improve the models:

- Use more specific patterns in which the shifts are additionally assigned to workdays, and introduce new sequence constraints in the formulation using one-week patterns to produce more detailed and ready-to-use rosters.

- Implement column generation to derive patterns, or use longer patterns, that are dynamically generated by column generation, to reduce computational times.

Acknowledgement

This work has been supported in part by the National Science Foundation under Grant No. DMI-0094462.

References

[1] K R Baker. Scheduling a full-time workforce to meet cyclic staffing requirements. *Management Science*, 20(12):1561–1568, 1974.

[2] K R Baker and M J Magazine. Workforce scheduling with cyclic demands and day-off constraints. *Management Science*, 24(2):161–167, 1977.

[3] J F Bard, C Binici, and A H DeSilva. Staff scheduling at the united states postal service. *Computers & Operations Research*, 30(5):745–771, 2003.

[4] J J Bartholdi III, J B Orlin, and H D Ratliff. Cyclic scheduling via integer programs with circular ones. *Operations Research*, 28(5):1074–1085, 1980.

[5] N Beamont. Using mixed-integer programming to design employee rosters. *Journal of the Operational Research Society*, 48(6):585–590, 1997.

[6] C Bentz. Conception des roulements pour le personnel de l'escale. SNCF Internship report (unpublished), 2002.

[7] E K Burke, P De Causmaecker, G Vanden Berghe, and H Van Landeghem. The state of the art of nurse rostering. *Journal of Scheduling*, 7(6):441–499, 2004.

[8] A Caprara, M Monaci, and P Toth. A global method for crew planning in railway applications. *Computer-Aided Transit Scheduling*, 505:17–36, 2001.

[9] A Caprara, P Toth, D Vigo, and M Fischetti. Modeling and solving the crew rostering problem. *Operations Research*, 46(6):820–830, 1998.

[10] B Cheang, H Li, A Lim, and B Rodrigues. Nurse rostering problems–a bibliographic survey. *European Journal of Operational Research*, 151(3):447 – 460, 2003.

[11] H Dawid, J König, and C Strauss. An enhanced rostering model for airline crews. *Computers and Operations Research*, 28(7):671–688, 2001.

[12] G De Pont. *Personalized crew rostering at Netherlands railways*. PhD thesis, University of Tilburg, The Netherlands, 2006.

[13] G Desaulniers, J Desrosiers, and M M Solomon. *Column generation*. Springer Science, New York, NY, 2005.

[14] A T Ernst, H Jiang, M Krishnamoorthy, H Nott, and D Sier. An integrated optimization model for train crew management. *Annals of Operations Research*, 108:211–224, 2001.

[15] A T Ernst, H Jiang, M Krishnamoorthy, B Owens, and D Shier. An annotated bibliography of personnel scheduling and rostering. *Annals of Operations Research*, 127(1):21–144, 2004.

[16] T Fahle, U Junker, S E Karisch, N Kohl, M Sellmann, and B Vaaben. Constraint programming based column generation for crew assignment. *Journal of Heuristics*, 8(1):59–81, 2002.

[17] R Freling, N Piersma, A P Wagelmans, and A van de Wetering. Rostering at a dutch security firm. Technical Report ERS-2001-37-LIS, Erasmus Research Institute of Management (ERIM), The Netherlands, June 2001.

[18] M Gamache, F Soumis, G Marquis, and J Desrosiers. A column generation approach for large-scale aircrew rostering problems. *Operations Research*, 47(2):247–262, February 1999.

[19] C A Glass and R A Knight. The nurse rostering problem: A critical appraisal of the problem structure. *European Journal of Operational Research*, 202(2):379 – 389, 2010.

[20] E D Gunes. Workforce scheduling. Technical report, Bilkent University, Ankara, Turkey, April 1999.

[21] IBM ILOG. IBM ILOG CPLEX Optimizer - High-performance mathematical programming solver for linear programming, mixed integer programming, and quadratic programming, 2012. See `www-01.ibm.com/software/integration/optimization/cplex-optimizer`.

[22] J König and C Strauss. Rostering-integrated service and crew efficiency. *Information Technology and Tourism*, 3(1):27–39, 2000.

[23] N Kohl and S E Karisch. Airline crew rostering: Problem types, modeling, and optimization. *Annals of Operations Research*, 127(1):223–257, 2004.

[24] M Lohatepanont and C Barnhart. Airline schedule planning: Integrated models and algorithms for schedule design and fleet assignment. *Transportation Science*, 38(1):19–32, 2004.

[25] A Mercier and F Soumis. An integrated aircraft routing, crew scheduling and flight retiming model. *Computers and Operations Research*, 34(8):2251–2265, 2007.

[26] H E Miller, W P Pierskalla, and G J Rath. Nurse scheduling using mathematical programming. *Operations Research*, 24(5):857–870, 1976.

[27] G L Nemhauser and L A Wolsey. *Integer and Combinatorial Optimization*. John Wiley and Sons, New York, second edition, 1999.

[28] Quintiq. Rail planning and scheduling solution - planning for success in a changing market, 2012. Commercial brochure available online at `http://connect.quintiq.com/wfo-rail-rail-rostering-buna.html?keyword=\%2Brail\%20\%2Brostering\&matchtype=b\&creative=18828281748\&gclid=CKiUvJv8xrACFUZN4AodX1jwXQ`.

[29] F Ramond. Optimized rostering of workforce subject to cyclic requirements. Master's thesis, Virginia Tech, USA, October 2003.

[30] H D Sherali, M H Ramahi, and Q J Saifee. Hospital resident scheduling problem. *Production Planning and Control*, 13(2):220–233, 2002.

[31] H D Sherali and J C Smith. Improving discrete model representations via symmetry considerations. *Management Science*, 47(10):1396–1407, 2001.

[32] SNCF. *Human Resources Department – Réglementation du travail RH-0077*, Paris, France, February 2000.

[33] SNCF. *Human Resources Department – Réglementation du travail RH-0677, Instruction d'application du décret n^{o}99-1161 du 29 décembre 1999*, Paris, France, December 2000.

[34] W Townsend. An approach to bus-crew roster design in London regional transport. *Journal of the Operational Research Society*, 39(6):543–550, 1988.

[35] P H Vance, C Barnhart, E L Johnson, and G L Nemhauser. Airline crew scheduling: A new formulation and decomposition algorithm. *Operations Research*, 45(2):188–200, 1997.

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 **Slove**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 submit a manuscript at: http://www.informatica.si/Editors/PaperUpload.asp. 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.

QUESTIONNAIRE

☐ Send Informatica free of charge

☐ Yes, we subscribe

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 eighteen 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 is free of charge for major scientific, educational and governmental institutions. Others should subscribe (see the last page of Informatica).

ORDER FORM – INFORMATICA

Name: ....

Title and Profession (optional): ....

....

Home Address and Telephone (optional): ....

....

Office Address and Telephone (optional): ....

....

E-mail Address (optional): ....

Signature and Date: ....

Informatica WWW:

http://www.informatica.si/

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, Vožarski pot 12, 1000 Ljubljana, Slovenia.
The subscription rate for 2012 (Volume 36) 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.
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):
Robotics Society of Slovenia (Jadran Lenarčič)
Slovene Society for Pattern Recognition (Franjo Pernuš)
Slovenian Artificial Intelligence Society; Cognitive Science Society (Matjaž Gams)
Slovenian Society of Mathematicians, Physicists and Astronomers (Bojan Mohar)
Automatic Control Society of Slovenia (Borut Zupančič)
Slovenian Association of Technical and Natural Sciences / Engineering Academy of Slovenia (Igor Grabec)
ACM Slovenia (Dunja Mladenič)

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

The issuing of the Informatica journal is financially supported by the Ministry of Higher Education, Science and Technology, Trg OF 13, 1000 Ljubljana, Slovenia.

Volume 36 Number 3 September 2012 ISSN 0350-5596

Informatica

An International Journal of Computing and Informatics

Informatica **36** (2012) Number 3, pp. 227–337