Volume 20 Number 1 February 1996 ISSN 0350-5596

Informatica

An International Journal of Computing and Informatics

The Slovene Society Informatika, Ljubljana, Slovenia

Informatica **20** (1996) Number 1, pp. 1–158

Informatica

An International Journal of Computing and Informatics

Basic info about Informatica and back issues may be FTP'ed from ftp.arnes.si in magazines/informatica ID: anonymous PASSWORD: <your mail address>
FTP archive may be also accessed with WWW (worldwide web) clients with
URL: `http://www2.ijs.si/~mezi/informatica.html`

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, 61000 Ljubljana, Slovenia.
The subscription rate for 1996 (Volume 20) is
- DEM 50 (US$ 35) for institutions,
- DEM 25 (US$ 17) for individuals, and
- DEM 10 (US$ 7) for students

plus the mail charge DEM 10 (US$ 7).
Claims for missing issues will be honored free of charge within six months after the publication date of the issue.

LaTeX Tech. Support: Borut Žnidar, DALCOM d.o.o., Stegne 27, 61000 Ljubljana, Slovenia.
Lectorship: Fergus F. Smith, AMIDAS d.o.o., Cankarjevo nabrežje 11, Ljubljana, Slovenia.
Printed by Biro M, d.o.o., Žibertova 1, 61000 Ljubljana, Slovenia.

Orders for subscription may be placed by telephone or fax using any major credit card. Please call Mr. R. Murn, Jožef Stefan Institute: Tel (+386) 61 1773 900, Fax (+386) 61 219 385, or use the bank account number 900–27620–5159/4 Ljubljanska banka d.d. Slovenia (LB 50101-678-51841 for domestic subscribers only).

According to the opinion of the Ministry for Informing (number 23/216-92 of March 27, 1992), the scientific journal Informatica is a product of informative matter (point 13 of the tariff number 3), for which the tax of traffic amounts to 5%.

Informatica is published 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 (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 (Janez Peklenik)

Informatica is surveyed by: AI and Robotic Abstracts, AI References, ACM Computing Surveys, Applied Science & Techn. Index, COMPENDEX*PLUS, Computer ASAP, Cur. Cont. & Comp. & Math. Sear., Engineering Index, INSPEC, Mathematical Reviews, Sociological Abstracts, Uncover, Zentralblatt für Mathematik und ihre Grenzgebiete., Linguistics and Language Behaviour Abstracts, Cybernetica Newsletter

The issuing of the Informatica journal is financially supported by the Ministry for Science and Technology, Slovenska 50, 61000 Ljubljana, Slovenia.

PROFILES

In the last few years, in some countries of the East Europe and the Central-East Europe, dramatic political and social changes have taken place. Scientists of these regions have been confronted with elementary professional, even surviving problems, when large research organizations, being supported by the state, closed their doors, when salaries at universities and research institutes highly decreased, and when in some countries even reduced salaries were payed with prolongated delays under conditions of rapid inflation. Though new sources for funding scientific researches have been engaged from abroad (e.g., National Science Foundation in U.S.A., European Communities' Foundations), they have improved the situation only partly, and very often the researchers in these countries need to undertake considerable additional efferts in order to carry out scientific studies, especially fundamental studies.

Professor Vladimir A. Fomichov belongs to the generation of scientists who, notwithstanding all difficulties, have continued to work persistently and fruitfully in the chosen directions. He has an excellent scientific potential which will—as we hope and wish—show the power of new concepts in the formal semantics of natural language. This should afford new important opportunities in the field of information processing concerning language understanding, language generation, knowledge representation, and knowledge transfer. The readers are advised to make them acquainted with a Fomichov's paper in this issue of *Informatica* entitled *A Mathematical Model for Describing Structured Items of Conceptual Level.*

Vladimir A. Fomichov (Fomitchev)

Vladimir A. Fomichov[1] is a professor of computer science at the Moscow State Aviation Technology and also State University and the Moscow State Institute of Electronics and Mathematics (Technical University). He holds a M.Sc. with honours in mathematics and Ph.D. in mathematical computer science (technical cybernetics and information theory) in 1973 and 1983, respectively. The title of his thesis was "Development and Application of Methods of Logic-Algebraic Modelling the Mechanisms of Speech-Formation Semantics".

In 1967, he finished a secondary school in Moscow specialized in mathematics with the gold medal. In 1973, he became a laureate of the All-Union competition of students' scientific works.

Besides in the Russian language, he speaks, reads, and writes in English, German, and French, reads in Polish and Italian.

Membership in Scientific Academies and Associations

V.A. Fomichov is an active member of the New York Academy of Sciences (founded in 1817) since December 1994, with the primary interest in computer science, linguistics, and science education.

He is a member of the International Cybernetics Academy "Ştefan Odobleja" (head office in Lugano, Switzerland; secretariats in Milano, Italy and Bucharest, Romania) since 1994. He is an active member of the Editorial Board of the international journal Informatica since 1992.

Further, he is a scientific member of the International Association for Cybernetics (Namur, Belgium), and a member of the Moscow Mathematical Society.

In October 1995, he was nominated for inclusion into the 14th edition of Marquis Who's Who in the World, scheduled for publication in December 1996. In January 1996, he passed successfully the initial screening, and his data are scheduled for publication.

V.A. Fomichov has been nominated by the Editorial Board of the American Biographical Institute, Inc. (Releigh, NC) for biographical inclusion in the Fifth Edition of "Five Hundred Leaders of Influence" (Publication: Early 1997).

Present and Previous Working Positions

V.A. Fomichov is a full professor of computer science at the Moscow State Aviation Technology University since February of 1966, an associate professor of computer science at the Moscow State Institute of Electronics and Mathematics (Technical University)

[1]The transcription of Russian names from the Cyrillic into Latin usually follows a French transcription. The original name Владимир Александрович Фомичёв, its last part, can be transcribed as Fomitchev, Fomitchov, and/or Fomichov (phonetically).

since 1984, and an associate professor of computer science at the Faculty of Mechanics and Mathematics of the Moscow State University since 1993.

In 1985, he was awarded the title "Associate Professor of Cybernetics" by the Highest Certifying Commission at the Council of Ministers of USSR. In 1973–84, he served as assistant professor of computer science at the Moscow Institute of Electronic Engineering, Faculty of Applied Mathematics, Department of Cybernetics.

His experience in delivering lectures concerns *discrete mathematics, theory of formal languages, theory of algorithms and computational processes, theoretical foundations of informational languages, mathematical foundations of natural-language-processing systems,* and *algorithmic languages and programming.*

International Activity in the Last Year

V.A. Fomichov was a member of the Program Committee of the First Workshop on Applications of Natural Languages to Data Bases (Versailles, France, June 1995).

He worked as the chairperson of the Symposium *From Ideas of Artificial Intelligence and Cognitive Science to New Theories and Methods of Teaching* being part of the 14th International Congress on Cybernetics, in Namur, Belgium, in August 1995.

He was an invited speaker of the First Workshop *Algebraic Methods in Language Processing—AMiLP '95* at the University of Twente, Enschede, The Netherlands, in December 1995.

Main Scientific Results and Interests

V.A. Fomichov is the author of 65 scientific papers and technical reports, including one monograph (1988) in Russian (В.А. Фомичёв: Представление информации средствами К-исчислений, Москва, 1988—Representing Information by Means of K-calculuses, Moscow, 1988).

A. His main innovation is the creation of an appropriate basis for developing a general mathematical theory of natural language (NL) use by intelligent systems (or of NL-communication).

As a consequence, flexible mathematical methods for describing semantics and pragmatics of NL-texts are elaborated which may be effectively applied to and provide *unique* large opportunities for designing *arbitrarily complicated* natural-language-processing systems, including, in particular, biomedical and legal textual (or full-text) databases (TDBs) and TDBs storing administrative, business, or scientific-technological documentation, and for developing powerful and flexible hybrid knowledge representation systems.

B. Proceeding from the ideas of the artificial intelligence theory and cognitive science, he developed together with Olga S. Fomichova a new, highly effective theory of teaching being, in essence, a theory of bridging gaps between the inner world's pictures (or conceptual systems) of a teacher and a learner. That theory, based on new methods of teaching, opens very large new prospects for teaching children and university students, training and retraining of teachers, building *Intelligent Tutoring Systems* with friendly interfaces.

Fomichov's Main Results and Interests in Computer Science and Mathematical Linguistics

Professor Fomichov developed a new, highly powerful and flexible approach to the formalization of semantics and pragmatics of natural language and to the use of mathematical methods in the design of NL-processing systems. This approach is logic-algebraic and is called Integral Formal Semantics (IFS). The central constituent of IFS is the theory of K-calculuses, algebraic systems of conceptual syntax, and K-languages (the KCL-theory).

Judging by the literature, the KCL-theory is now the *only* theory providing the possibility to describe effectively structured meanings (SMs) of arbitrarily complicated real discourses pertaining to science, technology, medicine, business, law, etc.

The KCL-theory opens large prospects to develop new, highly powerful and flexible *hybrid knowledge representation systems (terminological knowledge representation systems).*

The KCL-theory provides unique opportunities for building *semantic representations* of arbitrarily complicated *visual images* (drawings of technical objects, etc.) and hence may be considered as a powerful tool to design *visual information management systems.* A part of the KCL-theory is presented in the author's paper in this issue of *Informatica.*

The development of the KCL-theory can be divided into two periods, the first one (1975–83) and the second one (1984 to present).

In the first period, the theory of S-models, S-calculuses and S-languages, T-calculuses and T-languages (STCL-theory) was developed. This theory provided new powerful mathematical means for describing structured meanings of NL-sentences and complicated NL-discourses—in particular, more powerful than the means provided by Montague Grammars, Discourse Representation Theory, Situation Semantics, and Theory of Generalized Quantifiers.

In the second period, the theory of K-calculuses, algebraic systems of conceptual syntax, and K-languages (the KCL-theory) has been created on the basis of the STCL-theory.

The kernel of the KCL-theory is an original mathematical model describing many regularities of structured meanings of NL-sentences and NL-discourses of arbitrarily great length and complexity.

For the first time such a class of formulas is determined that the structured meaning of practically arbitrary NL-text of arbitrary great length can be represented by a compact formula or a set of formulas from this class. The opportunities to describe structured meanings of NL-sentences and NL-discourses afforded by the KCL-theory by far exceed the opportunities of main popular approaches to the formalization of NL-semantics—Montague Grammar and its non-dynamic extensions, Discourse Representation Theory, Situation Semantics, Theory of Generalized Quantifiers, Dynamic Predicate Logic, and Dynamic Montague Grammar. As a consequence, the KCL-theory provided in 1988 much more powerful mathematical methods to describe structured meanings of sentences and discourses than methods stated in the most complete until recently text-book on mathematical linguistics, B.H. Partee, A. ter Meulen & R.E. Wall: *Mathematical Methods in Linguistics,* Kluwer, Dordrecht, 1990.

The expressive power of standard K-languages exceeds the expressive power of Episodic Logic, suggested by L.K. Schubert and C.H. Hwang.

It is highly important that the KCL-theory provides effective mathematical means both for describing structured meanings of NL-texts, interrelations of surface and semantic structures of NL-texts and for representing knowledge about the reality, developing powerful knowledge representation systems, including terminological languages.

The key advantage of IFS consists in providing a widely-applicable *framework for mathematical modelling the use of natural languages,* i.e. for studying and formalizing the *logic of natural language use.* The analysis indicates that IFS may be effectively applied to the design of arbitrarily complicated NLPSs, including full-text DBs.

IFS provides means, principles, and methods for speeding-up the development of the mathematical theory of natural language use. The author is on the way to develop the foundations of a *theory of information retrieval in textual databases.* A possible direction of studies is also the elaboration of *powerful knowledge representation systems* and *models of knowledge bases.*

During the 1980's and 1990's, V.A. Fomichov has been the head of the research groups (at Technical University) applying the methods and models of IFS to the design of NL-processing systems: NL-text animation subsystems of training complexes for preventing the collision of ships, NL-interfaces to flexible manufacturing systems, expert systems for technical diagnostics, and autonomous intelligent agents (robots). The theoretical basis for these applied studies was provided by several original mathematical models of the correspondences

$$\text{Text}\begin{pmatrix}\text{a sentence,}\\ \text{a discourse,}\\ \text{a discourse}\\ \text{fragment}\end{pmatrix} + \text{Knowledge} \longrightarrow$$

$$\begin{pmatrix}\text{Semantic,}\\ \text{Informational}\end{pmatrix} \text{Representation of Text}$$

and several algorithms of semantic-syntactic analysis.

The KCL-theory affords new possibilities for the development of powerful and flexible hybrid knowledge representation systems processing and greater expressive power than the language developed in the framework of the known project LILOG[2].

The theory of K-calculuses, algebraic systems of conceptual syntax, and K-languages (KCL- theory) is a *tool of unique effectiveness* from the standpoint of designing textual (or full-text) databases (TDBs). The designers of TDBs for the first time get a universal instrument for formal describing (representing as formulas) the conceptual structures (including re-

[2]O. Herzog & C.-R. Rollinger (Eds.), *Text Understanding in LILOG.* Integrating Computational Linguistics and Artificial Intelligence. Final Report on the IBM Germany LILOG-Project, Springer-Verlag, Berlin, 1991.

ferential structures) of analyzed fragments of NL-discourses, the used notions, and knowledge items of other kinds.

Thus, the creation of the KCL-theory means the emergence of a universal effective formal tool for meditating and recording the results of mediating in the course of designing the TDBs.

Besides, Integral Formal Semantics (IFS) creates the preconditions for the development of a mathematical theory of conceptual information retrieval in TDBs as a theory stating *logical foundations* of constructing TDBs.

Fomichov's Main Results in the Field of AI Theory, Cognitive Science, and Theory of Teaching

Simultaneously with the work in the mathematical theory of NL-processing systems, Professor Fomichov has been developing during several years, together with Dr. Olga Fomichova, the foundations of a theory of teaching called the *Theory of Dynamic Conceptual Mappings* (DCM-theory) and being motivated by ideas of the AI theory, cognitive psychology, and cognitive linguistics. The DCM-theory bridges gaps between the inner world's pictures (conceptual systems) of a teacher and a learner. On this basis, highly effective methods of teaching children foreign languages have been elaborated, called the *Methods of Emotional-Imaginative Teaching* (EIT-methods). They may be effectively applied to teaching young children mathematics, other theoretical disciplines, and English as the mother language[3].

The obtained results open new, very large and unique prospects to construct *Intelligent Tutoring Systems* with friendly interfaces helping children and adults to learn foreign languages. DCM-theory and EIT-methods provide new opportunities for training and retraining teachers of varied disciplines.

[3]See, e.g., V.A. Fomichov and O.S. Fomichova: *The Role of the Artificial Intelligence Theory in Developing New, Highly Effective Methods of Foreign Languages Teaching*, in PEG '93, Proceedings of the Seventh International PEG Conference on AI Tools and the Classroom: Theory into Practice. Moray House Institute of Education, Heriot-Watt University, Edingburgh, 1993, pp. 319–329 and V.A. Fomichov and O.S. Fomichova: *The Theory of Dynamic Conceptual Mappings and Its Significance for Education, Cognitive Science, and Artificial Intelligence.* Informatica **18** (1994) [2] 131–148.

Professor Fomichov's Family

The family of V.A. Fomichov is, in a full sense of the word, the academic (scientific) one. His wife, Olga S. Fomichova (Fomitcheva), holds a Ph.D. in philology and is his scientific colleague in one of the fields he works. Her research interest is in the theory and methods of teaching, especially of English as a second language. She speaks Russian, English, and German. She has been nominated by the Editorial Board of the American Biographical Institute, Inc. (Releigh, NC) for biographical inclusion in the Fifth Edition of "Five Hundred Leaders of Influence" (Publication: Early 1997).

Their son Dmitri manifested very early mathematical abilities and skipped three grades after the third grade. In 1994, he finished a secondary school in Moscow specialized in mathematics and entered the Faculty of Computational Mathematics and Cybernetics of the Moscow State University being 13 years old. He likes mathematics, programming, sport swimming and running and is a student of the second year of the Moscow State University.

The mother-in-law, Dr. Ljudmila Udalova, who lives with the family, holds a Ph.D. in biology, and is retired.

Addresses:

Professor Vladimir A. Fomichov

Department of Information Technologies, Faculty of Applied Mathematics and Information Technologies, Moscow State Aviation Technology University, Petrovka Street 27, 103767 Moscow, Russia.

Department of Discrete Mathematics, Faculty of Mechanics and Mathematics, Moscow State University, Vorobyovy Hills, 119899 Moscow, Russia, E-mail: `vaf@nw.math.msu.su`, Phone: +7 (095) 930 98 97.

Address for correspondence: Department of Mathematical Provision of Information-Processing and Control Systems, Faculty of Applied Mathematics, Moscow State Institute of Electronics and Mathematics—Technical University, Bol. Tryokhsvyatitelsky pereulok, 3/12, 109028 Moscow, Russia, E-mail: `root@onti.miem.msk.su`, Fax:+7 (095) 916 28 07 (for Prof. V. Fomichov).

Compiled and commented by *A.P. Železnikar*

A Mathematical Model for Describing Structured Items of Conceptual Level

Vladimir A. Fomichov
An Active Member of the New York Academy of Sciences
Department of Discrete Mathematics, Faculty of Mechanics and Mathematics,
Moscow State University, Vorobyovy Hills, 119899 Moscow, Russia
E-mail: vaf@nw.math.msu.su, Phone: +7 (095) 930 98 97, and
Department of Math. Provision of Information-Processing and Control Systems,
Faculty of Applied Mathematics, Moscow State Inst. of Electronics and Math. - Techn. Univ.
Bol. Tryokhsvyatitelsky pereulok, 3/12, 109028 Moscow, Russia

Keywords: artificial intelligence, natural language processing, natural language semantics, conceptual level, discourse, structured meaning, semantic representation, mathematical model, formal language, integral formal semantics, restricted K-calculus, restricted standard K-language, terminological knowledge representation language, universal metagrammar of conceptual structures, universal stationary metagrammar of deep conceptual syntax

Edited by: Anton P. Železnikar
Received: January 10, 1995 **Revised:** August 16, 1995 **Accepted:** August 22, 1995

A mathematical model is determined and investigated which is destined for describing structured items of conceptual level, in particular, for describing structured meanings (SMs) of natural language (NL) sentences and discourses. It defines a new class of calculuses called restricted K-calculuses and a new class of formal languages called restricted standard K-languages. The model is a modified propositional component of the theory of K-calculuses and standard K-languages being the central constituent of Integral Formal Semantics of NL—an original approach to the mathematical study of NL use.
The formal means provided by the model are convenient, in particular, for describing SMs of discourses with references to the mentioned entities and to the meanings of phrases or larger fragments of texts, with complicated designations of sets, with definitions of concepts. It is shown that the possibilities of the model to describe SMs of sentences and discourses and to represent concepts considerably exceed the possibilities of Montague Grammar, Situation Theory, Theory of Generalized Quantifiers, Discourse Representation Theory, Theory of Conceptual Graphs, Episodic Logic, and several other approaches.
It is pointed out that the results may be used, in particular, for developing mathematical theory of NL use, or mathematical linguocybernetics, creating new hybrid knowledge representation systems, building very powerful and flexible languages of semantic representations of texts. The model is considered as an important contribution to the development of a Universal Metagrammar of Conceptual Structures. The hypothesis is put forward that the model may be interpreted as a variant of a Universal Stationary Metagrammar of Deep Conceptual Syntax.

1 Introduction

In the beginning of the nineties, at least three grandiose scientific-technical tasks were formulated raising new demands to the theory of natural language processing (NLP).

The first and second tasks are working up michov 1981, 1982; Fomitchov 1983, 1984; Fomichov 1988, 1992, 1993b, 1994).

If a model is convenient for describing arbitrary conceptual structures of NL-texts and representing arbitrary knowledge about the world, we'll say about a *Universal Metagrammar of Conceptual Structures*, or a *Universal Conceptual Metagrammar (UCM)*.

The reason to say about a metagrammar but not about a grammar is as follows. A *grammar* of conceptual structures is to be a formal model dealing with elements directly corresponding to some basic conceptual items (like "physical object", "space location", etc.). An example of such semi-formal grammar is provided by the known Conceptual Dependency theory of R. Schank. On the contrary, a *metagrammar* of conceptual structures is to postulate the existence of some classes of conceptual items, to associate in a formal way with arbitrary element from each class certain specific information, and to describe the rules to construct arbitrarily complicated structured conceptual items in a number of steps in accordance with such rules (proceeding from elementary conceptual items and specific information associated with arbitrary elements of considered classes of items).

All approaches mentioned above (including Montague Grammar) practically don't give the cues to the construction of a UCM. The same conclusion is valid also as concerns the Theory of Conceptual Graphs (Sowa 1984, 1991; Ellis 1995) and the Theory of Semantic Structures of R.S. Jackendoff (Jackendoff 1990). Conceptual graphs are rather close to the formulas of first-order logic, and its restrictions are well known. Semantic structures suggested by R.S. Jackendoff may be approximated by a special variant of first-order logic with other sorts of entities besides individuals, as it is shown in (Zwarts & Verkuyl 1994).

The present paper continues a long series of works on IFS aimed at constructing a UCM and including, in particular, the works (Fomichov 1981, 1982; Fomitchov 1983, 1984; Fomichov 1988, 1992, 1993b, 1994). The monograph (Fomichov 1988) contains the foundations of the theory of K-calculuses, algebraic systems of conceptual syntax, and K-languages (the KCL-theory). The kernel of the KCL-theory is the theory of K-calculuses and standard K-languages (the KCSL-theory). It appears that the KCSL-theory may be interpreted as the first approximation of a UCM. The papers (Fomichov 1992, 1993b, 1994) set forth many principal ideas of the KCSL-theory. In fact, these three publications state (without many mathematical details) a method to describe structured meanings (SMs) of practically arbitrary (very likely, arbitrary) NL-texts and to represent diverse pieces of knowledge about the world.

The aim of this paper in the narrow sense is to determine and investigate a widely-applicable CM considerably extending the spectrum of mathematical means for describing SMs of NL-texts and knowledge about the world provided by the approaches mentioned above (except IFS). The model to be defined is a modified and simplified propositional component of the KCSL-theory.

The aim of this paper in a broad sense is to overcome the existing "barrier of complexity" on the way of developing a Universal Metagrammar of Conceptual Structures and to make a considerable part of the work on that way; in particular, to suggest a general methodology for solving this problem.

The composition of the paper is as follows. Section 2 contains the task statement. Sections 3–12 introduce mathematical definitions and contain a mathematical investigation of the determined formal objects. Section 13 illustrates some expressive possibilities of defined languages and outlines the directions of extending the expressive power of the constructed model. Section 14 points out the advantages of the model in comparison with a number of related approaches. Section 15 indicates some possible applications of the described model.

2 Task Statement

There are some important aspects of formalizing NL-semantics which were underestimated or ignored until recently by the dominant part of researchers.

First of all (see also Section 3 of Fomichov 1993a), this applies to formal investigating SMs of (a) narrative texts including descriptions of sets; (b) discourses with references to the meaning of sentences and larger fragments of texts; (c) phrases where logical connectives "and", "or" are used in non-traditional manners and join not fragments expressing assertions but descriptions of objects, sets, concepts; (d) phrases with attributive clauses; (e) phrases with the words "concept", "notion".

Besides, the most popular approaches to the mathematical study of NL-semantics (mentioned in Section 1) practically don't take into account the role of knowledge about the world in NL comprehension and generation and hence do not study the problem of formal describing knowledge fragments (definitions of concepts, etc.). It should be added that texts have authors, may be published in one or other source, may be inputted from one or other terminal, etc. The information about these external ties of a text may be important for its conceptual interpreting. That's why it is expedient to consider a text as a structured item having a surface structure T, a set of meanings S (in most cases S consists of one meaning) corresponding to T, and some values $V_1, \ldots, V_N$ denoting the author (authors) of T, the date of writing (uttering) T, indicating the new information in T, etc. The main popular approaches to the mathematical study of NL-semantics provide no formal means to represent texts as structured items of the kind.

The analysis shows that the limits of traditional mathematical logic are too narrow for taking into account the enumerated restrictions in formalizing the use of NL. That's why the task of creating logical foundations of designating intellectually powerful NLPSs demands not only to extend the first order logic but rather to develop new mathematical systems compatible with the first order logic and allowing us to formalize the logic of NL use.

We'll proceed from the hypothesis that there is only one mental level for representing meanings of NL-expressions, which may be called a conceptual level but not the semantic and the conceptual levels. This hypothesis is advocated by a number of scientists (see, in particular, Meyer 1994).

Let's demand that the formal means of our model allow us:

1. To build the designations of structured meanings (SMs) of both phrases expressing assertions and of narrative texts; such designations are called usually semantic representations (SRs) of NL-expressions.
2. To build and to distinguish the designations of items corresponding to (a) objects, situations, processes of the real world and (b) concepts qualifying these objects, situations, processes.
3. To build and to distinguish the designations of (3.1) objects and sets of objects, (3.2) concepts and sets of concepts, (3.3) SRs of texts and sets of SRs of texts.
4. To distinguish in a formal manner concepts qualifying objects and concepts qualifying sets of objects of the same kinds.
5. To build compound representations of concepts, e.g., to construct formulas reflecting the surface semantic structure of the NL-expressions like "a person graduated from the Stanford University and being a biologist or a chemist".
6. To construct explanations of more general concepts by means of less general concepts; in particular, to build strings of the form $(a \equiv Des(b))$ where a designates some concept to be explained and $Des(b)$ designates a description of some concretization of the known concept b.
7. To build designations of ordered n-tuples of objects ($n > 1$).
8. To construct: (8.1) formal analogues of complicated designations of sets like the expression "this group consisting of 12 tourists being biologists or chemists", (8.2) designations of sets of tuples, (8.3) designations of sets of sets, etc.
9. To describe set-theoretical relationships.
10. To build designations of SMs of phrases containing, in particular: (10.1) the words "arbitrary", "some", "all", "every", etc.; (10.2) expressions formed by means of applying the connectives "and", "or" to the designations of (10.2a) things, events, (10.2b) concepts, (10.2c) sets; (10.3) expressions where the connective "not" is located just before a designation of a thing, event, etc.; (10.4) indirect speech; (10.5) the participle constructions and the attributive clauses; (10.6) the words "a concept", "a notion".

11. To build designations of SMs of discourses with references to the mentioned objects.

12. To indicate explicitly in SRs of discourses causal and time relationships between described situations (events).

13. To describe SMs of discourses with references to the *meanings* of of phrases and larger fragments of a considered text.

14. To express the assertions about the identity of two entities.

15. To build formal analogues of formulas of the first-order logic with the existential and/or universal quantifiers.

16. To build formulas with logical quantifiers on sets.

17. To consider non-traditional functions (and other non-traditional relations of the kind) with arguments or/and values being: (17.1) sets of things, situations (events); (17.2) sets of concepts; (17.3) sets of SRs of texts.

18. To build conceptual representations of texts as informational objects reflecting not only the meaning but also the values of external characteristics of a text: the author (authors), the date, the application domains of the stated results, etc.

This task statement develops the task statements from (Fomichov 1981,1982; Fomitchov 1983, 1984; Fomichov 1988, 1992).

3 Simplified Sort Systems

Let's begin to solve the task posed above. Suppose that we are to describe some application domain and we've decided to consider some entities as *elementary entities* (people, numbers, events, concepts, etc.). Then determine *compound entities* for the given domain as such entities which will be considered as sets or n-tuples ($n > 1$) formed by more simple entities. We'll interpret *concepts* as general designations of entities belonging to some distinguished by people classes of entities.

Objects are determined as such entities which are not treated as concepts. The class of objects includes, in particular, SRs of texts and sets of concepts.

Suppose that describing a domain begins with choosing some finite set of strings (sorts) denoted by St and containing the designations of the most general concepts qualifying elementary objects from our domain.

Let St contain a distinguished sort P qualifying SRs of propositions (assertions) expressed by sentences and discourses. Assume that some binary relation $\longleftarrow$ on St is given, and for $s, t \in St$ the designation $s \longleftarrow t$ means that either s and t are identical or a concept associated with t is a generalization of a concept associated with s.

Definition 3.1 *A triple S of the form*

$$(St,\ P,\ \longleftarrow) \qquad (1)$$

is called a simplified sort system *(s.s.s.), if St is a finite set of symbols, $P \in St$, $St \setminus \{P\} \neq \emptyset$, $\longleftarrow$ is a partial order on St, $St \neq \{u \in St \mid u$ and P are comparable with respect to $\longleftarrow\}$, and St doesn't include the symbols $\uparrow$, $\{,\}, (,)$, $[\uparrow ent]$, $[\uparrow ob]$, $[\uparrow conc]$, $[ent]$, $[ob]$, $[conc]$. The elements of the set St are called sorts, P is called the sort "sense of proposition", $\longleftarrow$ is called the generality relation. If $t, u \in St$, $t \longleftarrow u$, then we'll say that u is a generalization of t.* □

Example 3.1 Let A_1 be the alphabet consisting of small and capital Latin letters, ciphers 0,1,2,...,9, and the symbol "." (the dot), and $St_1 = \{Pro, sit, nat, int, real, ins, mom\}$. We'll interpret the elements of St_1 as the designations, respectively, of the concepts "semantic representation of a declarative text", "situation", "natural number", "integer", "real number", "intelligent system", "moment". Let the sets G_1, G_2, Gen_1 be defined as follows:

$G_1 = \{(u, u) | u \in St_1\}$,
$G_2 = \{(nat, int), (nat, real), (int, real)\}$,
$Gen_1 = G_1 \cup G_2$.

Then it is easy to see that the triple S_1 of the form (St_1, Pro, Gen_1) is an s.s.s. □

4 Types Generated by a Simplified Sort System

Let us define for any s.s.s. S a set of strings $Tp(S)$ whose elements are called the types of the system

S and are interpreted as characteristics of entities which are considered while reasoning in a selected domain. Agree that if in a reasoning about an entity z it is important that z is not a concept (a notion) then we'll say that z is an object.

Suppose that the strings $[\uparrow ent]$, $[\uparrow conc]$, $[\uparrow ob]$ are associated with the terms "an entity", "a concept", "an object", respectively, or, in other words, these strings are types of semantic items corresponding to the expressions "an entity", "a concept" ("a notion"), "an object".

Formalizing the reasonings, let's agree to proceed from the following recommendations. If the nature of an entity z considered in a reasoning is of no importance then we'll associate with z the type $[ent]$ in the course of reasoning. If all that is important concerning z is that z is an object then we'll associate with z the type $[ob]$. If, to the contrary, all that is important as concerns z is that z is a concept (a notion) then we'll associate with z the type $[conc]$. The destination of types $[ent]$, $[ob]$, $[conc]$ may be explained also by means of the following examples.

Let E_1 and E_2 be, respectively, the expressions "the first entity mentioned on the page 12 of the issue of the newspaper 'The Moscow Times' published on October 1, 1994" and "the first object mentioned on the page 12 of the issue of 'The Moscow Times' published on October 1, 1994". Then we may associate the types $[ent]$ and $[ob]$ with the entities referred in E_1 and E_2, respectively, in case we haven't read the page 12 of the indicated issue.

However, after reading the page 12 we'll get to know that the first entity and the first object mentioned on this page is the city Madrid. Hence, we may associate now with the mentioned entity (object) a more informative type *popul.area* (a populated area).

Let E_3 be the expression "the notion with the mark AC060 defined in the Longman Dictionary of Scientific Usage (Moscow, Russky Yazyk Publishers, 1989)". Not seeing this dictionary, we may associate with the notion mentioned in E_3 only the type $[conc]$. But after finding the definition with the mark AC060, we get to know that it is the definition of the notion "a tube" (a hollow cylinder with its length much greater than its diameter). Hence we may associate with the notion mentioned in E_3 a more informative type *ph* (designating the concept "a physical object").

Let's consider the strings $[\uparrow ent]$, $[\uparrow ob]$, $[\uparrow conc]$, $[ent]$, $[ob]$, $[conc]$ as symbols in the next definitions.

Definition 4.1 *Let S be an s.s.s. of the form (1),*

$$Spectp = \{[\uparrow ent], [\uparrow ob], [\uparrow conc]\},$$
$$Toptp = \{[ent], [ob], [conc]\}.$$

Then the set of types $Tp(S)$ is the least set M satisfying the following conditions:

1. $Spectp \cup Toptp \cup St \cup \{\uparrow s \mid s \in St\} \subseteq M$. *The elements of Spectp and Toptp are called* special types *and* top types, *respectively.*

2. *If $t \in M \setminus Spectp$, then the string of the form $\{t\}$ belongs to M.*

3. *If $n > 1$, for $i = 1, \ldots, n$ and $t_i \in M \setminus Spectp$, then the string of the form $(t_1, \ldots, t_n)$ belongs to M.*

4. *If $t \in M$ and t has the beginning $\{$ or $($, then the string $\uparrow t$ belongs to M.*

□

Definition 4.2 *If S is an s.s.s., then $Mtp(S) = Tp(S) \setminus Spectp$; the elements of the set $Mtp(S)$ are called* main types. □

Let's formulate the principles of establishing correspondences between the entities considered in a domain with an s.s.s. S of the form (1) and the types from the set $Mtp(S)$. The types of concepts, as distinct from types of objects, have the beginning $\uparrow$. With a concept denoted by a string s from St we associate a type $\uparrow s$. The type $\{t\}$ corresponds to any set of entities of type t. If $x_1, \ldots, x_n$ are the entities of types $t_1, \ldots, t_n$, then the type $(t_1, \ldots, t_n)$ is assigned to the n-tuple $(x_1, \ldots, x_n)$.

Example 4.1 We may assign the types from $Mtp(S_1)$ to some concepts and objects (including relations and other sets) with the aid of the following table:

ENTITY	TYPE
the concept "a set"	$\uparrow \{[ent]\}$
the concept "a set of objects"	$\uparrow \{[ob]\}$
the concept "a set of concepts"	$\uparrow \{[conc]\}$
the concept "a person"	$\uparrow ins$
Tom Soyer	ins
the concept "an Editorial Board"	$\uparrow \{ins\}$
the Editorial Board of "Informatica"	$\{ins\}$
the concept "a pair of integers"	$\uparrow (int, int)$
the pair (12, 144)	(int, int)

We can also associate with the relation "Less" on integers the type $\{(int, int)\}$, with the relation "To belong to a set" the type $\{([ent], \{[ent]\})\}$, with the relation "An object Y is qualified by a concept C" the type $\{([ob], [conc])\}$, and with the relation "A concept D is a generalization of a concept C" the type $\{([conc], [conc])\}$. □

Some types may be particular cases of other types. That's why formulate several definitions in order to determine the notion of a concretization of a type.

Definition 4.3 *Let S be an s.s.s. of the form (1),*

$$Tc(S) = \{t \in Tp(S) \setminus (Spectp \cup Toptp) \mid \text{the symbol } \uparrow \text{ is the beginning of } t\},$$
$$Tob(S) = Tp(S) \setminus (Spectp \cup Toptp \cup Tc(S)).$$

Then determine the transformations $tr_1, \ldots, tr_5$ partially applicable to the types of the set $Mtp(S)$ (called main types) in the following way:

1. *If $t \in Mtp(S)$, t includes a substring $[ent]$, then tr_1 is applicable to t and $tr_1(t)$ is the result of simultaneous replacing in t each occurrence of $[ent]$ by the type $[ob]$.*

2. *If $t \in Mtp(S)$, t includes a substring $[ent]$, then tr_2 is applicable to t and $tr_2(t)$ is the result of simultaneous replacing in t each occurrence of $[ent]$ by the type $[conc]$.*

3. *If $t \in Mtp(S)$, t includes a substring $[ob]$, $u \in Tc(S)$, $u \in Tob(S)$, w is the result of simultaneous replacing each occurrence of the symbol $[ob]$ by the type u, then w is a possible result of applying tr_3 to the type t.*

4. *If $t \in Mtp(S)$, t includes a substring $[conc]$, $u \in Tc(S)$, and w is the result of simultaneous replacing each occurrence of the symbol $[conc]$ by the type u, then w is a possible result of applying the transformation tr_4 to t.*

5. *If t belongs to $Mtp(S)$, r, u belong to St, t contains a substring r, $u \longleftarrow r$, $u \neq r$, w is the result of replacing in t some occurrence of r by the sort u, then w is a possible result of applying tr_5 to t.*

□

Example 4.2 If S_1 is the s.s.s. built in the Example 3.1, $t_1 = [ob]$, $w_1 = ins$, then w_1 is a possible result of applying the transformation tr_3 to t_1. If $t_2 = \{(ins, real)\}$, $w_2 = \{(ins, nat)\}$, then w_2 is a possible result of applying tr_5 to t_2 (since $nat \longleftarrow real$).

If S is an s.s.s., then denote by $Strings(S)$ the set of all strings in the alphabet $St \cup Toptp \cup \{`,', \uparrow, `\{', `\}', (,)\}$ (hence $Strings(S)$ contains the symbol "comma").

It is not difficult to prove the following proposition:
Let S be arbitrary s.s.s., $i \in \{1, \ldots, 5\}, t \in Mtp(S)$, the transformation tp_i may be applied to the type t, and the string $w \in Strings(S)$ be a possible result of applying just one time the transformation tr_i to the type t. Then $w \in Mtp(S)$. □

Definition 4.4 *Let S be an s.s.s., $t, u \in Mtp(S)$. Then the type u is called a* concretization *of the type t, and t is called a* generalization *of the type u (the equivalent designation is $t \vdash u$), if $u = t$ or there are such $x_1, \ldots, x_n \in Mtp(S)$, where $n > 1$, that $x_1 = t, x_n = u$, and for $i = 1, \ldots, n-1$ there is such $k \in \{1, \ldots, 5\}$ that the transformation tr_k is applicable to x_i and x_{i+1} is a possible result of applying tr_k to x_i.* □

Thus, a binary relation $\vdash$ is determined on the set of main types $Mtp(S)$ of arbitrary s.s.s. S.

Example 4.3 Let S_1 be the s.s.s. built in the Example 3.1. Then it is not difficult to show that

$$[ent] \vdash [ob],\ [ent] \vdash [conc],\ [ob] \vdash ins,$$
$$[ob] \vdash \{(\{ins\}, nat)\},\ [ob] \vdash (nat, nat),$$
$$[conc] \vdash \uparrow ins,\ [conc] \vdash \uparrow \{ins\}.$$

One can easily show that the following statement is true:

If S is an arbitrary s.s.s., then the relation $\vdash$ is a partial order on the set $Mtp(S)$. □

5 Concept-Object Systems

Let's proceed from the assumption that for describing an application domain on the conceptual level we should choose some sets of strings X and V. The first set is to contain, in particular, designations of concepts, physical objects (people, ships, books, etc.), events, n-ary relations ($n \geq 1$). The set V should consist of variables which will play in expressions of our knowledge representation language the roles of marks of diverse entities and, besides, will be used together with the quantifiers $\exists$ and $\forall$.

We'll distinguish in X some subset F containing the designations of diverse functions. Each function f with n arguments will be considered as some set of $n+1$-tuples $(x_1, \ldots, x_n, y)$, where $y = f(x_1, \ldots, x_n)$. Besides, we'll introduce a mapping tp assigning to each element $d \in X \cup V$ a type $tp(d)$ characterizing the entity denoted by d.

Definition 5.1 *Let S be any s.s.s. of the form (1). Then a four-tuple Ct of the form*

$$(X,\ V,\ tp,\ F) \qquad (2)$$

is called a concept-object system (c.o.s.) for S, iff the following conditions are satisfied:

(1) *X, V are countable non-intersecting sets of symbols;*

(2) *tp is a mapping from $X \cup V$ into $Tp(S)$;*

(3) *$\{v \in V \mid tp(v) = [ent]\}$ is countable;*

(4) *$F \subseteq \{r \in X \mid tp(r)$ is a string with the beginning $\{($and the ending$)\}\}$,*

(5) *$St \subset X$ and for each $s \in St$, $tp(s) = \uparrow s$.*

The set X is called a primary universe, *the elements of V and F are called* variables *and* functional symbols, *respectively.*

If $d \in X \cup V$, $tp(d) = t$, then we say that t is the type *of d.* □

Example 5.1 Construct some c.o.s. Ct_1 for the s.s.s. S_1 determined in Example 3.1. Let N be the set of all such strings *str* formed out of the ciphers 0,1, ..., 9 that the first symbol of *str* is distinct from "0" in case *str* contains more than one symbol. Let

$$U_1 = \{concept, person, chemist, biologist, tour.gr, expl1, fire1, J.Price, R.Scott, N.Cope, P.Somov, Friends, Numb, Less, Cause, Knows, Is1, Elem, Subset, Include1, Now, Before\}.$$

The strings of U_1 denote, respectively: the notions "a concept", "a person", "a chemist", "a biologist", "a tourist group", "an explosion", "a fire"; some concrete persons with the initial and surname J.Price, R.Scott, N.Cope, P.Somov; the function "Friends" assigning to a person z the set of all friends of z; the function "Numb" assigning to a set the number of elements in it; the relations "Less" (on the set of integers), "Cause" (on situations), "Knows" ("An intelligent system knows something"), "Is1" ("An object is qualified by a concept"), "Elem" ("An entity z_1 is element of a set z_2"), "Subset" ("To be a subset of a set of entities"), "Include1" ("An intelligent system z_1 includes an object z_2 into some set of objects z_3 at the moment z_4"), "Before". The string *Now* will be used in SRs of texts for denoting a current moment of time.

Define a mapping t_1 from U_1 into $Tp(S_1)$ by the following table:

x	$t_1(x)$
concept	$[\uparrow conc]$
person, chemist, biologist	$\uparrow ins$
tour.group	$\uparrow \{ins\}$
expl1, fire1	$\uparrow sit$
J.Price, R.Scott, N.Cope, P.Somov	ins
Friends	$\{(ins, \{ins\})\}$
Numb	$\{(\{[ent]\}, nat)\}$
Less	$\{(int, int)\}$
Cause	$\{(sit, sit)\}$
Knows	$\{(ins, Pro)\}$
Is1	$\{([ob], [conc])\}$
Elem	$\{([ent], \{[ent]\})\}$
Subset	$\{(\{[ent]\}, \{[ent]\})\}$

*Include*1	$\{(ins, [ob], \{[ob]\}, mom)\}$
Now	mom
Before	$\{(sit, sit)\}$

Let

$Vent = \{v \mid v = an,\ a$ is the letter "x"$\}$,
$Vset = \{v \mid v = bn,\ b$ is the letter "M"$\}$,
$Ve = \{v \mid v = dn,\ d$ is the letter "e"$\}$,
$Vmom = \{v \mid v = gn,\ g$ is the string "mt"$\}$,
$Vp = \{v \mid v = hn,\ h$ is the letter "P"$\}$,
where $n \in N$.

We'll interpret the strings from the sets $Vent, \ldots, Vp$ as variables and use them for marking out, respectively, arbitrary entities, sets of entities, situations (in particular, events), moments of time, and SRs of propositions.

Let

$$V_1 = Vent \cup Vset \cup Ve \cup Vmom \cup Vp;$$
$$X_1 = N \cup St_1 \cup U_1,$$

and the mapping tp_1 from $X_1 \cup V_1$ into $Tp(S_1)$ be defined as follows:

$z \in N \Longrightarrow tp_1(z) = nat;$
$s \in St_1 \Longrightarrow tp_1(s) = \uparrow s;$
$z \in U_1 \Longrightarrow tp_1(z) = t1(z);$
$v \in Vent \Longrightarrow tp_1(v) = [ent];$
$v \in Vset \Longrightarrow tp_1(v)$ is the string $\{[ent]\}$;
$v \in Ve \Longrightarrow tp_1(v) = sit;$
$v \in Vmom \Longrightarrow tp(v) = mom;$
$v \in Vp \Longrightarrow tp_1(v) = Pro.$

Let $F_1 = \{Friends, Numb\}$, $Ct_1 = (X_1, V_1, tp_1, F_1)$. Then it is not difficult to verify that Ct_1 is a c.o.s. for the s.s.s. S_1. □

6 Systems of Quantifiers and Logical Connectives. Simplified Conceptual Bases

Presume that we define a s.s.s S of the form (1) and a c.o.s. Ct of the form (2) for S in order to describe an application domain. Then it is proposed to distinguish in the primary universe X two non-intersecting and finite (hence non-void) subsets Int_1 and Int_2 in the following manner: we distinguish in St two sorts int_1 and int_2 and suppose that for $m = 1, 2, Int_m = \{x \in X \mid tp(x) = int_m\}$. The elements of Int_1 correspond to the meanings of expressions "every", "some", "any", "arbitrary", etc. (and, may be, "almost every", etc.) in situations when these expressions are parts of the word groups in singular. The elements of Int_2 are interpreted as semantic items corresponding to the expressions "all", "several", "almost all", "many", and so on; the minimal requirement is that Int_2 contains a semantic item corresponding to the word "all". Let Int_1 contain a distinguished element ref considered as an analogue of the word "some" (when it is associated in a context with singular) in the sense "some quite definite" (but, possibly, unknown). If Ct is a c.o.s. of the form (2), $d \in X$, d denotes a concept, and a SR of a text includes a substring of the form $ref\, d$ (e.g., the substring *sm chemist*, if $ref = sm$, $d = chemist$), then we'll suppose that this substring denotes some concrete entity (but not an arbitrary one) which is characterized by the concept d. Let X contain the elements $\equiv, \neg, \wedge, \vee$ interpreted as the connectives "is identical with", "not", "and", "or", and the elements $\forall$ and $\exists$ interpreted as universal and existential quantifiers.

Definition 6.1 *Let S be any s.s.s. of the form (1), Ct be any c.o.s. of the form (2) for S, $ref \in X$, the different elements $int_1, int_2, eq, neg, binlog, ext$ are some distinguished sorts from $St \setminus \{P\}$, and each pair of these sorts is incomparable with respect to the generality relation $\longleftarrow$. Then the 7-tuple Ql of the form*

$$(int_1, int_2, ref, eq, neg, binlog, ext) \qquad (3)$$

is called a system of quantifiers and logical connectives *(s.q.l.c.) for S and Ct, iff the following conditions are satisfied:*

1. *For each $m = 1, 2$, the set $Int_m = \{d \in X \mid tp(d) = int_m\}$ is a finite set; $ref \in Int_1$.*

2. *$\{\equiv, \neg, \wedge, \vee, \forall, \exists\} \subset X$; besides, $tp(\equiv) = eq$, $tp(\neg) = neg$, $tp(\wedge) = tp(\vee) = binlog$, $tp(\forall) = tp(\exists) = ext$.*

3. *There are no such $d \in X \setminus (Int_1 \cup Int_2 \cup \{\equiv, \neg, \wedge, \vee, \forall, \exists\})$ and no such $s \in \{int_1, int_2, eq, neg, binlog, ext\}$ that $tp(d)$ and s are comparable with respect to the relation $\longleftarrow$.*

4. *For each $u \in \{int_1, int_2, eq, neg, binlog, ext\}$, u and the sort P ("sense of proposition") are incomparable with respect to $\longleftarrow$.*

The elements of Int_1 *and* Int_2 *are called* intensional quantifiers, *ref is called* the referential quantifier, $\forall$ *and* $\exists$ *are called* extensional quantifiers. □

Example 6.1 Let $S_1 = (St_1, Pro, Gen_1)$ be the s.s.s. built in Example 3.1, $Ct_1 = (X_1, V_1, tp_1, F_1)$ be the c.o.s. for S_1 considered in Example 5.1. Suppose that A_2 is the alphabet consisting of small and capital Latin letters, ciphers 0,1, ..., 9, symbols

$$\text{"."}\ (\text{dot}), \equiv, \neg, \wedge, \vee, \forall, \exists;$$
$$Sr = \{qint1, qint2, eql, cnot, bin, exq\},$$
$$Gen_2 = Gen_1 \cup \{(s,s) \mid s \in Sr\},$$
$$St_2 = St_1 \cup Sr,\ S_2 = (St_2, Pro, Gen_2);$$
$$U_2 = \{sm, all, \equiv, \neg, \wedge, \vee, \forall, \exists\}$$

where the string *sm* is interpreted as the semantic item "some" associated with the singular; $X_2 = X_1 \cup Sr \cup U_2$. Since, obviously, S_2 is an s.s.s., determine a mapping tp_2 from $X_2 \cup V_1$ into $Tp(S_2)$ in the following way:

$$s \in Sr \Longrightarrow tp_2(s) = \uparrow s;$$
$$d \in X_1 \Longrightarrow tp_2(d) = tp_1(d);$$
$$v \in V_1 \Longrightarrow tp_2(V) = tp_1(v);$$
$$tp_2(sm) = qint1,\ tp_2(all) = qint2,$$
$$tp_2(\equiv) = eql,\ tp_2(\neg) = cnot,$$
$$tp_2(\wedge) = tp_2(\vee) = bin,$$
$$tp_2(\forall) = tp_2(\exists) = exq.$$

Let

$$Ct_2 = (X_2, V_1, tp_2, F_1),$$
$$Ql_1 = (qint1, qint2, sm, eql, cnot, bin, exq).$$

Then it is easy to see that Ct_2 is a c.o.s. for S_2, and Ql_1 is an s.q.l.c. for S_2 and Ct_2; besides, the element *sm* is interpreted as the referential quantifier of the system Ql_1. □

Definition 6.2 *A triple B of the form*

$$(S, Ct, Ql) \tag{4}$$

is called a simplified conceptual basis *(s.c.b.), iff S is an arbitrary s.s.s., Ct is an arbitrary c.o.s. of the form (2) for S, Ql is an arbitrary s.q.l.c. for S and Ct, and*

$$(X \cup V) \cap \{\text{','},\ (,\ ),\ :,\ *,\ \langle,\ \rangle, \&\} = \emptyset.$$

We'll denote by $S(B)$, $Ct(B)$, $Ql(B)$ the components of an arbitrary s.c.b. B of the form (4). Each component with the name h of the mentioned systems of the forms (1), (2), (3) will be denoted by $h(B)$. E.g., the set of sorts, the distinguished sort "sense of proposition", and the primary universe of B will be denoted by $St(B), P(B), X(B)$. We'll interpret simplified conceptual bases as formal enumerations of

(a) *primary items needed for building SRs of NL-texts and for describing knowledge about the reality,*

(b) *the information associated with these items and required for constructing such SRs of NL-texts and forming knowledge fragments.*

□

Example 6.2 Let S_2, Ct_2, Ql_1 be, respectively, the s.s.s., c.o.s., and s.q.l.c. determined in Example 6.1. Then, obviously, the triple $B_1 = (S_2, Ct_2, Ql_1)$ is an s.c.b., and

$$St(B_1) = St_2,\ P(B_1) = Pro,$$
$$X(B_1) = X_2,\ V(B_1) = V_1.$$

□

Proposition 1 *The set of all simplified conceptual bases is non-void.* □

Proof. The truth of this proposition follows immediately from the Example 6.2.

7 The Idea of Determining Three Classes of Formulas Generated by a Simplified Conceptual Basis

The notions introduced above allow us to determine for every s.c.b. *B* a set of formulas $Formrs(B)$ convenient for describing structured meanings (SMs) of NL-texts and operations on SMs.

Definition 7.1 *For arbitrary s.c.b. B,*

$$Drs(B) = X(B) \cup V(B) \cup \{`,', (,), :, *, \langle, \rangle\}$$

(hence $Drs(B)$ includes, in particular, the symbol "comma"), $Drs^+(B)$ is the set of all non-empty finite sequences of elements from $Drs(B)$. □

The essence of the approach to determining conceptual formulas proposed in this paper is as follows. Some assertions $P[0], P[1], \ldots, P[10]$ will be defined; they are interpreted as the rules of building SRs of NL-texts out of the elements of the primary universe $X(B)$, variables from $V(B)$, and several special symbols, if B is an s.c.b. destined for describing a considered domain. For any s.c.b. B, the rule $P[0]$ provides an initial stock of formulas.

Definition 7.2 *Denote by $P[0]$ the assertion*

"If $d \in X(B) \cup V(B), t \in Tp(S(B))$,
$tp = tp(B), tp(d) = t$,
then $d \in L(B)$ and the string of the form
$d \,\&\, t$ belongs to $T^0(B)$"

Let B be any s.c.b., $L(B)$ and $T^0(B)$ be the least sets defined by the assertion $P[0]$, $Lnr_0(B) = L(B)$ (the designation "Lnr" is deciphered as "L" numbered). Then, obviously,

$$Lnr_0(B) = X(B) \cup V(B),\ T^0(B) = \{b \mid b = d \,\&\, t,\ d \in X(B) \cup V(B),\ t \in Tp(S(B)),\ t = tp(d)\}.$$

□

E.g., for the s.c.b. B_1 defined in Example 6.2, $T^0(B_1)$ includes the formulas

$$J.Price \,\&\, ins,\ \ Friends \,\&\, \{(ins, \{ins\})\}.$$

If $1 \le i \le 10$, then for any s.c.b. B and for $k = 1, \ldots, i$ the assertions $P[0], P[1], \ldots, P[i]$ determine by conjoint induction some sets of formulas

$$Lnr_i(B) \subset Drs^+(B),\ T^0(B),\ Tnr_i^1(B), \ldots, Tnr_i^i(B) \ \text{and}\ Ynr_i^1(B), \ldots, Ynr_i^i(B).$$

The set $Lnr_i(B)$ is considered as the main subclass of formulas generated by $P[0], \ldots, P[i]$. The formulas from this set are destined for describing SMs of NL-texts.

If $1 \le k \le i$, then the set $Tnr_i^k(B)$ consists of strings of the form $b \,\&\, t$, where $b \in Lnr_i(B)$, $t \in Tp(S(B))$, and b is obtained by means of applying the rule $P[k]$ to some simpler formulas at the final step of an inference. It should be added that the inference of b out of the elements of $X(B)$ and $V(B)$ may use any of the rules $P[0], \ldots, P[k], \ldots, P[i]$; these rules may be applied arbitrarily many times. If s.c.b. B is chosen to describe some domain, then b can be interpreted as a SR of a text or as a fragment of a SR of a text pertaining to the selected domain. In this case, t may be considered as the designation of the kind of the entity characterized by such SR or by a fragment of SR. Besides, t may qualify b as a SR of a narrative text.

E.g., it will be shown below that $Lnr_4(B_1), \ldots, Lnr_{10}(B_1)$ include the formula $Elem(P.Somov, Friends(J.Price))$, and $Tnr_4^4(B_1), \ldots, Tnr_{10}^4(B_1)$ include the formula $Elem(P.Somov, Friends(J.Price)) \,\&\, Pro$, where Pro is the distinguished sort $P(B_1)$ ("sense of proposition").

Each string $c \in Ynr_i^k(B)$, where $1 \le k \le i$, can be represented in the form $c = a_1 \,\&\, \ldots \,\&\, a_m \,\&\, b$, where $a_1, \ldots, a_m, b \in Lnr_i(B)$. Besides, there is such $t \in Tp(S(B))$ that the string $b \,\&\, t$ belongs to $Tnr_i^k(B)$. The strings $a_1, \ldots, a_m$ are obtained by employing some rules from $P[0], \ldots, P[i]$, and b is constructed out of "blocks" $a_1, \ldots, a_m$ (some of them can be a little bit changed) by applying just one time the rule $P[k]$. The possible quantity of "blocks" $a_1, \ldots, a_m$ depends on k. Thus, the set $Ynr_i^k(B)$ fixes the result of applying the rule $P[k]$ just one time. E.g., we'll see below that the sets $Ynr_4^4(B_1), \ldots, Ynr_{10}^4(B_1)$ include the formula

$$Elem \,\&\, P.Somov \,\&\, Friends(J.Price) \,\&\, Elem(P.Somov, Friends(J.Price)).$$

Let for $i = 1, \ldots, 10$,

$$Tr_i(B) = T^0(B) \cup Tnr_i^1(B) \cup \ldots \cup Tnr_i^i(B);$$
$$Yr_i(B) = Ynr_i^1(B) \cup \ldots \cup Ynr_i^i(B);$$
$$Form_i(B) = Lnr_i(B) \cup Tr_i(B) \cup Yr_i(B)$$

We'll interpret $Form_i(B)$ as the set of all formulas generated by the s.c.b. B. This set is the union of three classes of formulas the principal class being $Lnr_i(B)$. The formulas from these three classes will be called ℓ-formulas, t-formulas, and y-formulas, respectively.

The class of t-formulas is needed for assigning a type from $Tp(S(B))$ to each $b \in Lnr_i(B)$, where $i = 1, \ldots, 10$. The class of y-formulas is considered on technical reasons: for describing (not in this article) an algebra of partial operations on n-tuples ($n \geq 1$) formed out of SRs of NL-texts and out of fragments of SRs. It is important that for $i = 0, \ldots, 9$, $Lnr_i(B) \subseteq Lnr_{i+1}(B)$. The formal language $Lnr_{10}(B)$ is a subset of some standard K-language in stationary basis with empty marking-out of variables. The class of such languages is defined in Fomichov (1988). That's why ℓ-formulas will be called also K-strings.

The pairs of the form (B, Rls), where B is a s.c.b., Rls is the set consisting of the rules $P[0], \ldots, P[10]$, will be called restricted K-calculuses.

8 The Use of Intensional Quantifiers in Formulas

The rule $P[1]$ allows us to join the intensional quantifiers to the designations of notions (concepts).

Definition 8.1 *If B is an s.c.b., then for $m = 1, 2$*

$$Int_m(B) = \{q \in X(B) \mid tp(q) = int_m(B)\},$$
$$Int(B) = Int_1(B) \cup Int_2(B),$$
$$Tconc(B) = \{t \in Tp(S(B)) \mid t \text{ has the beginning } \uparrow\} \cup Spectp$$

where (see Definition 4.1)

$$Spectp = \{[\uparrow ent], [\uparrow ob], [\uparrow conc]\}.$$

□

Using the rules $P[0]$ and $P[1]$, we can build some strings of the form $q\,d$, where $q \in Int(B)$, $d \in X(B)$, $tp(d) \in Tconc(B)$. E.g., if B_1 is the s.c.b. determined in Example 6.2, then we can construct the ℓ-formulas *sm person*, *sm tour.gr*, *sm concept*, *all person*, *all tour.gr*, *all concept* generated by B_1.

It is possible also to build more complex strings of the form $q\,a$, where q is an intensional quantifier, with the help of the rule $P[1]$ using the rule $P[8]$ (see Section 11) and, may be, some other rules (besides the rules $P[0]$ and $P[1]$) for constructing a. E.g., it will be possible to build the ℓ-formulas $sm\,tour.gr * (Numb, 12)$, $all\,tour.gr * (Numb, 12)$ in the s.c.b. B_1. These formulas are interpreted as SRs of the expressions "some tourist group consisting of 12 persons" and "all tourist groups consisting of 12 persons".

The transition from a ℓ-formula c designating a notion to some ℓ-formula $q\,c$, where q is an intensional quantifier, is described with the help of some special function h.

Definition 8.2 *Let B be any s.c.b., $S = S(B)$. Then the mapping $h : \{1,2\} \times Tp(S) \rightarrow Tp(S)$ is determined as follows: if $u \in Tp(S)$ and the string $\uparrow u$ belongs to $Tp(S)$, then*

$$\begin{array}{ll} h(1, \uparrow u) = u, & h(2, \uparrow u) = \{u\}; \\ h(1, [\uparrow ent]) = [ent], & h(2, [\uparrow ent]) = \{[ent]\}; \\ h(1, [\uparrow ob]) = [ob], & h(2, [\uparrow ob]) = \{[ob]\}; \\ h(1, [\uparrow conc]) = [conc], & h(2, [\uparrow conc]) = \{[conc]\}. \end{array}$$

From the standpoint of building SRs of texts, the mapping h describes the transformations of the types in the course of the transition:

(a) *from the notions "a person", "a tourist group" to the SRs of expressions "some person", "arbitrary person", "some tourist group", "arbitrary tourist group", etc. (in case the first argument of h is 1) and to the SRs of expressions "all people", "all tourist groups", etc. (if the first argument of h is 2);*

(b) *from the notions "an entity", "an object", "a concept" to the SRs of the expressions "some entity", "arbitrary entity", "some object", "arbitrary object", "some concept", "arbitrary concept", etc. (when the first argument of h is 1) and to the SRs of the expressions "all entities", "all objects", "all concepts", etc. (when the first argument of h is 2).*

□

Definition 8.3 *Denote by $P[1]$ the assertion*

"Let $a \in L(B) \setminus V(B)$, $u \in Tconc(B)$, $k \in \{0, 8\}$ and the string $a \,\&\, u$ belong to $T^k(B)$. Let $m \in \{1, 2\}$, $q \in Int_m$, $t = h(m, u)$, and b be the string of the form $q\,a$. Then $b \in L(B)$, the string $b \,\&\, t$ belongs to $T^1(B)$, and the string of the form $q \,\&\, a \,\&\, b$ belongs to $Y^1(B)$". □

Example 8.1 Let B be the s.c.b. B_1 considered in Example 6.2; $L(B)$, $T^0(B)$, $T^1(B)$, $Y^1(B)$ be the least sets of formulas jointly defined by the assertions $P[0]$ and $P[1]$. Then it is easy to verify that the following relationships take place:

$$\begin{aligned}
&\mathit{person} \in L(B) \setminus V(B),\\
&\quad \mathit{person}\,\&\uparrow \mathit{ins} \in T^0(B),\\
&\quad \mathit{sm} \in \mathit{Int}_1(B),\\
&\quad h(1,\uparrow \mathit{ins}) = \mathit{ins} \Longrightarrow \mathit{sm\,person} \in L(B),\\
&\quad \mathit{sm\,person}\,\&\,\mathit{ins} \in T^1(B),\\
&\quad \mathit{sm}\,\&\,\mathit{person}\,\&\,\mathit{sm\,person} \in Y^1(B);\\
&\mathit{all} \in \mathit{Int}_2(B),\\
&\quad h(2,\uparrow \mathit{ins}) = \{\mathit{ins}\} \Longrightarrow \mathit{all\,person} \in L(B),\\
&\quad \mathit{all\,person}\,\&\,\{\mathit{ins}\} \in T^1(B),\\
&\quad \mathit{all}\,\&\,\mathit{person}\,\&\,\mathit{all\,person} \in Y^1(B);\\
&\mathit{tour.gr} \in L(B) \setminus V(B),\\
&\quad \mathit{tour.gr}\,\&\uparrow \{\mathit{ins}\} \in T^0(B),\\
&\quad h(1,\uparrow \{\mathit{ins}\}) = \{\mathit{ins}\},\\
&\quad h(2,\uparrow \{\mathit{ins}\}) = \{\{\mathit{ins}\}\} \Longrightarrow \mathit{sm\,tour.gr},\\
&\quad \mathit{all\,tour.gr} \in L(B),\\
&\quad \mathit{sm\,tour.gr}\,\&\,\{\mathit{ins}\},\\
&\quad \mathit{all\,tour.gr}\,\&\,\{\{\mathit{ins}\}\} \in T^1(B),\\
&\quad \mathit{sm}\,\&\,\mathit{tour.gr}\,\&\,\mathit{sm\,tour.gr},\\
&\qquad \mathit{all}\,\&\,\mathit{tour.gr}\,\&\,\mathit{all\,tour.gr} \in Y^1(B);\\
&\mathit{concept} \in L(B) \setminus V(B),\\
&\quad \mathit{concept}\,\&\,[\uparrow \mathit{conc}] \in T^0(B),\\
&\quad h(1,[\uparrow \mathit{conc}]) = [\mathit{conc}],\\
&\qquad h(2,[\uparrow \mathit{conc}]) = \{[\mathit{conc}]\} \Longrightarrow\\
&\qquad \mathit{sm\,concept}, \mathit{all\,concept} \in L(B),\\
&\quad \mathit{sm\,concept}\,\&\,[\mathit{conc}],\\
&\qquad \mathit{all\,concept}\,\&\,\{[\mathit{conc}]\} \in T^1(B),\\
&\quad \mathit{sm}\,\&\,\mathit{concept}\,\&\,\mathit{sm\,concept},\\
&\qquad \mathit{all}\,\&\,\mathit{concept}\,\&\,\mathit{all\,concept} \in Y^1(B).
\end{aligned}$$

□

Notation. We'll define in what follows the rules $P[2], \ldots, P[10]$. For $k = 1, \ldots, 10$, the rule $P[k]$ asserts that some formula b belongs to $L(B)$, some formula $b\,\&\,t$ belongs to $T^k(B)$, where $t \in \mathit{Tp}(S(B))$, and some formula c belongs to $Y^k(B)$. If $1 \le i \le 10$, B is any s.c.b., then the rules $P[0], P[1], \ldots, P[i]$ determine by conjoint induction the sets of formulas

$$L(B), T^0(B), T^1(B), \ldots, T^i(B),\\ Y^1(B), \ldots, Y^i(B).$$

Denote these sets by $\mathit{Lnr}_i(B), T^0(B), \mathit{Tnr}_i^1(B), \ldots, \mathit{Tnr}_i^i(B), \mathit{Ynr}_i^1(B), \ldots, \mathit{Ynr}_i^i(B)$ and denote the family consisting of all these sets by $\mathit{Globset}_i(B)$.

Let $n \ge 1$, $Z_1, \ldots, Z_n \in \mathit{Globset}_i(B)$, $w_1 \in Z_1, \ldots, w_n \in Z_n$. Then, if these relationships for the formulas $w_1, \ldots, w_n$ are the consequence of employing some rules $P[\ell_1], \ldots, P[\ell_m]$, where $m \ge 1$, we'll denote this fact by the expression of the form $B(\ell_1, \ldots, \ell_m) \Rightarrow w_1 \in Z_1, \ldots, w_n \in Z_n$. The sequence $\ell_1, \ldots, \ell_m$ may contain the repeated numbers.

In the expressions of the kind we'll often omit the symbol B in the designations of the sets $Z_1, \ldots, Z_n$; besides, we'll use the expressions $w_\ell, w_2 \in Z_1$, $w_3, w_4, w_5 \in Z_2$, and so on. E.g., it was shown in Example 8.1 that

$$\begin{aligned}
&B_1(0,1) \Rightarrow \mathit{all\,person},\ \mathit{all\,tour.gr} \in \mathit{Lnr}_1(B_1),\\
&\mathit{all\,person}\,\&\,\{\mathit{ins}\}, \mathit{all\,tour.gr}\,\&\,\{\{\mathit{ins}\}\} \in\\
&\quad \mathit{Tnr}_1^1(B_1),\\
&\mathit{all}\,\&\,\mathit{person}\,\&\,\mathit{all\,person} \in \mathit{Ynr}_1^1(B_1),\\
&\mathit{all}\,\&\,\mathit{tour.gr}\,\&\,\mathit{all\,tour.gr} \in \mathit{Ynr}_1^1(B_1).
\end{aligned}$$

The expression

$$B_1(0,1) \Rightarrow \mathit{all}\,\&\,\mathit{person}\,\&\,\mathit{all\,person} \in \mathit{Ynr}_1^1(B_1)$$

is equivalent to the expression

$$B_1(0,1) \Rightarrow \mathit{all}\,\&\,\mathit{person}\,\&\,\mathit{all\,person} \in \mathit{Ynr}_1^1.$$

9 The Use of Relational Symbols and Marking Formulas

The rule $P[2]$ allows us, in particular, to construct K-strings of the form $f(a_1, \ldots, a_n)$, where f is the designation of a function with n arguments $a_1, \ldots, a_n$. The rule $P[3]$ is destined for building K-strings of the form $(a_1 \equiv a_2)$, where a_1 and a_2 denote the entities characterized by types comparable with respect to the relation $\vdash$. Using consecutively $P[2]$ and $P[3]$, we can build the K-strings of the form $(f(a_1, \ldots, a_n) \equiv b)$, where b is the value of f for $a_1, \ldots, a_n$.

Let's recall that for arbitrary s.s.s. S, the set of main types

$$\mathit{Mtp}(S) = \mathit{Tp}(S) \setminus \{[\uparrow \mathit{ent}], [\uparrow \mathit{ob}], [\uparrow \mathit{conc}]\}$$

(according to the definitions 4.1 and 4.2).

Definition 9.1 *Let B be any s.c.b., $S = S(B)$. Then:*

(a) $R_1(B) = \{d \in X(B) \mid$ *there is such* $t \in$ *Mtp*(S) *that* t *has no beginning '(' and* $tp(d)$ *is the string of the form* $\{t\}\}$;

(b) *for arbitrary* $n > 1$,

$$R_n(B) = \{d \in X(B) \mid \text{there are such } t_1, \ldots, t_n \in Mtp(S) \text{ that } tp(d) \text{ is the string of the form } \{(t_1, \ldots, t_n)\}\};$$

(c) *for arbitrary* $n \geq 1$,

$$F_n(B) = F(B) \cap R_{n+1}(B).$$

If $n \geq 1$, *then the elements of* $R_n(B)$ *will be called n-ary relational symbols, and the elements of* $F_n(B)$ *will be called additionally n-ary functional symbols.*

□

It is easy to show that for arbitrary s.c.b. B and arbitrary $k, m > 1$, it follows from $k \neq m$ that $R_k(B) \cap R_m(B) = \emptyset$.

Definition 9.2 *Denote by* $P[2]$ *the assertion "Let*

$$n \geq 1,\ f \in F_n(B),\ tp = tp(B),\ u_1, \ldots, u_n,\ t \in Mtp(S(B)),\ tp(f) = \{(u_1, \ldots, u_n, t)\};$$

for

$$j = 1, \ldots, n,\ 0 \leq k_j \leq i,\ z_j \in Mtp(S(B)), a_j \in L(B),$$

the string $a_j \& z_j$ *belongs to* $T^{k_j}(B)$; *if* $a_j \neq V(B)$, *then* $u_j \vdash z_j$ *(i.e.* z_j *is a concretization of* u_j*); if* $a_j \in V(B)$, *then* u_j *and* z_j *are comparable with respect to* $\vdash$.

Let b *be the string of the form* $f(a_1, \ldots, a_n)$. *Then*

$$b \in L(B), b \,\&\, t \in T^2(B),\ f \,\&\, a_1 \,\&\, \ldots \,\&\, a_n \,\&\, b \in Y^2(B)\text{''}.$$

□

It should be recalled before formulating the next definition that, according to the definitions 6.1, 6.2, the symbol $\equiv$ is an element of the primary universe $X(B)$ for arbitrary s.c.b. B.

Definition 9.3 *Denote by* $P[3]$ *the assertion "Let*

$$a_1, a_2 \in L(B),\ u_1, u_2 \in Mtp(S(B)),$$

u_1 *and* u_2 *are comparable with respect to the relation* $\vdash$.

Let for $m = 1, 2$, $0 \leq k_m \leq i$, $a_m \& u_m \in T^{k_m}(B)$; P *be the sort "sense of proposition" of* B, b *be the string* $(a_1 \equiv a_2)$. *Then*

$$b \in L(B),\ b \,\&\, P \in T^3(B),$$

and the string $a_1 \,\&\, \equiv \,\&\, a_2 \,\&\, b$ *belongs to the set* $Y^3(B)$". □

Example 9.1 Let B_1 be the s.c.b. built in Example 6.2; $i = 3$;

$$\begin{aligned} b_1 &= Friends(J.Price), \\ b_2 &= Numb(Friends(J.Price)), \\ b_3 &= (Numb(Friends(J.Price)) \equiv 2), \\ b_4 &= Numb(all\,concept), \\ b_5 &= Numb(all\,chemist), \\ b_6 &= (all\,chemist \equiv M_1), \\ b_7 &= Numb(M_1). \end{aligned}$$

Then one can easily verify the validity of the following relationships (taking into account the agreement in Section 8 about the used notation):

$$\begin{aligned} & B_1(0) \Rightarrow\ Friends \,\&\, \{(ins, \{ins\})\} \in T^0, \\ & J.Price \,\&\, ins,\ Numb \,\&\, \{(\{[ent]\}, nat)\} \in T^0; \\ & B_1(1,2) \Rightarrow b_1 \,\&\, \{ins\} \in Tnr_3^2; \\ & B_1(0,2,2) \Rightarrow b_2 \in Lnr_3, b_2 \,\&\, nat \in Tnr_3^2, \\ & Numb \,\&\, b_1 \,\&\, b_2 \in Ynr_3^2; \\ & B_1(0,2,2,3) \Rightarrow b_3 \in Lnr_3,\ b_3 \,\&\, Pro \in Tnr_3^3; \\ & B_1(0,1,2) \Rightarrow b_4, b_5 \in Lnr_3, \\ & b_4 \,\&\, nat, b_5 \,\&\, nat \in Tnr_3^2; \\ & B_1(0,1,3) \Rightarrow b_6 \in Lnr_3,\ b_6 \,\&\, Pro \in Tnr_3^3; \\ & M_1 \in V(B_1), \\ & tp(M_1) = \{[ent]\} \Longrightarrow B_1(0,2) \Rightarrow b_7 \in Lnr_3, \\ & b_7 \,\&\, nat \in Tnr_3^2. \end{aligned}$$

□

Definition 9.4 *Denote by* $P[4]$ *the assertion "Let*

$$n \geq 1,\ r \in R_n(B) \setminus F(B),\ u_1, \ldots, u_n \in Mtp(S(B)),\ tp = tp(B),$$

$tp(r)$ *be the string*

$$\{(u_1, \ldots, u_n)\} \text{ or } \{u_1\} \text{ for } n = 1;$$

for

$$j = 1, \ldots, n,\ 0 \le k_j \le i,$$
$$z_j \in Mtp(S(B)),\ a_j \in L(B),$$

the string $a_j \,\&\, z_j$ belongs to $T^{k_j}(B)$;

if $a_j \notin V(B)$, then $u_j \vdash z_j$; if $a_j \in V(B)$, then u_j and z_j are comparable with respect to the relation $\vdash$.

Let b be the string $r(a_1, \ldots, a_n)$, $P = P(B)$ be the sort "sense of proposition" of B. Then

$$b \in L(B),\ b \,\&\, P \in T^4(B), \text{ and}$$
$$r \,\&\, a_1 \,\&\, \ldots \,\&\, a_n \,\&\, b \in Y^4(B)\text{''}.$$

□

Example 9.2 Let B_1 be the s.c.b. considered in Example 6.2;

$$i = 4;$$
$$b_8 = Less(10000, Numb(all\,chemist)),$$
$$b_9 = Less(50000, Numb(all\,concept)),$$
$$b_{10} = Elem(person, all\,concept),$$
$$b_{11} = Elem(R.Scott, Friends(J.Price)),$$
$$b_{12} = Subset(all\,chemist, all\,person),$$
$$b_{13} = Knows(P.Somov, (Numb(Friends(J.Price)) \equiv 2)),$$
$$b_{14} = Cause(sm\,expl_1, sm\,fire_1),$$
$$b_{15} = Less(10000, Numb(M_1)).$$

Taking into account the definition of the c.o.s. $Ct(B_1)$ (see Example 5.1) and employing the rules $P[0], \ldots, P[4]$, we have, obviously, the following relationships:

$$B_1(0,1,2,3,4) \Rightarrow b_8, \ldots, b_{15} \in Lnr_4,$$
$$b_8 \,\&\, Pro, \ldots, b_{15} \,\&\, Pro \in Tnr_4^4,$$
$$Subset \,\&\, all\,chemist \,\&\, all\,person \,\&\, Subset(all\,chemist, all\,person) \in Ynr_4^4.$$

The rule $P[5]$ is destined, in particular, for marking by variables in SRs of texts: (a) the descriptions of diverse entities mentioned in a text (physical objects, events, notions, etc.), (b) the fragments being SRs of sentences and of larger parts of texts to which a reference is given in any part of a text. □

Definition 9.5 *Denote by $P[5]$ the assertion*

"Let

$$a \in L(B) \setminus V(B),$$
$$0 \le k \le i,\ k \ne 5,\ t \in Mtp(S(B)),$$
$$a \,\&\, t \in T^k(B);\ v \in V(B),\ u \in Mtp(S(B)),$$
$$v \,\&\, u \in T^0(B),\ u \vdash t,$$

v be not a substring of the string a. Let b be the string of the form $a : v$. Then

$$b \in L(B),\ b \,\&\, t \in T^5(B),$$
$$a \,\&\, v \,\&\, b \in Y^5(B)\text{''}.$$

□

Example 9.3 Consider, as before, the s.c.b. B_1 built in Example 6.2.

Let $i = 5$, $a_1 = b_3 = (Numb(Friends(J.Price)) \equiv 2), k_1 = 3, t_1 = Pro = P(B_1)$. Then, obviously, $a_1 \,\&\, t_1 \in Tnr_5^3(B_1)$.

Suppose that $v_1 = P_1$, $u_1 = Pro = P(B_1)$. Then, according to the definition of B_1,

$$v_1 \,\&\, u_1 \in T^0(B_1);$$

besides, $u_1 \vdash t_1$(since $u_1 = t_1$), and v_1 is not a substring of a_1.

Let $b_{16} = (Numb(Friends(J.Price)) \equiv 2) : P_1$. Then, according to the rule $P[5]$,

$$b_{16} \in Lnr_5(B_1),\ b_{16} \,\&\, Pro \in Tnr_5^5(B_1),$$
$$a_1 \,\&\, v_1 \,\&\, b_{16} \in Ynr_5^5(B_1).$$

Let

$$b_{17} = sm\,expl1 : e_1, b_{18} = sm\,fire1 : e2,$$
$$b_{19} = Cause(sm\,expl1 : e_1, sm\,fire1 : e2),$$
$$b_{20} = Friends(N.Cope) : M_2$$

Then it is easy to see that

$$B_1(0,1,5) \Rightarrow b_{17},\ b_{18} \in Lnr_5,$$
$$b_{17} \,\&\, sit,\ b_{18} \,\&\, sit \in Tnr_5^5;$$
$$B_1(0,1,5,0,1,5,4) \Rightarrow b_{19} \in Lnr(5),$$
$$b_{19} \,\&\, Pro \in Tnr_5^4,$$
$$Cause \,\&\, b_{17} \,\&\, b_{18} \,\&\, b_{19} \in Ynr_5^4;$$
$$B_1(0,2,5) \Rightarrow b_{20} \in Lnr_5,$$
$$b_{20} \,\&\, \{ins\} \in Tnr_5^5,$$
$$Friends(N.Cope) \,\&\, M_2 \,\&\, b_{20} \in Ynr_5^5.$$

It should be added that the rule $P[5]$ is very important for building SRs of discourses. It allows us to form SRs of discourses in such a manner that the SRs reflect the referential structure of discourses. The examples of the kind may be found, in particular, in (Fomichov 1992, 1993b, 1994). □

10 The Use of Logical Connectives "not", "and", "or"

The rule $P[6]$ describes the use of the connective $\neg$ ("not").

Definition 10.1 *Denote by $P[6]$ the assertion*

"Let $a \in L(B)$, $t \in Mtp(S(B))$, $0 \le k \le i$, k be not in the set Forbid, $a \,\&\, t \in T^k(B)$, b be the string $\neg a$. Then $b \in L(B)$, $b \,\&\, t \in T^6(B)$, the string of the form $\neg \,\&\, a \,\&\, b$ belongs to $Y^6(B)$". □

In this assertion, the expression *Forbid* designates the set consisting of several forbidden values for k. We'll use in this article the set *Forbid* $= \{2, 5, 10\}$. This means the following: if an ℓ-formula a is inferred in some way with the help of some rules from the list $P[0], P[1], \ldots, P[i]$, then the rule $P[2]$ or $P[5]$ or $P[10]$ can't be applied at the last step of the inference.

Example 10.1 If B_1 is the s.c.b. determined in Example 6.2, $i = 6$, *Forbid* $= \{2, 5, 10\}$, then one can easily verify that

$$\begin{aligned}
&B_1(0,6) \Rightarrow \neg chemist \in Lnr_6,\\
&\neg chemist \,\&\, \uparrow ins \in T_6^6;\\
&B_1(0,6,4,4) \Rightarrow Knows(P.Somov,\\
&\quad Is1(N.Cope, \neg chemist)) \in Lnr_6;\\
&B_1(0,4,4,6) \Rightarrow \neg Knows(P.Somov,\\
&\quad Is1(N.Cope, biologist)) \in Lnr_6.
\end{aligned}$$

We'll interpret the built ℓ-formulas as possible SRs of the NL-expressions "not a chemist", "P.Somov knows that N.Cope is not a chemist", and "P.Somov doesn't know that N.Cope is a biologist".

The rule $P[7]$ describes the manners to use the connectives $\wedge$ ("and") and $\vee$ ("or") while constructing SRs of NL-texts. □

Definition 10.2 *Denote by $P[7]$ the assertion*

"Let

$$\begin{aligned}
&n > 1,\ t \in Mtp(S(B)),\ \text{for}\\
&m = 1, \ldots, n,\ 0 \le k_m \le i,\ a_m \in L(B),\\
&a_m \,\&\, t \in T^{k_m}(B),\ s \in \{\wedge, \vee\}.
\end{aligned}$$

Let b be the string of the form

$$(a_1\, s\, a_2\, s\, \ldots\, s\, a_n).$$

Then

$$\begin{aligned}
&b \in L(B),\ b \,\&\, t \in T^7(B),\\
&s \,\&\, a_1 \,\&\, \ldots \,\&\, a_n \,\&\, b \in Y^7(B)".
\end{aligned}$$

□

Example 10.2 Suppose that B_1 is the s.c.b. considered in Example 6.2, $i = 7$, *Forbid* $=$ $\{2, 5, 10\}$. Let

$$\begin{aligned}
&b_1 = (R.Scott \wedge N.Cope),\\
&b_2 = (chemist \vee biologist),\\
&b_3 = ((Numb(Friends(J.Price)) \equiv 2) :\\
&\quad P_1 \wedge Knows(P.Somov, P_1) \wedge\\
&\quad \neg Knows(P.Somov, Is1(J.Price, (chemist\\
&\quad \vee biologist)))),\\
&b_4 = Knows(P.Somov, Elem((R.Scott \wedge\\
&\quad N.Cope), Friends(J.Price))),\\
&b_5 = (Elem(R.Scott, Friends(N.Cope) :\\
&\quad M_2) \wedge \neg Elem(P.Somov, M_2)).
\end{aligned}$$

It is not difficult to show that

$$\begin{aligned}
&B_1(0,7) \Rightarrow b_1, b_2 \in Lnr_7,\\
&b_1 \,\&\, ins, b_2 \,\&\, \uparrow ins \in Tnr_7^7,\\
&B_1(0,2,2,3,5,4,7,4,4,6,7) \Rightarrow b_3 \in Lnr_7,\\
&B_1(0,7,2,4,4) \Rightarrow b_4 \in Lnr_7,\\
&b_4 \,\&\, Pro \in Tnr_7^4;\\
&B_1(0,2,5,4,4,6,7) \Rightarrow b5 \in Lnr_7,\\
&b_5 \,\&\, Pro \in Tnr_7^7.
\end{aligned}$$

□

11 Building Compound Designations of Concepts and Objects. The Use of Extensional Quantifiers. Representing n-tuples

Consider the rule $P[8]$ destined for constructing compound representations of concepts, e.g., such as

$$\begin{aligned}
&text.book * (Field, biology),\\
&concept * (Name.conc, \text{"}molecule\text{"}),\\
&tourist.group * (Number.of.persons,\\
&\quad 12)(Composition, (chemist \vee biologist)).
\end{aligned}$$

Together with the rule $P[1]$ and other rules, it will enable us to build compound designations of things and sets of things in the form $q\,des$, where q is an intensional quantifier (see Section 6), des is a compound designation of a notion formed with the help of $P[8]$ at the last step of the inference. E.g., the formulas

$$\begin{aligned}&all\,person * (Age, 18.year),\\ &some\,person * (Age, 18.year),\\ &some\,tourist.group * (Number.of.persons, 12)\end{aligned}$$

etc. It should be recalled that for arbitrary s.c.b. B (according to Definition 8.1),

$$\begin{aligned}&Tconc(B) = \{t \in Tp(S(B)) \mid t \text{ has the beginning } \uparrow\} \cup Spectp,\\ &\text{where } Spectp = \{[\uparrow ent], [\uparrow ob], [\uparrow conc]\}\end{aligned}$$

Each element c of the primary universe $X(B)$ such that $tp(c) \in Tconc(B)$ is interpreted as a designation of a concept.

The set $R_2(B)$ consists of binary relational symbols (some of them may correspond to functions with one argument); $F(B)$ is the set of functional symbols. The element $ref = ref(B)$ from $X(B)$ is called the referential quantifier (see Section 6) and is interpreted as a semantic item corresponding to the meaning of the word "some" in expressions in singular ("some book", etc.). $P(B)$ is the distinguished sort "sense of proposition" of s.c.b. B.

Definition 11.1 *Denote by $P[8]$ the assertion "Let*

$$\begin{aligned}&a \in X(B),\ tp = tp(B),\ t = tp(a),\\ &t \in Tconc(B),\ P = P(B),\ ref = ref(B).\end{aligned}$$

Let $n \geq 1$, for $m = 1, \ldots, n$, $r_m \in R_2(B)$, c_m be the string of the form $ref\ a$, and d_m, $h_m \in L(B)$;

if $r_m \in R_2(B) \cap F(B)$, then let h_m be the string of the form $(r_m(c_m) \equiv d_m)$ and, besides, $h_m \,\&\, P \in T^3(B)$;

if $r_m \in R_2(B) \setminus F(B)$, then let

$$h_m = r_m(c_m, d_m) \text{ and } h_m \,\&\, P \in T^4(B).$$

Let b be the string of the form

$$a * (r_1, d_1) \ldots (r_n, d_n).$$

Then

$$\begin{aligned}&b \in L(B),\ b \,\&\, t \in T^8(B),\\ &a \,\&\, h_1 \,\&\, \ldots \,\&\, h_n \,\&\, b \in Y^8(B)\text{''}.\end{aligned}$$

□

The rule $P[8]$ is a weakened formulation of the rule P_8 from (Fomichov 1988).

Example 11.1 Suppose that B_1 is the s.c.b. defined in Example 6.2, $i = 8$, $Forbid = \{2, 5, 10\}$. Then consider a possible way of constructing the formulas b_1 and b_2 (defined below) corresponding to the notion "a tourist group consisting of 12 persons" and to the expression "some biologist from a tourist group consisting of 12 persons". We'll use for this the denotations introduced in the end of Section 8.

Let

$$\begin{aligned}&a = tour.gr,\ t = tp(a) = \uparrow \{ins\},\\ &P = Pro, ref = sm,\\ &n = 1,\ r_1 = Numb,\\ &c_1 = ref\ a = sm\,tour.gr,\\ &d_1 = 12,\\ &h_1 = (r_1(c_1) \equiv d_1) =\\ &\quad (Numb(sm\,tour.gr) \equiv 12).\end{aligned}$$

Then

$$\begin{aligned}&B_1(0, 1, 2, 3) \Rightarrow h_1 \in Lnr_3(B_1),\\ &h_1 \,\&\, Pro \in Tnr_8^3(B_1).\end{aligned}$$

Let $b_1 = a * (r_1, d_1) = tour.gr * (Numb, 12)$. Then it follows from the rule $P[8]$ that

$$\begin{aligned}&b_1 \in Lnr_8(B_1),\\ &b_1 \,\&\, \uparrow \{ins\} \in Tnr_8^8(B_1).\end{aligned}$$

Let $b_2 = sm\,biologist{*}(Elem, sm\,tour.gr{*}(Numb, 12))$. Then

$$\begin{aligned}&B_1(0, 1, 2, 3, 8, 1, 1, 4, 8, 1) \Rightarrow b_2 \in Lnr_8(B_1),\\ &\quad b2 \,\&\, ins \in Tnr_8^8(B_1).\end{aligned}$$

□

Example 11.2 Proceeding from the same assumptions as in Example 11.1, consider a way to build a possible SR of the phrase "N.Cope has included J.Price into a tourist group consisting of 12 persons". Let

$$\begin{aligned}b3 = (&Include1(N.Cope, J.Price,\\ &sm\,tour.gr*\\ &(Numb, 12), mt1) \wedge\\ &Before(mt1, Now)).\end{aligned}$$

Then

$$\begin{aligned}&B_1(0, 1, 2, 3, 8, 1, 4, 4, 7) \Rightarrow b3 \in Lnr_8(B_1),\\ &\quad b3 \,\&\, Pro \in Tnr_8^7(B_1).\end{aligned}$$

The rule $P[9]$ describes how to join extensional quantifiers $\forall$ and $\exists$ to the SRs of propositions. E.g., we'll be able to build a SR of the sentence "For each country in Europe, there is a city with the number of habitants exceeding 50000" in the form

$$\begin{aligned}&\forall x_1 (country * (Location, Europe))\\ &\exists x_2 (city) ((Location(x_2, x_1) \wedge\\ &\quad Less(50000, Numb(Habitants(x_2))))).\end{aligned}$$

Here the expression $country * (Location, Europe)$ restricts the domain where the variable x_1 can take values.

This rule mentions the mapping $h : \{1,2\} \times Tp(S(B)) \rightarrow Tp(S(B))$ introduced in Definition 8.2, a subset of types $Tconc(B)$ (see Definition 8.1), and the relation $\vdash$ (see Definition 4.4). The expression $Numb$ used in the rule $P[9]$ designates the set consisting of numbers of some rules from the list $P[1], \ldots, P[i]$. In this paper, we'll consider for each i, $1 \leq i \leq 10$, the set $Numb = Numb(i) = \{3,4,9\} \cap \{1, \ldots, i\}$. □

Definition 11.2 *Denote by* $P[9]$ *the assertion "Let*

$$\begin{aligned}&qex \in \{\forall, \exists\},\\ &a \in L(B) \setminus V(B),\\ &P = P(B),\\ &k \in Numb,\\ &a \,\&\, P \in T^k(B),\\ &t \in Mtp(S(B)),\\ &v \in V(B),\\ &tp = tp(B),\\ &tp(v) = t,\end{aligned}$$

v be a substring of a and, besides, a don't include the substrings of the form $s\,v$, where s is one of the symbols ":", "$\forall$", "$\exists$".

Besides, let $des \in L(B) \setminus V(B)$, v be not a substring of des, u be a type from $Tconc(B)$; for some $m \in \{0, 8\}$, the string $des \,\&\, u$ belong to $T^m(B)$; $w = h(1,u)$, $t \vdash w$.

Let b be the string of the form $qex\, v\, (des)\, a$. Then

$$\begin{aligned}&b \in L(B),\ b \,\&\, P \in T^9(B),\\ &qex \,\&\, v \,\&\, des \,\&\, a \,\&\, b \in Y^9(B)\text{''}.\end{aligned}$$

□

Example 11.3 Construct a possible SR of the phrase T = "There are such a moment $mt1$ and a tourist group M_3 consisting of 12 persons that N.Cope included J.Price into the group M_3 at the moment $mt1$". Let B_1 be the s.c.b. determined in Example 6.2,

$$\begin{aligned}&i = 9,\\ &Forbid = \{2, 5, 10\},\\ &Numb = \{3, 4, 9\},\\ &b_4 = \exists\, mt1\, (mom)\\ &\quad \exists\, M_3\, (tour.gr * (Numb, 12))\\ &\qquad (Include1\, (N.Cope, J.Price, M_3, mt1) \wedge\\ &\qquad Before\, (mt1, Now)).\end{aligned}$$

Then it is easy to show that

$$\begin{aligned}&B_1(0,4,4,7,0,1,2,3,8,9,9) \Rightarrow b_4 \in Lnr_9,\\ &b_4 \,\&\, Pro \in Tnr_9^9.\end{aligned}$$

The formula b_4 is to be interpreted as a possible SR of T. □

Remark. If a basis B is chosen in such a way that the primary universe $X(B)$ includes an element $conc.ent$ having the type $tp(conc.ent) = [\uparrow ent]$ (corresponding to the conceptual item "an entity"), then for the formulas $\exists v(conc.ent)a$ and $\forall v(conc.ent)a$ the domain for v is the set of all entities of the considered application domain.

The rule $P[10]$ is destined for building representations of n-tuples ($n > 1$) in the form $\langle a_1, \ldots, a_n \rangle$, where $a_1, \ldots, a_n$ denote the components of a n-tuple.

Definition 11.3 *Denote by* $P[10]$ *the assertion "Let*

$$\begin{aligned}&n > 1,\\ &for\ m = 1, \ldots, n,\\ &\quad a_m \in L(B),\\ &\quad u_m \in Mtp(S(B)),\\ &0 \leq k_m \leq i,\\ &\quad a_m \,\&\, u_m \in T^{k_m}(B).\end{aligned}$$

Let t be the string of the form $(u_1, \ldots, u_n)$, b be the string of the form $\langle a_1, \ldots, a_n \rangle$. Then

$$\begin{aligned}&b \in L(B),\ b \,\&\, t \in T^{10}(B),\\ &a_1 \,\&\, \ldots \,\&\, a_n \,\&\, b \in Y^{10}(B)\text{''}.\end{aligned}$$

□

Example 11.4 Let B_1 be the s.c.b. from Example 6.2, $i = 10$, *Forbid* $= \{2,5,10\}$, *Numb* $= \{3,4,9\}$, and $b_5 = (Elem(x_4, M_4) \equiv ((x_4 \equiv \langle sm\,int : x_1, sm\,int : x_2\rangle) \wedge (Less(x_1,x_2) \vee (x_1 \equiv x_2))))$, where the string b_5 is to be interpreted as a possible formal definition of the binary relation "Not greater" on the set of integers. Then one can easy show that

$$B_1(0,4,1,5,10,3,4,3,7,7,3) \Rightarrow b_5 \in Lnr_{10},\ b_5\,\&\,Pro \in Tnr^3_{10}$$

□

12 Restricted K-Calculuses, Restricted Standard K-Languages, and Mathematical Investigation of Their Properties

Definition 12.1 *Let B be any s.c.b.,*

$$Forbid = \{2,5,10\},\ 1 \le i \le 10,$$
$$Numb = \{3,4,9\} \cap \{1,\ldots,i\}$$

and the sets of strings

$$L(B),$$
$$T^0(B), T^1(B), \ldots, T^k(B), \ldots, T^i(B),$$
$$Y^1(B), \ldots, Y^k(B), \ldots, Y^i(B)$$

are the least sets determined jointly by the rules $P[0], P[1], \ldots, P[i]$. Then denote these sets, respectively, by

$$Lnr_i(B),$$
$$T^0(B), Tnr^1_i(B), \ldots, Tnr^k_i(B), \ldots, Tnr^i_i(B),$$
$$Ynr^1_i(B), \ldots, Ynr^k_i(B), \ldots, Ynr^i_i(B)$$

and denote the family consisting of all these sets by $Globset_i(B)$.

Besides, let for $i = 1, \ldots, 10$,

$$Tr_i(B) = T^0(B) \cup Tnr^1_i(B) \cup \ldots \cup Tnr^i_i(B), \quad (5)$$
$$Yr_i(B) = Ynr^1_i(B) \cup \ldots \cup Ynr^i_i(B), \quad (6)$$
$$Form_i(B) = Lnr_i(B) \cup Tr_i(B) \cup Yr_i(B). \quad (7)$$

□

Definition 12.2 *If B is any s.c.b., then*

$$Lrs(B) = Lnr_{10}(B), \quad (8)$$
$$Trs(B) = Tr_{10}B), \quad (9)$$
$$Yrs(B) = Yr_{10}(B), \quad (10)$$
$$Formrs(B) = Form_{10}(B),$$
$$Krs(B) = (B, Rls),$$

where Rls is the set consisting of the rules $P[0], P[1], \ldots, P[10]$.

The ordered pair $Krs(B)$ is called the restricted K-calculus in the s.c.b. *B; the elements of the set $Formrs(B)$ are called inferred formulas of the restricted K-calculus $Krs(B)$. The formulas from the sets $Lrs(B)$, $Trs(B)$, and $Yrs(B)$ are called, respectively,* ℓ-formulas, t-formulas, *and* y-formulas. *The set of ℓ-formulas $Lrs(B)$ is called the* restricted standard K-language in the s.c.b. *B.* □

Proposition 2 *If B is an arbitrary s.c.b., then:*

(a) *the set $Lnr_0(B) \neq \emptyset$;*

(b) *for $i = 1, \ldots, 10$,*

$$Lnr_{i-1}(B) \subseteq Lnr_i(B).$$

□

Proof. (a) For arbitrary s.c.b. B, $X(B)$ includes the non-empty set $St(B)$ (see the Definitions 3.1 and 5.1). Then it follows from the rule $P[0]$ that $St(B) \subseteq Lnr_0(B)$. (b) The structure of definition 12.1 and assertions $P[0], P[1], \ldots, P[i]$ shows that the addition of the rule $P[i]$ to the list $P[0], \ldots, P[i-1]$ either enlarges the set of ℓ-formulas or doesn't change it (if the set of functional symbols $F(B)$ is empty, then $Lnr_1(B) = Lnr_2(B)$). Thus, we have the validity of Proposition 2b.

Taking into account Proposition 2, it is easy to see that Sections 8–11 provide numerous examples of ℓ-, t-, and y-formulas of restricted K-calculuses and, as a consequence, of the strings of restricted standard K-languages.

Proposition 3 *If B is any s.c.b., then*

$$Lrs(B), Trs(B), Yrs(B) \neq \emptyset.$$

□

Proof. Let B be a s.c.b. Then $Lrs(B) \neq \emptyset$ according to the Proposition 2 and the relationship (8). It follows from Definition 5.1 that the set of variables $V(B)$ includes a countable subset of such variables v that $tp(v) = [ent]$. Let v_1 and v_2 be two variables from that subset, b be the string $(v_1 \equiv v_2)$, c be the string $v_1 \& \equiv \& v_2 \& (v_1 \equiv v_2)$, $P = P(B)$ be the sort "sense of proposition" of B. Then, according to the rules $P[0]$ and $P[3]$, $b \& P \in Tnr^3_i(B)$, $c \in Ynr^3_i(B)$ for each $i = 3, \ldots, 10$. Hence [with respect to (5), (6), (9),(10)] $Trs(B)$ and $Yrs(B)$ are non-empty.

Proposition 4 *If B is a s.c.b., then:*

(a) *if $w \in Trs(B)$, then w is a string of the form $a \& t$, where $a \in Lrs(B)$, $t \in Tp(S(B))$, and such a representation depending on w is unique for each w;*

(b) *if $y \in Yrs(B)$, then there are such $n > 1$ and $a_1, \ldots, a_n$, $b \in Lrs(B)$ that $y = a_1 \& \ldots \& a_n \& b$; besides, such a representation depending on y is unique for each y.*

□

Proof. The structure of the rules $P[0], \ldots, P[10]$ and the definitions 6.2, 12.1, 12.2 immediately imply the truth of this proposition.

Proposition 5 *If B is a s.c.b., $d \in X(B) \cup V(B)$, then there are no such k, where $1 \leq k \leq 10$, $n > 1$, and $a_1, \ldots, a_n \in Lrs(B)$ that*

$$a_1 \& \ldots \& a_n \& d \in Ynr^k_{10}(B) \qquad (11)$$

□

Proof Assume that there are such $k \in \{1, \ldots, 10\}$, $n > 1$, $a_1, \ldots, a_n \in Lrs(B)$ that the relationship (11) takes place. For arbitrary $m \in \{1, \ldots, 10\}$, the set $Ynr^m_{10}(B)$ may include a string $a_1 \& \ldots a_n \& d$, where d contains no occurrences of the symbol &, only in case d is obtained out of $a_1, \ldots, a_n$ by means of applying one time the rule $P[m]$. It follows from the structure of the rules $P[1], \ldots, P[10]$ that d must contain at least two symbols. But we consider the elements of the set $X(B) \cup V(B)$ as symbols. Hence we get a contradiction, since we assume that $d \in X(B) \cup V(B)$. Q.E.D.

It would be not difficult also to prove the following two propositions, but this goes beyond the scope of the present paper.

Proposition 6 *Let B be an s.c.b., $b \in Lrs(B) \setminus (X(B) \cup V(B))$. Then there is one and only one such a sequence $(k, n, a_1, \ldots, a_n)$, where $1 \leq k \leq 10$, $n > 1$, $a_1, \ldots, a_n \in Lrs(B)$, that $a_1 \& \ldots \& a_n \& b \in Ynr^k_{10}(B)$.* □

Proposition 7 *Let B be a s.c.b., $b \in Lrs(B)$. Then there is one and only one such type $t \in Tp(S(B))$ that $b \& t \in Trs(B)$.* □

This proposition is a direct consequence of Proposition 5 and Proposition 6. Proposition 7 enables us to define a mapping tl from $Lrs(B)$ into $Tp(S(B))$ in the following way: for each $b \in Lrs(B)$, $tl(b)$ is equal to such $t \in Tp(S(B))$ that $b \& t \in Trs(B)$. Obviously, for each $b \in X(B) \cup V(B)$, $tl(b) = tp(b)$. Hence each string of a restricted standard K-language will be associated with some type.

13 Expressive Possibilities of Restricted Standard K-Languages

For formulating and explaining numerous definitions in preceding sections, very simple examples were deliberately chosen. The collection of these examples is far from demonstrating the real power of the constructed model. That's why let's consider several additional examples illustrating some important possibilities of restricted standard K-languages.

Suppose that E is some NL-expression, *Semp* is a string of the restricted standard K-language in some s.c.b. and, besides, *Semp* is a possible SR of E. Then we'll say that *Semp* is a K-representation (KR) of E.

Example 13.1 Let T_1 = "Somebody hadn't switched off a knife-switch. As a result, the Laboratory No. 11 was burnt down". Then the string

$$\begin{aligned}
&(\exists\, mt1\, (mom * (Before, Now))\\
&\quad \neg\, Switch.off(sm\, person : x_1,\\
&\quad sm\, knife.switch : x_2, mt1) : P_1 \wedge\\
&\quad Descr.sit(P_1, e_1) \wedge\\
&\quad \exists\, mt2\, (mom * (Before, Now))\\
&\quad Burn.down(sm\, lab*\\
&\quad (No, 11) : x_3, mt2) :\\
&\quad P_2 \wedge Descr.sit(P_2,\ e_2) \wedge\\
&\quad Cause(e_1, e_2))
\end{aligned}$$

is a possible KR of T_1. Here P_1 and P_2 are variables of the type being the distinguished sort "sense of proposition"; e_1 and e_2 are to be interpreted as variables designating, respectively, two situations (events) and having the type being the sort "situation". □

Example 13.2 Let T_2 = "Yves said to Mary that he was occupied with rowing and painting. The new for Mary was that Yves was occupied with painting". Then we may consider the formula

$$\begin{aligned}
&\exists\, mt1\, (mom * (Before, Now))\\
&\quad (Say(\langle Agent, some\, person*\\
&\qquad (First\, name, \text{“}Yves\text{”}) : y_1\rangle,\\
&\qquad \langle Addressee, some\, person*\\
&\qquad (First\, name, \text{“}Mary\text{”}) : y2\rangle,\\
&\qquad \langle Moment, mt1\rangle,\\
&\qquad \langle Info, Be.occupied(y1,\\
&\qquad (rowing \wedge painting)) : P_1\rangle) \wedge\\
&\qquad New(y_2, P_1, mt1,\\
&\qquad Be.occupied(y_1, painting)))
\end{aligned}$$

as a possible KR of T_2. □

Example 13.3 Let T_3 = "The notion "a molecule" is used in physics, chemistry, and biology". One can determine such an s.c.b. B that the set of sorts $St(B)$ includes the elements *areal* and *string*, the primary universe $X(B)$ includes the elements

$$\begin{aligned}
&areal,\ string,\ concept,\ \text{“}molecule\text{”},\\
&Is.used,\ Name.conc,\ physics,\ some,\\
&chemistry,\ biology,\ tp(concept) = [\uparrow conc],\\
&tp(\text{“}molecule\text{”}) = string,\\
&tp(physics) = tp(chemistry) =\\
&\quad tp(biology) = areal,\\
&tp(Is.used) = \{([conc], areal)\},\\
&tp(Name.conc) = \{([conc], string)\},
\end{aligned}$$

some is the referential quantifier of B, *Is.used* and *Name.conc* are not functional symbols. Let

$$\begin{aligned}
&s_1 = Name.conc(some\, concept, \text{“}molecule\text{”}),\\
&s_2 = concept * (Name.conc, \text{“}molecule\text{”});\\
&s_3 = Is.used(some\, concept\ *\\
&\qquad (Name.conc, \text{“}molecule\text{”}),\\
&\qquad (physics \wedge chemistry \wedge biology)).
\end{aligned}$$

Then

$$\begin{aligned}
&B(0,1,4) \Rightarrow s_1 \in Lrs(B);\\
&B(0,1,4,8) \Rightarrow s_2 \in Lrs(B);\\
&B(0,1,4,8,1,7,4) \Rightarrow s_3 \in Lrs(B).
\end{aligned}$$

The formula s_3 is a possible KR of T_3. □

Example 13.4 Let T_4 = "Teenager is a person having the age from 12 to 19 years". Let u be the string

$$\begin{aligned}
&((teenager \equiv person * (Age, x_1)) \wedge\\
&\quad \neg\, Less(x_1, 12.year) \wedge\\
&\quad \neg\, Greater(x_1, 19.year)).
\end{aligned}$$

Then u is a possible KR of T_4. □

Example 13.5 Nebel and Peltason (1991) formulate the following definition: a small and medium enterprise (*sme*) is a company with at most 50 employees. This definition may have the KR

$$\begin{aligned}
&Definition(sme,\\
&\quad \forall\, x_1\, (company1)\\
&\qquad (Is1(x_1, sme) \equiv\\
&\qquad \neg\, Greater(Number(Employees(x_1)),\\
&\qquad 50)))).
\end{aligned}$$

or the KR

$$\begin{aligned}
&((sme \equiv company1 * (Description, P_1)) \wedge\\
&\quad (P_1 \equiv \forall\, x_1 (company1)\\
&\quad (Is1(x_1, sme) \equiv\\
&\qquad \neg\, Greater(Number(Employees(x_1)),\\
&\qquad 50))))
\end{aligned}$$

□

Example 13.6 We can build complex descriptions of diverse objects and sets of objects. E.g., we can build the following KR of the description of "Informatica":

$$\begin{aligned}&some\, int.sc.journal*\\ &\quad (Title, \text{``}Informatica\text{''})\\ &\quad (Country, Slovenia)\\ &\quad (City, Ljubljana)\\ &\quad (Fields, (artif.intel \wedge\\ &\qquad cogn.science \wedge\\ &\qquad databases)) : k225\end{aligned}$$

where $k225$ is the mark of the knowledge module with the data about "Informatica". □

Example 13.7 In a similar way, we can construct a knowledge module stating the famous Pythagorean Theorem and indicating also its author and field. For instance, such a module may be the following expression of some restricted standard K-language:

$$\begin{aligned}&some\, textual.object*\\ &\quad (Kind, theorem)\\ &\quad (Fields, geometry)\\ &\quad (Authors, Pythagoras)\\ &\quad (Meaning,\\ &\qquad \forall x_1(geom)\forall x_2(geom)\\ &\qquad \forall x_3(geom)\forall x_4(geom)\\ &\qquad If\text{-}then((Is1(x_1,\\ &\qquad right\text{-}triangle)\wedge\\ &\qquad Hypotenuse(x_2, x_1)\wedge\\ &\qquad Leg((x_3 \wedge x_4), x_1)),\\ &\qquad (Square(Length(x_2)) \equiv\\ &\qquad Sum(Square(Length(x_3)),\\ &\qquad Square(Length(x_4)))))) : k81.\end{aligned}$$

□

Example 13.8 The meaning of the question "Is Ghent located in Belgium ?" may be represented by the ℓ-formula

$$Question1(Location(Ghent, Belgium))$$

where $Question1$ is a special unary relational symbol (r.s.) with the type P, and P is the sort "sense of proposition". □

Example 13.9 The question "What and whom has John told?" may have a KR

$$\begin{aligned}&Question2((x_2 \wedge x3),\\ &\quad (Tell(\langle Agent1,\\ &\qquad some\, person*\\ &\qquad (Name, \text{``}John\text{''}) : x_1\rangle,\\ &\quad \langle Content, x_2\rangle,\\ &\quad \langle Moment, x_3\rangle)\wedge\\ &\quad Before(x_3, Now)))\end{aligned}$$

where x_1, x_2, x_3 are variables of the type $[ent]$, $Question2$ is a binary r.s. with the type $\{([ent], P)\}$. □

Example 13.10 The phrase "Professor P. Jones advised M. Smith to enter the Stanford University and prepare a Ph.D. thesis on physics" may have a deep KR

$$\begin{aligned}&(Pose\text{-}goal(\langle Agent1,\\ &\qquad some\, person*\\ &\qquad (Name, \text{``}P.Jones\text{''})\\ &\qquad (Qualif, prof) : x_1\rangle,\\ &\quad \langle Adressee,\\ &\qquad some\, person*\\ &\qquad (Name, \text{``}M.Smith\text{''}) : x_2\rangle,\\ &\quad \langle Form, advice\rangle,\\ &\quad \langle Goal, (Entering1*\\ &\qquad (Inst, Stanford.Univ)\wedge\\ &\qquad Preparing1*\\ &\qquad (Goal\text{-}product,\\ &\qquad certain\, ph.d.thesis *\\ &\qquad (Field1, physics))) : x_3\rangle,\\ &\quad \langle Moment, x_4\rangle)\wedge\\ &\quad Before(x_4, Now)).\end{aligned}$$

In a similar manner, one can build KRs of phrases with the verbs "to order", "to request", etc.

Thus, restricted standard K-languages are convenient, in particular, for: .

(a) reflecting casual relationships in SRs of discourses (Example 13.1); one can reflect time relationships between described situations in a similar way;

(b) explicating references to the meanings of phrases or larger parts of discourses (example 13.2);

(c) building SRs of phrases with the words "notion", "concept" (example 13.3);

(d) explicit indicating in SRs conceptual cases, or semantic cases, or deep cases(examples 13.2, 13.9, 13.10);

(e) building formal definitions of notions (examples 13.4, 13.5);

(f) building complex designations of diverse objects (example 13.6);

(g) storing information pieces in object-like knowledge modules representing both the meaning of an information piece (a notion, a theorem, a rule, an abstract, a patent, etc.) and the values of its external characteristics: the authors, the date of inputting into computer system or publishing a piece, fields of application, etc.(example 13.7);

(h) constructings SRs of questions with the answers "Yes" or "No" (example 13.8) and with the interrogative words (example 13.9);

(i) building semantic analogues of complicated goals and SRs of phrases with orders, advices, etc. (Example 13.10).

□

Numerous other examples illustrating the expressive power of standard K-languages and as a consequence, of restricted standard K-languages (in many cases) may be found in (Fomichov 1992, 1993b, 1994).

The analysis of the constructed model and of the examples considered in Sections 3–13 enables us to draw the conclusion that the task stated in Section 2 is solved.

Introduce an informal notion of a stationary NL-text. Assume that K is some knowledge base (k.b.). If T is an arbitrary NL-text, then we'll say that T is a *stationary text with respect to* K in case T doesn't contain fragments *introducing* new concepts and/or relationships (but T may include phrases *explaining* such concepts and/or relationships that the considered k.b. *contains* their designations as informational items).

It appears that there are reasons to put forward the following

Hypothesis. *The expressive possibilities of restricted standard K-languages are sufficient and these languages are convenient for representing on some* deep *conceptual sublevel the structured meanings (SMs) of such arbitrarily complicated real sentences and discourses which are considered as stationary texts with respect to any k.b.* □

In other words, there are reasons *to conjecture* that the constructed model may be interpreted as a variant of a *Universal Stationary Metagrammar of Deep Conceptual Syntax*. Naturally, it is necessary to carry out further studies in order to prove or to correct the formulated hypothesis.

It was shown above that the built model affords the opportunity to reflect in SRs many peculiarities of surface semantic structure of texts. But one can essentially extend its expressive possibilities in this direction proceeding from the ideas set forth in (Fomichov 1992, 1993b, 1994). The *second hypothesis* is as follows: modifying the rule $P[8]$ and adding the rules $P11, P12, P13, P14$ shortly characterized in (Fomichov 1992), we are able to develop a variant of a *Universal Stationary Metagrammar of Conceptual Structures* (or of *Conceptual Syntax*).

At last, the *third hypothesis* is that proceeding from the ideas stated in this paper and in (Fomichov 1992, 1993b, 1994), we can construct a variant of a *Universal Metagrammar of Conceptual Structures* (being convenient for describing both stationary and dynamic semantics of NL-texts).

The initial version of the model described in this paper was published in (Fomichov 1988). That monograph was also the basis for preparing the articles (Fomichov 1992, 1993b, 1994). That's why it seems that the work (Fomichov 1988) provides the first variants of metagrammars mentioned in the formulated second and third hypotheses.

14 Related Approaches

In comparison with Montague Grammar (MG), Theory of Generalized Quantifiers (TGQ), Situation Theory (ST), Discourse Representation Theory (DRT), Dynamic Predicate Logic (DPL) set forth in (Groenendijck & Stokhof 1991), the constructed model has a lot of common advantages as concerns formal describing SMs of NL-texts and representing knowledge. These common advantages are the properties 2–9, 10.2, 10.3, 10.6, 12, 13, 16–18 formulated in the task statement in Section 2.

Comparing the model with the Theory of Conceptual Graphs (TCG), we can see that the advantages of the model are the properties 3–5, 7–9, 10.2, 10.3, 10.6, 12, 13, 16–18.

The Theory of Semantic Structures (Jackendoff 1990) is stated in an absolutely non-mathematical form. But in (Zwarts & Verkuyl 1994) a mathematical interpretation of this theory is suggested. In comparison with that mathematical interpretation, the elaborated model possesses additionally, in particular, the properties 3–9, 10.2, 10.3, 10.6, 12, 13, 16–18.

The constructed model is an important component of Integral Formal Semantics (IFS)—a powerful approach to mathematical studying the use of NL; the basic principles and composition of IFS are set forth in (Fomichov 1994). The model enables us to approximate all manners suggested by the mentioned approaches to build SRs of NL-texts and represent knowledge and provides numerous additional expressive possibilities. However, both the model and IFS as a whole are very far from all enumerated approaches to investigating NL-semantics if we take into account the posed goals and used mathematical methods.

Meanwhile, there is one approach to the study of NL-semantics with the help of formal methods which is rather close to IFS as concerns the general goals of researches (but far from the standpoint of technical aspects). It is Episodic Logic (EL) developed by L.K. Schubert and C.H. Hwang (Schubert & Hwang 1989, Hwang 1992, Hwang & Schubert 1993a, 1993b, 1993c). Both approaches advocate the need of highly expressive (natural-language-like) formal systems of semantic and knowledge representations as the ground of a comprehensive framework for developing the theory of NLPSs capable to understand complicated NL-discourses. EL is qualified by its authors as an extended first order, intensional, highly expressive logic; the structure of its formulas called logical forms (LFs) and quasi logical forms (QLFs) reflects many peculiarities of NL-texts.

It is possible to show that the elaborated model allows us to approximate all formulas of EL considered in (Hwang & Schubert 1993b). One of the principal ideas how to do this underlies the example 13.1. Besides, the model has a number of advantages: the properties 2, 3.1 (concerning complicated designations of sets), 3.2, 3.3, 4–9, 10.2, 10.3, 10.6, 13, 16–18.

It may be pointed out also that the principal idea of EL as a tool for building NL-like semantic and knowledge representations of texts was anticipated as far back as in the works (Fomichov 1981, 1982; Fomitchov 1983, 1984) setting forth the theory of free S-models, restricted S-languages of types 1–4, S-calculuses and S-languages of types 1–5 (see also Fomichov 1994).

Moreover, the expressive power of restricted S-languages of type 4 (Fomichov 1982), S-calculuses and S-languages of types 4 and 5 (Fomitchov 1983, 1984) exceeds the expressive power of EL. In particular, one can show that expressive possibilities of LFs and QLFs considered in all examples in (Hwang & Schubert 1993b) may be modelled by formulas of restricted S-languages of type 4 (Fomichov 1982) or by formulas of S-calculuses and S-languages of type 4 or 5 (Fomitchov 1983, 1984).

It should be added that the model has at least three global distinctive features as concerns its structure and destination in comparison with EL. The first feature is as follows. In fact, the purpose of this paper is to represent in a mathematical form a *hypothesis* about the general mental mechanisms (or operations) underlying the formation of complicated conceptual structures (or semantic structures, or knowledge structures) out of basic conceptual items. EL doesn't undertake an attempt of the kind, and 21 Backus-Naur forms used in (Hwang 1992) for defining the basic logical syntax (b.l.s.) rather disguise such mechanisms (operations) in comparison with more general 11 inference rules of the model determined above and 15 rules described in (Fomichov 1988, 1992).

The second global distinctive feature is that this paper formulates a hypothesis about a *complete collection of operations* of some deep conceptual sublevel providing the possibility to build effectively the conceptual structures corresponding to arbitrarily complicated real sentences and discourses pertaining to science, technology, business, medicine, law, etc.

The third global distinction is that the form of describing in EL the b.l.s. is *not a strictly mathematical one.* E.g., the collection of Backus-Naur forms used for defining b.l.s. in (Hwang 1992) contains the expressions

$$\langle \textit{1-place-pred-const} \rangle := \textit{happy} \mid \textit{person} \mid \textit{certain} \mid \textit{probable} \mid \ldots,$$
$$\langle \textit{1-fold-pred-modifier-const} \rangle := \textit{plur} \mid \textit{very} \mid \textit{former} \mid \textit{almost} \mid \textit{in-manner} \mid \ldots$$

The only way to escape the use of three dots in productions is to define some analogue of the notion of a simplified conceptual basis introduced in the present paper.

A large stock of formal expressions for describing surface semantic structure of texts is provided by the Core Language Engine (CLE)—a domain independent system for translating a wide range of English sentences into formal representations of their literal meanings (Alshawi & van Eijck 1989; Alshawi 1990; Alshawi (Ed.) 1992). The CLE has two representation languages, their expressions are called logical forms (LFs) and quasi logical forms (QLFs). The model described in this paper enables us to approximate all expressive possibilities of LFs and QLFs. The model possesses additionally the properties 3.2, 3.3, 4–8, 10.3, 10.6, 16–18. Besides, the model allows us to build much more complicated designations of sets (the property 3.1). So the model has essential advantages from the standpoint of practice. As for theoretical aspects, the orientations of the model and CLE are very different, and the model has the same three global distinctive features in comparison with CLE as in comparison with EL.

The model described in Sections 3–13 has important advantages in comparison with the terminological knowledge representation language (TKRL) $L_{\rm LILOG}$ developed by a number of German universities and research institutes (Herzog & Rollinger (Eds.) 1991). First, it is possible to show that the model allows us to approximate all expressive possibilities of $L_{\rm LILOG}$ (Section 5 of Fomichov 1994 may be of help for this). Second, the model has the advantages 3.2, 3.3, 4, 7, 10.6, 13, 16, 17 and enables us to build much more complicated definitions of concepts and designations of objects.

It appears that the most important integral advantage of the constructed model in comparison with all approaches discussed above is that only the model presented in this paper together with the ideas set forth in (Fomichov 1988, 1992, 1993b, 1994) provides a sound mathematical basis for studying the regularities of conveying information by arbitrarily complicated real NL-texts pertaining to diverse areas of human activity and formalizing arbitrary kinds of NL-dialogue.

15 Some Possible Applications of Results

15.1 Theoretical Applications

In (Fomichov 1992, 1993a; Martin-Vide 1993) it is pointed out at the necessity of mathematical studying NL in a broader context than it is being done traditionally: at the expediency of creating a mathematical theory of NL-communication between active intelligent systems.

The ideas described above and in (Fomichov 1988, 1992, 1993a, 1993b, 1994) provide the ground for developing *mathematical linguocybernetics*, or *mathematical theory of natural language use*, or *mathematical theory of NL-communication*—in other terms, mathematical foundations of designing arbitrarily complicated text-based intelligent systems (full-text databases, etc.) and robust NL-interfaces to autonomous intelligent agents and other applied intelligent systems. In particular, the results may be applied to formalizing abductive inference (see Hobbs, Stickel, Appelt, & Martin 1993) as inference aimed at the construction of the best explanation of conceptual ties between fragments of a discourse.

In comparison with the state of affairs reflected in (Partee, ter Meulen, & Wall 1990, ICML'93), this paper considerably extends the limits of mathematical linguistics, opens a door to constructing *generative algebraic models* of complicated natural-language conceptual structures in addition to *analytical algebraic models* of language characterized in (Marcus 1993).

The constructed model provides very rich formal means to describe knowledge about the world and, as a consequence, to describe various situations and relations between situations. That's why it seems that the model may give a new strong impulse to the studies in Situation Theory.

The results of this paper may be used also for building models of knowledge bases, formalizing common-sense reasoning and heterogeneous reasoning, integrating multiple knowledge sources.

15.2 Significance for Practice

The model enables the designers of applied intelligent systems to work up formal languages providing one or several of the following possibilities: to build SRs of complicated real NL-texts pertaining to diverse areas of human activity; to represent the intermediate results of the conceptual analysis of NL-texts; to describe background knowledge; to construct high-level conceptual representations of complicated visual images; to describe lexical semantics; to construct ontologies of application domains; to develop more powerful and flexible hybrid knowledge representation systems. The model opens a lot of new prospects in all these directions.

16 Conclusions

The model determined and studied in this paper enriches in a leap-like manner the stock of formal means for describing structured meanings (SMs) of NL-texts and representing other conceptual items. Combining this model with the ideas set forth in English in (Fomichov 1992, 1993b, 1994), we have also an effective method of constructing formal models being convenient for describing SMs of practically arbitrary (very likely, arbitrary) real discourses and for representing knowledge about the world. That method was published in full in Russian in the monograph (Fomichov 1988), which, it seems, provided the first variant of a Universal Conceptual Metagrammar.

The results stated above bridge a gap between formal semantics of NL and the demands of the theory of text-based intelligent systems and of several other subfields of computer science. They open a way for mathematical studying the regularities of conveying information by arbitrarily complicated real discourses pertaining to science, technology, medicine, business, law, etc. Besides, it may be hoped that obtained results will be effectively used for studying a number of actual theoretical and practical problems in computer and cognitive science.

Acknowledgements

I thank two anonymous referees for constructive comments helped to improve the presentation style of the paper.

I am deeply grateful to Prof. A.P. Železnikar for the kind help (going beyond the scope of the Editor's duties) in improving the submitted LaTEX file of this paper.

References

[1] Aczel, P., Israel, D., Katagiri, Y., & Peters, S. (Eds) (1993): Situation Theory and Its Applications, Vol. 3. CSLI Lecture Notes No. 37. Stanford: CSLI Publications.

[2] Ahrenberg,L.(1992): On the integration and scope of segment-based models of discourse. In Papers from the Third Nordic Conf. on Text Comprehension in Man and Machine, Linkoeping Univ., 1–16.

[3] Alshawi, H. & van Eijck, J. (1989): Logical Forms in the Core Language Engine. In Proc. 27th Ann. Meeting of the ACL, Vancouver, Canada, 25–32.

[4] Alshawi, H. (1990): Resolving quasi logical forms. Computational Linguistics, 16(3), 133–144.

[5] Alshawi, H. (Ed.)(1992): The Core Language Engine, MIT Press, Cambridge, MA.

[6] A Plan (1992): A Plan for the Knowledge Archives Project, The Economic Res. Inst.; Japan Soc. for the Promotion of Machine Industry; Systems Res. & Dev. Inst. of Japan, Tokyo, March, 1–78.

[7] Barwise, J. (1989): The Situation in Logic. CSLI Lecture Notes No. 17. Stanford., Calif.: CSLI.

[8] Barwise, J. (1992): Information links in domain theory. In Proc. of the Math. Foundations of Programming Semantics Conf. (1991), ed. S. Brookes et al., Lecture Notes in Computer Science, No. 598, Springer, 168–192.

[9] Barwise, J. (1993): Constraints, channels, and the flow of information. In Aczel, Israel, Katagiri, & Peters (1993), 3–27.

[10] Barwise, J. & Cooper, R. (1993): Extended Kamp notation: a graphical notation for Situation Theory. In Aczel, Israel, Katagiri, & Peters (1993), 29–53.

[11] Barwise, J. & Etchemendy, J. (1990): Information, infons, and inference. In Situation Theory and Its Applications 1, ed. R. Cooper, K. Mukai, and J. Perry, CSLI Lecture Notes No. 22. Stanford, Calif: CSLI, 33–78.

[12] van Benthem, J. (1992): Fine-structure in categorial semantics. In Rosner & Johnson (1992), 127–157.

[13] Cooper, R. (1991): Three lectures on Situation Theoretic Grammar. In Natural Language Processing, ed. M. Filgueiras et al. Berlin etc.: Springer-Verlag, 102–140.

[14] Devlin, K. (1991): Logic and Information. Cambridge: Cambridge Univ. Press.

[15] Ellis, G. (1995): Compiling conceptual graphs. IEEE Trans. on Knowl. and Data Eng., 7(1), 68–81.

[16] Fenstad, J.E., Langholm, T., & Vestre, E. (1992): Representations and interpretations. In Rosner & Johnson (1992), 31–95.

[17] Fomichov, V.A. (1981): To the theory of logic-algebraic modelling the speech-forming mechanisms of conceptual level. 1. Task statement and the idea of an approach to its resolving. Moscow Inst. of Electronic Eng., 85 pages.—Paper deposited in AUISTI of Ac. Sc. USSR (All-Union Inst. of Sc. and Techn. Information; in Russian—Vsesoyuzny Institut Nauchnoy i Technicheskoy Informatsii, or VINITI) 27/10/81, No. 4939-81 Dep.—The abstract 2B1386DEP in the Abstracts J. "Mathematica" of AUISTI, 1982, No. 2.

[18] Fomichov, V.A. (1982): Development and application of methods of logic-algebraic modelling the mechanisms of speech-formation semantics. Ph.D. dissertation. Moscow: Moscow Inst. of Electr. Eng.

[19] Fomitchov, V.A. (1983): Formal systems for natural language man-machine interaction modelling. In Int. Symp. on Artif. Intel. IFAC, Vol. 1. Leningrad: Ac. Sc. USSR, 223–243 (in two variants: in Russian and in English). Paper presented at the First Symp. of the Intern. Feder. of Autom. Control on Artif. Intell.

[20] Fomitchov, V.A. (1984): Formal systems for natural language man- machine interaction modelling. In Artificial Intelligence. Proc. of the IFAC Symp. Leningrad, USSR, 4–6 Oct. 1983, ed. V.M. Ponomaryov (IFAC Proc. Series, 1984, No. 9). Oxford, UK; New York, etc.: Pergamon Press, 203–207.

[21] Fomichov, V.A. (1988): Representing Information by Means of K-calculuses. Text-book. Moscow: The Moscow Institute of Electronic Engineering Press (in Russian).

[22] Fomichov, V. (1992): Mathematical models of natural-language-processing systems as cybernetic models of a new kind. Cybernetica (Belgium), XXXV (1), 63–91.

[23] Fomichov, V.A. (1993a): Towards a mathematical theory of natural-language communication. Informatica. An Intern. J. of Computing and Informatics (Slovenia), 17(1), 21–34.

[24] Fomichov, V.A. (1993b): K-calculuses and K-languages as powerful formal means to design intelligent systems processing medical texts. Cybernetica, XXXVI (2), 161–182.

[25] Fomichov, V. (1994): Integral Formal Semantics and the design of legal full-text databases. Cybernetica, XXXVII (2), 145–177.

[26] Groenendijk, J. & Stokhof, M. (1991): Dynamic Predicate Logic. Linguistics and Philosophy, 14(1), 39–101.

[27] Herzog, O. & Rollinger, C.-R. (Eds.)(1991): Text Understanding in LILOG. Integrating Computational Linguistics and Artificial Intelligence. Final Report on the IBM Germany LILOG-Project. Berlin etc.: Springer-Verlag.

[28] Hobbs, J.R., Stickel, M.E., Appelt, D.E., & Martin, P. (1993): Interpretation as abduction. Artif. Intelligence, 63(1–2), 69–142.

[29] Hwang, C.H (1992): A Logical Approach to Narrative Understanding. Ph.D. Dissertation, U. of Alberta, Edmonton, Canada.

[30] Hwang, C. H. & Schubert, L.K (1993a): Meeting the interlocking needs of LF-computation, deindexing, and inference: An

organic approach to general NLU. Proc. 13th Int. Joint Conf. on AI (IJCAI'93). Chambery, France, 28 Aug.–3 Sept. 1993.

[31] Hwang, C.H. & Schubert, L.K. (1993b): Episodic Logic: a comprehensive, natural representation for language understanding. Minds and Machines, 3 , 381–419.

[32] Hwang, C.H. & Schubert, L.K. (1993c): Episodic Logic: a situational logic for natural language processing. In Aczel, Israel, Katagiri, & Peters (1993), 303–338.

[33] ICML'93 (1993): 1 Intern. Conf. on Math. Linguistics. ICML'93. Tarragona (Catalonia, Spain), March 30–31, 1993. Ed.: Carlos Martin-Vide.

[34] Jackendoff, R.S. (1990): Semantic Structures. MIT Press, Cambridge, Mass.

[35] Kamp, H. & Reyle, U. (1993): Introduction to Modeltheoretic Semantics of Natural Language, Formal Logic and Discourse Representation Theory. Dordrecht: Kluwer Academic Publishers.

[36] Markus, S. (1993): The status of research in the field of analytical algebraic models of language. In ICML'93, 47–48.

[37] Martin-Vide, C. (1993): How to infer nonformal conclusions from mathematical models in the analysis of natural language: a preliminary case study in communication theory. In ICML'93, 49–50.

[38] Meyer, R. (1994): Probleme von Zwei-Ebenen-Semantiken. Kognitionswissenschaft, Band 4, Heft 1, 32–46.

[39] Nebel, B. & Peltason, C. (1991): Terminological reasoning and information management. In Information Systems and Artif. Intel.: Integration Aspects. First Workshop. Ulm, FRG, March 1990, Proc., ed. D. Karagiannis. Berlin etc.: Springer-Verlag, 181–212.

[40] Partee B. H., ter Meulen, A., & Wall, R. (1990): Mathematical Methods in Linguistics. Dordrecht etc.: Kluwer.

[41] Rosner, M. & Johnson, R. (Eds) (1992): Computational Linguistics and Formal Semantics. Cambridge: Cambridge Univ. Press.

[42] Schubert, L. K. & Hwang, C. H. (1989): An episodic knowledge representation for narrative texts. Proc. 1st Int. Conf. on Principles of Knowledge Repres. and Reasoning (KR'89), Toronto, Canada, 444–458.

[43] Sowa, J.F. (1984): Conceptual Structures: Information Processing in Mind and Machine. Addison-Wesley Publ. Comp.: Reading, MA.

[44] Sowa, J.F. (1991): Toward the expressive power of natural language. In Sowa, J.F. (Ed.), Principles of Semantic Networks. Explorations in the Representation of Knowledge. Morgan Kaufman Publ., Inc.

[45] Wah, B.W., Huang, T.S., Joshi, A.K. et al. (1993): Report on Workshop on High Performance Computing and Communications for Grand Challenge Applications: Computer Vision, Speech and Natural Language Processing, and Artificial Intelligence. IEEE Trans. on Knowl. and Data Eng., 5(1), 138–154.

[46] Železnikar, A.P. (1993): Mission and Research Reports. Informatica, 17(1), 81–100.

[47] Zwarts, J. & Verkuyl, H. (1994): An algebra of conceptual structure; an investigation into Jackendoff's conceptual semantic. Linguistics & Philosophy, 17(1), 1–28.

A Logic for Conditional Recommendation

S. K. Das
William Penney Laboratory, Imperial College, London SW7 2AZ
Phone: +44 171 594 8424, Fax: +44 171 594 8449
E-mail: `skd@doc.ic.ac.uk`
AND
P. Hammond
Department of Computer Science, Brunel University, Middlesex UB8 3PH
Phone: +44 1895 203393, Fax: +44 1895 251686
E-mail: `P.Hammond@brunel.ac.uk`

Keywords: conditional, decision support system, logic, recommendation

Edited by: Johann Eder
Received: November 22, 1994 **Revised:** December 15, 1995 **Accepted:** December 22, 1995

A decision support system is a computerized system which utilizes knowledge about a particular application area to help decision makers by recommending suitable actions. An action is conditionally recommendable when a set of conditions associated with it is expected to be brought about in the state resulting from performing the action. Conditional recommendations occur frequently in practical decision-making contexts but their formal treatment is yet to be explored. Our approach is to develop a modal propositional logic $\mathcal{L}_{cr}$ for reasoning about conditional recommendations. The semantics of the logic is studied in terms of possible worlds where transition from one state to another is made by performing an action. The soundness and completeness of the logic is established. We also discuss the exploitation of $\mathcal{L}_{cr}$ to formalize update and integrity constraint verification in knowledge bases.

1 Motivation

The formalization of conditional recommendation is important in the implementation of safety-critical systems, those decision support systems [3] where malfunction or operator error may give rise to severe injury, death or financial/environmental catastrophe with the potential for legal liability. The work presented here arises from our involvement in a project addressing the technology and legal implications of knowledge-based decision support systems in safety-critical domains. To illustrate and motivate the abstract notions, we give examples in the domain of decision support in oncology, a specific interest of one of the project partners.

Because of the complexity of treatment performed and the amount of data captured during oncology trials, computerized decision support is essential. The safety-critical aspect arises from the narrow window between efficacy and toxicity of many of the treatments, but especially in chemotherapy [9].

A decision concerned with the selection of a particular treatment plan, or protocol, can be expressed as a high-level recommendation [6]:

> If the patient has operable bone cancer then consideration of treatment under protocol BO03 can be recommended.

This kind of sentence has the general form

> If condition ϕ (the patient has operable bone cancer) holds in the present state of the knowledge base then an action α (consideration of treatment under protocol BO03) can be recommended.

Symbolically these sentences are of the form

$$\phi \rightarrow \mathcal{R}(\alpha)$$

where $\mathcal{R}(\alpha)$ is read as "action α is recommendable".

The condition ϕ is sometimes complex and involves conjunctions and disjunctions. For example, consider the following recommendation of an action:

> If the patient has operable bone cancer and the patient's renal function is adequate then the inclusion of the drug cisplatin in chemotherapy can be recommended.

In this case the condition ϕ is a conjunction of two subconditions that the patient has operable bone cancer and the patient's renal function is adequate. The action α is the inclusion of the drug cisplatin in chemotherapy. On the other hand, sentences like

> If the patient is eligible for treatment under protocol BO03 and folinic acid rescue and prehydration schemes are instituted before chemotherapy and posthydration afterwards and the patient is available for subsequent toxicity monitoring then the inclusion of high-dose methotrexate in chemotherapy can be recommended.

involves an action with two associated conditions. The general form of such sentences is

> If a set of conditions ϕ (the patient is eligible for treatment under protocol BO03 and folinic acid rescue and prehydration schemes are instituted before chemotherapy) holds in the present state of the knowledge base then an action α (inclusion of high-dose methotrexate in chemotherapy) can be recommended provided another condition C (posthydration is instituted and the availability of patient for toxicity monitoring) is brought about in the state obtained by taking the action.

Symbolically these sentences are of the form

$$\phi \rightarrow \mathcal{CR}(\alpha, C)$$

where $\mathcal{CR}(\alpha, C)$ is called a *conditional* and read as "action α is conditionally recommendable provided condition C". These sentences are of the most general kind. The conditions ϕ and C will be called respectively the *precondition* and the *postcondition* for the recommendation of α. We aim to formalize the behaviour of postconditions by developing a logic $\mathcal{L}_{cr}$, an extension of propositional logic. The reason for ignoring preconditions and concentrating on postconditions is that a precondition can be verified on the current state in a straightforward manner using a theorem prover. On the other hand, the assumption of a postcondition is a prediction and therefore its study beforehand is important.

Now if the postcondition in a conditional is a property (for example, the patient's availability or low temperature) then it can be brought about by performing, in general, more than one type of action in the transformed state (see figure 1). For example, if the patient's low temperature is to be brought about after an operation, then for each type of treatment for bringing down temperature a state exists in which the patient does have low temperature. On the other hand, if an action (for example, posthydration) is to brought about, then it can be implemented by performing the action itself and another state is obtained after performing the action (see figure 2). It is our assumption that if a property is to be brought about then there must exist at least one action whose execution in the present state will make the property true in the transformed world.

The rest of the paper is organized as follows. Sections 2, 3 and 4 present respectively the syntax, axioms and semantics of $\mathcal{L}_{cr}$. The soundness and completeness result of $\mathcal{L}_{cr}$ is presented in section 5. Section 6 provides a detailed guideline for formalizing update and constrains verification in a knowledge base. Throughout the rest of the paper, unless otherwise stated, conditions in conditionals will be denoted by C, D, ...; property symbols by p, q, ...; action symbols by α, β, ...; formulae by F, G, ...; worlds by w, w', ...

2 Syntax

In this section, we present the syntax of the language of $\mathcal{L}_{cr}$ which makes provision for representing conditionals of the form discussed in the previous section. The language of $\mathcal{L}_{cr}$ is the usual propositional one extended with a binary modal operator.

Due to the dynamic nature of the applications

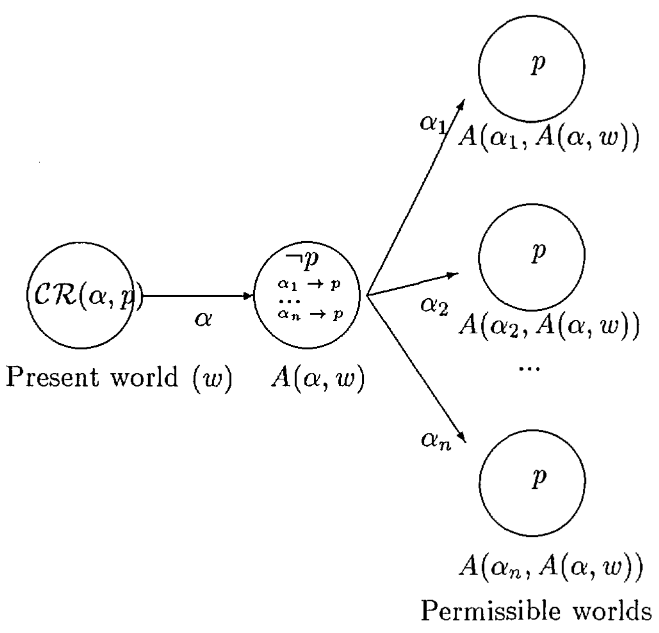


Figure 1: Relationship between the present world and permissible worlds in the presence of the conditional $\mathcal{CR}(\alpha,p)$, where p is a property and $A(\alpha,w)$ is the world obtained from w by executing α.

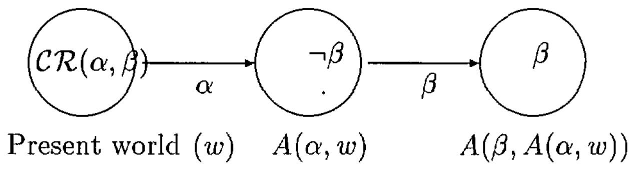


Figure 2: Relation between the present world and the final world in the presence of the conditional $\mathcal{CR}(\alpha,\beta)$, where β is an action.

we have dealt with, the description of the universe of discourse of $\mathcal{L}_{cr}$ is considered to have two aspects: *static* and *dynamic*. The static aspect of the universe of discourse is described by *properties* (e.g., the patient has a bone cancer) and dynamic aspects by *occurrences* (e.g., the patient is given an injection). For a detailed discussion of properties, occurrences, etc., readers are referred to [1]. An occurrence is either an event or a process. Some occurrences involve animate agents (i.e., decision makers) performing *actions*. We shall not consider occurrences where animate agents do not perform any actions. In view of this, the set of propositional symbols P_0, P_1, ... is partitioned into two disjoint subsets as $\mathcal{P}$ and $\mathcal{A}$ to represent respectively properties and actions.

The domain of propositions includes the special property symbol $\top$ (true). The modal operator of $\mathcal{L}_{cr}$ is $\mathcal{CR}$. The *formulae* $\mathcal{F}$ of $\mathcal{L}_{cr}$ are as follows:

- All propositional formulae are modal formulae.
- An atomic conditional $\mathcal{CR}(\alpha,C)$ is a modal formula, where α is an action and C is a formula.
- $\neg\phi$, $\phi \wedge \psi$ are modal formulae, where ϕ and ψ are modal formulae.

Note that we consider formulae that are more general than the form $\phi \to \mathcal{CR}(\alpha,C)$ as suggested in section 1. We take $\perp$ (false) to be an abbreviation of $\neg\top$. Other logical connectives are defined using $\neg$ and $\wedge$ in the usual manner. $\mathcal{CR}(\alpha,C)$ is read as "action α is conditionally recommendable provided condition C" and the formula $\mathcal{CR}(\alpha,\top)$ is simply written as $\mathcal{R}(\alpha)$ and is read as "action α is recommendable". If β is an action then the conditional $\mathcal{CR}(\alpha,\beta)$ states that the action α is conditionally recommendable provided β is performed in the state obtained from the current state by performing α (figure 2). Note that the presence of an action β in the knowledge base represents the occurrence that β has been taken.

3 Axioms

To present the axioms of $\mathcal{L}_{cr}$, consider first the propositional logic and we have

all instances of propositional tautologies (1)

The above set includes instances of propositional tautologies that may involve any number of the modal operators, for example, $\mathcal{CR}(\alpha,C) \to \mathcal{CR}(\alpha,C)$. The *modus ponens* rule

MP: if $\vdash F$ and $\vdash F \to G$ then $\vdash G$

is the only rule of inference, where '$\vdash$' denotes derivability in $\mathcal{L}_{cr}$. As a first step of extending propositional logic to meet our objectives, we have to cover situations where no action can bring $\perp$ about. Therefore, we consider the following axiom as part of $\mathcal{L}_{cr}$:

$$\neg\mathcal{CR}(\alpha,\perp) \tag{2}$$

For any action α, equivalent conditions C and D guarantee the conditionals $\mathcal{CR}(\alpha,C)$ and $\mathcal{CR}(\alpha,D)$ are equivalent. This can be represented by the following *equivalence* inference rule:

EQ: if $\vdash C \leftrightarrow D$ then $\vdash \mathcal{CR}(\alpha, C) \leftrightarrow \mathcal{CR}(\alpha, D)$

Consider the possibility of replacing the above inference rule by a more general kind of inference rule "if $\vdash D \rightarrow C$ then $\vdash \mathcal{CR}(\alpha, C) \rightarrow \mathcal{CR}(\alpha, D)$". This can be justified by the fact that if bringing about D in a state implies bringing about C then α being conditionally recommendable provided C will yield that α is also conditionally recommendable provided D. We exclude this possibility for the following reason. If $D \rightarrow C$ holds then bringing about D in a state implies bringing about at least C and possibly more, if $C \rightarrow D$ does not hold. Correspondingly, the additional actions may affect the objective for which α is recommended.

Readers may have observed the similarity between axiom (2) and inference rule EQ and the axiom and inference rule of minimal deontic logic [2, 4], which can be obtained by replacing each $\mathcal{CR}(\alpha, C)$ by the obligation $\mathcal{O}(C)$. Later, in section 5, we shall take advantage of this to develop the soundness and completeness of $\mathcal{L}_{cr}$ in line with that of minimal deontic logic. The major difference is the variation of the modal operator $\mathcal{CR}$ for every action. Also, we have two more axioms in $\mathcal{L}_{cr}$ and they are presented below.

In any particular state of a knowledge base more than one kind of action can be conditionally recommendable and we choose the most suitable one to execute. Therefore, in a particular state and for a particular action α we may be able to derive $\mathcal{CR}(\alpha, C), \mathcal{CR}(\alpha, D)$, and so on. Thus, to say that α is conditionally recommendable provided condition C and α is conditionally recommendable provided condition D is equivalent to saying that α is conditionally recommendable provided condition C or D. This statement provides the following axiom of $\mathcal{L}_{cr}$:

$$\mathcal{CR}(\alpha, C) \wedge \mathcal{CR}(\alpha, D) \rightarrow \mathcal{CR}(\alpha, C \vee D) \tag{3}$$

The above axiom provides us with a view of alternative conditions available to satisfy when a specific action is conditionally recommended. Note that, if we have $\mathcal{CR}(\alpha, C)$ and $\mathcal{CR}(\alpha, \neg C)$ then the axiom (3) together with the inference rule EQ generate $\mathcal{CR}(\alpha, \top)$, that is, $\mathcal{R}(\alpha)$. This is quite expected because either C or $\neg C$ holds in any state and thus α can simply be recommended without any postcondition.

In our knowledge base we may have actions in different names which are equivalent to each other. By an equivalence of two actions α and β (written as $\alpha \leftrightarrow \beta$) we mean that the cost, benefit, outcome, etc., involved in executing α is exactly the same as that of β. For example, if the school is located 100 metres in a northerly direction and the only road to the school is in this direction then the action of walking to the school and the action of walking 100 metres north are equivalent. This motivates us to consider the following inference rule as part of $\mathcal{L}_{cr}$:

$$(\alpha \leftrightarrow \beta) \rightarrow (\mathcal{CR}(\alpha, C) \leftrightarrow \mathcal{CR}(\beta, C)) \tag{4}$$

The axiom states that if two actions are equivalent then the recommendability of one action provided some condition is equivalent to the recommendability of the other provided the same condition. The above axiom generates equivalent conditionals with respect to actions. The axiom also provides us with a way of renaming actions in a knowledge base.

This completes our presentation of axioms and inference rules of $\mathcal{L}_{cr}$. Now we show that the system is consistent by means of the following proposition.

Proposition 1 *The axiom system of $\mathcal{L}_{cr}$ is consistent. In other words, $\nvdash_{\mathcal{L}_{cr}} \perp$.*

Proof The propositional subset of $\mathcal{L}_{cr}$ is consistent. Axiom (2) is the source of only one conditional. Therefore, the resultant system is still consistent. Inference rule EQ fails to generate a positive conditional of the form $\mathcal{CR}(\alpha, C)$ which could have been inconsistent with axiom (2). Therefore consistency is still preserved. We now add axiom (3). This axiom does generate a positive conditional of the form $\mathcal{CR}(\alpha, C \vee D)$ given $\mathcal{CR}(\alpha, C)$ and $\mathcal{CR}(\alpha, D)$. But neither C nor D is equivalent to $\perp$ due to the presence of axiom (2) and inference rule EQ. Therefore the addition of this axiom cannot result in inconsistency. Finally, when axiom (4) is added to obtain $\mathcal{L}_{cr}$, we are still unable to derive an atomic conditional. □

We now provide an example to illustrate the formalization and the deduction process within $\mathcal{L}_{cr}$.

Example 1

Consider the following sentence of conditional recommendation which has already been cited in section 1:

> If the patient is eligible for treatment under protocol BO03 and folinic acid rescue and prehydration schemes are instituted before chemotherapy then the action of the inclusion of high-dose methotrexate in chemotherapy is recommended provided posthydration is instituted and the patient is available for toxicity monitoring.

Consider another sentence which is a rule of the knowledge base:

> If the patient has satisfactory diagnosis for protocol then the patient is eligible for treatment under protocol.

The symbolic representations of the above two sentences are as follows:

$q \wedge \beta \rightarrow \mathcal{CR}(\alpha, \gamma \wedge r)$
$p \rightarrow q$

where α, β and γ are action symbols whose meanings are described as follows:

α - inclusion of high-dose methotrexate.
β - institution of fol. acid rescue and prehyd.
γ - institution of posthydration scheme.

The symbols p, q and r represent different properties of patients as described below:

p - has satisfactory diagnosis for protocol.
q - eligible for treatment under protocol.
r - patient is available for toxicity monitoring.

Now, consider the following state of a knowledge base:

$\{\ p \rightarrow q,\ q \wedge \beta \rightarrow \mathcal{CR}(\alpha, \gamma \wedge r)\ \}$

Suppose we have evidence that the patient has a satisfactory diagnosis for the protocol and folinic acid rescue and prehydration scheme are instituted. Then the above state is changed to the following state:

$\{\ p,\ \beta,\ p \rightarrow q,\ q \wedge \beta \rightarrow \mathcal{CR}(\alpha, \gamma \wedge r)\ \}$

Now $\mathcal{CR}(\alpha, \gamma \wedge r)$ becomes derivable which conditionally recommends α. If we execute α, the state of the knowledge base is changed to the following:

$\{\ \alpha,\ p,\ \beta,\ p \rightarrow q,\ q \wedge \beta \rightarrow \mathcal{CR}(\alpha, \gamma \wedge r)\ \}$

Since α has been executed, $\mathcal{CR}(\alpha, \gamma \wedge r)$ now acts as an obligation on $\gamma \wedge r$. In other words, γ should be executed and r should be brought about. If these are carried out, the final state without any obligation is transformed to the following:

$\{\ \gamma,\ r,\ \alpha,\ p,\ \beta,\ p \rightarrow q,\ q \wedge \beta \rightarrow \mathcal{CR}(\alpha, \gamma \wedge r)\ \}$

Suppose we have $s \vee \gamma \wedge r$ as the postcondition instead of $\gamma \wedge r$ and s has been brought about. The following would have been the final state without any obligations:

$\{\ s,\ \alpha,\ p,\ \beta,\ p \rightarrow q,\ q \wedge \beta \rightarrow \mathcal{CR}(\alpha, s \vee \gamma \wedge r)\}$

4 Semantics

The description of the universe of discourse of $\mathcal{L}_{cr}$ is interpreted as a conceptual model which consists of a set of possible worlds [10]. In this section we study conceptual models of $\mathcal{L}_{cr}$ in terms of minimal models.

Definition 1 *A minimal model for the logic $\mathcal{L}_{cr}$ can be defined as*

$$\mathcal{M}_{cr} = \langle W, \{R_\alpha\}_{\alpha \in \mathcal{A}}, B \rangle$$

where W is a set of possible worlds and B maps a proposition P to a set of worlds for which P is true. Each R_α is a mapping that selects a set of formulae, for each world w, such that, if $C \in R_\alpha(w)$ then α is conditionally recommendable in w provided condition C.

A proposition or a formula can be uniquely characterized by a set of worlds in each of which it is true. Therefore, the mapping R_α can be defined as

$$R_\alpha : W \rightarrow \Pi(\Pi(W))$$

where $\Pi(X)$ is the power set of X. The set of worlds from W for which C is true is a member of $\Pi(\Pi(W))$ and will be denoted as $\parallel C \parallel_{\mathcal{M}_{cr}}$ or simply $\parallel C \parallel$, when the name of the model is clear from the context. On the other hand, a member S of $\Pi(\Pi(W))$ is of the form $\parallel C \parallel_{\mathcal{M}_{cr}}$, where C is the member of $\mathcal{F}$ which is uniquely characterized by S.

For every w in W, the set $\{R_\alpha\}_{\alpha\in\mathcal{A}}$ satisfies the following three properties:

P1: $\emptyset \notin R_\alpha(w)$.
P2: if $S_1,\ S_2 \in R_\alpha(w)$ then $S_1 \cup S_2 \in R_\alpha(w)$.
P3: if $\|\alpha\| = \|\beta\|$ then $R_\alpha(w) = R_\beta(w)$.

These three properties are due to axioms (2), (3) and (4) respectively. Property P1 makes each $R_\alpha(w)$ a collection of non-empty sets. Property P2 makes $R_\alpha(w)$ closed under union.

Not every action can be recommended in every world. Each world $w \in W$ has a certain number of defined actions and each defined action when executed on w transforms it to another world. When an action α is undefined in a world w then $R_\alpha(w)$ is empty. If an action α is defined in every world then each $R_\alpha(w)$ contains $\top$.

The rules for determining truth values of formulae in $\mathcal{M}_{cr}$ and a world w are as follows:

$\models^w_{\mathcal{M}_{cr}} \top$
$\models^w_{\mathcal{M}_{cr}} p$ iff $w \in B(p)$
$\models^w_{\mathcal{M}_{cr}} \neg F$ iff $\not\models^w_{\mathcal{M}_{cr}} F$
$\models^w_{\mathcal{M}_{cr}} \mathcal{CR}(\alpha, C)$ iff $\|C\| \in R_\alpha(w)$
$\models^w_{\mathcal{M}_{cr}} F \wedge G$ iff $\models^w_{\mathcal{M}_{cr}} F$ and $\models^w_{\mathcal{M}_{cr}} G$

Definition 2 *A formula F is said to be true in model $\mathcal{M}_{cr}$, written as $\models_{\mathcal{M}_{cr}} F$, if and only if $\models^w_{\mathcal{M}_{cr}} F$, for every world w in W. A formula F is said to be valid, written as $\models F$, if F is true in every model.*

5 Soundness and Completeness

This section contains the soundness and completeness results for $\mathcal{L}_{cr}$ using the concept of minimal model [4]. $\mathcal{L}_{cr}$ is said to be sound with respect to a class of models Γ if and only if for every model $\mathcal{M}$ in Γ and for every sentence $F \in \mathcal{F}$, if $\vdash F$ then $\models_{\mathcal{M}} F$. $\mathcal{L}_{cr}$ is said to be complete with respect to Γ if and only if for every $\mathcal{M}$ in Γ and for every sentence $F \in \mathcal{F}$, if $\models_{\mathcal{M}} F$ then $\vdash F$. The following two propositions prove that the validity in a class of models is preserved by the use of the rule of inference and the axioms of $\mathcal{L}_{cr}$.

Proposition 2 *Let $\mathcal{M}_{cr} = \langle W, \{R_\alpha\}_{\alpha\in\mathcal{A}}, B\rangle$ be a minimal model. Then, $\models_{\mathcal{M}_{cr}} C \leftrightarrow D$ implies $\models_{\mathcal{M}_{cr}} \mathcal{CR}(\alpha, C) \leftrightarrow \mathcal{CR}(\alpha, D)$.*

Proof If $\models_{\mathcal{M}_{cr}} C \leftrightarrow D$ then $\|C\| = \|D\|$. Therefore, for any world w in $\mathcal{M}_{cr}$, $\|C\| \in R_\alpha(w)$ iff $\|D\| \in R_\alpha(w)$. So for any world w in $\mathcal{M}_{cr}$, $\models^w_{\mathcal{M}_{cr}} \mathcal{CR}(\alpha, C)$ iff $\models^w_{\mathcal{M}_{cr}} \mathcal{CR}(\alpha, D)$. Thus, $\models_{\mathcal{M}_{cr}} \mathcal{CR}(\alpha, C) \leftrightarrow \mathcal{CR}(\alpha, D)$. $\square$

Proposition 3 *For every formula F and G the axioms of $\mathcal{L}_{cr}$ are valid, that is,*

1. $\models \neg\mathcal{CR}(\alpha, \bot)$.

2. $\models \mathcal{CR}(\alpha, C) \wedge \mathcal{CR}(\alpha, D) \rightarrow \mathcal{CR}(\alpha, C \vee D)$

3. $\models (\alpha \leftrightarrow \beta) \rightarrow (\mathcal{CR}(\alpha, C) \leftrightarrow \mathcal{CR}(\beta, C))$.

Proof Let $\mathcal{M}_{cr} = \langle W, \{R_\alpha\}_{\alpha\in\mathcal{A}}, B\rangle$ be a model and $w \in W$.

1. By property P1 of $\mathcal{M}_{cr}$, $\emptyset \notin R_\alpha(w)$. Thus, $\|\bot\| \notin R_\alpha(w)$; otherwise, $R_\alpha(w)$ would contain the $\emptyset$ to satisfy $\|\bot\| \in R_\alpha(w)$. Thus, $\not\models^w_{\mathcal{M}_{cr}} \mathcal{CR}(\alpha, \bot)$, that is $\models^w_{\mathcal{M}_{cr}} \neg\mathcal{CR}(\alpha, \bot)$.

2. If $\models_{\mathcal{M}_{cr}} \mathcal{CR}(\alpha, C) \wedge \mathcal{CR}(\alpha, D)$ then $\models_{\mathcal{M}_{cr}} \mathcal{CR}(\alpha, C)$ and $\models_{\mathcal{M}_{cr}} \mathcal{CR}(\alpha, D)$. Therefore, for any world w, $\|C\|$, $\|D\| \in R_\alpha(w)$ and thus, by property P2 of $\mathcal{M}_{cr}$, $\|C\| \cup \|D\| \in R_\alpha(w)$. This means that $\|C \vee D\| \in R_\alpha(w)$, that is, $\models_{\mathcal{M}_{cr}} \mathcal{CR}(\alpha, C \vee D)$.

3. If $\models^w_{\mathcal{M}_{cr}} \alpha \leftrightarrow \beta$ then, by property P3 of $\mathcal{M}_{cr}$, $R_\alpha(w) = R_\beta(w)$. Therefore, $\models^w_{\mathcal{M}_{cr}} \mathcal{CR}(\alpha, C)$ iff $\|C\| \in R_\alpha(w)$ iff $\|C\| \in R_\beta(w)$ iff $\models^w_{\mathcal{M}_{cr}} \mathcal{CR}(\beta, C)$. Therefore, $\models^w_{\mathcal{M}_{cr}} \mathcal{CR}(\alpha, C) \leftrightarrow \mathcal{CR}(\beta, C)$. $\square$

Propositions 2 and 3 establish the basis of the soundness result. In order to prove the completeness result, the following class of models is relevant.

Definition 3 *A minimal model $\mathcal{M}_{cr} = \langle W, \{R_\alpha\}_{\alpha\in\mathcal{A}}, B\rangle$ for the logic $\mathcal{L}_{cr}$ is called a canonical minimal model, written as $\mathcal{M}^c_{cr}$, if and only if*

- *$W = \{w$: w is a maximal consistent set in logic $\mathcal{L}_{cr}\}$. (N.B. The maximal consistent sets exist because $\mathcal{L}_{cr}$ is consistent).*

- *For every w in W, $\mathcal{CR}(\alpha, C) \in w$ if and only if $|C|_{\mathcal{L}_{cr}} \in R_\alpha(w)$, where $|C|_{\mathcal{L}_{cr}}$ (or simply $|C|$) is the set of maximal consistent set of sentences containing C in logic $\mathcal{L}_{cr}$.*

– *For each $n = 0, 1, 2, \ldots$, the set of all possible worlds for which the proposition P_n is true is $\mid P_n \mid_{\mathcal{L}_{cr}}$.*

In the above definition of the canonical minimal model we have considered the set of worlds W as the set of maximal consistent sets of sentences. Therefore, we should make sure that when an action α is performed on a maximal consistent set w, leads uniquely to another world $A(\alpha, w)$ which is also a maximal consistent set. The mapping A in this case can be defined as follows. $A(\alpha, w)$ is constructed initially from α itself and all non-atomic sentences of w. An atomic proposition P (resp. $\neg P$) from w is added to $A(\alpha, w)$ if it is consistent with $A(\alpha, w)$; otherwise, $\neg P$ (resp. P) is added to $A(\alpha, w)$.

Proposition 4 *Let $\mathcal{M}^c_{cr} = \langle W, \{R_\alpha\}_{\alpha \in \mathcal{A}}, B\rangle$ be a canonical minimal model for $\mathcal{L}_{cr}$. Then, for every $w \in W$ and $F \in \mathcal{F}$, $\models_{\mathcal{M}^c_{cr}} F$ iff $F \in w$.*

Proof Since each $w \in W$ is a maximal consistent set of $\mathcal{L}_{cr}$, then $\models^w_{\mathcal{M}^c_{cr}} F$ iff $F \in w$ for every $F \in \mathcal{F}$ without any occurrences of $\mathcal{CR}$. Consider the conditional $\mathcal{CR}(\alpha, C)$.
Then, by induction hypothesis, we have
$\models^w_{\mathcal{M}^c_{cr}} C$ iff $C \in w$.
So for any $w \in W$,
$\| C \| \in R_\alpha(w)$ iff $\mid C \mid \in R_\alpha(w)$.
Since $\mathcal{M}^c_{cr}$ is a minimal model,
$\models^w_{\mathcal{M}^c_{cr}} \mathcal{CR}(\alpha, C)$ iff $\| C \| \in R_\alpha(w)$.
Therefore, from the last two sentences,
$\models^w_{\mathcal{M}^c_{cr}} \mathcal{CR}(\alpha, C)$ iff $\mid C \mid \in R_\alpha(w)$.
By the definition of the canonical minimal model,
$\mathcal{CR}(\alpha, C) \in w$ iff $\mid C \mid \in R_\alpha(w)$.
Therefore, from the previous two sentences,
$\models^w_{\mathcal{M}^c_{cr}} \mathcal{CR}(\alpha, C)$ iff $\mathcal{CR}(\alpha, C) \in w$.
Therefore, for every $F \in \mathcal{F}$,
$\models^w_{\mathcal{M}^c_{cr}} F$ iff $F \in w$ □

Therefore, the worlds in a canonical minimal model for a system of modal logic will always verify just those sentences they contain. In other words, the sentences which are true in such a model are precisely the theorems of the system. Thus, we have the following theorem.

Theorem 1 *Let $\mathcal{M}^c_{cr} = \langle W, \{R_\alpha\}_{\alpha \in \mathcal{A}}, B\rangle$ be a canonical minimal model for $\mathcal{L}_{cr}$. Then, $\models_{\mathcal{M}^c_{cr}} F$ if and only if $\vdash F$.*

Proof Follows from the above remark. □

Theorem 2 *If $\mathcal{M}^c_{cr}$ is a canonical minimal model of $\mathcal{L}_{cr}$ then the model satisfies properties P1, P2 and P3.*

Proof To show that $\mathcal{M}^c_{cr}$ satisfies property P1, if possible, let $\emptyset \in R_\alpha(w)$. This means that $\|\bot\| \in R_\alpha(w)$, that is, $\models^w_{\mathcal{M}^c_{cr}} \mathcal{CR}(\alpha, \bot)$. By proposition (4), $\mathcal{CR}(\alpha, \bot) \in w$. Also, $\models^w_{\mathcal{M}^c_{cr}} \neg\mathcal{CR}(\alpha, \bot)$, that is, $\neg\mathcal{CR}(\alpha, \bot) \in w$, which yields a contradiction as w is consistent.

To prove that $\mathcal{M}^c_{cr}$ satisfies property P2, suppose $\| C_1 \|$, $\| C_2 \| \in R_\alpha(w)$, that is, $\models^w_{\mathcal{M}^c_{cr}} \mathcal{CR}(\alpha, C_1)$ and $\models^w_{\mathcal{M}^c_{cr}} \mathcal{CR}(\alpha, C_2)$. By proposition (4), $\mathcal{CR}(\alpha, C_1)$, $\mathcal{CR}(\alpha, C_2) \in w$. Since w is maximal consistent, $\mathcal{CR}(\alpha, C_1) \wedge \mathcal{CR}(\alpha, C_2) \in w$. Now, axiom (3) is a member of w and w is maximal consistent. These suggest that $\mathcal{CR}(\alpha, C \vee D) \in w$, that is, by proposition (4), $\models^w_{\mathcal{M}^c_{cr}} \mathcal{CR}(\alpha, C \vee D)$. By definition, $\| C_1 \vee C_2 \| \in w$, that is, $\| C_1 \|$, $\| C_2 \| \in w$.

To establish that $\mathcal{M}^c_{cr}$ satisfies property P3, suppose $\| \alpha \| = \| \beta \|$, that is, $\models^w_{\mathcal{M}^c_{cr}} \alpha \leftrightarrow \beta$. By proposition (4), $\alpha \leftrightarrow \beta \in w$. Now, axiom (4) is a member of w and w is maximal consistent. These suggest that $\mathcal{CR}(\alpha, C) \leftrightarrow \mathcal{CR}(\beta, C) \in w$. Again, by proposition (4), $\models^w_{\mathcal{M}^c_{cr}} \mathcal{CR}(\alpha, C) \leftrightarrow \mathcal{CR}(\beta, C)$, that is, $\models^w_{\mathcal{M}^c_{cr}} \mathcal{CR}(\alpha, C)$ iff $\models^w_{\mathcal{M}^c_{cr}} \mathcal{CR}(\beta, C)$. Thus, by definition, $\| C \| \in R_\alpha(w)$ iff $\| C \| \in R_\beta(w)$. Therefore, $R_\alpha(w) = R_\beta(w)$. □

Suppose Γ is the class of all models satisfying the properties P1, P2 and P3. Then the following soundness and completeness theorem establishes the fact that $\mathcal{L}_{cr}$ is determined by Γ.

Theorem 3 *For every formula F in $\mathcal{L}_{cr}$, $\vdash F$ if and only if $\models F$.*

Proof The first half follows from propositions 2 and 3. To prove the converse, suppose $\models F$. Theorem 1 implies that there is a canonical minimal model $\mathcal{M}^c_{cr}$, satisfying properties P1, P2 and P3, such that $\vdash F$ if and only if $\models_{\mathcal{M}^c_{cr}} F$. Now, theorem 2 establishes that $\mathcal{M}^c_{cr} \in \Gamma$. Thus, $\models F$ implies $\models_{\mathcal{M}^c_{cr}} F$. Therefore, $\vdash F$. □

6 Discussion

In this section we discuss a special case of $\mathcal{L}_{cr}$ to handle updates to knowledge bases formally and the corresponding integrity constraint verification. Following the traditional approach in databases, a knowledge-based decision support system can be defined as a knowledge base together with a set of integrity constraints [5]. Each integrity constraint must be satisfied by every state of the knowledge base. Following an update to the knowledge base the relevant constraints are evaluated; this can be achieved through meta-level reasoning. An alternative way of handling this is by the notion of what we call *conditional possession.* A property p can be possessed conditionally by a knowledge base provided condition C must hold in the state after possessing the property. This is expressed as $\mathcal{CR}(p, C)$.

There are some differences between a conditional possession and a conditional recommendation. The latter is more general than the former. First of all, conditional recommendation of an action implies conditional possession of a set of properties or occurrences which reflect the change due to the action. Secondly, if a conditionally recommended action is executed then it cannot be reversed. On the other hand, a conditionally possessed property can be retracted. Finally, in the case of a conditional possession, the condition C in a conditional must hold or must be brought about in the state after possessing the said property; otherwise, it does not make sense to continue using the possessed property and therefore the property must be retracted. In contrast, in the case of conditional recommendation, the condition C in a conditional is to be brought about, failing which it may result in undesirable consequences.

Consider an example of an integrity constraint that a person's height must be less than 3 metres. If the property to be possessed is the person's height then the condition is that for every person the height must be less than 3 metres. If we enter 3.2 as a person's height then we shall be able to derive $\perp$ from the knowledge base and this contradicts axiom (2). If we continue using values contradicting constraints we may arrive at an absurd drug dosage or result such as a probability greater than one. The former arises because drug dosage is typically computed according to height and weight. Therefore the possessed property must be retracted.

Consider another example where $\perp$ can be derived after possessing the property and we bring about the condition and therefore restore consistency by another possession. Consider the conditional $\mathcal{CR}(p, C)$ where the property p is a student's record and C is the integrity constraint that for every student there must exist a supervisor. If we now possess a new student record before assigning its supervisor then the knowledge base becomes inconsistent, that is, $\perp$ is derivable from the knowledge base. To bring back the consistency, that is, to bring about C, we have an option to enter a supervisor for that student rather than retracting the student's record.

What has just been discussed reflects the sensitivity of a property to be possessed by allowing conditionals of the form $\mathcal{CR}(p, C)$, that is, the conjunction of relevant integrity constraints C (which may be the empty condition $\top$) to be evaluated upon possessing the property. If every possible property to be possessed is annotated with its conjunction of relevant integrity constraints then the knowledge-based system can be viewed as a single entity consisting of sentences involving propositions as well as conditionals.

Although we have named the primitive construct $\mathcal{CR}(\alpha, C)$ of $\mathcal{L}_{cr}$ as conditional, there is no similarity between a conditional logic and $\mathcal{L}_{cr}$. (for an account of conditional logics, see [11]). The construct $\mathcal{CR}(\alpha, C)$ will not be equivalent to the traditional type of conditional $\alpha > C$. The idea of embedded implication [7, 8] in N-Prolog is similar to our concept of conditional possession discussed in this section. An embedded implication $D \Rightarrow G$, where G is a goal and D is a set of clauses, succeeds from an N-Prolog program P if G succeeds from the enlarged program P$\cup\{D\}$. On the other hand, condition C arising from a conditional $\mathcal{CR}(p, C)$ due to the possession of p is said to have been fulfilled in the context of a knowledge base KB provided C succeeds from KB$\cup\{p\}$.

7 Conclusion

We have developed a modal logic extending propositional logic for reasoning with conditionals. It would be more natural to consider an underlying logic more general than propositional such as temporal first-order and then introduce the conditional modal operator on it. We have shown that a special case of the logic is useful to formalize update and integrity constraint verification in knowledge bases.

Acknowledgements

SKD completed this work while working at the Computer Science Department of Queen Mary and Westfield College, London. Similarly, PH was previously at the Advanced Computation Laboratory, Imperial Cancer Research Fund, London. The authors would like to thank their colleagues in the RED project and two anonymous reviewers for helpful comments. The project was supported under the DTI/SERC project ITD 4/1/9053: Safety-Critical Systems Initiative.

References

[1] J. F. Allen. Towards a general theory of action and time. *Artificial Intelligence*, 23:123–154, 1984.

[2] L. Aqvist. Deontic logic. In D. Gabbay and R. Guenthner, editors, *Extensions of Classical Logic*, volume 2 of *Handbook of Philosophical Logic*, pages 605–714. D. Reidel Publishing Company, 1985.

[3] R. H. Bonczek, C. W. Holsapple and A. B. Whinston. Development in decision support systems. *Advances in Computers*, 23:123–154, 1984.

[4] B. Chellas. *Modal Logic.* Cambridge University Press, 1980.

[5] S. K. Das. *Deductive Databases and Logic Programming.* Addison-Wesley, 1992.

[6] S. K. Das and J. Fox. A logic for reasoning about safety in decision support systems. In Clarke et al., editors, *Proceedings of the Second European Conference on Symbolic and Quantitative Approaches to Reasoning and Uncertainty*, pages 79–84. Springer-Verlag, November 1993.

[7] D. M. Gabbay. N-Prolog: an extension of prolog with hypothetical implications. II. logical foundations and negation as failure. *Journal of Logic Programming*, 4:251–283, 1985.

[8] D. M. Gabbay and U. Reyle. N-prolog: an extension of prolog with hypothetical implications. I. *Journal of Logic Programming*, 4:319–355, 1984.

[9] P. Hammond, A. L. Harris, S. K. Das and J. C. Wyatt. Safety and decision support in oncology. *Methods of Information in Medicine*, 33:371–381, 1994.

[10] S. A. Kripke. Semantical analysis of modal logic I: normal modal propositional calculi. *Zeitschr. f. math. Logik und Grundlagen d. Math.*, 9:67–96, 1963.

[11] D. Nute. Conditional logic. In D. Gabbay and R. Guenthner, editors, *Extensions of Classical Logic*, volume 2 of *Handbook of Philosophical Logic*, pages 387–440. D. Reidel Publishing Company, 1985.

Rule-Generating Abduction for Recursive Prolog

Kouichi Hirata
Kyushu Institute of Technology, Kawazu 680-4, Iizuka 820, Japan
Phone: +81 948 29 7884, Fax: +81-948-29-7600
E-mail: hirata@ai.kyutech.ac.jp

Keywords: rule-generating abduction, inductive logic programming, generalization

Edited by: Rudi Murn
Received: June 14, 1995 **Revised:** September 9, 1995 **Accepted:** October 31, 1995

The rule-generating abduction is a kind of abduction which generates a rule and proposes a hypothesis from a surprising fact. In general, there may exist infinitely many rules and hypotheses to explain such a surprising fact. Hence, we need to put some restriction on the class of rules. In rule-generating abduction, only one surprising fact is given. Hence, we also need to generalize the concept of a surprising fact. When we deal with such generalizations we must avoid overgeneralization. It should be determined whether or not a generalization is overgeneral by an intended model. However, it is hard to give such an intended model in advance in our rule-generating abduction. In this paper, we introduce a syntactical formulation of generalization, in which it can be determined whether or not a generalization is overgeneral by the forms of atoms and substitutions. On the other hand, by the restriction of rules, it suffices to consider only two types of terms, constants and lists, and two types of substitutions with these two terms. By using the above generalizations and substitutions, we design an algorithm for rule-generating abduction, which generates rules and proposes hypotheses in polynomial time with respect to the length of a surprising fact. The number of rules and hypotheses is at most the number of common terms in a surprising fact. Furthermore, we show that a common term in some argument of a surprising fact also appears in the same argument of the proposed hypothesis by this algorithm.

1 Introduction

C. S. Peirce, who was a philosopher, scientist and logician, asserted that every scientific inquiry consists of three stages, *abduction*, *deduction*, and *induction* (Peirce 1965, Yonemori 1982). According to him, abduction is an inference which begins with an observation of *a surprising fact*, and proposes a hypothesis to explain why the fact arises. Then, an inference schema by abduction is described by the following three steps (Peirce 1965, Yonemori 1982).

1. A surprising fact C is observed.
2. If A were true, then C would be a matter of course.
3. Hence, there is reason to suspect that A is true.

In general, the above inference schema is depicted by a syllogism:

$$\frac{C \quad C \leftarrow A}{A}.$$

In computer science, especially in computational logic and logic programming, many researchers have extensively studied abduction from various viewpoints. Plotkin (1971) has studied abduction together with inductive generalization. It is considered that Shapiro's model inference system (Shapiro 1981) and inductive logic programming (Ling 1989, Ling & Ungar 1989, Muggleton 1992) are the extensions of Plotkin's work. Duval (1991) and Genest *et al.* (1990) have suggested abduction for *explanation-based generalization*, which is an efficient technique for obtaining a general concept from examples and a background knowledge. Thagard (1988) has introduced *analogical abduction*, which is a kind of abduction incorporating with analogical reasoning. They are also related to machine learning and knowledge acquisition.

On the other hand, Pople has given another direction for the researches of abduction (Inoue 1992, Kunifuji 1987). It is considered that Poole's Theorist (Poole 1988, Inoue 1992, Kunifuji 1987), Kunifuji's hypothesis-based reasoning (Kunifuji 1987), and abductive logic programming (Dung 1991, Eshghi & Kowalski 1989, Kakas & Mancarella 1990, Kakas *et al.* 1992) are the extensions of Pople's approach. They are also related to knowledge representation.

In order to systematically understand these various researches of abduction, we have classified abduction into five types by the interpretation of syllogism (Hirata 1993); *rule-selecting abduction*, *rule-finding abduction*, *rule-generating abduction*, *theory-selecting abduction*, and *theory-generating abduction*. They are characterized as the following inference schema. More detail would be found in (Hirata 1993).

rule-selecting abduction :

$$\begin{array}{l} C\text{: } \textit{surprising fact} \text{ wrt } P \\ \textit{Select} \text{ a rule } C \leftarrow A \text{ in } P \\ \hline \textit{Propose} \text{ a hypothesis } A \text{ in } P \end{array}$$

rule-finding abduction :

$$\begin{array}{l} C\text{: } \textit{surprising fact} \text{ wrt } P \\ \textit{Find} \text{ a rule } C \leftarrow A \text{ in } P' \ (\neq P) \\ \hline \textit{Propose} \text{ a hypothesis } A \text{ in } P \end{array}$$

rule-generating abduction :

$$\begin{array}{l} C\text{: } \textit{surprising fact} \text{ wrt } P \\ \hline \textit{Generate} \text{ a rule } C \leftarrow A \text{ in } P \\ \textit{Propose} \text{ a hypothesis } A \text{ in } P \end{array}$$

theory-selecting abduction :

$$\begin{array}{l} C\text{: } \textit{surprising fact} \text{ wrt } B \\ \textit{Select} \text{ a theory } A \text{ such that } A \text{ makes } C \text{ true} \\ \hline \textit{Propose} \text{ a hypothesis } A \end{array}$$

theory-generating abduction :

$$\begin{array}{l} C\text{: } \textit{surprising fact} \text{ wrt } B \\ \hline \textit{Generate} \text{ a theory } A \text{ such that } A \text{ makes } C \text{ true} \\ \textit{Propose} \text{ a hypothesis } A \end{array}$$

By this classification, the above researches are placed in the following positions (Hirata 1993): Abductive logic programming (Dung 1991, Eshghi & Kowalski 1989, Kakas & Mancarella 1990, Kakas *et al.* 1992) is a sort of rule-selecting abduction. Abduction for explanation-based generalization (Duval 1991, Genest *et al.* 1990) and analogical abduction (Thagard 1988) are a sort of rule-finding abduction. The constructive operators such as V and W operators (Muggleton 1992) in inductive logic programming are a sort of rule-generating abduction. Poole's Theorist (Poole 1988) and hypothesis-based reasoning (Kunifuji 1987) are a sort of theory-selecting abduction. Shapiro's model inference system (Shapiro 1981) and inductive logic programming (Ling 1989, Ling & Ungar 1989, Muggleton 1992) are a sort of theory-generating abduction.

In this paper, we investigate the *rule-generating abduction*. In the inference schema of rule-generating abduction, only one surprising fact is given. Hence, we need to generalize the concept of a surprising fact.

A *generalization* is an important tool for inductive logic programming and program synthesis. Plotkin (1970, 1971) has introduced and developed the *least generalization* and the *relative least generalization*. Arimura *et al.* (1991) have developed Plotkin's least generalization as the *minimal multiple generalization*. Note that all of them are researches on the generalization of at least two atoms. Thus, the following problem arises: Is the generalization of one atom worthwhile or not? Hirowatari and Arikawa (1994) have introduced the *partially isomorphic generalization* and answered this problem affirmatively in the framework of analogical reasoning.

When we deal with generalizations, we should avoid overgeneralization. It should be determined whether or not a generalization is overgeneral by an intended model. However, it is hard to give such an intended model in advance. Hence, we introduce a syntactical generalization of one atom, called a *safe generalization*. In general, an atom is regarded as a relation between its arguments. Then, for safe generalizations, common ground terms are replaced by common variables.

In rule-generating abduction, if the class of definite programs is not restricted to some subclass, there may be infinitely many meaningless hypotheses. Hence, we introduce the subclass of definite programs, called *single recursive programs on lists*. Many typical Prolog programs are included in this class. However, without any heuristic, the number of single recursive rules on lists also increases exponentially with respect to the length of the surprising fact. In order to obtain the hypotheses efficiently by using safe generalizations, we design an algorithm for rule-generating abduction in single recursive programs on lists.

This paper is organized as follows: In Section 2, we introduce the concepts of *single recursive rules on lists* and *partially isomorphic generalization* necessary for the following sections. In Section 3, we formulate a *safe generalization*, which is based on the forms of atoms and substitutions instead of an intended model, and show some properties of safe generalizations. In Section 4, we show that the number of hypotheses in single recursive rules on lists also increases exponentially with respect to the length of the surprising fact. In Section 5, by using the above generalizations and substitutions, we design an algorithm for rule-generating abduction, which generates rules and proposes hypotheses in polynomial time with respect to the length of the surprising fact. The number of rules and hypotheses is at most the number of common terms in a surprising fact. Furthermore, we show that a common term in some argument of a surprising fact also appears in the same argument of the proposed hypothesis by this algorithm. In Section 6, we discuss the several examples for this algorithm.

2 Preliminary

For *lists*, a constant symbol is necessary to terminate recursion. This "empty list" is denoted by [ ]. We also need a functor [|] of arity 2, called a *list constructor*. Note that $[W|X]$ is a list adding W to the top of a list X. Historically, a list $[W|X]$ is formulated by a dotted pair $.(W, X)$. In this paper, we assume that a list constructor [|] and an empty list [ ] are included in first order language $\mathcal{L}$.

For a term t, $|t|$ denotes the length of t. In particular, for a list l, the length $|l|$ of l is defined as follows: $|l| = 0$, if l is an empty list [ ]. Otherwise $|l| = n + 1$, if t is a list $[a|list]$ and $|list| = n$.

First, we introduce the subclass of definite programs in order to realize rule-generating abduction.

Definition 1 *Let P be the following definite program:*

$$P = \left\{ \begin{array}{l} p(t_1, \cdots, t_n) \leftarrow p(X_1, \cdots, X_n) \\ p(s_1, \cdots, s_n) \end{array} \right\}.$$

Then, P is called single recursive on lists *if P satisfies the following conditions:*

1. *for any i $(1 \leq i \leq n)$, t_i is of the form either $[W_i|X_i]$ or X_i, where W_i is a variable,*
2. *for at least one j, t_j is of the form $[W_j|X_j]$,*
3. *for any i and j, $(1 \leq i, j \leq n, i \neq j)$, X_i and X_j are mutually distinct,*
4. *for any i, s_i is of one of the forms [ ], X, $[Y]$ or $[W|Z]$.*

A rule $p(t_1, \cdots, t_n) \leftarrow p(X_1, \cdots, X_n)$ is called *single recursive on lists* if it satisfies the above conditions 1, 2, and 3.

Example 1 *The following Prolog rules in Sterling and Shapiro (1986) are single recursive rules on lists:*

member$(X, [W|Y]) \leftarrow$ member(X, Y),
prefix$([W|X], [W|Y]) \leftarrow$ prefix(X, Y),
suffix$(X, [W|Y]) \leftarrow$ suffix(X, Y),
append$([W|X], Y, [W|Z]) \leftarrow$ append(X, Y, Z),
concat$(X, [W|Y], [W|Z]) \leftarrow$ concat(X, Y, Z).

Note that the rules of member *and* suffix *have the same form. The first argument of* member *is a constant symbol, while that of* suffix *is a list.*

Let α, β, and γ be atoms. Then, β is said to be *more general than* α, or a *generalization* of α if there exists a substitution θ such that $\beta\theta = \alpha$. Furthermore, β is called a *maximal* generalization if there exists no atom γ such that $\gamma\theta = \beta$ for any substitution θ. Also β is called the *greatest* generalization if there exists a substitution θ such that $\beta\theta = \gamma$ for any atom γ with the predicate symbol of β.

Hirowatari and Arikawa (1994) have introduced the concept of a *partially isomorphic generalization*, which is the special generalization of one atom and is a useful tool for analogical reasoning. Let α be an atom. A term t is a *replaceable term* of α if t is a constant symbol or a term $f(X_1, \cdots, X_n)$, where f is a function symbol and each X_i is a variable which does not appear in the other terms in α. For a replaceable term t of α, let $\alpha[t]$ be an atom obtained by replacing each t in α by a new variable Z which does not appear in α. Then we write $\beta \rightarrow \alpha$ when $\alpha[t]$ is a variant of β. We define $\rightarrow^*$ as the reflexive and transitive closure of $\rightarrow$.

Definition 2 (Hirowatari & Arikawa 1994) *Let α and β be atoms. Then, β is a* partially isomorphic generalization *of α if $\beta \rightarrow^* \alpha$.*

For a set of atoms S, let $[S]$ be the set of all equivalence classes of atoms in S. In particular, for any $\alpha \in [S]$ and $\beta \in [S]$, α is a variant of β.

Hirowatari and Arikawa (1994) have shown the following two theorems.

Theorem 1 (Hirowatari & Arikawa 1994) *Let α be an atom and S be the set of all partially isomorphic generalizations of α. Then, $[S]$ is a lattice whose partial order is $\rightarrow^*$, join operator is the greatest instantiation, and meet operator is the least generalization.*

Theorem 2 (Hirowatari & Arikawa 1994) *Let α be a ground atom $p(t_1, \cdots, t_n)$ and $k = |t_1| + \cdots + |t_n|$. Then, a partially isomorphic generalization of α can be computed in $O(k^2)$ time.*

3 Safe Generalization

It is an important problem to avoid overgeneralization when we deal with generalizations. In general, whether or not a generalization β of α is overgeneral is determined by an intended model. For an atom α, suppose that $\beta\theta = \alpha$ and M is an intended model. Then, β is an *overgeneralization* of α if there exists a ground atom γ such that $\forall\beta \vdash \gamma$ and $M \not\models \gamma$ for M. However, the decision problem of whether or not there exists such ground atom γ is undecidable. In rule-generating abduction, only one surprising fact is given, and it is hard to give in advance an intended model. To overcome these difficulties, we introduce a syntactical generalization of one atom.

Let θ be a ground substitution, that is, $\theta = \cup_{i=1}^{n}\{X_i := t_i\}$ and every term t_i is ground. Let α be a ground atom and β be an atom such that $\beta\theta = \alpha$. Note that, throughout this paper, if $\beta\theta = \alpha$, then a variable $X_i \in dom(\theta)$ appears in β and $t_i \neq [\ ]$. A substitution θ is *well-defined* if, for any t_i, there exists no term t_j which is a subterm of t_i.

Example 2 *Let α be a ground atom $p([a, b], [b])$. Let β_i be the following atoms* $(1 \leq i \leq 5)$*:*

$$\beta_1 = p([a, X], [b]),\ \beta_2 = p([a|X], [b]),$$
$$\beta_3 = p([X|Y], Y),\ \beta_4 = p([a|X], [Y]),$$
$$\beta_5 = p([a|X], Y).$$

For any β_i, there exist the following substitutions θ_i such that $\beta_i\theta_i = \alpha$ $(1 \leq i \leq 5)$:

$$\theta_1 = \{X := b\},\ \theta_2 = \{X := [b]\},$$
$$\theta_3 = \{X := a, Y := [b]\},$$
$$\theta_4 = \{X := [b], Y := b\},$$
$$\theta_5 = \{X := [b], Y := [b]\}.$$

Then, θ_1, θ_2, and θ_3 are well-defined, while θ_4 and θ_5 are not.

Let α be a ground atom and β be an atom such that $\beta\theta = \alpha$. If a substitution $\theta = \cup_{i=1}^{n}\{X_i := t_i\}$ is well-defined, then we can define a *reversal* $\theta^{-1} = \cup_{i=1}^{n}\{t_i := X_i\}$. Note that, if θ is well-defined, then, for any t_i and X_i, there exists no term t_j such that t_j is a subterm of t_i and no variable X_i such that $X_i = X_j (j \neq i)$. However, even if θ is well-defined, β is not always $\alpha\theta^{-1}$.

Example 3 *For β_1 and β_2 in Example 2,*

$$\begin{aligned} \alpha\theta_1^{-1} &= p([a, b], [b])\{b := X\} \\ &= p([a, X], [X]) \neq \beta_1, \\ \alpha\theta_2^{-1} &= p([a, b], [b])\{[b] := X\} \\ &= p([a|X], X) \neq \beta_2. \end{aligned}$$

On the other hand,

$$\begin{aligned} \alpha\theta_3^{-1} &= p([a, b], [b])\{a := X, [b] := Y\} \\ &= p([X|Y], Y) = \beta_3. \end{aligned}$$

For the reversal θ^{-1}, the following lemma holds.

Lemma 1 *Let α be a ground atom and β be an atom such that $\beta\theta = \alpha$. Suppose that a substitution $\theta = \cup_{i=1}^{n}\{X_i := t_i\}$ is well-defined. Then, $\beta = \alpha\theta^{-1}$ if and only if no term t_i appears in β.*

Proof Suppose that $\beta = \alpha\theta^{-1}$. By the definition of θ^{-1}, $\alpha\theta^{-1}$ is an atom which replaces all the terms t_i in α with the variable X_i. Then, no term t_i appears in $\alpha\theta^{-1} = \beta$.

For simplicity, suppose that $\theta = \{X := t\}$. If a term t appears once in α, that is, α has the form of $p(\cdots t \cdots)$, then $\alpha\theta^{-1} = p(\cdots t \cdots)\{t := X\} = p(\cdots X \cdots)$. Since α is ground and $\beta\theta = \alpha$, $\alpha\theta^{-1} = \beta$.

If a term t appears at least twice in α, that is, α has the form of $p(\cdots t \cdots t \cdots)$, then β has the form of $p(\cdots X \cdots Y \cdots)$. If $X \neq Y$, then, by $\beta\theta = \alpha$, θ has the form of $\{X := t, Y := t\}$. This θ is not well-defined. Hence, it is a contradiction. Then, $X = Y$, $\theta = \{X := t\}$, and β has the form of $p(\cdots X \cdots X \cdots)$. Hence, $\beta = \alpha\theta^{-1}$. □

A ground term t ($\neq$ [ ]) is a *common term* in α if t appears at least twice in α. In particular, if a common term is a ground list, it is called a *common list*. Then, we formulate a *safe generalization*.

Definition 3 *Let α be a ground atom, θ be a substitution $\cup_{i=1}^{n}\{X_i := t_i\}$, and β be an atom such that $\beta\theta = \alpha$. An atom β is a* safe generalization *of α if (β, θ) satisfies the following* safeness conditions*:*

1. *θ is well-defined,*
2. *$\beta = \alpha\theta^{-1}$, and*
3. *if there exists a common term in α, then there exists a ground term $t_j \in \cup_{i=1}^{n}\{t_i\}$ such that t_j is a common term in α.*

Let α be a ground atom and β be an atom such that $\beta\theta = \alpha$. Let t be some common term in α. If θ is well-defined and θ^{-1} has the form of $\{t := X\}$, then β is safe on α.

Example 4 *Let α be a ground atom $p([a, b], [b])$ and β_i be an atom such that $\beta_i\theta_i = \alpha$ ($6 \leq i \leq 8$). Then, the common terms in α are $[b]$ and b.*

1. *Let β_6 be an atom $p(X, Y)$ and θ_6 be a substitution $\{X := [a, b], Y := [b]\}$. By the safeness condition 1, β_6 is not safe on α.*
2. *Let β_7 be an atom $p([X, b], [Y])$ and θ_7 be a substitution $\{X := a, Y := b\}$. By the safeness condition 2, β_7 is not safe on α.*
3. *Let β_8 be an atom $p(X, [b])$ and θ_8 be a substitution $\{X := [a, b]\}$. By the safeness condition 3, β_8 is not safe on α.*

For the above α, atoms $p([a|X], X)$, $p([a, X], [X])$, $p([Y, X], [X])$, and $p([Y|X], X)$ are safe on α.

In general, an atom is regarded as a relation between its arguments. Thus, the syntactical generalization of one atom should be obtained by replacing common terms with common variables. The safe generalization is an example of such generalizations. On the other hand, in single recursive rules on lists it suffices to consider only two types of terms, constant symbols and lists. Then, we also define the following two types of substitutions.

Let θ be a substitution $\cup_{i=1}^{n}\{X_i := t_i\}$. Then, θ is a *constant substitution* (*resp.*, a *list substitution*) if every t_i is a constant symbol (*resp.*, a ground list) without an empty list [ ].

In particular, a constant substitution is related to *partially isomorphic generalizations* which have been introduced by Hirowatari and Arikawa (1994). Note that, though a replaceable term includes an empty list [ ], the definition of a replaceable term is independent of the proof of Theorem 1. Let RT be the set of all replaceable terms of α and $T \subseteq RT$. Then, we can reformulate a partially isomorphic generalization by using T instead of RT, and show that Theorem 1 also holds for the set T of replaceable terms. Let α be a ground atom, θ_c be a constant substitution, and β be an atom such that $\beta\theta_c = \alpha$. Thus, we assume that *β is a partially isomorphic generalization of α, whose replaceable terms are all constant symbols in α except an empty list* [ ].

In Section 5, we apply rule-generating abduction to a list substitution θ_l and a constant substitution θ_c in the following way: Let α be a ground atom, that is, a surprising fact. First, by using a list substitution, we obtain an atom β such that $\beta\theta_l = \alpha$ and β is safe on α. Secondly, by using a constant substitution, we obtain an atom γ such that $\gamma\theta_c = \beta$ and γ is safe on β. By the above assumption, γ is also a partially isomorphic generalization of β.

Unfortunately, $\theta_c\theta_l$ is not always well-defined, and γ is not always safe on α. For example, let α be a ground atom $p([a,b,c],[b,c],[a,b,c])$. Then, there exist the following atoms β_i such that $\beta_i\theta_i = \alpha$ and θ_i is a well-defined list substitution ($9 \leq i \leq 11$):

$$\begin{aligned}
\beta_9 &= p([a,b|X],[b|X],[a,b|X]),\\
\theta_9 &= \{X := [c]\};\\
\beta_{10} &= p([a|Y],Y,[a|Y]),\\
\theta_{10} &= \{Y := [b,c]\};\\
\beta_{11} &= p(Z,[b,c],Z),\\
\theta_{11} &= \{Z := [a,b,c]\}.
\end{aligned}$$

Then, there exist no generalization γ of β and substitution $\sigma(\neq \varepsilon)$ such that $\gamma\sigma = \beta_{11}$ and $\theta_{11}\sigma$ is well-defined. Note that there does not exist the greatest list generalization of α.

Let α be a ground atom. Let β and γ be atoms such that $\beta\theta_l = \alpha$ and $\gamma\theta_c = \beta$. Suppose that both (β,θ_l) and (γ,θ_c) satisfy the safeness conditions. Then, the following two theorems hold.

Theorem 3 *If $\theta_c\theta_l$ is well-defined, then γ is a safe generalization of α.*

Proof Suppose that $\theta_c\theta_l$ is well-defined. Then, $(\gamma,\theta_c\theta_l)$ satisfies the safeness condition 1.

Since both (β,θ_l) and (γ,θ_c) satisfy the safeness conditions, $(\gamma,\theta_c\theta_l)$ satisfies the safeness condition 3.

By the supposition, $\beta = \alpha\theta_l^{-1}$ and $\gamma = \beta\theta_c^{-1}$. The list substitution θ_l replaces the common lists in α by variables. The constant substitution θ_c replaces the same constant symbols in β by the same variables and other constant symbols by other variables. Hence, the composition $\theta_c\theta_l$ replaces the common lists in α by variables, the same constant symbols in α except common lists by the same variables, and other constant symbols by other variables. By Lemma 1 and since $\theta_c\theta_l$ is well-defined, then $(\gamma,\theta_c\theta_l)$ satisfies the safeness condition 2. □

Theorem 4 *Suppose that any constant symbol appearing in common lists in α does not appear elsewhere in α except in the lists. If $\beta = \alpha\theta_l^{-1}$ and $\gamma = \beta\theta_c^{-1}$, then $\theta_c\theta_l$ is well-defined. Hence, γ is a safe generalization of α.*

Proof Suppose that $\theta_l = \cup_{i=1}^{n}\{X_i := l_i\}$, where l_i is a common list in α. For any j-th argument's term t_j of α, if t_j includes l_i, then $t_j = [a_1^j, a_2^j, \cdots, a_{n_j}^j|l_i]$, and no constant symbols $a_1^j, a_2^j, \cdots, a_{n_j}^j$ appear in l_i. Then, θ_c does not include the binding $X := c$ such that c appears in l_i. Hence, $\theta_c\theta_l$ is well-defined. By Theorem 3, $(\gamma,\theta_c\theta_l)$ satisfies the safeness conditions. Then, for γ such that $\gamma\theta_c\theta_l = \alpha$, γ is a safe generalization of α. □

Hence, we can conclude that the condition of Theorem 4 is regarded as one of a *good* surprising fact for the discussion in Section 5. In other words, we consider that the atoms which does not satisfy the condition, such as $p([a],[a],[a,a])$ and $p([a,b,a],[b,a])$, have no enough information to determine which constant symbols or common lists should be generalized.

4 Number of Hypotheses

An inference schema of rule-generating abduction is described by the following three steps:

1. A surprising fact C is observed.
2. A rule $C \leftarrow A$ is generated in P.
3. A hypothesis A is proposed.

In this paper, we deal with the most simple rule-generating abduction such that $P = \phi$. Suppose that a ground atom $p(t_1,\cdots,t_n)$ is given as a surprising fact. Then, by rule-generating abduction, the following single recursive rule on lists

$$p(t_1',\cdots,t_n') \leftarrow p(s_1',\cdots,s_n')$$

is generated and the hypothesis $p(s_1,\cdots,s_n)$ is proposed. Here,

$$\begin{aligned}
p(t_1',\cdots,t_n')\theta &= p(t_1,\cdots,t_n),\\
p(s_1',\cdots,s_n')\theta &= p(s_1,\cdots,s_n).
\end{aligned}$$

The above inference schema is depicted by the following syllogism:

$$\frac{p(t_1,\cdots,t_n)}{\begin{array}{r}p(t_1',\cdots,t_n') \leftarrow p(s_1',\cdots,s_n')\\ p(s_1,\cdots,s_n)\end{array}}$$

In definite programs, if the class of programs is not restricted to some subclass, then there are also infinitely many meaningless hypotheses.

In this section, we investigate the number of rules generated by rule-generating abduction. Unfortunately, the following Theorem 5 claims that, even if the generated rule is single recursive on lists then the number of hypotheses increases in exponential order with respect to the length of an atom.

Theorem 5 *Let $p(t_1, \cdots, t_n)$ be a surprising fact. Then, the number of single recursive rules on lists which satisfies the above syllogism is at most 3^n.*

Proof See Appendix. □

By using safe generalizations from Section 3, we design an algorithm to generate single recursive rules on lists as follows: Suppose that a ground atom α is given. First, by generalizing α with a list substitution θ_l, we obtain an atom β such that $\beta\theta_l = \alpha$ and β is safe on α. We call such a generalization β a *list generalization* of α. Secondly, by generalizing β with a constant substitution θ_c, we obtain an atom γ such that $\gamma\theta_c = \beta$ and γ is safe on β. We call such a generalization γ a *constant generalization* of β. Note that γ is also assumed to be a *partially isomorphic generalization* of β. For this algorithm, the number of rules is at most the number of generalizations. Then we investigate the number of generalizations, in particular, the number of maximal generalizations.

Let α be a ground atom $p(t_1, \cdots, t_n)$. For all common lists in α, we can classify them by the *sublist relation*. For example, let α be the following ground atom:

$$p([a,b,c,d],[c,d],[b,c],[c],[b,c,d]) \\ (= p(t_1,t_2,t_3,t_4,t_5)),$$

and t_i be the i-th argument's term of α. Then, t_2, t_4, and t_5 are common lists in α. By the sublist relation, we classify them into $\{t_2, t_5\}$ and $\{t_4\}$.

The number of maximal generalizations is characterized as follows.

Theorem 6 *Let α be a ground atom $p(t_1, \cdots, t_n)$ and l be the number of classes in t_i by the sublist relation. Then, the number of the maximal generalizations is at most*

$$\left(\frac{\left(\sqrt{2}\right)^n}{l}\right)^l.$$

Even when $l = 1$, the number of the maximal list generalizations of α is at most $\left(\sqrt{2}\right)^n$.

Proof See Appendix. □

By Theorem 6, the number of single recursive rules on lists, even if we adopt the maximal list generalizations, increases exponentially with respect to n. Hence, in the next section, we restrict the forms of generalizations, and design the algorithm for rule-generating abduction whose number of hypotheses is at most n.

5 Algorithm PROPOSE

In this section, we design an efficient algorithm for rule-generating abduction, which is called *PROPOSE*. In the algorithm *PROPOSE* illustrated in Figure 1, we restrict the reversal for list generalizations to the form of $\{t := X\}$, where t is both a common list in α and some argument's term of α. Obviously, the list generalization $\alpha\{X := t\}$ is safe on α. Then, the following lemma holds.

Lemma 2 *The number of single recursive rules on lists generated by the algorithm* PROPOSE *is at most n.*

Proof The number of terms that are both a common term and some argument's term is at most n. Then, the number of elements of L is at most n. Hence, the number of hypotheses is also at most n. □

An important feature of the algorithm *PROPOSE* is that, *if the i-th argument's term is some common list in α, then the i-th argument's term of the head of the generated rule is a variable; otherwise, it is a list.* Furthermore, by the algorithm *PROPOSE*, the rules in Example 1 are constructed from one ground atom.

In Figure 1, $rs\_abd(fact, head \leftarrow body, hyp)$, which is rule-selecting abduction, is a procedure to propose a hypothesis hyp such that $(head)\sigma = fact$ and $(body)\sigma = hyp$ for some substitution σ.

For the algorithm *PROPOSE*, the following two theorems hold.

Theorem 7 *Let α be a ground atom $p(t_1, \cdots, t_n)$ and $k = |t_1| + \cdots + |t_n|$. Then, the algorithm* PROPOSE *computes the rules and hypotheses in $O(k^3)$ time.*

```
Algorithm $PROPOSE$
input $\alpha = p(t_1, \cdots, t_n)$ : a ground atom
output $head \leftarrow body$ : a rule
        $\delta$ : a hypothesis

$L := \{\beta \mid \beta = \alpha\{t_i := V_i\}, t_i$: common list in $\alpha\}$
        $\cup\{\alpha\}$; /* $\beta$ : safe on $\alpha$ */
while $L \neq \phi$ do
    select $\beta \in L$;
    $\gamma :=$ the greatest constant generalization
            $p(s_1, \cdots, s_n)$ of $\beta$;
    for $i = 1$ to $n$
        if $s_i = [\,]$ then
            output $\gamma \leftarrow true$
            output $true$
            halt;
        else if $s_i$ is a variable then
            $head\_arg_i := X_i$;
            /* $X_i$ is a new variable */
        else /* $s_i = [W_1^i, \cdots]$ */
            $head\_arg_i := [W_1^i | X_i]$;
            /* $X_i$ is a new variable */
        end
    end
    $head := p(head\_arg_1, \cdots, head\_arg_n)$;
    /* $head\_arg_i = [W_1^i | X_i]$ or $X_i$ */
    $body := p(X_1, \cdots, X_n)$;
    output $head \leftarrow body$
    $rs\_abd(\alpha, head \leftarrow body, \delta)$;
    output $\delta$
    $L := L - \{\beta\}$;
end
```

Figure 1: Algorithm *PROPOSE*

Proof For any t_i, it can be determined whether or not t_i is a sublist of t_j in $O(|t_i|)$. Then, for any i, it can be determined whether or not t_i is a sublist of any $t_j (j \neq i)$ in $O((n-1)|t_i|)$. Hence, the set L in the algorithm *PROPOSE* can be constructed in $O((n-1)k)$.

For the selected β in L, the greatest constant generalization of β is also a partially isomorphic generalization of β. By Theorem 2, a partially isomorphic generalization γ of β can be computed in $O(k^2)$. Since the procedures in the for-loop can be computed in $O(n)$, the for-loop terminates in $O(n^2)$. Then, the procedures in while-loop can be computed in $O(k^2 + n^2)$. Since the number of elements in L is at most n by Lemma 2, the while-loop terminates in $O(k^2 n + n^3)$. Hence, the algorithm *PROPOSE* terminates in $O((n-1)k + k^2 n + n^3)$.

Since $n \leq k$, the algorithm *PROPOSE* computes rules and hypotheses in $O(k^3)$ time. □

Theorem 8 *Let α be a ground atom $p(t_1, \cdots, t_n)$ and δ be the proposed hypothesis $p(s_1, \cdots, s_n)$ by* PROPOSE. *If there exists a selected common list l in α by the algorithm* PROPOSE *and l appears in t_i, then l also appears in s_i.*

Proof Let l be the selected common list in α by *PROPOSE*. If the i-th argument's term t_i of α is l itself, then the i-th argument's terms of both the head and the body of the generated rules are variables X_i. Then, the i-th argument's term s_i of δ is also l.

Suppose that l appears in another argument's term of α, and t_i has the form of $[a_1^i, a_2^i, \cdots, a_{n_i}^i | l]$. By the algorithm *PROPOSE*, the i-th argument's term of the head of the generated rule is a list $[Y_1^i | X_i]$, where Y_1^i is a variable corresponding to a_1^i, while one of the body is a variable X_i. Then, for the hypothesis δ, the i-th argument's term s_i of δ has the form of $[a_2^i, \cdots, a_{n_i}^i | l]$.

Hence, the selected common list l also appears in the i-th argument's term s_i of δ. □

Theorem 8 claims that, if a given ground atom satisfies the relation on common lists, then the proposed hypothesis by the algorithm *PROPOSE* also satisfies it.

6 Examples

In this section, we discuss the several examples for the algorithm *PROPOSE*.

Example 5 *Let α be a ground atom $p([a,b],[c,d],[a,b,c,d])$. The list $[c,d]$ is both a common list in α and the second argument's term of α. By the construction of L, $\beta = p([a,b],V_2,[a,b|V_2])$ is a* safe *list generalization of α and $\gamma = p([X,Y],V_2,[X,Y|V_2])$ is the greatest constant generalization of β. The first argument of γ is a list which begins with X. The second argument is a variable V_2, and the third argument is also a list which begins with X. By the for-loop in* PROPOSE, *we obtain a head $p([X|X_1],X_2,[X|X_3])$ and a body $p(X_1,X_2,X_3)$. Hence,* PROPOSE *generates a rule*

$$p([X|X_1],X_2,[X|X_3]) \leftarrow p(X_1,X_2,X_3),$$

and proposes a hypothesis $p([b],[c,d],[b,c,d])$. The predicate p means the append *in Example 1.*

Since L includes α, then $\beta = \alpha$, and $\gamma = p([X,Y],[Z,W],[X,Y,Z,W])$. The first argument of γ is a list which begins with X, the second argument is a list which begins with Z, and the third argument is also a list which begins with X. By the for-loop in PROPOSE, *we obtain a head $p([X|X_1],[Z|X_2],[X|X_3])$ and a body $p(X_1,X_2,X_3)$. Hence,* PROPOSE *generates a rule*

$$p([X|X_1],[Z|X_2],[X|X_3]) \leftarrow p(X_1,X_2,X_3),$$

and proposes a hypothesis $p([b],[d],[b,c,d])$. The predicate p means that all arguments' terms are lists and the first argument's term is the prefix of the third argument's one.

Furthermore, each of the rules and hypotheses by *PROPOSE* respectively satisfies the following syllogisms:

$$\frac{p([a,b],[c,d],[a,b,c,d])}{p([X|X_1],X_2,[X|X_3]) \leftarrow p(X_1,X_2,X_3)} \quad p([b],[c,d],[b,c,d])$$

$$\frac{p([a,b],[c,d],[a,b,c,d])}{p([X|X_1],[Z|X_2],[X|X_3]) \leftarrow p(X_1,X_2,X_3)} \quad p([b],[d],[b,c,d])$$

Example 6 *Let α be a ground atom $p(a,[a,b])$. Since there exist no common lists in α, then $\beta = p(a,[a,b])$, and $\gamma = p(X,[X,Y])$. The first argument of γ is a variable X, and the second argument is a list which begins with X. By the for-loop in* PROPOSE, *we obtain a head $p(X_1,[X|X_2])$ and a body $p(X_1,X_2)$. Hence,* PROPOSE *generates a rule*

$$p(X_1,[X|X_2]) \leftarrow p(X_1,X_2),$$

and proposes a hypothesis $p(a,[b])$. Note that the predicate p means the member *in Example 1.*

Let α be a ground atom $p([a],[a,b])$. Since there exist no common lists in α, $\beta = p([a],[a,b])$ and $\gamma = p([X],[X,Y])$. The first argument of γ is a list which begins with X, and the second argument is also a list which begins with X. By the for-loop in PROPOSE, *we obtain a head $p([X|X_1],[X|X_2])$ and a body $p(X_1,X_2)$. Hence,* PROPOSE *generates a rule*

$$p([X|X_1],[X|X_2]) \leftarrow p(X_1,X_2),$$

and proposes a hypothesis $p([\,],[b])$. Note that the predicate p means the prefix *in Example 1.*

Let α be a ground atom $p([b],[a,b])$. The list $[b]$ is both a common list in α and the first argument's term of α. Then, $\beta = p(V_1,[a|V_1])$ and $\gamma = p(V_1,[X|V_1])$. The first argument of γ is a variable V_1, and the second argument is a list which begins with X. By the for-loop in PROPOSE, *we obtain a head $p(X_1,[X|X_2])$ and a body $p(X_1,X_2)$. Hence,* PROPOSE *generates a rule*

$$p(X_1,[X|X_2]) \leftarrow p(X_1,X_2),$$

and proposes a hypothesis $p([b],[b])$. Note that the predicate p means the suffix *in Example 1. On the other hand, since L includes $\alpha = p([b],[a,b])$, then $\beta = \alpha$, and $\gamma = p([X],[Y,X])$. The first argument of γ is a list which begins with X, and the second argument is a list which begins with Y. By the for-loop in* PROPOSE, *we obtain a head $p([X|X_1],[Y|X_2])$ and a body $p(X_1,X_2)$. Hence,* PROPOSE *generates a rule*

$$p([X|X_1],[Y|X_2]) \leftarrow p(X_1,X_2),$$

and proposes a hypothesis $p([\,],[b])$. Note that the predicate p means the defining lists, that is, all arguments' terms are lists.

If $\alpha = p([a,b],[c,d],[a,b,c,d])$, then *PROPOSE* generates a rule

$$p([X|X_1], X_2, [X|X_3]) \leftarrow p(X_1, X_2, X_3),$$

and proposes a hypothesis $p([b],[c,d],[b,c,d])$. If $\alpha = p([a,b],[a,b,c,d],[c,d])$, then *PROPOSE* also generates a rule

$$p([X|X_1], [X|X_2], X_3) \leftarrow p(X_1, X_2, X_3),$$

and proposes a hypothesis $p([b],[b,c,d],[c,d])$. Hence, the algorithm *PROPOSE* is independent of the order of arguments.

Furthermore, by the construction of *member* and *suffix* in Example 6, the algorithm *PROPOSE* is also independent of the types, such as lists and constant symbols, of argument. In other words, *PROPOSE* needs no types of arguments.

We can realize the algorithm *PROPOSE* in a Prolog program. The predicate propose returns a generated rule as the third argument. It also returns a hypothesis proposed by the generated rule as the second argument.

For the following five surprising facts

$$p([a,b],[c,d],[a,b,c,d]),\ p(a,[a,b]),\ p([a],[a,b]),$$
$$p([b],[a,b]),\ p([a,b,c],[b,c],[a,b,c]),$$

the results of propose are as follows:

```
: ?- propose(p([a,b],[c,d],[a,b,c,d]),X,Y).
X = p([b],[d],[b,c,d]),
Y = p([_2|_5],[_1|_4],[_2|_3]):-p(_5,_4,_3);
X = p([b],[c,d],[b,c,d]),
Y = p([_1|_4],_2,[_1|_3]):-p(_4,_2,_3);
no
: ?- propose(p(a,[a,b]),X,Y).
X = p(a,[b]),
Y = p(_2,[_1|_3]):-p(_2,_3);
no
: ?- propose(p([a],[a,b]),X,Y).
X = p([],[b]),
Y = p([_1|_3],[_1|_2]):-p(_3,_2);
no
: ?- propose(p([b],[a,b]),X,Y).
X = p([],[b]),
Y = p([_2|_4],[_1|_3]):-p(_4,_3);
X = p([b],[b]),
Y = p(_2,[_1|_3]):-p(_2,_3);
no
: ?- propose(p([a,b,c],[b,c],[a,b,c]),X,Y).
X = p([b,c],[c],[b,c]),
Y = p([_2|_5],[_1|_4],[_2|_3]):-p(_5,_4,_3);
X = p([b,c],[b,c],[b,c]),
Y = p([_1|_4],_2,[_1|_3]):-p(_4,_2,_3);
X = p([a,b,c],[c],[a,b,c]),
Y = p(_3,[_1|_4],_2):-p(_3,_4,_2);
no
```

In the last example, two atoms $p([a|V_2],V_2,[a|V_2])$ and $p(V_1,[b,c],V_1)$ are the safe list generalizations of $p([a,b,c],[b,c],[a,b,c])$ obtained by the algorithm *PROPOSE*. Hence, for a ground atom $p([a,b,c],[b,c],[a,b,c])$, there are three hypotheses and rules. The first rule obtained by *PROPOSE* means that all arguments' terms are lists, and the first and the third arguments' terms begin with the same constant symbols. The second rule means that the second argument's term is the sublist of the first and the third arguments' terms. The third rule means that the first argument's term is equal to the third argument.

On the other hand, for a ground atom $p([a,b,c],[d,e],[a,b,c])$, the predicate propose returns the following two rules and hypotheses:

```
: ?- propose(p([a,b,c],[d,e],[a,b,c]),X,Y).
X = p([b,c],[e],[b,c]),
Y = p([_2|_5],[_1|_4],[_2|_3]):-p(_5,_4,_3);
X = p([a,b,c],[e],[a,b,c]),
Y = p(_3,[_1|_4],_2):-p(_3,_4,_2);
no
```

From this surprising fact, the rule whose second argument's term is the sublist of the first and the third arguments' terms is not generated by *PROPOSE*.

7 Conclusion

In this paper, we have introduced *single recursive programs on lists* for rule-generating abduction. We have also discussed a *safe generalization*, which is a generalization of one atom whose common terms are replaced by common variables. We have given some properties of safe generalizations. In rule-generating abduction for single recursive programs on lists, we have shown that the number of hypotheses increases exponentially with respect to the length of a surprising fact. On the other hand, by using safe generalizations, we have designed an algorithm called *PROPOSE* to construct single recursive rules on lists. The number of hypotheses by *PROPOSE* is at most the length of a surprising fact. We have shown that the algorithm *PROPOSE* generates rules and proposes hypotheses in polynomial time with respect to the length of a surprising fact. Also we have shown that the selected common list in some argument of a surprising fact appears in the same argument of the hypothesis proposed by *PROPOSE*.

We have left several problems as future work.

Abduction is an inference to propose hypotheses to be used before deduction and induction. We have left as a future work to combine abduction and inductive logic programming. Ling (1989, Ling & Ungar 1989) has introduced the constructive inductive logic programming. There may exist a relationship between this work and this paper.

The results in this paper depend on the simplicity of the hypothesis space for rule-generating abduction. Hence, we need to extend the result of this paper to the wider class, for example, the programs consisting of multiple recursions, including local variables, and dealing with terms except lists. Furthermore, as to the inductive logic programming, we have left the problem of predicate invention. The number of rules and hypotheses becomes exponentially large. Hence, we need to introduce some heuristics such as on a distinction between a necessary and useful intermediate term, the number of local variables, invented predicate symbols and rules, and so on.

All researches on abduction in computer science are based on computational logic, and the abduction beyond the scope of computational logic such as abduction for formal language or numerical data has not yet been studied. On the other hand, abduction begins with a surprising fact. In machine learning, a surprising fact is considered as a good example. Hence, abduction is regarded as *learning from good examples*. We need to study the abduction for various frameworks together with machine learning. This should be one of the most important future tasks.

Acknowledgment

The author would like to thank Prof. Setsuo Arikawa and Dr. Eiju Hirowatari for many insightful comments and precious discussions. He also thanks anonymous AII'94 and Informatica reviewers for valuable comments.

References

[1] Arimura, H., Shinohara, T. & Otsuki, S. (1991) A Polynomial Time Algorithm for Finite Unions of Tree Pattern Languages. *Proceedings of ALT'91*, p105–114.

[2] Dung, P. M. (1991) Negation as Hypothesis: An Abductive Foundation for Logic Programming. *Proceedings of ICLP'91*, p 3–17.

[3] Duval, B. (1991) Abduction for Explanation-Based Learning. *Proceedings of EWSL'91* LNAI 482, p 348–360.

[4] Eshghi, K. & Kowalski, R. A. (1989) Abduction Compared with Negation by Failure. *Proceedings of ICLP'89*, p 234–254.

[5] Genest, J., Matwin, S. & Plante, B. (1990) Explanation-Based Learning with Incomplete Theories: A Three-Step Approach. *Proceedings of ML'90*, p 286–294.

[6] Hirata, K. (1993) A Classification of Abduction: Abduction for Logic Programming. *Machine Intelligence* 14 (to appear).

[7] Hirowatari, E. & Arikawa, S. (1994) Partially Isomorphic Generalization and Analogical Reasoning. *Proceedings of ECML'94*, LNAI 784, p 363–366.

[8] Inoue, K. (1992) Principles of Abduction. *Journal of Japanese Society for Artificial Intelligence* 7, p 48–59 (in Japanese).

[9] Kakas, A. C. & Mancarella, P. (1990) Generalized Stable Models: A Semantics for Abduction. *Proceedings of ECAI'90*, p 385–391.

[10] Kakas, A. C., Kowalski, R. A. & Toni, F. (1992) Abductive logic programming. *Journal of Logic and Computation* 2, p 719–770.

[11] Kunifuji, S. (1987) Hypothesis-Based Reasoning. *Journal of Japanese Society for Artificial Intelligence* 2, p 22–87 (in Japanese).

[12] Ling, X. (1989) Learning and Invention of Horn Clause Theories - A Constructive Method. *Methodologies for Intelligent Systems* 4, p 323–331.

[13] Ling, X. & Ungar, L. (1989) Inventing Theoretical Terms in Inductive Learning of Functions - Search and Constructive Methods. *Methodologies for Intelligent Systems* 4 p 332–341.

[14] Muggleton, S. (ed.) (1992) *Inductive Logic Programming*. Academic Press.

[15] Peirce, C. S. (1965) *Collected papers of Charles Sanders Peirce (1839-1914)*. Hartshone, C. S., Weiss, P.(eds.), The Belknap Press.

[16] Plotkin, G. D. (1970) A Note on Inductive Generalization. *Machine Intelligence* 5, p 153–163.

[17] Plotkin, G. D. (1971) A Further Note on Inductive Generalization. *Machine Intelligence* 6, p 101–124.

[18] Poole, D. (1988) A Logical Framework for Default Reasoning. *Artificial Intelligence* 36, p 27–47.

[19] Shapiro, E. Y. (1981) Inductive Inference of Theories from Facts. Research Report 192, Yale University.

[20] Sterling, L. & Shapiro, E. (1986) *The Art of Prolog*. The MIT Press.

[21] Thagard, P. (1988) *Computational philosophy of science*. The MIT Press.

[22] Yonemori, Y. (1982) *Peirce's Semiotics*. Keisou Syobou (in Japanese).

Appendix

Proof of Theorem 5

Suppose that the generated rule is of the form $p(u_1,\cdots,u_n) \leftarrow p(v_1,\cdots,v_n)$ and the generated rule is single recursive on lists. The number of combinations such that u_i has the form of $[W_i|X_i]$ for j arguments in n arguments is at most ${}_nC_j$. In such j arguments, the number of combinations such that at least one v_i has the form of X_i is at most

$$\sum_{i=2}^{j} {}_jC_i,$$

where ${}_1C_2$ is assumed to be 1.

Then, the total number of hypotheses is at most

$$\sum_{j=1}^{n} {}_nC_j \times \left(\sum_{i=2}^{j} {}_jC_i\right).$$

Hence, we obtain the following formulas:

$$\begin{aligned}\sum_{j=1}^{n} {}_nC_j \times \left(\sum_{i=2}^{j} {}_jC_i\right) &= \sum_{j=1}^{n} {}_nC_j \times (2^j - j - 1)\\ &\le \sum_{j=1}^{n} {}_nC_j \times 2^j\\ &\le 3^n.\end{aligned}$$

Here, we have used the following formula:

$$\sum_{i=0}^{n} {}_nC_i \times x^i = (1+x)^n.$$

Proof of Theorem 6

Let α be a ground atom $p(t_1,\cdots,t_n)$ and K_n be the number of the maximal list generalizations of α. If $l = 1$, then we can find the upper bound of K_n in the following way.

For simplicity, suppose that the common lists in α are $t_1, t_2, \cdots, t_{n-1}$, and $|t_1| > |t_2| > \cdots > |t_{n-1}|$. We denote the generalization $\alpha\{t_{j_1} := X_{j_1}, \cdots, t_{j_f} := X_{j_f}\}$ by $\beta_{(j_1,\cdots,j_f)}$. Note that t_{j_i+1} is not a common list in $\beta_{(j_i)}$. Furthermore, for $\beta_{(j_i,j_i+a,j_i+a+b)}$ $(a, b = 2 \text{ or } 3)$, there exist substitutions θ_{j_i+a+b}, θ_{j_i+a}, and θ_{j_i} such that

$$\begin{aligned}\beta_{(j_i,j_i+a)} &= \beta_{(j_i,j_i+a,j_i+a+b)}\theta_{j_i+a+b},\\ \beta_{(j_i,j_i+a+b)} &= \beta_{(j_i,j_i+a,j_i+a+b)}\theta_{j_i+a},\\ \beta_{(j_i+a,j_i+a+b)} &= \beta_{(j_i,j_i+a,j_i+a+b)}\theta_{j_i}.\end{aligned}$$

Hence, $\beta_{(j_i,j_i+a)}$, $\beta_{(j_i,j_i+a+b)}$, and $\beta_{(j_i+a,j_i+a+b)}$ are not maximal list generalizations.

By using the indices of β, K_n is equal to the number of the sequences $(j_1,\cdots,j_f)$ which satisfy the following conditions:

1. $j_1 = 1$ or 2,
2. $j_f = n-1$ or n, and
3. the adjacent number of j_i is either $j_i + 2$ or $j_i + 3$.

For example, if $n = 8$, then the following seven sequences

$$(1,3,5,7),\ (1,3,6),\ (1,4,6),\ (1,4,7),\\ (2,4,6),\ (2,4,7),\ (2,5,8)$$

satisfy the above conditions. For the sequence $(j_1, \cdots, j_f)$ which satisfies the above conditions, the number of sequences such that $j_1 = 1$ is greater than $j_1 = 2$. Let A_n be the set of the sequences $(j_1, \cdots, j_f)$ which satisfy the above conditions and $j_1 = 1$. Then, $K_n \leq 2|A_n|$. Furthermore, for $n \geq 6$, we can construct the set A_n in the following way:

1. if $(j_1, \cdots, j_f) \in A_{n-2}$, then $(j_1, \cdots, j_f, n) \in A_n$,
2. if $(j'_1, \cdots, j'_f) \in A_{n-3}$, then $(j'_1, \cdots, j'_f, n-1) \in A_n$.

Hence, $|A_n|$ satisfies the following equations:

$$|A_3| = 1,\ |A_4| = 1,\ |A_5| = 2,$$
$$|A_n| = |A_{n-2}| + |A_{n-3}|\ (n \geq 6).$$

By mathematical induction on n, we obtain the following formula:

$$\left(\sqrt[3]{2}\right)^{n-2} \leq |A_n| \leq \left(\sqrt{2}\right)^{n-2} \quad (n \geq 6).$$

Hence, the number K_n of the maximal list generalizations is characterized by the following formula:

$$K_n \leq 2\left(\sqrt{2}\right)^{n-2} = \left(\sqrt{2}\right)^{n}.$$

Note that this formula also holds for any $n \geq 1$.

Let l be the number of classes by the sublist relation and C_j be such a class for $1 \leq j \leq l$. For any C_j, the number of the sequences which satisfy the above conditions is at most $\left(\sqrt{2}\right)^{|C_j|}$. Then, the number K_n of the maximal generalizations is at most

$$\left(\sqrt{2}\right)^{|C_1|} \times \cdots \times \left(\sqrt{2}\right)^{|C_l|}.$$

Hence, K_n is also characterized as the following formula:

$$K_n \leq \left(\frac{\left(\sqrt{2}\right)^{n}}{l}\right)^{l}.$$

Bottom-up Layout Generation

Walter Hower
Department of Computer Science, University College Cork, National University of Ireland,
College Rd, Cork, Ireland
E-mail: walter@cs.ucc.ie

Keywords: layout design, constraint satisfaction

Edited by: Rudi Murn

Received: June 20, 1995 **Revised:** August 29, 1995 **Accepted:** October 25, 1995

The constraint satisfaction problem (CSP) comprises a set of n variables with associated finite domains, and some combinations of value assignments ("constraints") to the variables; then, in order to get the globally consistent solution, we need to compute the set of all n-tuples consistent with the given constraints. Here we consider a CSP modeling of a layout problem where we have to place rectangles ("objects") inside a target frame such that the objects do not overlap. However, we further constrain the problem to obey the following requirements: (a) Try to compute all *layout possibilities. (Just the answer whether a layout exists at all or to compute only one possibility should not suffice.) (b) The user shall be able to interact with the (semi-) automatic layout system in a way such that s/he may pick an object to place it in a subarea, offered by the system, in an arbitrary manner without the need to think about the placement of the other objects. (After an algorithm's termination a globally consistent layout should be guaranteed.)*

1 Preliminaries

The constraint satisfaction problem (CSP) comprises a set of n variables with associated finite domains, and some combinations of value assignments ("constraints") to the variables; then, in order to get the globally consistent solution, we need to compute the set of all n-tuples consistent with the given constraints.[1]

There are a lot of papers which exploit specific CSP features, and others focus on local constraint propagation. However, finally, the global solution is wanted and in this regard, we would like to introduce here a general framework which really yields this globally consistent solution. Anyway, to start directly with a higher-level mechanism may not be a bad idea; see also [SF94] (p. 126) in this regard. ([HJ94] illustrates a distributed CSP approach. [BD95] reports on a CSP solving approach using a genetic algorithm component.)

By and large, there are mainly two different qualities of constraint solving techniques: *local constraint propagation* and *global constraint satisfaction*. Briefly speaking, the distinction is as follows: Local propagation just considers those parts of the problem which are closely related to each other and works only on a subset of the problem;[2] global satisfaction however copes with the matter as a whole. Or from another point of view: a local technique usually tries to rule out values which are identified to be of no further use and thereby potentially reduces the domains of the variables;[3] a global algorithm however keeps track of the interrelation of the values and maintains complete combinations. The result of local propagation, the so-called "locally consistent solution", is an n-tuple of (domain) sets, whereas the result of global satisfaction, the so-called "globally consistent solution", is a set of n-tuples enumerating

[1]Striving for *all* combinations is important in a lot of real-world applications; for instance, modelling a set of traffic lights requires all possible colour assignments; see [How95]. Furthermore, in computer-aided layout design all potential configurations should be offered to the user; this feature is one of the reasons to employ a constraint-based approach. (You may consult [Van95].)

[2]also pointing to an assignment mode mentioned below
[3]herewith we point to the label filtering mode

all possible combinations.[4] Although local propagation is a fast technique we would like to note the fact that local consistency does not imply global consistency; cf. also [Kra92] (p. 62, footnote).[5]

Thus, the locally consistent solution just serves as a pre-processing state; the correct instantiations of the variables must still be tried. Besides, in the graphics domain, local propagation often works in an assignment mode to just inform other variables about some variable instantiation(s). Then, when k ($\leq n$) variables are given in a subnetwork, we need the instantiation of $k-1$ variables to determine the value of the remaining one.[6] (However, some systems even allow the propagation of algebraic expressions.) In the papers taken into consideration local propagation has been the method mostly preferred—probably due to the efficiency of the procedure. Unfortunately, it sometimes leaves to the reader to imagine how global consistency is achieved.[7] (Backtracking-like algorithms are discussed in [ZH96].)

In view of the necessity to compute the globally consistent solution we would like to discuss a general framework (working on constraints with arbitrary arities) incorporating the *global* constraint satisfaction view ([How95][8]). The main point to employ such an exhaustive approach is the need of the user of a computer-aided design (CAD) layout system to choose among various or even all options one may have to place the objects. The procedure works in the domain of rectangles, and its employment even in the domain of triangles[9] has just been completed. (In CAD and layout design it is enough to consider rectangles and triangles; we may refer to the triangulation in the area of computer graphics.)

Also [AB93] deals with rectangles. In [Bor81][10] the weakness of local propagation has already been recognized, and it refers to the need of a global analysis—see also [HHLM92] (p. 302). [FS93] states the fact that binary constraint networks are often not expressive enough and prefers user interaction when some modifications come up (pp. 1454/1456 there). ([Ros93] points to "un-do" operations in CAD.) [Szw94] has presented the use of constraint processing in the area of graphics. [HF94] deals with continuous domains. [Fal94] nicely motivated a constraint-based approach to CAD; see also [HHR94].

There are already special purpose routines for the examples introduced herein. The intent of this paper however is to briefly illustrate how a general global constraint solver is able to function well in CAD. ([HRB93][11] may be regarded as an initial precursor.) One of its main features is its ability to interact with the user in a way such that after the termination of the algorithm (employed for layout design) the user may arbitrarily place any object inside a subarea offered by the system while it is guaranteed that an arbitrary placement of a different object in another subarea (again offered by the system) in a following step is always possible; this guarantee is ensured by the *global* consistency of the solver.

In order to point to related work in a really broad sense we have compiled about 200 papers and collected this part in a separate work: [HG95]. Therefore, only an unusually brief list of appreciated articles is included here. (This fact hopefully does not matter because we do not compare the various appproaches; we would just like to illustrate a global constraint satisfaction view.)

2 Algorithmic framework

The procedure employs a global constraint satisfaction solver where all the various layout possibilities are generated by a bottom-up approach. The CSP algorithm in its use here just employs an upward phase starting from the variables' domains (the 1-ary constraints) and proceeds until the n-ary constraint (the globally consistent solution) is finally computed.

As already introduced n is the number of the variables of the CSP. Let the index set I associated with a constraint C_I denote the variables involved in C_I; the cardinality of this index set is the arity of the constraint. The maximum arity

[4]Please note that "local" and "global" may both refer to the way how to relate constraints and to the solution quality of a CSP.

[5]Please consider [Dv95] regarding the way of the quality of a locally consistent solution to a globally consistent one.

[6]typically for equational constraints

[7]In some cases just simple assignment techniques seem to get employed.

[8]which I have already further improved in the meantime

[9]where at the moment two sides must still be in parallel to the co-ordinate axes

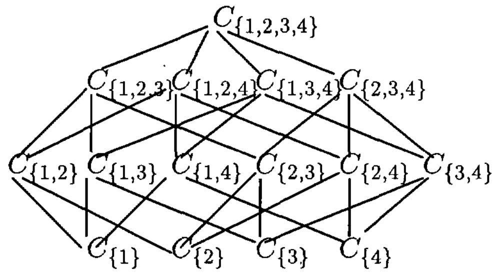
[10]see also [San94] w.r.t. further developments

[11]see also [How94]

of the constraints is the highest index level: $h := max_{C_I}\{|I| \mid I \subseteq \{1,...,n\}\}$. (The "natural equijoin" operation $\bowtie$ (see also [Bib88] and [JCG95]) may get understood as in the database field.) We use the following abbreviations:
CS_i := entire set of constraints of arity i whose index sets are proper subsets of the index set of a specific constraint of arity $i+1$. (See the formal description of the algorithm. $|CS_i| = i+1$.)
$\bowtie_{\forall C \in CS_i} C \;:=\; C_{I_1} \bowtie C_{I_2} \bowtie \ldots \bowtie C_{I_{|CS_i|}}$.

We do not have constraints with an arity larger than h; so, when it is possible to establish consistency among h variables arbitrarily chosen ("h-consistency") then we may try to compute the globally consistent solution ("n-consistency"). The algorithm already realizes this idea.

The following lattice structure symbolizes a CSP with four variables:



```
algorithm global_upward_skip
for $i \leftarrow 1$ to $h-1$
  for each $C_J : |J| = i+1$
    $CS_i \leftarrow \{C_I \mid |I| = i, I \subset J\}$
    $C_J \leftarrow C_J \cap (\bowtie_{\forall C \in CS_i} C)$
if for each $C_H$ $(|H| = h)$ $|C_H| \neq 0$
  then skip else INCONSISTENCY
```

Algorithm 1: Global upward skip

```
procedure skip
$C_{\{1,\ldots,n\}} \leftarrow \bowtie_{\forall C \in CS_h} C$
if $C_{\{1,\ldots,n\}} \neq \{\}$
  then CONSISTENCY else INCONSISTENCY
```

In order to reformulate a problem to get a discrete search space we group corresponding values to form an interval; the reasoning then takes place along the boundaries of the value intervals. (See also [EBH93] where interval domains represent ranges of values.)

3 Illustration

Let the *top*ological[12] *CSP* solver TopCsp work on the following short example.[13] The problem is similar to the one once found in the newsgroups *comp.ai* and *comp.graphics.algorithms*: "configuration/encapsulation problem": given a larger rectangle and several smaller ones we would like to know all the possibilities of the smaller rectangles to fit into the larger one. To perform this task we introduce a two-dimensional variable for each rectangle indicating the range of the lower left corner point (P^l) and the range of the upper right corner point (P^r); each point itself has two dimensions (along both the x- and y-axis).

For instance, a given larger rectangle has the dimensions 9×11. This results in the following co-ordinates in a fixed co-ordinate frame:

$$P^l : (0,0),\ P^r : (9,11).$$

Let further three other (smaller) rectangles r_1, r_2, r_3 be given with the following information on width (w) and height (h):

$w_1 := 5, h_1 := 2;\ w_2 := 3, h_2 := 1;\ w_3 := 4, h_3 := 10.$

Now we denote the various possibilities (initially given by the dimension of the frame rectangle along with the width and height information) for these corner points of the smaller rectangles by an interval notation which yields the following ranges:

$$P^l_1: ([0..4],[0..9]),\ P^r_1: ([5..9],[2..11]),$$
$$P^l_2: ([0..6],[0..10]),\ P^r_2: ([3..9],[1..11]),$$
$$P^l_3: ([0..5],[0..1]),\ P^r_3: ([4..9],[10..11]),$$

which corresponds to the computation of the 1-ary constraints.[14]

The procedure computes the 2-ary constraints by checking the different choices of the placement of two rectangles: whether it is possible to place a rectangle left (or right) of another one or below (or above) it. Therefore, we have the following possibilities connected among each other via "<u>or</u>" w.r.t. the rectangles r_1 and r_2:

[12]Winfried Graf has proposed this term to me; see also [Gra95].

[13]Please note that such a simple example allows to focus the discussion on the main points; a more "real-world" example would veil the focal point and would be much harder to introduce.

[14]In fact, we only maintain the lower left corner point; the upper right one is just a convenience to us.

$x_{P^r_1} \leq x_{P^l_2}$ (r_1 *is left of* r_2)
$x_{P^r_2} \leq x_{P^l_1}$ (r_2 *is left of* r_1)
$y_{P^r_1} \leq y_{P^l_2}$ (r_1 *is below* r_2)
$y_{P^r_2} \leq y_{P^l_1}$ (r_2 *is below* r_1).

In the following we take the lower left corner point as the reference point.

To re-compute the boundaries of the intervals we introduce the following notation: min_{x_i} denotes the smallest possible x-value for the reference point of object i (the left bound of the x-interval), and max_{x_i} denotes its largest possible x-value (the right bound of the x-interval). (In the example above, $min_{x_1} := 0$, $max_{x_1} := 4$; $min_{x_2} := 0$, $max_{x_2} := 6$.) Obviously, min_{y_i} denotes the smallest possible y-value for the reference point of object i (the left bound of the y-interval), and max_{y_i} denotes its largest possible y-value (the right bound of the y-interval). (In the example above, $min_{y_1} := 0$, $max_{y_1} := 9$; $min_{y_2} := 0$, $max_{y_2} := 10$.)

The following two computation rules are needed in the first case ("r_1 *is left of* r_2"):

<u>if</u> $min_{x_1} + w_1 > min_{x_2}$ <u>then</u> $min_{x_2} := min_{x_1} + w_1$;

<u>if</u> $max_{x_1} > max_{x_2} - w_1$ <u>then</u> $max_{x_1} := max_{x_2} - w_1$.

Now it becomes visible that in fact just the lower left corner points are maintained.

To deal with the second case ("r_2 *is left of* r_1") you just interchange the numbers "1" and "2" in both rules.

The following two computation rules are needed in the third case ("r_1 *is below* r_2"):

<u>if</u> $min_{y_1} + h_1 > min_{y_2}$ <u>then</u> $min_{y_2} := min_{y_1} + h_1$;

<u>if</u> $max_{y_1} > max_{y_2} - h_1$ <u>then</u> $max_{y_1} := max_{y_2} - h_1$..

(Here, x has been replaced by y, and w has been replaced by h, compared to the rules for the first case.)

To deal with the fourth case ("r_2 *is below* r_1") you again interchange the numbers "1" and "2".

As an example we show the re-computation of the interval boundaries concerning the topological relations mentioned first and last. Testing $x_{P^r_1} \leq x_{P^l_2}$ results in the following:

P^l_1: ([0..1], [0..9]), P^r_1: ([5..6], [2..11]);
P^l_2: ([5..6], [0..10]), P^r_2: ([8..9], [1..11]).

Checking $y_{P^r_2} \leq y_{P^l_1}$ yields:

P^l_1: ([0..4], [1..9]), P^r_1: ([5..9], [3..11]);
P^l_2: ([0..6], [0..8]), P^r_2: ([3..9], [1..9]).

Concerning the rectangles r_1 and r_3 the boundaries are computed as follows when testing $x_{P^r_1} \leq x_{P^l_3}$:

P^l_1: ([0..0], [0..9]), P^r_1: ([5..5], [2..11]);
P^l_3: ([5..5], [0..1]), P^r_3: ([9..9], [10..11]).

Relating r_2 and r_3 when checking $y_{P^r_2} \leq y_{P^l_3}$ yields the following:

P^l_2: ([0..6], [0..0]), P^r_2: ([3..9], [1..1]);
P^l_3: ([0..5], [1..1]), P^r_3: ([4..9], [11..11]).

The final step is to check consistency among all three rectangles. Thereby, we "intersect" the intervals and test consistency. In our example 44 "abstract" solutions are computed (and presented in an interval-like notation) where each abstract solution represents several single solutions.[15] For instance, the following solution space is presented corresponding to one of the 44 abstract solutions:

P^l_1: ([0..0], [1..9]), P^r_1: ([5..5], [3..11]);
P^l_2: ([0..6], [0..0]), P^r_2: ([3..9], [1..1]);
P^l_3: ([5..5], [1..1]), P^r_3: ([9..9], [11..11]);
<u>and</u> $x_{P^r_1} \leq x_{P^l_3}$ (r_1 *is left of* r_3)
<u>and</u> $y_{P^r_2} \leq y_{P^l_3}$ (r_2 *is below* r_3)
<u>and</u> $y_{P^r_2} \leq y_{P^l_1}$ (r_2 *is below* r_1).

The complete computation of all solutions of the example discussed here has been performed in a few milli-seconds.

Working just on the boundaries of the intervals enables a reasonable processing. This fact that the domain sizes do no longer be a crucial parameter, due to a kind of qualitative reasoning ([Her94]) on an interval-like representation, is one of the advantages of this approach; hence, it actually does not matter whether we consider a discrete or a continuous domain.

An initial implementation also positions pieces of furniture in a room. ([Mac92] mentions to employ CSP techniques in the domain of furniture

[15]Those solutions which are obtained by interchanging height and width of the "working" rectangles are included.

layout, too.) Even a deployment for VLSI design is conceivable—we have already prepared the algorithm to consider some space ("channels") between objects (for the wiring).

Furthermore, an early detection of global inconsistency is possible; if in the example a proper constraint is already present forcing, for instance, r_1 to be below r_3 the lower bound of the y-interval of the lower left corner point of r_3 would have to get incremented to the value 2 producing an empty interval of values for y: P_3^l: ([0..5], [2..1]). We may notice that this is an important efficiency feature. (It also contributes to the completeness of the approach—in contrast to the lack of such a feature in general evolutionary approaches.)

4 Final remarks

The actual implementation is already based on a more flexible version of the algorithm.

In the future, intelligent heuristics should be considered to select an appropriate order of the constraints during the computation process and to exploit problem-dependent specialties.

Just recently we have experimented with some evolutionary computing ideas[16] (thereby, however, abandoning the exhaustive completeness feature) to gain in efficiency in the case when only a few placements suffice; you may consult [HKR95].

Acknowledgement

Many thanks to Derrick Köstner and Manfred Rosendahl for their kind assistance as well as to the referees for their helpful comments.

References

[AB93] Abderrahmane Aggoun and Nicolas Beldiceanu. Extending CHIP in order to solve complex scheduling and placement problems. *Mathematical and Computational Modelling*, 17(7):57–73, 1993. Pergamon Press Ltd.

[BD95] James Bowen and Gerry Dozier. Solving Constraint Satisfaction Problems Using A Genetic/Systematic Search Hybrid That Realizes When to Quit. In *The 6th International Conference on Genetic Algorithms (ICGA-95)*, University of Pittsburgh, Pennsylvania, USA, July 15–19, 1995.

[BH93] Roland Berling and Walter Hower. Solving Geometrical Constraint Systems. *AI Communications*, 6(3/4):232–234, Sept./Dec. 1993. IOS Press, Amsterdam, The Netherlands. Book review of [Kra92].

[Bib88] Wolfgang Bibel. Constraint Satisfaction from a Deductive Viewpoint. *Artificial Intelligence*, 35(3):401–413, 1988. Elsevier Science Publishers B.V., Amsterdam, The Netherlands.

[Bor81] Alan Borning. The Programming Language Aspects of ThingLab, a Constraint-Oriented Simulation Laboratory. *ACM Transactions on Programming Languages and Systems*, 3(4):353–387, October 1981.

[Bor94] Alan Borning, editor. *Principles and Practice of Constraint Programming*, volume 874 of *Lecture Notes in Computer Science*. Proceedings, Springer-Verlag, Berlin/Heidelberg, May 2–4, 1994. Second International Workshop, PPCP '94, Rosario, Orcas Island, WA, U.S.A.

[Dv95] Rina Dechter and Peter van Beek. Local and Global Relational Consistency — Summary of Recent Results. In [MR95], pages 240–257, 1995.

[EBH93] D.G. Elliman, E.K. Burke, and M.I. Heard. A Constraint Based Approach to Engineering Design. *AISB Quarterly*, 83/84: 35–38, Spring/Summer 1993. Newsletter of the Society for the Study of Artificial Intelligence and Simulation of Behaviour, Brighton, UK.

[Fal94] Boi Faltings. Constraint Satisfaction Problems in CAD Systems, August 14, 1994. Invited talk, International Workshop on *Constraint Processing in Computer-Aided Design (CoPiCAD-94)*, Swiss Federal Institute of Technology, Lausanne, Switzerland.

[FS93] Boi Faltings and Kun Sun. Computer-aided Creative Mechanism Design. In Ruzena Bajcsy, editor, *IJCAI-93, Proceedings*

[16]see [KTN94], for instance

of the Thirteenth International Joint Conference on Artificial Intelligence, pages 1451–1457, Chambéry, Savoie, France, August 28 – September 3, 1993. IJCAII. Volume 2; distributed by Morgan Kaufmann Publishers, San Mateo, California, USA.

[Gra95] W. H. Graf. *Constraintbasiertes automatisches Layout multimedialer Präsentationen: Semantikorientierte Plazierung, Typographie, Visualisierung und Design.* PhD thesis, Fachbereich Informatik, Technische Fakultät, Universität des Saarlandes, Saarbrücken, 1995. Dissertation.

[Her94] Daniel Hernández. *Qualitative Representation of Spatial Knowledge*, volume 804 of *Lecture Notes in Artificial Intelligence, Subseries of Lecture Notes in Computer Science.* Springer-Verlag, Berlin/Heidelberg, 1994.

[HF94] Djamila Haroud and Boi Faltings. Global Consistency for Continuous Constraints. In [Bor94], pages 40–50, 1994.

[HG95] Walter Hower and Winfried H. Graf. Research in Constraint-Based Layout, Visualization, CAD, and Related Topics: A Bibliographical Survey. In *International Workshop on Constraints for Graphics and Visualization (CGV '95)*, Cassis, France, September 18, 1995. DFKI Research Report RR-95-12, Deutsches Forschungszentrum für Künstliche Intelligenz GmbH, Saarbrücken.

[HHLM92] Richard Helm, Tien Huynh, Catherine Lassez, and Kim Marriott. A Linear Constraint Technology for Interactive Graphic Systems. In *Graphics Interface '92*, pages 301–309, Canada, 1992. Proceedings, Morgan Kaufmann Publishers, USA.

[HHR94] Walter Hower, Djamila Haroud, and Zsófia Ruttkay, editors. *Constraint Processing in Computer-Aided Design (CoPiCAD-94)*, August 14, 1994. Workshop notes, Third International Conference on Artificial Intelligence in Design (AID'94), Swiss Federal Institute of Technology, Lausanne, Switzerland, 15–18 August 1994.

[HJ94] Walter Hower and Stephan Jacobi. A distributed realization for constraint satisfaction. In H. Kitano et al., editors, *Parallel Processing for Artificial Intelligence 2*, volume 15 of *Machine Intelligence and Pattern Recognition series*, pages 107–116. Elsevier Science Publishers B.V., Amsterdam, The Netherlands, 1994.

[HKR95] Walter Hower, Derrick Köstner, and Manfred Rosendahl. Computer-aided layout by evolutionary computing. In Remco C. Veltkamp and Edwin H. Blake, editors, *Proceedings of the fifth Eurographics workshop on Programming Paradigms in Graphics, EUROGRAPHICS '95*, pages 251–269, Maastricht, September 2–3, 1995. CWI, Amsterdam, The Netherlands.

[How94] Walter Hower. 5th International Conference on Human-Computer Interaction jointly with 9th Symposium on Human Interface (Japan), 8–13 August 1993, Orlando, Florida, USA. *AISB Quarterly*, 86:46–47, 1994. Winter 1993/94, Newsletter of the Society for the Study of Artificial Intelligence and Simulation of Behaviour, Brighton, UK; conference report.

[How95] Walter Hower. Constraint satisfaction — Algorithms and complexity analysis. *Information Processing Letters*, 55(3):171–178, 11 August 1995. Elsevier Science Publishers B.V.

[HRB93] Walter Hower, Manfred Rosendahl, and Roland Berling. Constraint processing in human-computer interaction with an emphasis on intelligent CAD. In Michael J. Smith and Gavriel Salvendy, editors, *Human-Computer Interaction: Applications and Case Studies*, volume 19A of *Advances in Human Factors / Ergonomics*, pages 243–248. Elsevier Science Publishers B.V., Amsterdam, The Netherlands, August 8–13, 1993. Proceedings of the Fifth International Conference on Human-Computer Interaction (HCI International '93, Orlando, Florida, USA), Volume 1.

[JCG95] Peter Jeavons, David Cohen, and Marc Gyssens. A Unifying Framework for Tractable Constraints. In [MR95], pages 276–291, 1995.

[Kra92] Glenn A. Kramer. *Solving Geometric Constraint Systems: A Case Study in Kinematics.* Artificial Intelligence series. The MIT Press, Cambridge, Massachusetts, USA / London, England, UK, 1992. Reviewed in [BH93].

[KTN94] Takashi Kido, Kazuo Takagi, and Masakazu Nakanishi. Analysis and Comparisons of Genetic Algorithm, Simulated Annealing, TABU Search, and Evolutionary Combination Algorithm. *Informatica*, 18(4):399–410, December 1994. An International Journal of Computing and Informatics. Special Issue: Artificial Life.

[Mac92] Alan K. Mackworth. Constraint Satisfaction. In Stuart C. Shapiro, editor, *Encyclopedia of Artificial Intelligence*, volume 1, pages 285–293. John Wiley & Sons, 1992. Second edition.

[MR95] Ugo Montanari and Francesca Rossi, editors. *Principles and Practice of Constraint Programming — CP '95*, volume 976 of *Lecture Notes in Computer Science*. Proceedings, Springer-Verlag, Berlin/Heidelberg, September 19–22, 1995. First International Conference, Cassis, France.

[Ros93] Manfred Rosendahl. UNDO in CAD Systems. In W. Cellary, K. Vidyasankar, and G. Vossen, editors, *Versioning in Database Management Systems*, page 12, Dagstuhl-Seminar-Report 55, IBFI GmbH, Internationales Begegnungs- und Forschungszentrum für Informatik, Schloß Dagstuhl, Wadern, February 1–5, 1993.

[San94] Michael Sannella. Analyzing and Debugging Hierarchies of Multi-Way Local Propagation Constraints. In [Bor94], pages 63–77, 1994.

[SF94] Daniel Sabin and Eugene C. Freuder. Contradicting Conventional Wisdom in Constraint Satisfaction. In A.G. Cohn, editor, *ECAI 94, 11th European Conference on Artificial Intelligence*, pages 125–129, Amsterdam, The Netherlands, August 8-12, 1994. Proceedings, John Wiley & Sons, Ltd., Chichester, England.

[Szw94] Gerd Szwillus. Graphical Constraints. CHI'94 Tutorial, Conference on Human Factors in Computing Systems, April 25, 1994. Boston, MA, U.S.A.

[Van95] Pascal Van Hentenryck. Constraint Solving for Combinatorial Search Problems: A Tutorial. In [MR95], pages 564–587, 1995.

[ZH96] Martin Zahn and Walter Hower. Backtracking along with constraint processing and their time complexities. *Journal of Experimental & Theoretical Artificial Intelligence*, 8(1), 1996. Taylor & Francis, London, UK.

Informational Frames and Gestalts

Anton P. Železnikar
An Active Member of the New York Academy of Sciences
Volaričeva ulica 8
SI 61111 Ljubljana, Slovenia (anton.p.zeleznikar@ijs.si)

Keywords: informational frame—arrangement, concatenation, demarked, disharmonious, edge syntax, harmonious, operator composition, parenthesized, possibility, syntax; informational framing; informational gestalt—axiomatic, causal, circular, inference, informational machine, intelligent, metaphysicalistic, parallel, phenomenalistic, reverse, serial, star; informational gestaltism; number of formulas in a gestalt

Edited by: Jíři Šlechta

Received: September 28, 1995 **Revised:** November 6, 1995 **Accepted:** November 20, 1995

This article deals with two characteristic and widely useful notions, called the informational frame (of representation) and the informational gestalt. A preliminary discussion of both notions was already presented in [17]. Informational framing is nothing else than a part of informational gestaltism by which various causal possibilities of formulas come into existence. Although a frame is everything which can be put in a frame within a well-formed informational formula, the concatenation of frames must preserve the so-called possibility of a frame to be a part of a well-formed formula. On the other hand, the gestalt structure represents a parallel array of informational formulas, that is, an informational system of causally different formulas proceeding from a given formula. To this, circular gestalting can induce the reversely circular properties, so different forms of gestalts become possible. The most complex and free gestalt called star gestalt is a consequence of an initial circular formula and its graph, where operator transitions from one to the next operand are possible in an arbitrary manner of repeated looping.

1 Introduction

Both the notion of informational frame and informational gestalt[1] have been introduced in a superficial form, in connection with the informational Being-of [17] as a phenomenon of informational functionalism. In this article, both informational frame and informational gestalt will be tackled in a more fundamental manner using some particular means of informational formalism to make them informationally accessible and formally effective.

An informational frame is everything which can be framed within an informational formula, from a single parenthesis $\boxed{)}$, operand $\boxed{\alpha}$ or operator $\boxed{\models}$ to an arbitrarily complex part of the formula, say $\boxed{) \models \alpha}$. In a similar way, the principle of framing can be applied to elements of a demarked formula [12], for example, $\boxed{.}$ or to any complex part of the demarked formula, e.g. $\boxed{. \models \alpha}$. A gestalt of any informational formula is everything formally, especially causally, hidden behind the structure of the formula which represents an arbitrarily complex informational entity. We could say that the gestaltistic nature of a serial formula comes in the foreground when this formula is roughly sketched by the corresponding informational graph (a graphical scheme of operand circles and operator arrows) in which no parenthesis pairs or demarcation points of a formula are considered. Therefore, the gestalt of an informational entity remains the hidden other, formally, the adequately transformed formula system concerning the original operands and operators in an unchanged

[1]This paper is a private author's work and no part of it may be used, reproduced or translated in any manner whatsoever without written permission except in the case of brief quotations embodied in critical articles.

order within a formula, describing the entity in question. In general, an informational entity can appear (be understood, interpreted, grasped) as a gestalt of different possible entities (various formulas), strictly corresponding to the initial order of the operand-operator structure of the entity in question leaving parenthesizing or demarcation completely open.

As the reader will see, the informational frame (of representation [4]) can be any part of an informational formula (system) and, in this respect, the frame does not follow a structurally limited and traditionally organized syntactic formalism. The concept of the informational frame shows also how traditional syntactic schemes can be managed in a natural frame-appearing form leaving open the concatenation of frames in, certainly, a regular form. Any part of the formula means that the formula can be broken off at any place and that a part of it does not necessarily represent an operand or operator structure.

On the other hand, an informational gestalt will represent the possibility of the variety or, the whole variety or the whole variety of an operand, formula or formula system in a structural sense, where different structures (possible formulas to a given formula) can be forecasted automatically to some determined extent.

2 Two Cases of Informational Enframing

The usefulness of framing was demonstrated for a verbal case concerning the predicate of existing of something in [18]. Framing of the informational entities can become an effective and transparent aid in transforming linguistic forms (especially, sentences) into informational ones (e.g., cognitively relevant internal states, representations) and vice versa.

2.1 Enframing of a Sentence and Its Translation

Let us take an example of framing a sentence ([5] p. 161) in German, its formal transcription, and retranslation into English. The enframed original German sentence is

$$\boxed{\begin{array}{c}\boxed{\begin{array}{c}\boxed{\textit{Die Rede}\ \boxed{\textit{ist mit}}_{\phi_1}\ \textit{Befindlichkeit}}\\ \boxed{\textit{und}}\ \textit{Verstehen}\end{array}}\\ \boxed{\textit{existenzial gleichursprünglich}}_{\phi_1}\end{array}}$$

The dot at the end of the sentence will be replaced by the semicolon at the end of a formula, when the formula appears in a parallel informational system. In the last frame the main operator composition (verbally, *ist existenzial gleichursprünglich mit*) is split within the two frames marked consecutively by ϕ_1 and, to this, still verbally transposed. In this sense, the substitutional enframed German sentence, also suitable for formal translation, becomes

$$\boxed{\begin{array}{c}\boxed{\begin{array}{l}\textit{Die Rede}\\ \boxed{\textit{ist existenzial gleichursprünglich mit}}_{\phi_1}\\ \textit{Befindlichkeit}\end{array}}\\ \boxed{\textit{und}}\ \textit{Verstehen}\end{array}}.$$

Let us introduce the following marks for the operands and operators in the given two sentences: $\mathfrak{r}$ for 'Rede' (discourse δ), $\mathfrak{b}$ for 'Befindlichkeit' (state-of-mind σ) and $\mathfrak{v}$ for 'Verstehen' (understanding v) as operands, and $\models_{\text{ex}}$ for 'ist existenzial' ('is existential' or 'informs existentially', operator $\models_{\text{exist}}$), $\models_{\text{gl-ur}}$ for 'ist gleichursprünglich' ('is equiprimordial' or 'informs equiprimordially', operator $\models_{\text{eq-prim}}$) and $\models_{\text{mit}}$ for 'ist mit' ('is with' or 'informs with', operator $\models_{\text{with}}$) as operators. An interpretation of the first enframed form is

$$(\mathfrak{r} \models_{\text{mit}} \mathfrak{b}, \mathfrak{v}) \models_{\text{ex}} \circ \models_{\text{gl-ur}}$$

where

$$(\mathfrak{r} \models_{\text{mit}} \mathfrak{b}, \mathfrak{v}) \equiv \begin{pmatrix} \mathfrak{r} \models_{\text{mit}} \mathfrak{b}; \\ \mathfrak{r} \models_{\text{mit}} \mathfrak{v} \end{pmatrix}$$

when the German 'und' ('and') was interpreted by comma (a short form) and by semicolon (a long form) within a parallel system of two formulas. Both kinds of expression are equivalent (informational operator $\equiv$).

The second form of the sentence enframing (frame-informational interpretation) delivers, formally,

$$\mathfrak{r}\ (\models_{\mathrm{ex}}\circ\models_{\mathrm{gl\text{-}ur}})\circ\models_{\mathrm{mit}}\ \mathfrak{b},\mathfrak{v}$$

being a parallel system (as shown in the preceding case) in regard to Befindlichkeit $\mathfrak{b}$ and Verstehen $\mathfrak{v}$.

The first formula is externalistically open via the operator composition $\models_{\mathrm{ex}}\circ\models_{\mathrm{gl\text{-}ur}}$ while in the second formula this composition is brought into a formula interior operator composition with $\models_{\mathrm{mit}}$, that is, $(\models_{\mathrm{ex}}\circ\models_{\mathrm{gl\text{-}ur}})\circ\models_{\mathrm{mit}}$.

Let us take the second form of the enframing of the German sentence (the last formula) with the English-adequate operand and operator markers, that is,

$$\rho\ (\models_{\mathrm{exist}}\circ\models_{\mathrm{eq\text{-}prim}})\circ\models_{\mathrm{with}}\ \sigma, v$$

Considering this formula, the English translation of the German sentence becomes (in an adequately enframed form)

$$\boxed{\begin{array}{c}\boxed{\begin{array}{l}\textit{Discourse}\\ \boxed{\textit{is existentially equiprimordial with}}_{\phi_1}\\ \textit{state-of-mind}\end{array}}\\ \boxed{\textit{and}}\ \textit{understanding}\end{array}}\ .$$

This sentence appears in the English translation of [5], that is in [6] (p. 203). A 'weaker' (so-called literal) translation would follow the first form of the German sentence enframing, that is,

$$\boxed{\begin{array}{l}\boxed{\begin{array}{c}\boxed{\textit{Discourse}\ \boxed{\textit{is with}}_{\phi_1}\ \textit{state-of-mind}}\\ \boxed{\textit{and}}\ \textit{understanding}\end{array}}\\ \boxed{\textit{existentially equiprimordial}}_{\phi_1}\end{array}}\ .$$

Another scheme of framing for the discussed sentence, depending upon an intuitive understanding, could take the enframing form

$$\boxed{\begin{array}{l}\boxed{\textit{Discourse}}\\ \boxed{\textit{is existentially equiprimordial with}}_{\phi_1}\\ \boxed{\textit{state-of-mind}\ \boxed{\textit{and}}\ \textit{understanding}}\end{array}}\ .$$

In the parallel approach, as formally expressed, the $\boxed{\text{and}}$ disappears and so two parallel sentences are coming to the surface, that is,

$$\boxed{\begin{array}{l}\boxed{\begin{array}{l}\textit{Discourse}\\ \boxed{\textit{is existentially equiprimordial with}}_{\phi_1}\\ \textit{state-of-mind}\end{array}}\ .\\ \boxed{\begin{array}{l}\textit{Discourse}\\ \boxed{\textit{is existentially equiprimordial with}}_{\phi_2}\\ \textit{understanding}\end{array}}\ .\end{array}}$$

Now, a dot appears at the end of each of the two enframed sentences. In the formal case, we have got the possibility with the comma between σ and v, or the semicolon between the two formulas. It is to stress that substantial differences can exist between the operators subscripted by ϕ_1 and ϕ_2 because the first one concerns internally the state-of-mind while the second one pertains internally to understanding. Here, the informational character of an equally marked operator in different contexts comes to the worth. Namely, in an informational transition of the form $\alpha\models\beta$, where α is the informer and β is the observer, operator $\models$ is to be understood as operator composition $\models_\alpha\circ\models_\beta$, where $\models_\alpha$ has to be decomposed according to the α's informingness and $\models_\beta$ according to the β's informedness.

The demonstrated translation of the German sentence into informational formula was only that which could be called the first informational approximation. Introducing the concept of the so-called informational graph it is possible to give several kinds of interpretation of an informational formula. In Fig. 1 an informational graph of the widening of this approximation is presented

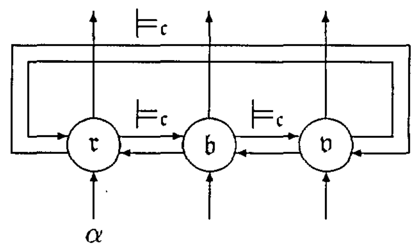


Figure 1: *A graphical interpretation of the bidirectionally and circularly (5-loop) structured parallel decomposed system* $\mathfrak{r}$ *concerning the input entity* α.

where the semantic scope of the word 'gleichursprünglich' (equiprimordial) is taken into consideration. Let us mark the operator composition

$(\models_{\mathrm{ex}} \circ \models_{\mathrm{gl\text{-}ur}}) \circ \models_{\mathrm{mit}}$ corresponding to 'ist existenzial gleichursprünglich mit' by operator $\models_{\mathfrak{c}}$. By this operator the arrows between entities $\mathfrak{r}, \mathfrak{d}, \mathfrak{v}$ in Fig. 1 are marked. The vertical input and output arrows are unmarked and represent the general operator $\models$.

Which is the parallel informational system describing the graph in Fig. 1? If discourse (die Rede) $\mathfrak{r}$ concerns an entity α, that is $\mathfrak{r}(\alpha)$, the so-called $\mathfrak{r}$-solution of the system graphically presented is

$$\mathfrak{r}(\alpha) \rightleftharpoons \begin{pmatrix} \alpha \models \mathfrak{r};\ \models \mathfrak{d}, \mathfrak{v}; \\ \mathfrak{r}, \mathfrak{d}, \mathfrak{v} \models; \\ \mathfrak{r} \models_{\mathfrak{c}} \mathfrak{d};\ \mathfrak{d} \models_{\mathfrak{c}} \mathfrak{v};\ \mathfrak{v} \models_{\mathfrak{c}} \mathfrak{r}; \\ \mathfrak{r} \models_{\mathfrak{c}} \mathfrak{v};\ \mathfrak{v} \models_{\mathfrak{c}} \mathfrak{d};\ \mathfrak{d} \models_{\mathfrak{c}} \mathfrak{r} \end{pmatrix}$$

In the first row of the parallel array the input paths are located and in the second row the output paths. The third and the fourth row represent the main two loops (the left and the right one) of the graph. These two loops, joined, hide three more elementary loops existing between operands $\mathfrak{r}$ and $\mathfrak{d}$, $\mathfrak{d}$ and $\mathfrak{v}$, and $\mathfrak{v}$ and $\mathfrak{r}$. Thus, one of the so-called serial type solutions (a specific one) for $\mathfrak{r}(\alpha)$ can be expressed as

$$\mathfrak{r}^1(\alpha) \rightleftharpoons \begin{pmatrix} \alpha \models \mathfrak{r};\ \models \mathfrak{d}, \mathfrak{v}; \\ \mathfrak{r}, \mathfrak{d}, \mathfrak{v} \models; \\ \mathfrak{r} \models_{\mathfrak{c}} (\mathfrak{d} \models_{\mathfrak{c}} (\mathfrak{v} \models_{\mathfrak{c}} \mathfrak{r})); \\ \mathfrak{r} \models_{\mathfrak{c}} (\mathfrak{v} \models_{\mathfrak{c}} (\mathfrak{d} \models_{\mathfrak{c}} \mathfrak{r})); \\ \mathfrak{r} \models_{\mathfrak{c}} (\mathfrak{d} \models_{\mathfrak{c}} \mathfrak{r});\ \mathfrak{d} \models_{\mathfrak{c}} (\mathfrak{v} \models_{\mathfrak{c}} \mathfrak{d}); \\ \mathfrak{v} \models_{\mathfrak{c}} (\mathfrak{r} \models_{\mathfrak{c}} \mathfrak{v}) \end{pmatrix}$$

This solution formula system considers the input and output structure of the graph in Fig. 1 in the first and the second row, two long-size (bidirectional) loops in the third and fourth row, and three short loops in the last two rows, respectively.

The discussed examples of the natural text translation into informational formulas, and vice versa, demonstrate the hidden (intuitive) importance of the informational framing. As a consequence, framing concerns the so-called gestalting, where as the gestalt of a serially structured formula the parallel system of all possible formulas is meant, emerged from an initial formula by all possible displacements of the parenthesis pairs.

2.2 Sentential Paradigm and Possible Frames and a Gestalt

The ideas for the subsequent example of using informational frames and a gestalt go back to Fodor [2, 3] being sketched in a condensed form in Churchland ([1], pp. 385–388). We will study this particular case on the basis of the graph presented in Fig. 2. This graph is an exact representation of the parallel formula system (as a free-standing system without a specific marker) consisting of primitive transitions (from an operand to another), which is,

$$\begin{pmatrix} \sigma \models \iota; & & & \\ \iota \models \mathfrak{I}_\iota; & \mathfrak{I}_\iota \models \mathfrak{i}_\iota; & \mathfrak{i}_\iota \models \mathfrak{C}_\iota; & \mathfrak{C}_\iota \models \gamma_\iota; \\ & \gamma_\iota \models \mathfrak{E}_\iota; & \mathfrak{E}_\iota \models \rho_\iota; & \rho_\iota \models \mathfrak{c}_\iota; \\ \mathfrak{c}_\iota \models \iota; & \gamma_\iota \models \mathfrak{I}_\iota; & \mathfrak{c}_\iota \models \mathfrak{C}_\iota; & \mathfrak{c}_\iota \models \mathfrak{E}_\iota; \\ & \mathfrak{i}_\iota \models \mathfrak{I}_\iota; & \gamma_\iota \models \mathfrak{C}_\iota; & \rho_\iota \models \mathfrak{E}_\iota \end{pmatrix}$$

This array includes three types of transitions: the input, forward, and backward ones. In the first row, the only input transition informs the internal state about the sentence. In the second and third row seven forward transitions are listed. In the last two rows, seven backward (feedback) transitions appear, enabling a complex cycling of the system when the input sentence is identified in a processing way by the arising representation ρ_ι and content $\mathfrak{c}_\iota$.

On the basis of this graph we will present a solution for content $\mathfrak{c}_\iota$ and some possible, for the discussion relevant frames. The gestalt and possible interpretations of content $\mathfrak{c}_\iota$ will be presented (and discussed later on).

The concept of sentential attitudes α_σ belongs to the autonomy of human being; on the other side, the logical inference defends co-evolution and interdependence, that is, informing of entities, in informational terms. Beliefs, desires, thoughts, intentions, etc. inform between a being's internalism and externalism and play a crucial role in causation of living behavior. Sentential attitudes approach the nature of the so-called representations and of the rules that govern the transitions between representations. All this sounds informationally familiar where sentential attitudes and representations may function as distinguished informational entities.

There exists an interplay of the sentential attitudes and representations, a parallel and serial informing among entities. The theory postulates that sentential attitudes can also play a role in nonconscious processes and, in this way, can be involved in cognitive processes, e.g. in the form

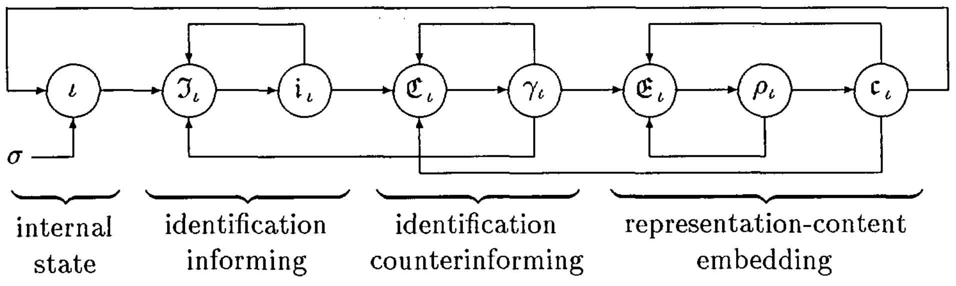


Figure 2: *A sententially paradigmatic, metaphysicalistically (7-tuple-loop) structured internal state system ι giving the input sentence(s) σ the sentence(s) representation $\rho_\iota(\sigma)$ and sentence(s) internal content (understanding with meaning) $\mathfrak{c}_\iota(\sigma)$.*

of inductive and deductive logic and decision making (the theory of internal states).

Let operand σ denote a sentence or a set of sentences, and operand ρ a representation or a set of representations. Transitions of sentences $\sigma_i \models \sigma_j$ and representations $\rho_i \models \rho_j$ have the standard informational form, where

$$(\sigma_i \models \sigma_j) \Longrightarrow (\rho_i \models \rho_j)$$

Other entities are the sentential attitude,

$$\alpha_\sigma(\sigma_1, \ldots, \sigma_n) \rightleftharpoons \begin{pmatrix} \sigma \rightleftharpoons (\sigma_1, \ldots, \sigma_n); \\ \iota \rightleftharpoons (\iota_1, \ldots, \iota_n); \\ \mathfrak{i}_\iota(\mathfrak{c}(\iota) \longleftrightarrow \sigma); \\ \rho_\iota \rightleftharpoons (\rho_1, \ldots, \rho_n); \\ \mathfrak{c}_\iota \rightleftharpoons \mathfrak{c}_\iota(\iota_1), \ldots, \mathfrak{c}_\iota(\iota_n) \end{pmatrix}$$

and the internal state ι. Cognitively relevant internal states can be comprehended as $\iota \rightleftharpoons \iota_1, \ldots, \iota_m$. An internal state ι has, in general, a content $\mathfrak{c}(\iota)$ (or in a short form, $\mathfrak{c}_\iota$), thus, $\mathfrak{c}_\iota \rightleftharpoons \mathfrak{c}_\iota(\iota_1), \ldots, \mathfrak{c}_\iota(\iota_n)$, where content components can be identified via sentences, that is,

$$\mathfrak{c}_\iota(\iota_1) \rightleftharpoons \sigma_1; \ldots; \mathfrak{c}_\iota(\iota_m) \rightleftharpoons \sigma_m$$

In this case, operator '$\rightleftharpoons$' can be replaced by operator $\models_{\text{identically}}$ which reads 'informs identically' (that is, circularly between the $\rightleftharpoons$-involved operands). Such an operator is bidirectional (constituting a loop, graphically), that is,

$$\mathfrak{c}_\iota(\iota_i) \models_{\text{identically}} \sigma_i; \; \sigma_i \models_{\text{identically}} \mathfrak{c}_\iota(\iota_i)$$

The identification, $\mathfrak{i}_\iota(\mathfrak{c}_\iota(\iota_i) \longleftrightarrow \sigma_i)$, is possible by virtue of the isomorphism (operator '$\longleftrightarrow$') between ι_i and σ_i.

Let us discuss the solution for content $\mathfrak{c}_\iota(\sigma)$ on the basis of the graph in Fig. 2, where

$$\mathfrak{c}_\iota(\sigma) \rightleftharpoons$$
$$\begin{pmatrix} \sigma \models \iota; \\ \iota \models \boxed{(}\mathfrak{I}_\iota \models \boxed{(}\mathfrak{i}_\iota \models \boxed{(}\mathfrak{C}_\iota \models \boxed{(}\gamma_\iota \models \\ \quad \boxed{(}\mathfrak{E}_\iota \models \boxed{(}\rho_\iota \models \boxed{(}\mathfrak{c}_\iota \models \iota \boxed{)))))))}; \\ \mathfrak{I}_\iota \models (\mathfrak{i}_\iota \models (\mathfrak{C}_\iota \models (\gamma_\iota \models \mathfrak{I}_\iota))); \\ \boxed{(}(\mathfrak{C}_\iota \models \gamma_\iota) \models \mathfrak{E}_\iota \boxed{) \models^* (}\rho_\iota \models (\mathfrak{c}_\iota \models \mathfrak{C}_\iota)\boxed{)}; \\ \mathfrak{E}_\iota \models (\rho_\iota \models (\mathfrak{c}_\iota \models \mathfrak{E}_\iota)); \\ \boxed{\mathfrak{I}_\iota \models (\mathfrak{i}_\iota \models \mathfrak{I}_\iota); \; \mathfrak{C}_\iota \models (\gamma_\iota \models \mathfrak{C}_\iota);} \\ \mathfrak{E}_\iota \models (\rho_\iota \models \mathfrak{E}_\iota) \end{pmatrix}$$

The same solution is, in the demarked form,

$$\mathfrak{c}_\iota(\sigma) \,.\!\rightleftharpoons\!.$$
$$\begin{array}{l} \sigma \models \iota; \\ \iota \models \boxed{.}\mathfrak{I}_\iota \models \boxed{.}\mathfrak{i}_\iota \models \boxed{.}\mathfrak{C}_\iota \models \boxed{.}\gamma_\iota \models \\ \quad \boxed{.}\mathfrak{E}_\iota \models \boxed{.}\rho_\iota \models \boxed{.}\mathfrak{c}_\iota \models \iota; \\ \mathfrak{I}_\iota \models .\mathfrak{i}_\iota \models .\mathfrak{C}_\iota \models .\gamma_\iota \models \mathfrak{I}_\iota; \\ \mathfrak{C}_\iota \models \gamma_\iota . \models \mathfrak{E}_\iota \boxed{. \models^* .}\rho_\iota \models .\mathfrak{c}_\iota \models \mathfrak{C}_\iota; \\ \mathfrak{E}_\iota \models .\rho_\iota \models .\mathfrak{c}_\iota \models \mathfrak{E}_\iota; \\ \boxed{\mathfrak{I}_\iota \models .\mathfrak{i}_\iota \models \mathfrak{I}_\iota; \; \mathfrak{C}_\iota \models .\gamma_\iota \models \mathfrak{C}_\iota;} \\ \mathfrak{E}_\iota \models .\rho_\iota \models \mathfrak{E}_\iota \end{array}$$

In the third and fourth line, there is the circular formula describing a causal situation of the longest loop. In the remaining rows, systematically all loops are formalized, so that the entire graph in Fig. 2 is systematically covered. The reader can compare the parenthesized, and equivalent to the demarked formula systems, for one particular situation of the graph. In the next section, the demarked formula system will be discussed in detail.

The gestalt of a circular formula is determined by Definition 19. How many formulas does the content gestalt $\Gamma(\mathfrak{c}_\iota(\sigma))$ include? The first row delivers only one formula, $\sigma \models \iota$. The length of the formula in the second and the third row is $\ell_{2,3} = 8$ (the number of the formula's binary operators). Thus, $N_{2,3} = \frac{1}{9}\binom{16}{8} = 1430$. For the next lines there is: $N_4 = \frac{1}{5}\binom{8}{4} = 14, N_5 = \frac{1}{6}\binom{10}{5} = 42, N_6 = \frac{1}{4}\binom{6}{3} = 5, N_7 = \frac{2}{3}\binom{4}{2} = 4$, and $N_8 = \frac{1}{3}\binom{4}{2} = 2$. The sum is 1498 formulas in the gestalt of the discussed formula system.

3 The Role of the Demarcation Point Replacing a Pair of Parentheses in an Informational Formula

Let us introduce the demarcation (delimiting, separating) point [12] in the context of an informational formula replacing a pair of parentheses. Let us list some characteristic examples of parenthesized formulas to get the feeling how the demarking points are uniquely set instead of parentheses pairs in such a way that the parenthesized formula can be reconstructed, uniquely. Let have the following examples for parenthesized and equivalently demarked formulas, respectively:

(α);	α;
$(\alpha) \models (\beta)$;	$\alpha \models \beta$;
$(\alpha \models \beta) \models$;	$\alpha \models \beta . \models$;
$((\alpha) \models (\beta)) \models$;	$\alpha \models \beta . \models$;
$\models (\alpha \models \beta)$;	$\models . \alpha \models \beta$;
$\models ((\alpha) \models (\beta))$;	$\models . \alpha \models \beta$;
$(\alpha) \models (\beta \models \gamma)$;	$\alpha \models . \beta \models \gamma$;
$(\alpha \models \beta) \models (\gamma)$;	$\alpha \models \beta . \models \gamma$;
$\alpha_1 \models (\alpha_2 \models (\alpha_3 \models \alpha_4))$;	$\alpha_1 \models . \alpha_2 \models . \alpha_3 \models \alpha_4$;
$((\alpha_1 \models \alpha_2) \models \alpha_3) \models \alpha_4$;	$\alpha_1 \models \alpha_2 . \models \alpha_3 . \models \alpha_4$;
$(\alpha_1 \models \alpha_2) \models (\alpha_3 \models \alpha_4)$;	$\alpha_1 \models \alpha_2 . \models . \alpha_3 \models \alpha_4$

The question is whether simple rules can be set for the mastering of a unique reading of a demarcated formula. Must it be read from the left to the right or can the reading be performed uniquely also in the opposite direction? In demarked formulas, the direction of formula reading is essential (from the left to the right or vice versa) and must remain the same as the direction in which demarcation points have been set when replacing particular parenthesis pairs. One must keep the same direction in which demarcation points have been set.

The rules of a unique replacement of parenthesis pairs by demarcation points (one point replacing the pair) in an informational formula are the following:

1. Before conversion, all superfluous parenthesis pairs in an informational formula have to be eliminated (crossed out).

2. Any left parentheses, $((\cdots($, at the beginning, as well as any right parentheses, $))\cdots)$, at the end of an informational formula (if any), are ignored and crossed out (within the final result).

3. Each left parenthesis, '('—and to it corresponding right part ')'—staying directly after an informational operator, $\models$ (or a particularized operator), is replaced by the demarcation point, '.'. E.g., $\models$ (____) is replaced by $\models$.____ (with one point only).

4. Each right parenthesis, ')'—and to it corresponding left part '('—staying directly before an informational operator, $\models$ (or a particularized operator), is replaced by the demarcation point, '.'. E.g., (____) $\models$ is replaced by ____. $\models$ (with one point only).

5. All remaining parentheses, if any, with exception of those used in functional forms (e.g., $\varphi(\xi)$), in an informational formula, irrespective of the the left or the right type, are ignored and crossed out.

6. By the procedure following the previous rules, all proper parenthesis pairs in a formula are removed and uniquely replaced by demarcation points.

7. Semicolon ';' and comma ',' are in regard to demarking a formula exceptional operators which remain as they are. By semicolons separated formulas are in parallel informing formulas of a system and the formula parallelism remains preserved in both parenthesized and demarked formulas.

On the other hand, the rules of a unique replacement of demarcation points by parenthesis pairs (one parenthesis pair replacing one demarcation point) in an informational formula are the following:

1. The first point on the left side of operator in a formula is replaced by the parenthesis pair at the beginning of formula (left parenthesis), and at the place of the point (right parenthesis).

2. The first point on the right side of operator is replaced by the parenthesis pair with the left parenthesis '(' at the place of the point and with the right parenthesis ')' at the place of the first point at the left side of operator (lying right of the discussed operator).

3. After parenthesizing by rule 1 or rule 2, the next point is already within a parenthesis pair. Satisfying rule 1, the left parenthesis is set behind the existing left parentheses and the right one at the place of the point. Satisfying rule 2, the right parenthesis is set before the existing right parenthesis and the left one at the place of the point.

4. In the way of rules 1, 2, and 3, all the points can be uniquely replaced by parentheses pairs.

4 The Basic Notion of the Informational Frame within an Informational Entity

Informational frame is a very basic notion which, as a special case, functions between two informational operands irrespective of the parentheses pairs, which are disposed within an informational formula. Informational frame is everything which can be enframed in an informational formula. Enframing means simply to put a frame around certain consequent elements in an informational formula.

In this section we must answer the question what does such a regular, or irregular, structure, named informational frame represent. We shall see how the introduced notion of frame deviates from any other, traditionally fortified constituents of mathematical formulas. This strange notion originates from a fully legal style of formula writing in metamathematics—using the so-called demarcation point (quadratic dot '.')—introduced in Principia mathematica by Whitehead and Russel [12].

An informational frame is a formula or anything which can be put into the frame within a formula. Thus, one has to investigate which characteristic frames can occur within a formula or even a formula system, where the system is comprehended as a complex formula structure.

4.1 Possibilities of Informational Frames

To get an intuitive insight of the possibilities of occurring informational frames we have to look into the formula structure and detach all possible typical frames. This insight can begin with looking into a complex formula and mark or list the characteristic frames. As we shall learn, some frames will be senseful, being informational entities by themselves, the other will be artificial and corresponding to specific views and analyses. In this context harmonious and disharmonious frames will be treated and classified down to necessary details.

In principle, any part of a formula can constitute the so-called informational frame. The problem is how to choose a sufficiently complex formula in which all possible forms of informational frames do occur.

We can begin to write formulas from the left to the right and observe how structurally different frames come into existence. So, let us make a trial beginning with an operand:

$$\begin{array}{ll}
\alpha & \alpha \\
\alpha \models & \alpha \models \\
\alpha \models (& \alpha \models . \\
\alpha \models (\beta & \alpha \models . \beta \\
\alpha \models (\beta \models & \alpha \models . \beta \models \\
\alpha \models (\beta \models (& \alpha \models . \beta \models . \\
\alpha \models (\beta \models (\gamma & \alpha \models . \beta \models . \gamma \\
\vdots & \vdots
\end{array}$$

In the right column, formulas with demarcation points instead of left parentheses are listed.

In a similar manner, we can write formulas from the right to the left and observe how structurally different frames come into existence. So, let us make a trial ending with an operand:

$$\begin{array}{rr}
\alpha & \alpha \\
\models \alpha & \models \alpha \\
) \models \alpha & . \models \alpha \\
\beta) \models \alpha & \beta . \models \alpha \\
\models \beta) \models \alpha & \models \beta . \models \alpha \\
) \models \beta) \models \alpha & . \models \beta . \models \alpha \\
\gamma) \models \beta) \models \alpha & \gamma . \models \beta . \models \alpha \\
\vdots & \vdots
\end{array}$$

Within the listed parenthesized or demarked frames arbitrary parts can be chosen as frames. Some of them are operand frames, that is, informationally regular formulas or subformulas. Other frames are the so-called operator frames, which can be put between two operands, or, in case of parenthesized formulas, at the beginning and the end of formulas or inside of them, representing the so-called parenthesis frames [17].

4.2 Informational Operand Frames

Definition 1 [Operand Frame] *An informational operand frame is nothing else than an enframed informational formula which is either a marker, an autonomous formula, in a formula occurring subformula, or a formula system. An informational operand is, by definition, an informationally well-structured formula of formula system.* □

By inspection, it can be clearly recognized which syntactic structures in a formula perform as operands. And not only this: the semantic character of operands can be recognized to the extent of their decomposition which takes place in a serial extension or in one or more parallel, that is, additional interpretations. Operand frames need not to be treated separately because the notion of the informational formula is fundamental and firmly informationally determined.

4.3 Informational Operator Frames

Informational operator frames are twofold. The most primitive operator frames are informational operators as they are (or appear) by themselves. For a complex and, to some extent unusual, concept of operator frame we introduce a new definition. The sense of the introduction of the notion of the operator frame is to identify a direct informational connection between two operands in an informational formula.

Definition 2 [Operator Frame of a Parenthesized Formula] *Operator frame in a parenthesized formula is the part between arbitrary two operands, and to this operand-intermediate part belonging left-parenthesis and right-parentheses frames.* □

Example 1 [Some Operator Frames of a Parenthesized Formula] Let the metaphysicalistic parenthesized informational formula

$$((((\alpha \models \mathcal{I}_\alpha) \models \iota_\alpha) \models \mathcal{C}_\alpha) \models \gamma_\alpha) \models^* (\mathcal{E}_\alpha \models (\varepsilon_\alpha \models \alpha))$$

be given. Let us have examples of the following three parenthesized frames

$$\boxed{((((}\ \alpha\ \boxed{\begin{array}{l} \models \mathcal{I}_\alpha) \models \iota_\alpha) \models \mathcal{C}_\alpha) \\ \qquad \models \gamma_\alpha) \models^* (\mathcal{E}_\alpha \models (\varepsilon \models \end{array}}\ \alpha\ \boxed{))}\ ;$$

$$\boxed{((((}\ \alpha\ \boxed{\begin{array}{l} \models \mathcal{I}_\alpha) \models \iota_\alpha) \models \mathcal{C}_\alpha) \\ \qquad \models \gamma_\alpha) \models^* (\mathcal{E}_\alpha \models (\end{array}}\ \varepsilon_\alpha \models \alpha\ \boxed{))}\ ;$$

$$(\ \boxed{(((}\ \alpha \models \mathcal{I}_\alpha\ \boxed{) \models \iota_\alpha) \models \mathcal{C}_\alpha) \models}\ \gamma_\alpha) \models^* (\mathcal{E}_\alpha \models (\varepsilon_\alpha \models \alpha))$$

between operands α at the beginning and α at the end, α at the beginning and ε_α, and $\mathcal{I}_\alpha$ and γ_α, respectively. In the first two cases, the operator frame is an entity of three parts, namely, the left, middle and the right one, the third case has a two-part operator frame. □

The last example shows how parenthesized frames can become badly transparent and mutually dependent.

Definition 3 [Operator Frame in a Demarked Formula] *Operator frame in a demarked formula is the part between arbitrary two operands.* □

Example 2 [Some Operator Frames of a Demarked Formula] Let the formula in Example 1 be expressed in the demarked form, that is,

$$\alpha \models \mathcal{I}_\alpha . \models \iota_\alpha . \models \mathcal{C}_\alpha . \models \gamma_\alpha . \models . \mathcal{E}_\alpha \models . \varepsilon_\alpha \models \alpha$$

The enframed examples, discussed in Example 1, become evidently,

$$\alpha \boxed{\begin{array}{l}\models \mathcal{I}_\alpha \cdot\!\models \iota_\alpha \cdot\!\models \mathcal{C}_\alpha \cdot\!\models \\ \qquad \gamma_\alpha \cdot \models\!\cdot \mathcal{E}_\alpha \models\!\cdot \varepsilon_\alpha \models\end{array}} \alpha;$$

$$\alpha \boxed{\begin{array}{l}\models \mathcal{I}_\alpha \cdot\!\models \iota_\alpha \cdot\!\models \mathcal{C}_\alpha \cdot\!\models \\ \qquad \gamma_\alpha \cdot \models\!\cdot\end{array}} \mathcal{E}_\alpha \models\!\cdot \varepsilon \models \alpha;$$

$$\alpha \models \mathcal{I}_\alpha \boxed{\cdot\!\models \iota_\alpha \cdot\!\models \mathcal{C}_\alpha \cdot\!\models} \gamma_\alpha \cdot \models\!\cdot \mathcal{E}_\alpha \models\!\cdot$$
$$\varepsilon_\alpha \models \alpha$$

Each case, irrespective of the operator frame structure (and position), possesses only one demarked frame. □

4.4 Harmonious Informational Frames

We have to determine harmonious and disharmonious parenthesized and demarked informational frames. Which are the various possible frame forms and in which manner do they appear as parenthesized and demarked frames? Which are the advantages of frames in one or the other form?

At the first glance, it seems that harmonious frames always appear in one piece, that is, they are not split within a formula. We shall see how this principle can have different consequences comparing the parenthesized and the demarked frames.

4.4.1 Harmonious Parenthesized Frames

Parenthesized frames in formulas of Example 1 are all harmonious. A frame, although split in two or three parts, is harmonious if it is functionally closed.

Definition 4 [Harmonious Parenthesized Frame] *A split, or unsplit, parenthesized frame is harmonious if it is functionally closed. The functional closeness of a parenthesized frame means that after concatenation of its parts the resulting frame presents a well-formed formula, that is, an operand frame. In the procedure of frame concatenation the empty parenthesized parts of the form () are replaced by an empty part (informational nothing), λ. At such places, the rule $() \leftarrow \lambda$ is applied. Also, the outmost parenthesis pair can be omitted.* □

The concatenation of frame parts in Example 1 gives

$$\boxed{((((} \boxed{\begin{array}{l}\models \mathcal{I}_\alpha) \models \iota_\alpha) \models \mathcal{C}_\alpha) \\ \qquad \models \gamma_\alpha) \models^* (\mathcal{E}_\alpha \models (\varepsilon \models\end{array}} \boxed{))};$$

$$\boxed{((((} \boxed{\begin{array}{l}\models \mathcal{I}_\alpha) \models \iota_\alpha) \models \mathcal{C}_\alpha) \\ \qquad \models \gamma_\alpha) \models^* (\mathcal{E}_\alpha \models (\end{array}} \boxed{))};$$

$$\boxed{(((} \boxed{) \models \iota_\alpha) \models \mathcal{C}_\alpha) \models}$$

which results in three well-formed formulas, that is,

$$((((\models \mathcal{I}_\alpha) \models \iota_\alpha) \models \mathcal{C}_\alpha) \models \gamma_\alpha) \models^* (\mathcal{E}_\alpha \models (\varepsilon \models));$$
$$((((\models \mathcal{I}_\alpha) \models \iota_\alpha) \models \mathcal{C}_\alpha) \models \gamma_\alpha) \models^* (\mathcal{E}_\alpha \models);$$
$$((\models \iota_\alpha) \models \mathcal{C}_\alpha) \models$$

One can see how the concatenated frame-harmonious structures are reduced in respect of the 'goal' entities, so that they preserve the necessary formula well-formedness.

Example 3 [Split Parenthesized Frames] Split parenthesized frames can be harmonious and disharmonious. They are divided in at least two parts and each part of a split harmonious frame is disharmonious. But the parts of a harmonious frame can be concatenated into a unique (well-formed) frame formula. For instance, the split frames in parenthesized formulas

$$\boxed{((} \alpha \boxed{\models \beta) \models \gamma)} \models \delta;$$
$$\boxed{((\alpha \models} \beta \boxed{) \models \gamma)} \models \delta;$$
$$\boxed{((\alpha \models \beta) \models} \gamma \boxed{)} \models \delta$$

are all harmonious. On the other hand, the frame parts are all disharmonious, that is,

$$\boxed{((}\,; \quad \boxed{\models \beta) \models \gamma)}\,; \quad \boxed{((\alpha \models}\,; \quad \boxed{) \models \gamma)}\,;$$
$$\boxed{((\alpha \models \beta) \models}\,; \quad \boxed{)}$$

The reader can recognize the disharmonious structures of the listed frames by himself/herself. □

4.4.2 Harmonious Demarked Frames

What is the difference between a harmonious parenthesized and demarked frame in concern to a frame splitting?

Definition 5 [Harmonious Demarked Frame] *An unsplit demarked frame is harmonious if it is functionally closed. The functional closeness of a demarked frame means that the frame itself presents a demarked well-formed formula.* □

Because the demarcation point replaces the parenthesis pair, the split harmonious frames in case of parenthesized formulas appear as unique frames in cases of demarked formulas. This is quite true for Example 2, where the demarked harmonious frames are

$$\boxed{\begin{array}{l} \models \mathcal{I}_\alpha \mathbin{.}\models \iota_\alpha \mathbin{.}\models \mathcal{C}_\alpha \mathbin{.}\models \\ \qquad \gamma_\alpha \mathbin{.} \models \mathbin{.} \mathcal{E}_\alpha \models \mathbin{.} \varepsilon_\alpha \models \end{array}};$$

$$\boxed{\models \mathcal{I}_\alpha \mathbin{.}\models \iota_\alpha \mathbin{.}\models \mathcal{C}_\alpha \mathbin{.}\models \gamma_\alpha \mathbin{.} \models \mathbin{.}};$$

$$\boxed{\mathbin{.}\models \iota_\alpha \mathbin{.}\models \mathcal{C}_\alpha \mathbin{.}\models}$$

The rightmost operator combination $\mathbin{.}\subset\mathbin{.}$ in the second frame means $\mathbin{.}\subset()$, so, it can be reduced into $\mathbin{.}\subset$ for the sake of simplicity. Similarly, the leftmost operator combination $\mathbin{.}\subset$ in the third frame can be replaced by operator $\subset$ to keep the frame in a common form.

4.4.3 A Syntax Comparison between Harmonious Parenthesized and Harmonious Demarked Frames

How can harmonious parenthesized frames be recognized at once? The answer is: by stating that they represent well-formed formulas. The correct form of a formula can be proved by the usual syntax analysis, taking into account the general, context-free grammar for well-formed informational formulas. It is instructive to determine such grammars for parenthesized and demarked informational formulas.

Designing a syntax, one can consider that a formula or formula system is nothing other than an operand. It means that the initial (starting) grammatical variable in a formula development (generation) is the operand, symbolized grammatically (and as a terminal) by o. A general context-free grammar for the parenthesized formula systems can be constructed by the following items (syntax categories and terminals): o as operand; $\models$ as operator; $\circ$ as a separator in an operator composition; semicolon ';' as the operator of formula parallelism; comma ',' as the operator of alternativeness; and '(' and ')' as parenthesis pair. A preliminary context-free grammar is a construct $G = (N, T, R, o)$, where $N = \{o, \models\}$ is the alphabet of nonterminals, $T = \{\text{'('}, \text{')'}, \circ, \text{';'}, \text{','}, o, \models\}$ denotes the terminal alphabet, R is a set of context-free rules (see below) and o marks the initial symbol. The set of rules is determined by two syntax rules, which are

$$\begin{array}{lcl} o & \leftarrow & o \models \mid\; \models o \mid o \models o \mid (o) \mid o; o \mid o, o \\ \models & \leftarrow & \models \circ \models \mid (\models) \end{array}$$

In case of formula systems using the demarcation points instead of the parenthesis pairs, the demarked grammar is $G^\bullet = (N, T^\bullet, R^\bullet, o)$, the alphabet of terminals is $T^\bullet = \{\text{'.'}\circ, \text{';'}, \text{','}, o, \models\}$ and the rules of $R^\bullet$ are

$$\begin{array}{lcl} o & \leftarrow & o \models \mid\; \models o \mid o \models o \mid \mathbin{.} o \mid o \mathbin{.} \mid o; o \mid o, o \\ \models & \leftarrow & \models \circ \models \mid \mathbin{.} \models \mid\; \models \mathbin{.} \end{array}$$

Rules with demarcation point in the second line (operator composition case) can be used only in such a way that the point is inside of an operator composition. For example, in an operator generation process, there is

$$\begin{array}{c} \models\; \twoheadrightarrow\; \models \circ \models\; \twoheadrightarrow\; \models \mathbin{.}\circ \models\; \twoheadrightarrow\; \models \circ \models \mathbin{.}\circ \models\; \twoheadrightarrow \\ \models \circ \models \mathbin{.}\circ\mathbin{.} \models\; \twoheadrightarrow\; \models \circ \models \mathbin{.}\circ\mathbin{.} \models \circ \models \end{array}$$

where the end result would correspond to the operator composition $(\models \circ \models) \circ (\models \circ \models)$. Symbol $\twoheadrightarrow$ represents the derivation (generation) step.

4.5 Disharmonious Informational Frames

The concept of the disharmonious informational frame (DIF) enables the treatment of arbitrary formula parts which do not fit harmonious informational frames. DIFs, in this way, contribute to the possibility to treat arbitrary parts of formulas as entities which may, in special cases, be of essential interest.

Definition 6 [Disharmonious Frame] *An informational frame is disharmonious if it is a part of a well-formed formula or formula system but it does not represent a well-formed formula by itself.* □

Disharmonious frames are parenthesized incompletely within a parenthesized formula and demarked within a demarked formula.

4.5.1 Disharmonious Parenthesized Frames

In a parenthesized formula, disharmonious parenthesized frames are the most arbitrary (enframed) entities. They can be parts of harmonious frames on one side, on the other side they can embrace any imaginable sequence of adequately serried operands, operators and parentheses, which do not constitute a harmonious frame. So, both types of frames exclude each other.

Definition 7 [Disharmonious Parenthesized Frame] *An informational frame is a disharmonious parenthesized frame if it is a part of a well-formed parenthesized formula or formula system but it does not represent a well-formed parenthesized formula by itself.* □

By this definition, a disharmonious parenthesized frame is not an arbitrary sequence of operands, operators and parentheses, but is an arbitrary sequence of the mentioned entities which constitute a part of a well-formed parenthesized formula or formula system. For example, frames $\boxed{(}$, $\boxed{)}$, $\boxed{(\models}$, $\boxed{\models)}$, $\boxed{\models(\models}$, etc. are disharmonious parenthesized frames because they can be completed to the harmonious parenthesized frames.

4.5.2 Disharmonious Demarked Frames

In a demarked formula, disharmonious demarked frames are enframed entities being arbitrary parts of the formula. They can represent parts of demarked harmonious frames and, in this way, can include any imaginable sequence of adequately composed operands, operators and demarcation points, which in this sequence appear in a harmonious frame. Demarked harmonious, and demarked disharmonious frames, exclude each other.

Definition 8 [Disharmonious Demarked Frame] *An informational frame is a disharmonious demarked frame if it is a part of a well-formed demarked formula or formula system but it does not represent a well-formed demarked formula by itself.* □

By this definition, a disharmonious demarked frame is not an arbitrary sequence of operands, operators and demarcation points, but is an arbitrary sequence of the mentioned entities which constitute a part of a well-formed demarked formula or formula system. E.g., frames $\boxed{\cdot}$, $\boxed{\cdot\models}$, $\boxed{\models\cdot}$, $\boxed{\models\cdot\models}$, $\boxed{\cdot\models\cdot}$, etc. are disharmonious demarked frames because they can be completed to the harmonious demarked frames.

4.5.3 A Comparison between Disharmonious Parenthesized and Disharmonious Demarked Frames

The comparison between disharmonious parenthesized and disharmonious demarked frames concerns the so-called of the frame's left and right edge development (see Subsubsection 5.6.1 and Table 1). In designing a frame (harmonious as well as disharmonious one), the designer (designing entity) proceeds from that part of the frame which already exists, developing the frame at its edges in such a way that the emerging disharmonious frame will become a part of a possible harmonious frame or of well-formed formula. Syntax rules for generation of disharmonious parenthesized and disharmonious demarked frames differ, of course, essentially between the parenthesized and the demarked case.

4.6 Functional Frames

Functional frames are characteristic in such a way that they can be clearly recognized in comparison with other formula frames. Informational function as an informational operand has the form $\varphi(\alpha)$, where the left parenthesis appears between two operands (the only possible case for such an appearance), and the right parenthesis appears at the end of argument α. There are not demarked functional frames, but within a function and its argument formulas can be expressed in the demarked form.

4.6.1 Harmonious Functional Frames

Functional frames are parenthesized structures which occur as such (roughly parenthesized) also in cases using demarcation points.

Definition 9 [Harmonious Functional Frame] *Harmonious functional frames have the general forms*

$$\boxed{(\varphi)(\alpha)},\ \boxed{(\varphi)}\boxed{(\alpha)},\ (\boxed{\varphi})(\boxed{\alpha}),\ (\varphi)\boxed{(\alpha)},$$
$$(\varphi)(\boxed{\alpha}),\ \boxed{(\varphi)}(\alpha),\ (\boxed{\varphi})(\alpha),\ \boxed{\boxed{(\boxed{\varphi})}\boxed{(\boxed{\alpha})}}$$

where, in principle, both φ and α are complex informational formulas, so, they must be parenthesized as (φ) and (α), respectively. A common form of informational functionalism is $\varphi(\alpha)$, where φ stands for a marker (a single operand name). Thus,

$$\boxed{\varphi(\alpha)},\ \boxed{\varphi(\boxed{\alpha})},\ \text{and}\ \boxed{\varphi\boxed{(\alpha)}}$$

where α is an argument formula and φ is a function representative informing upon the argument [17]. □

The following comment could be useful: frames $\boxed{\alpha}$ and $\boxed{(\alpha)}$ are harmonious and can appear as such also outside a functional context. On the other side, the possibility of framing both the function entity φ, and the function argument α, makes them visible for further informational investigation.

4.6.2 Disharmonious Functional Frames

A disharmonious functional frame offers an especially characteristically visible case which explicitly concerns the syntactic structure pertaining solely to the concept of informational function.

Definition 10 [Disharmonious Functional Frame] *Characteristic disharmonious functional frames take the general forms as*

$$\boxed{)(},\ \boxed{\varphi)(},\ \boxed{\varphi)(\alpha},\ \boxed{)(\alpha},\ \boxed{(\varphi)(},\ \boxed{\models)(},\ \boxed{)(\models},$$
$$\boxed{\models)(\models},\ \boxed{\varphi)(\models},\ \boxed{\varphi)(\models (},\ \boxed{)\models)(\models (}$$

etc. □

As stressed, combination ')(' is the most significant mark of the presence of an informational function which can be searched in the left and the right direction of the mark.

4.7 Frames Concerning the Operator Composition

Operator composition belongs to distinguished operator structure. It enables to join several operators systematically into one resulting operator. Similarly, as a function, operator composition can be treated as an autonomous framing problem. Harmonious and disharmonious framings of operator compositions can be studied and clarified.

4.7.1 Harmonious Frames for Operator Compositions

An operator composition is a structure consisting of operators $\models$ (differently particularized), operator separators '$\circ$' (uniquely determined) and parenthesis pairs.

Definition 11 [Harmonious Parenthesized Frame of Operator Composition] *Harmonious parenthesized frames of an operator composition can take the general forms*

$$\boxed{\models\circ\models},\ \boxed{(\models\circ\models)},\ \boxed{(\models\circ\models)\circ\models},\ \boxed{\models\circ(\models\circ\models)}$$

etc. They must satisfy the operator syntax described in Subsection 4.4.3. □

In a similar way, frames for the demarked operator compositions can be determined.

Definition 12 [Harmonious Demarked Frame of Operator Composition] *Harmonious demarked frames of an operator composition can take the general forms*

$$\boxed{\models\circ\models},\ \boxed{\models\circ .\models\circ\models},\ \boxed{\models\circ\models .\circ\models},$$
$$\boxed{\models\circ\models .\circ . \models\circ\models}$$

etc. They must satisfy the operator syntax described in Subsection 4.4.3. □

In a syntactically regular way, harmonious operator composition frames can be arbitrarily adequately nested.

4.7.2 Disharmonious Frames for Operator Compositions

Which are the main forms of disharmonious parenthesized and demarked operator composition frames? At least, a harmonious operator composition frame must begin and end by an operator $\models$ (which, obviously, follows from the previous definition). A general answer is given in the form of the following two definitions for a parenthesized and demarked case, respectively.

Definition 13 [Disharmonious Parenthesized Frame of Operator Composition] *Disharmonious parenthesized frames of an operator composition can take the general forms*

$$\boxed{\circ},\ \boxed{)\circ},\ \boxed{\circ(},\ \boxed{)\circ(},\ \boxed{\models)\circ},\ \boxed{\circ(\models},$$
$$\boxed{\models)\circ(\models},\ \boxed{\circ\models)\circ(\models\circ},\ \boxed{\circ\models)\circ(\models\circ\models}$$

etc. □

On the other hand, frames of disharmonious demarked operator compositions are determined in the following manner.

Definition 14 [Disharmonious Demarked Frame of Operator Composition] *Disharmonious demarked frames of an operator composition can take the general forms*

$$\boxed{\circ},\ \boxed{.\circ},\ \boxed{\circ .},\ \boxed{.\circ .},\ \boxed{\circ\models .\circ},\ \boxed{\circ .\models\circ},$$
$$\boxed{\circ\models .\circ .\models\circ},\ \boxed{\models\circ\models .\circ .\models\circ},$$
$$\boxed{\circ\models .\circ .\models\circ\models .}$$

etc. □

5 A Frame-analytical Comprehension of Informational Transition and Possible Frame Concatenation

5.1 Operator Frame

Informational operator is that entity which appears between two informational operands, forming the so-called basic informational transition, irrespective of the complexity of operands. On the other hand, the operator can be understood as an arbitrarily complex entity, composed of many other entities, that is, operators, operands, and parenthesis pairs, which do not constitute a well-formed formula. In such a sense, operator can be

(a)

Left operand	Operator	Right operand

(b)

Left operand	Left operator	Separator	Right operator	Right operand

Figure 3: *Frame sequences for transition of type* (a) $\alpha \models \beta$ *and* (b) $\alpha \models\circ\models \beta$.

comprehended as a the most essential informational frame—the operator frame—which has to be studied carefully, and exhaustively.

Informational transition, of the form (a) $\alpha \models \beta$, or, operator-compositionally, in the form (b) $\alpha \models\circ\models \beta$ can be comprehended by the general schemes in Fig. 3, respectively. How can informational frames be consistently (well-formedly) joined together?

Let us start with the following frame structures concerning the transition $\alpha \models \beta$. There are, evidently, some possible frame configurations:

$$\alpha\,\boxed{\models}\,\beta; \qquad \alpha\,\boxed{\models_\alpha\circ\models_\beta}\,\beta;$$
$$\alpha\,\boxed{\models_\alpha}\,\boxed{\circ}\,\boxed{\models_\beta}\,\beta; \qquad \boxed{\alpha\models_\alpha}\,\circ\,\boxed{\models_\beta\beta};$$
$$\boxed{\alpha\models_\alpha}\,\boxed{\circ}\,\boxed{\models_\beta\beta}; \qquad \boxed{\alpha}\,\boxed{\models_\alpha}\,\boxed{\circ}\,\boxed{\models_\beta}\,\boxed{\beta}$$

Let us study particular framings of informational transition.

5.2 Frame Concatenation and Frame Parallelism

Under certain circumstances, informational frames can be concatenated into new frames and can be stacked in a parallel manner within frames and, then, the stacked frames concatenated, etc.

Definition 15 [Frame Concatenation] *Two frames ϕ_α, and ϕ_β, can be concatenated building up a frame ϕ, that is, $\phi \rightleftharpoons \phi_\alpha\phi_\beta$, if both ϕ_α, and ϕ_β, are parts of a well-formed formula, and such is ϕ.* □

Definition 16 [Frame Parallelism and Concatenation] *Frames ϕ_α and ϕ_β, being arrays of frames*

$\phi_{\alpha,1}, \phi_{\alpha,2}, \cdots, \phi_{\alpha,k}$ and $\phi_{\beta,1}, \phi_{\beta,2}, \cdots, \phi_{\beta,k}$, respectively, can be concatenated into frame array ϕ with frame components $\phi_1, \phi_2, \cdots, \phi_k$ in such a way, that $\phi \rightleftharpoons \phi_\alpha \phi_\beta$, where

$$\boxed{\begin{array}{c}\boxed{\phi_1}\\ \boxed{\phi_2}\\ \vdots\\ \boxed{\phi_k}\end{array}} \rightleftharpoons \boxed{\begin{array}{c}\boxed{\phi_{\alpha,1}}\\ \boxed{\phi_{\alpha,2}}\\ \vdots\\ \boxed{\phi_{\alpha,k}}\end{array}}\boxed{\begin{array}{c}\boxed{\phi_{\beta,1}}\\ \boxed{\phi_{\beta,2}}\\ \vdots\\ \boxed{\phi_{\beta,k}}\end{array}} \quad \text{that is,} \quad \boxed{\begin{array}{cc}\boxed{\phi_{\alpha,1}}&\boxed{\phi_{\beta,1}}\\ \boxed{\phi_{\alpha,2}}&\boxed{\phi_{\beta,2}}\\ \vdots & \\ \boxed{\phi_{\alpha,k}}&\boxed{\phi_{\beta,k}}\end{array}}$$

(and $\phi_i \rightleftharpoons \phi_{\alpha,i}\phi_{\beta,i}$ for $i = 1, 2, \cdots, k$). □

5.3 Framing the Transition $\alpha \models \beta$

In the framed transition $\alpha \boxed{\models} \beta$, the question what could an operator $\models$ represent, and which is the degree of its complexity in a serial and parallel sense, comes to the surface. Initially, $\models$ is an arbitrarily structured operator and its structure has now to be clarified.

Evidently, the serial parenthesized decomposition of a transition $\alpha \boxed{\models} \beta$ can have the form

$$\boxed{(\cdots((}\ \alpha\ \boxed{\begin{array}{l}\models \alpha_1) \models \alpha_2) \models \cdots \alpha_m) \models^* \\ (\beta_1 \models (\beta_2 \models \cdots (\beta_n \models\end{array}}\ \beta\ \boxed{)\cdots))}$$

The complex operator frame is split into three parts and one can understand how a symbol $\models$ between operands can become as complex as possible. Operator $\models^*$ is the main operator and signals the transition decomposition process is never ended. Simultaneously, it marks, how the α-part of length $\ell_\alpha = m$ of the operator belongs to α and how the β-part of length $\ell_\beta = n$ belongs to β. Thus, the length of decomposed transition is $\ell_{\alpha\models\beta} = m+n+1$. The transition gestalt includes (see Subsection 6.3)

$$N_{\alpha\models\beta} = \frac{1}{m+n+2}\binom{2m+2n+2}{m+n+1}$$

possible decompositions of transition $\alpha \models \beta$ of length $\ell_\alpha + \ell_\beta + 1$.

The demarked form of the discussed operator decomposition of transition has a compact shape, with only one (unsplit) frame, that is,

$$\alpha\ \boxed{\begin{array}{l}\models \alpha_1 \cdot \models \alpha_2 \cdot \models \cdots \alpha_m \cdot \models \cdot \\ \beta_1 \models \cdot \beta_2 \models \cdot \cdots \models \cdot \beta_n \models\end{array}}\ \beta$$

where the care of parenthesis pairs is left over the the mechanism of the demarcation point.

The next enframing shows a clear separation between the left, and the right, part of transition $\alpha \models \beta$ in regard of the main operator $\models^*$. There is

$$\boxed{\boxed{(\cdots((}_{\phi_{\alpha,l}}\ \alpha\ \boxed{\begin{array}{l}\models \alpha_1) \models \alpha_2)\\ \models \cdots \alpha_m)\end{array}}_{\phi_{\alpha,r}}}_{\phi_\alpha} \quad \models^*$$
$$\boxed{\boxed{\begin{array}{l}(\beta_1 \models (\beta_2 \models\\ \cdots (\beta_n \models\end{array}}_{\phi_{\beta,l}}\ \beta\ \boxed{)\cdots))}_{\phi_{\beta,r}}}_{\phi_\beta}$$

where ϕ_α is a harmonious left frame and ϕ_β a harmonious right frame. Within these frames, frame pairs and $(\phi_{\alpha,l}, \phi_{\alpha,r})$ and $(\phi_{\beta,l}, \phi_{\beta,r})$ are disharmonious. Additionally, the main operator does not belong explicitly either to operand α or to operand β. It stays between both of them and can be decomposed in a further way in the left, and the right, direction.

The demarked case of the discussed example brings a clear evidence how the informer, and the observer, part in a transition can be separated. There is

$$\boxed{\alpha\ \boxed{\begin{array}{l}\models \alpha_1 \cdot \models \alpha_2\\ \cdot \models \cdots \alpha_m\end{array}}}_{\phi_\alpha} \cdot \models \cdot \boxed{\boxed{\begin{array}{l}\beta_1 \models \cdot \beta_2 \models \cdot\\ \cdots \models \cdot \beta_n \models\end{array}}\ \beta}_{\phi_\beta}$$

Certainly, there can exist several different, that is parallel, decompositions of transition $\alpha \models \beta$. In this case, instead of a single operator enframed formula, there are, say, k parallel formulas of the form

$$\boxed{(\cdots((}\ \alpha\ \boxed{\begin{array}{l}\models \alpha_{11}) \models \alpha_{12}) \models \cdots \alpha_{1m})\\ \models^*\\ (\beta_{11} \models (\beta_{12} \models \cdots (\beta_{1n} \models\end{array}}\ \beta\ \boxed{)\cdots))};$$
$$\boxed{(\cdots((}\ \alpha\ \boxed{\begin{array}{l}\models \alpha_{21}) \models \alpha_{22}) \models \cdots \alpha_{2m})\\ \models^*\\ (\beta_{21} \models (\beta_{22} \models \cdots (\beta_{2n} \models\end{array}}\ \beta\ \boxed{)\cdots))};$$
$$\vdots$$
$$\boxed{(\cdots((}\ \alpha\ \boxed{\begin{array}{l}\models \alpha_{k1}) \models \alpha_{k2}) \models \cdots \alpha_{km})\\ \models^*\\ (\beta_{k1} \models (\beta_{k2} \models \cdots (\beta_{kn} \models\end{array}}\ \beta\ \boxed{)\cdots))}$$

which results into a compact interpretation of the transition decomposition by

$$\boxed{\begin{array}{l}\boxed{(\cdots((}\\ \boxed{(\cdots((}\\ \vdots\\ \boxed{(\cdots((}\end{array}}\ \alpha\ \boxed{\begin{array}{l}\boxed{\begin{array}{l}\models\alpha_{11})\models\alpha_{12})\models\cdots\\ \alpha_{1m})\models^*(\beta_{11}\\ \models(\beta_{12}\models\cdots(\beta_{1n}\models\end{array}}\\ \boxed{\begin{array}{l}\models\alpha_{21})\models\alpha_{22})\models\cdots\\ \alpha_{2m})\models^*(\beta_{21}\\ \models(\beta_{22}\models\cdots(\beta_{2n}\models\end{array}}\\ \vdots\\ \boxed{\begin{array}{l}\models\alpha_{k1})\models\alpha_{k2})\models\cdots\\ \alpha_{km})\models^*(\beta_{k1}\\ \models(\beta_{k2}\models\cdots(\beta_{kn}\models\end{array}}\end{array}}\ \beta\ \boxed{\begin{array}{l}\boxed{)\cdots))}\\ \boxed{)\cdots))}\\ \vdots\\ \boxed{)\cdots))}\end{array}}$$

The reader can comprehend in which sense the complexity of a transition operator $\models$ can develop. In the last enframing example the split parts of a parallel decomposed transition operator are separately enframed, so that operands α and β appear only once, like in an informational graph. In the last complex parallel case, the transition gestalt includes (see Subsection 6.3)

$$N^{\|}_{\alpha\models\beta}=\sum_{i=1}^{k}\frac{1}{m_i+n_i+2}\binom{2m_i+2n_i+2}{m_i+n_i+1}$$

possible decompositions of transition $\alpha\models\beta$ of parallel lengths $\ell_{\alpha,i}+\ell_{\beta,i}+1$ for $i=1,2,\ldots,k$.

As already ascertained, the demarked form does not need an explanation concerning the parallel frames within a frame. The demarked formula system for multiple operator (e.g. interpretative) decomposed transition is (without serially split frames)

$$\alpha\ \boxed{\begin{array}{l}\boxed{\begin{array}{l}\models\alpha_{11}\centerdot\models\alpha_{12}\centerdot\models\cdots\\ \alpha_{1m}\centerdot\models\centerdot\beta_{11}\\ \models\centerdot\beta_{12}\models\centerdot\cdots\beta_{1n}\models\end{array}}_{\phi_1}\\ \boxed{\begin{array}{l}\models\alpha_{21}\centerdot\models\alpha_{22}\centerdot\models\cdots\\ \alpha_{2m}\centerdot\models\centerdot\beta_{21}\\ \models\centerdot\beta_{22}\models\centerdot\cdots\beta_{2n}\models\end{array}}_{\phi_2}\\ \vdots\\ \boxed{\begin{array}{l}\models\alpha_{k1}\centerdot\models\alpha_{k2}\centerdot\models\cdots\\ \alpha_{km}\centerdot\models\centerdot\beta_{k1}\\ \models\centerdot\beta_{k2}\models\centerdot\cdots\beta_{kn}\models\end{array}}_{\phi_k}\end{array}}\ \beta$$

In a compact, however virtually artificial way, the separation into the left and the right part frames can be expressed in the form

$$\boxed{\begin{array}{l}\boxed{(\cdots((}_{\phi^1_{\alpha,l}}\\ \boxed{(\cdots((}_{\phi^2_{\alpha,l}}\\ \vdots\\ \boxed{(\cdots((}_{\phi^k_{\alpha,l}}\end{array}}_{\phi_{\alpha,l}}\ \alpha\ \boxed{\begin{array}{l}\boxed{\begin{array}{l}\models\alpha_{11})\models\alpha_{12})\\ \models\cdots\alpha_{1m})\end{array}}_{\phi^1_{\alpha,r}}\\ \boxed{\begin{array}{l}\models\alpha_{21})\models\alpha_{22})\\ \models\cdots\alpha_{2m})\end{array}}_{\phi^2_{\alpha,r}}\\ \vdots\\ \boxed{\begin{array}{l}\models\alpha_{k1})\models\alpha_{k2})\\ \models\cdots\alpha_{km})\end{array}}_{\phi^k_{\alpha,r}}\end{array}}_{\phi_{\alpha,r}}\ \boxed{\begin{array}{c}\models^*_1\\ \models^*_2\\ \vdots\\ \models^*_k\end{array}}$$

$$\boxed{\begin{array}{l}\boxed{\begin{array}{l}(\beta_{11}\models(\beta_{12}\models\\ \cdots(\beta_{1n}\models\end{array}}_{\phi^1_{\beta,l}}\\ \boxed{\begin{array}{l}(\beta_{21}\models(\beta_{22}\models\\ \cdots(\beta_{2n}\models\end{array}}_{\phi^2_{\beta,l}}\\ \vdots\\ \boxed{\begin{array}{l}(\beta_{k1}\models(\beta_{k2}\models\\ \cdots(\beta_{kn}\models\end{array}}_{\phi^k_{\beta,l}}\end{array}}_{\phi_{\beta,l}}\ \beta\ \boxed{\begin{array}{l}\boxed{)\cdots))}_{\phi^1_{\beta,r}}\\ \boxed{)\cdots))}_{\phi^2_{\beta,r}}\\ \vdots\\ \boxed{)\cdots))}_{\phi^k_{\beta,r}}\end{array}}_{\phi_{\beta,r}}$$

The reader can find the demarked form of the last formula system in Section 11 and Subsections 5.4 and 5.5 where the philosophy around transition $\alpha\models\circ\models\beta$ is debated. A kind of the system simplification will become evident.

5.4 Framing the Transition $\alpha\models\circ\models\beta$

At the first look, there is a minimal difference between the presentation of transition decomposition $\alpha\models\beta$ and $\alpha\models\circ\models\beta$. However, as it is pointed out in Subsection 5.5, the difference is essential, because in the second case the informer part (α) and the observer part (β) can be separated up to the operator composition separator '$\circ$'.

In general, for the parenthesized case, the split, composed operator ($\models\circ\models$) frame example is

$$\boxed{(\cdots((}\ \alpha\ \boxed{\begin{array}{l}\models\alpha_1)\models\alpha_2)\models\cdots\alpha_m)\models\\ \circ\\ \models(\beta_1\models(\beta_2\models\cdots(\beta_n\models\end{array}}\ \beta\ \boxed{)\cdots))}$$

The demarked form of the same formula becomes

$$\alpha\ \boxed{\begin{array}{l}\models\alpha_1\centerdot\models\alpha_2\centerdot\models\cdots\alpha_m\centerdot\models\\ \circ\\ \models\centerdot\beta_1\models\centerdot\beta_2\models\centerdot\cdots\models\centerdot\beta_n\models\end{array}}\ \beta$$

The next two examples of possible enframing come near to the goal of separation between the informer and the observer. In case of parenthesized formula, one obtains the enframing

$$\boxed{\boxed{(\cdots((}_{\phi_{\alpha,l}}\ \alpha\ \boxed{\begin{matrix}\models \alpha_1) \models \alpha_2)\\ \models \cdots\alpha_m) \models\end{matrix}}_{\phi_{\alpha,r}}}_{\phi_\alpha} \circ$$

$$\boxed{\boxed{\begin{matrix}\models (\beta_1 \models (\beta_2 \models\\ \cdots(\beta_n \models\end{matrix}}_{\phi_{\beta,l}}\ \beta\ \boxed{)\cdots))}_{\phi_{\beta,r}}}_{\phi_\beta}$$

and in case of demarked formula the enframing

$$\boxed{\alpha\ \boxed{\begin{matrix}\models \alpha_1 \mathbin{\centerdot} \models \alpha_2\\ \mathbin{\centerdot} \models \cdots\alpha_m \mathbin{\centerdot} \models\end{matrix}}}_{\phi_\alpha} \circ \boxed{\boxed{\begin{matrix}\models \mathbin{\centerdot}\beta_1 \models \mathbin{\centerdot}\beta_2 \models \mathbin{\centerdot}\\ \cdots \models \mathbin{\centerdot}\beta_n \models\end{matrix}}\ \beta}_{\phi_\beta}$$

comes into the separation foreground.

Further examples, discussed in Subsection 5.3, can easily be constructed for the $\alpha \models\circ\models \beta$ case.

5.5 Interpreting the Transition Framing $\boxed{\alpha \models_\alpha} \circ \boxed{\models_\beta \beta}$

Framing of the form $\boxed{\alpha \models_\alpha} \circ \boxed{\models_\beta \beta}$ is essential for a proper understanding and informational regularity of the separator '$\circ$', functioning as a regular operator. Evidently,

$$(\boxed{\alpha \models_\alpha} \circ \boxed{\models_\beta \beta}) \equiv ((\alpha \models_\alpha) \circ (\models_\beta \beta))$$

In this formula, subformulas $(\alpha \models_\alpha)$ and $(\models_\beta \beta)$ are well-formed formulas, and between them an informational operator, that is, also, operator '$\circ$', can appear. In this way, separator '$\circ$' is a regular informational operator. On the other hand, in α $\boxed{\models_\alpha \circ \models_\beta}$ β, entity '$\circ$' is a member of the operator composition $\models_\alpha \circ \models_\beta$, that is, the separator between operators $\models_\alpha$ and $\models_\beta$. In both cases, the meaning of '$\circ$' is a sort of informational concatenation between the informing operand α and by it informed operand β.

5.6 A Consistent Concatenation of Frames in Complex Transition Formulas

How can arbitrary frames be linked together to keep the possibility that a final frame concatenation will represent a well-formed formula? A particular question concerns the frame concatenation which would lead to an operator frame between arbitrary two operands in a serial (or serially circular) formula. What are the characteristics of such an disharmonious (irregular, non-well-formed) operator frame?

5.6.1 A Conditional Frame Edge Syntax

How can frames be composed beginning from an initial frame $\aleph$? Let the initial frame $\aleph$ be replaced by a single basic alternative frame which is nothing else than a general symbol appearing in a well-formed formula. Thus, we introduce the initial replacement rule in the form

$$\aleph \leftarrow \alpha \mid \models \mid \circ \mid (\mid) \mid , \mid ;$$

where α represents all possible operands, $\models$ all possible operators, $\circ$ composition operator for operators. We can also introduce operand ϕ as the current occurrence of a frame which is still arising or is already arisen by the use of syntax rules. These rules can be used only at the left and the right edge of arising ϕ in such a way, that the final result of different frame concatenation delivers, at the end, a well-formed informational formula, after an arbitrary initial rule $\aleph \leftarrow \phi$ was chosen. A complete collection of frame edge syntax rules is listed in Table 1.

In this table, the aesthetical (obligatory) space symbol, ␣, is introduced, which explicitly marks the usual space between formula components. As one can see, there are seven syntactic types of symbols as given by the $\aleph$-rule. The use of rules from Table 1 is conditional. The condition which must be satisfied at the generation of frame ϕ is that ϕ is a syntactically correct part of the arising informational formula. Thus, additional conditions in the form of context dependent rules can be constructed in the following way:

$$\circ\models \leftarrow \models\circ\models \mid \models)\circ\models$$

We must not forget that the rules can be applied only at the edge of the current frame ϕ.

5.6.2 Explanations Concerning the Use of Rules in Table 1

Explanations concerning the use of concrete rules in Table 1 is necessary. As said at the very beginning of this section, the application of concrete

#	0		1	2	3	4	5	6	7	8	9	10	11	12
0	$\aleph$	$\leftarrow$	α	$\models$	$\circ$	(	)	,	;					
1	α	$\leftarrow$	$\alpha\models$	$\models\alpha$	$(\alpha$	$\alpha)$	$\alpha,$	$,\sqcup\alpha$	$\alpha;$	$;\sqcup\alpha$	$\alpha($	$\alpha\sqcup($	$)\alpha$	$)\sqcup\alpha$
2	$\models$	$\leftarrow$	$\alpha\models$	$\models\alpha$	$(\models$	$\models)$	$\models,$	$,\sqcup\models$	$\models;$	$;\sqcup\models$	$\models($	$)\models$	$\models\circ$	$\circ\models$
3	$\circ$	$\leftarrow$	$\circ\models$	$\models\circ$	$\circ\models)$	$(\models\circ$	$\circ($	$)\circ$	$\alpha\models\circ$	$\circ\models\alpha$	$\alpha\sqcup(\models\circ$	$\circ\models)\sqcup\alpha$	$)\circ\models$	
4	(	$\leftarrow$	$(\alpha$	$\alpha($	$(\models$	$\models($	$)($	$,\sqcup($	$;\sqcup\models$	((	$\circ($	$\alpha\sqcup($		
5	)	$\leftarrow$	$\alpha)$	$)\alpha$	$\models)$	$)\models$	$)($	$),$	$);$	))	$)\circ$	$)\sqcup\alpha$		
6	,	$\leftarrow$	$,\sqcup\alpha)$	$\alpha,$	$,\sqcup\models$	$\models,$	$,\sqcup($	$),$						
7	;	$\leftarrow$	$;\sqcup\alpha)$	$\alpha;$	$;\sqcup\models$	$\models;$	$;\sqcup($	$);$						

Table 1: *A conditional frame-edge context-free syntax in which some meanings are the following:* $\leftarrow$ *reads* is_replaced_by; *for example,* $(1:0\leftarrow 1)$ *marks* $\alpha\leftarrow\alpha\models$, *or* $(4:0\leftarrow 4)$ *marks* "$(\leftarrow\ \models($"; *the visible space symbol marks an aesthetical (also obligatory) space between some elements of a formula.*

rules is conditional: the arising frame ϕ must be a part of a well-formed formula. We will refer to particular rules by (x,z) markers, for example, $(2,12)$ $\models\leftarrow\circ\models$. The most interesting cases are those in which for a given edge symbol several rules for this symbol could be applied, but only one (or several) can generate a well-formed formula. The use of a certain rule depends on the context on the right or the left side of the edge symbol.

Additional explanations to the use of some nonevident rules (replacements) for the edge symbols are given in Table 2. The application of rules is conditional in the sense that the result of a symbol replacement must stay within the well-formed formula. The preceding (already existing) context and the intention of a formula development determine the choice of a concrete rule.

5.6.3 Examples: the Application of Conditional Frame Edge Syntax

Let us show several examples of the discussed frame syntax. This syntax enables a straightforward generation of frames from the left to the right and vice versa, but also from wherever in the middle of an arising formula and then proceeding on its left and its right side. Within such a frame generation the way to the well-formed formula as the final result must be considered, that is, the design of an adequate (syntactically correct) frame ϕ.

We can distinguish two characteristic cases of the design by the conditional frame syntax. In the first case, we are confronted with the formation of a complex operator composition, for which certain conditions of generation have to be satisfied. In the second case we discuss a general case and point out the conditions where syntax rules could violate the emerging of a well-formed formula.

Example 4 [Generation of a Complex Operator Composition] An operator composition can be begun by several rules of Table 1. The beginning symbol is $\aleph$ and rules $\aleph\leftarrow\models$ and $\aleph\leftarrow\circ$ are both adequate. The used rules can be marked by $(x:y\leftarrow z)$ where x is the line number, and y and z are the column numbers in Table 1. The shortened marker is simply (x,z) and marks uniquely the rule of the table. Thus, we will use the marked deduction arrow of the form $\xmapsto{x,z}$. In this way, the final form of a frame ϕ can be generated in the following way:

$$\aleph \xmapsto{0,2} \models \xmapsto{2,11} \models\circ \xmapsto{2,12} \models\circ\models \xmapsto{2,3} (\models\circ\models \xmapsto{4,8} \\ ((\models\circ\models \xmapsto{2,4} ((\models\circ\models) \xmapsto{5,9} ((\models\circ\models)\circ \xmapsto{3,1} \\ ((\models\circ\models)\circ\models \xmapsto{2,4} ((\models\circ\models)\circ\models) \xmapsto{5,9} \\ ((\models\circ\models)\circ\models)\circ \xmapsto{3,1} ((\models\circ\models)\circ\models)\circ\models \xmapsto{2,2} \\ ((\models\circ\models)\circ\models)\circ\models\alpha \xmapsto{4,10} \alpha\,((\models\circ\models)\circ\models)\circ\models\alpha$$

with the current frame ϕ at the end of the deduction chain. □

Example 5 [General Formula Generation Violating the Formula Syntax] How, by using the rules, the well-formedness of the emerging formula can be violated? Somebody being acquainted or having the feeling of formula well-formedness can immediately sense the mentioned violation. According to the Table 1, the following illegal (syntactically incorrect) frames (derivation results or their parts) can be generated:

(x,z)	Rule	Explanation
$(1,3)$	$\alpha \leftarrow (\alpha$	α begins a function argument formula or a regular serial subformula
$(1,4)$	$\alpha \leftarrow \alpha)$	α ends a function argument formula or a regular serial subformula
$(1,9)$	$\alpha \leftarrow \alpha($	α is a function of that which will follow after '(' to the corresponding ')'
$(1,10)$	$\alpha \leftarrow \alpha_\sqcup($	α with '$\sqcup$' before '(' marks the beginning of an operator composition
$(1,11)$	$\alpha \leftarrow)\alpha$	α is a simple argument (marker) of a function arising before ')'
$(1,12)$	$\alpha \leftarrow)_\sqcup\alpha$	α marks the beginning of a subformula after an operator composition
$(2,11)$	$\models \leftarrow \models\circ$	$\models$ is the left of an operator composition
$(2,12)$	$\models \leftarrow \models\circ$	$\models$ is the right of an operator composition
$(3,3)$	$\circ \leftarrow \circ\models)$	')' marks the end of a complex operator composition
$(3,4)$	$\circ \leftarrow (\models\circ$	'(' marks the beginning of a complex operator composition
$(3,5)$	$\circ \leftarrow \circ($	after '(' an operator composition will follow
$(3,6)$	$\circ \leftarrow)\circ$	before ')' an operator composition will appear
$(3,9)$	$\circ \leftarrow \alpha_\sqcup(\models\circ$	after α, a more complex operator composition can follow by '$\circ \leftarrow \circ($'
$(3,10)$	$\circ \leftarrow \circ\models)_\sqcup\alpha$	before α, a more complex operator composition can follow by '$\circ \leftarrow)\circ$'
$(3,11)$	$\circ \leftarrow)\circ\models$	before ')', an arbitrarily complex operator composition can appear
$(4,1)$	$(\leftarrow (\alpha$	α begins a subformula or a function argument in the context $)(\alpha$
$(4,2)$	$(\leftarrow \alpha($	α is a single marker of a function after '(', ')', or $\models$
$(4,5)$	$(\leftarrow)($	$)($ marks the concatenation of a function formula and function argument
$(4,10)$	$(\leftarrow \alpha_\sqcup($	α with '$\sqcup$' before '(' marks the beginning of an operator composition
$(5,5)$	$) \leftarrow)($	$)($ marks the concatenation of a function formula and function argument
$(5,10)$	$(\leftarrow)_\sqcup\alpha$	$)$ with '$\sqcup$' before 'α' marks the end of an operator composition

Table 2: *Explanation of some contextually nonevident (critical) replacement formulas from Table 1.*

$$\aleph \xmapsto{0,1} \alpha \xmapsto{1,4} \alpha) \xmapsto{5,1} \boxed{\alpha\alpha});$$

$$\aleph \xmapsto{0,1} \alpha \xmapsto{2,1} \boxed{\alpha\alpha};$$

$$\aleph \xmapsto{0,2} \models, \xmapsto{2,5} \models \boxed{,\,,};$$

$$\aleph \xmapsto{0,1} \alpha \xmapsto{1,1} \alpha \models \xmapsto{1,2} \boxed{\models \alpha \models};$$

$$\aleph \xmapsto{0,1} \alpha \xmapsto{1,2} \models \alpha \xmapsto{1,2} \boxed{\models \alpha \models};$$

$$\aleph \xmapsto{0,1} \alpha \xmapsto{1,3} (\alpha \xmapsto{1,3} ((\alpha \xmapsto{4,3} (\models (\alpha \xmapsto{4,9}$$
$$\circ(\models (\alpha \xmapsto{1,4} \boxed{\circ(\models (}\alpha)$$

etc. Through these examples one can understand the conditionality of the edge syntax. In this way, the intermediate results of formula generation must be proved on syntactic correctness, However, this mode of correct frame generation enables a spontaneous approach in emerging of formulas, connected with semantic (interpretive) concepts of the spontaneously arising formulas. □

6 The Basic Notion of the Gestalt Belonging to an Informational Entity

6.1 Introduction

Gestalts are a kind of interpretative possibilities to a given formula. As one will learn, gestalts can be classified in various directions, the formal and the applied ones. In some respect, the concept of informational gestalt approaches the concept of an informational graph, but in a formal, especially causal and circular sense.

6.2 Informationally Phenomenalistic Gestalts

Informational entity on the formalistic level is nothing else then an informational formula. The simplest form of a formula is a marking operand which marks an entity. Thus, in the very beginning of our discourse we have to put the question concerning the gestalt of a formula (or formula system) on an intuitive level. Later, we will answer the question in an informationally formalistic way, that is, by an adequate formula expression for informational gestalts.

Gestalt of a formula describes the entire possible *structure of causality* in the framework of the informational logical consistency, that is, the well-formedness of formulas which follow (can be derived) from the original formula by all possible displacements of the parenthesis pairs.

For example, a sentence in a natural language is a grammatically correct sequence of words and each word in the sentence performs (more exactly, informs) as an autonomous and with other words informationally connected entity, that is, formally, as an informational operand or informational operator (the property, quality of operands which it concerns). Such a sentence hides all possible causal choices (cases, example) of the sentence and this sentence presentation potentiality is called the gestalt of the sentence.

In this way, a sentence can also be understood as an informational graph, which is an ordered structure (sequence, loop) of operands and operators without any parenthesis pairs (grouped informational connections) between words (operands and operators). In a practical case of a sentence understanding, only few of the possible cases of the setting of parentheses pairs are realized (constructed) by the observer of the sentence, that is, only those which fit in the best possible manner the given discourse of involved observers.

6.3 Definition of the Gestalt of a Formula and of a Formula System

In this subsection, we have to define gestalts concerning a serial and circular formula, and gestalts of parallel formula systems.

Definition 17 [Length of a Serial Formula] *The length ℓ of a serial formula is an integer being equal to the number of binary informational operators (of type $\models$) in the formula.* □

For the length ℓ, unary operators in a formula do not count. They represent the so-called internalism (input) or externalism (output) of operands occurring in the formula. For instance, the length of formula

$$\alpha \models ((\alpha_1 \models; \models \alpha_1) \models \alpha_2)$$

is, evidently, $\ell = 2$, where operand α_1 disposes of its own input and output. Evidently, the last formula can be expressed by a parallel formula system in the form

$$\alpha \models (\alpha_1 \models \alpha_2);\ \alpha_1 \models;\ \models \alpha_1$$

where from the original serial formula the unary parts are removed.

Definition 18 [Gestalt of a Serial Formula] *Let*

$$\alpha \models (\alpha_1 \models (\alpha_2 \models \cdots (\alpha_{n-1} \models \alpha_n) \cdots))$$

be a serial formula $\varphi_{\rightarrow}$ of length $\ell = n$. Then, the parallel formula system

$$\Gamma(\varphi_{\rightarrow}) \rightleftharpoons \begin{pmatrix} \alpha \models (\alpha_1 \models (\alpha_2 \models \cdots (\alpha_{n-1} \models \alpha_n) \cdots)); \\ (\alpha \models \alpha_1) \models (\alpha_2 \models \cdots (\alpha_{n-1} \models \alpha_n) \cdots); \\ \vdots \\ (\cdots ((\alpha \models \alpha_1) \models \alpha_2) \models \cdots \alpha_{n-1}) \models \alpha_n \end{pmatrix}$$

consisting of exactly

$$N_{\Gamma(\varphi_{\rightarrow})} = \frac{1}{n+1}\binom{2n}{n}$$

formulas obtained from formula $\varphi_{\rightarrow}$ by all possible replacements of the parenthesis pairs, including $\varphi_{\rightarrow}$, is called the gestalt *of serial formula $\varphi_{\rightarrow}$.* □

A circular serial formula $\varphi_{\rightarrow}^{\circlearrowleft}$ differs from a serial formula $\varphi_{\rightarrow}$ only in its last (the rightmost) operand which is equal to its first (the leftmost) one and, in this way, performs the cycle concerning the distinguished operand or its circularity (its circular closeness).

Definition 19 [Gestalt of a Circular Formula] *Let*

$$\alpha \models (\alpha_1 \models (\alpha_2 \models \cdots (\alpha_{n-1} \models (\alpha_n \models \alpha)) \cdots))$$

be a circular serial formula $\varphi_{\rightarrow}^{\circlearrowleft}$ of the length $\ell = n+1$. Then, the parallel formula system

$$\Gamma(\varphi_{\rightarrow}^{\circlearrowleft}) \rightleftharpoons \begin{pmatrix} \alpha \models (\alpha_1 \models (\alpha_2 \models \cdots (\alpha_{n-1} \models (\alpha_n \models \alpha)) \cdots)); \\ (\alpha \models \alpha_1) \models (\alpha_2 \models \cdots (\alpha_{n-1} \models (\alpha_n \models \alpha)) \cdots); \\ \vdots \\ ((\cdots ((\alpha \models \alpha_1) \models \alpha_2) \models \cdots \alpha_{n-1}) \models \alpha_n) \models \alpha \end{pmatrix}$$

consisting of exactly

$$N_{\Gamma(\varphi^{\circlearrowleft}_{\rightarrow})} = \frac{1}{n+1}\binom{2n}{n}$$

formulas obtained from formula $\varphi^{\circlearrowleft}_{\rightarrow}$ by all the possible replacements of the parenthesis pairs, including $\varphi^{\circlearrowleft}_{\rightarrow}$, is called the gestalt *of serial formula $\varphi^{\circlearrowleft}_{\rightarrow}$.* □

In the next two definitions the adjective *basic* will concern the serial formula in the form of transition $\xi \models \eta$. So, the named *parallel* formula systems will consist of basic transitions. On the other hand, we have learned the gestalts of complex serial formulas are not basic in the described sense, and they will be marked in other ways.

Definition 20 [Length of a Parallel Basic Serial Formula System] *The length $\ell_{\parallel}$ of a parallel serial formula system is an integer being equal to the number of serially connected parallel basic transitions (of type $\xi \models \eta$) in the formula system*

$$\varphi_{\parallel} \rightleftharpoons \begin{pmatrix} \alpha \models \alpha_1; \\ \alpha_1 \models \alpha_2; \\ \vdots \\ \alpha_{n-1} \models \alpha_n \end{pmatrix}$$

including n serially operand-coupled basic transitions). Thus, evidently, $\ell_{\parallel} = n$. □

Definition 21 [Gestalt of a Parallel Basic Serial Formula System] *Let $\varphi_{\parallel}$ be a parallel basic serial formula system of the length $\ell_{\parallel} = n$ (Definition 20). The gestalt Γ of formula system $\varphi_{\parallel}$ is given by*

$$\Gamma(\varphi_{\parallel}) \rightleftharpoons \varphi_{\parallel}$$

The formula system $\varphi_{\parallel}$ means the gestalt of itself. □

This definition says a basic parallel system of the form $\varphi_{\parallel}$ represents the final achievement with respect to other possibilities of informational presentation (interpretation). In $\varphi_{\parallel}$ already all its informational possibilities are included.

Definition 22 [Gestalt of a Parallel Circular Basic Serial Formula System] *Let $\varphi^{\circlearrowleft}_{\parallel}$ be a parallel circular basic serial formula system of length $\ell^{\circlearrowleft}_{\parallel} = n+1$ where*

$$\varphi^{\circlearrowleft}_{\parallel} \rightleftharpoons \begin{pmatrix} \alpha \models \alpha_1; \\ \alpha_1 \models \alpha_2; \\ \vdots \\ \alpha_{n-1} \models \alpha_n; \\ \alpha_n \models \alpha \end{pmatrix}$$

The gestalt Γ of formula system $\varphi_{\parallel}$ is given by

$$\Gamma(\varphi^{\circlearrowleft}_{\parallel}) \rightleftharpoons \varphi^{\circlearrowleft}_{\parallel}$$

Circular formula system $\varphi^{\circlearrowleft}_{\parallel}$ means the gestalt of itself. □

This definition is in accordance with Definition 21 for the serial noncircular case.

6.4 Parallelization of Serial Formulas and Serialization of Parallel Formula Systems

A serial formula $\varphi_{\rightarrow}$ (each single formula is in a way serial) demonstrates possibilities of its parallelization by comprehending it into detail, where the transition from one operand to another occurs from one to the next operand in the formula sequence. For instance, in $\alpha \models (\beta \models \gamma)$, as a serial formula specimen, the detailed analysis concerns the transition from α to β and, then, from β to γ, that is, formally, $\alpha \models \beta$ and $\beta \models \gamma$, respectively. Within this view,

$$(\alpha \models (\beta \models \gamma)) \Longrightarrow \begin{pmatrix} \alpha \models \beta; \\ \beta \models \gamma \end{pmatrix}$$

But, the consequence $(\alpha \models \beta; \beta \models \gamma)$ of this implication delivers all the possible cases (serial interpretations) of the original formula $\alpha \models (\beta \models \gamma)$. Thus, a further implication is

$$\begin{pmatrix} \alpha \models \beta; \\ \beta \models \gamma \end{pmatrix} \Longrightarrow ((\alpha \models \beta) \models \gamma)$$

and, implication transitively,

$$(\alpha \models (\beta \models \gamma)) \Longrightarrow ((\alpha \models \beta) \models \gamma)$$

6.4.1 Parallelization of a Serial Formula

In which way a serial formula $\varphi_{\rightarrow}(\alpha, \alpha_1, \ldots, \alpha_n)$ of the length n could be possibly parallelized (when looking into its details on the operand level, and ignoring the set parenthesis pairs)? Such a view, ignoring the parenthesis pairs, searches for all possible causal cases of a serial formula presentation when operands and operator keep their places and meanings of the original formula, but the informational relations concerning the parenthesis pairs are changed in all possible manners.

Definition 23 [Parallelization of a Basic Serial Formula] *Let Pi mark parallelization and let $\varphi_{\rightarrow}(\alpha, \alpha_1, \ldots, \alpha_n)$ be a serial formula. Then,*

$$\Pi(\varphi_{\rightarrow}(\alpha, \alpha_1, \ldots, \alpha_n)) \rightleftharpoons \begin{pmatrix} \alpha \models \alpha_1; \\ \alpha_1 \models \alpha_2; \\ \vdots \\ \alpha_{n-1} \models \alpha_n \end{pmatrix}$$

is called the parallelization of the serial formula. □

On the other hand, the discussed serial formula $\varphi_{\rightarrow}(\alpha, \alpha_1, \ldots, \alpha_n)$ has the standardized gestalt $\Gamma(\varphi_{\rightarrow}(\alpha, \alpha_1, \ldots, \alpha_n))$, given by Definition 18.

Definition 24 [Graphical Equivalence between Parallelization and Gestalt of a Basic Serial Formula] *Two formula or formula systems are graphically equivalent (operator $\equiv_{\mathfrak{G}}$) if they have one and the same informational graph $\mathfrak{G}$, where informational graph is presented by circles (or ovals) for operands and by arrows for operators, ignoring the positions of parenthesis pairs in both formulas and formula systems.*

Under such circumstances, parallelization of a serial formula is graphically equivalent to gestalt of the serial formula, that is,

$$\Pi(\varphi_{\rightarrow}(\alpha, \alpha_1, \ldots, \alpha_n)) \equiv_{\mathfrak{G}} \Gamma(\varphi_{\rightarrow}(\alpha, \alpha_1, \ldots, \alpha_n))$$

□

This definition shows that a gestalt of a serial formula represents, according to the causal possibilities (causal structure), nothing more than the parallelization of the formula. As a consequence, the last definition and the previous ones deliver, evidently, for basic serial formulas the following graphical equivalences:

$$\begin{array}{ll} \Gamma(\varphi_{\rightarrow}) \equiv_{\mathfrak{G}} \varphi_{\parallel}; & \Pi(\varphi_{\rightarrow}) \equiv_{\mathfrak{G}} \varphi_{\parallel}; \\ \Gamma(\varphi_{\rightarrow}) \equiv_{\mathfrak{G}} \varphi_{\parallel}; & \Gamma(\varphi_{\parallel}) \equiv_{\mathfrak{G}} \varphi_{\parallel}; \\ \Pi(\varphi_{\rightarrow}) \equiv_{\mathfrak{G}} \Gamma(\varphi_{\rightarrow}); & \Gamma(\varphi_{\parallel}) \equiv_{\mathfrak{G}} \varphi_{\parallel} \end{array}$$

For circular serial formulas, there is, analogously,

$$\begin{array}{ll} \Gamma(\varphi^{\circlearrowleft}_{\rightarrow}) \equiv_{\mathfrak{G}} \varphi^{\circlearrowleft}_{\parallel}; & \Pi(\varphi^{\circlearrowleft}_{\rightarrow}) \equiv_{\mathfrak{G}} \varphi^{\circlearrowleft}_{\parallel}; \\ \Gamma(\varphi^{\circlearrowleft}_{\rightarrow}) \equiv_{\mathfrak{G}} \varphi^{\circlearrowleft}_{\parallel}; & \Gamma(\varphi^{\circlearrowleft}_{\parallel}) \equiv_{\mathfrak{G}} \varphi^{\circlearrowleft}_{\parallel}; \\ \Pi(\varphi^{\circlearrowleft}_{\rightarrow}) \equiv_{\mathfrak{G}} \Gamma(\varphi^{\circlearrowleft}_{\rightarrow}); & \Gamma(\varphi^{\circlearrowleft}_{\parallel}) \equiv_{\mathfrak{G}} \varphi^{\circlearrowleft}_{\parallel} \end{array}$$

This completes the discussion concerning parallelization and gestalts of serial and circular formulas.

6.4.2 Serialization of a Parallel System of Basic Transition Formulas

Particularized causation can be extracted from parallel formula systems by their serialization. If the parallelization of serial formulas is a sort of causal universalization, the serialization of a parallel formula systems has the meaning in a causal specialization, proceeding from a universal situation to a very particular one.

Definition 25 [Graphical Implication between Serialization of a Parallel Formula and Serial Formula Belonging to the Gestalt] *If $\varphi_{\parallel}$ and $\varphi_{\rightarrow}$ are parallel and serial formulas according to Definition 20 and Definition 18, respectively. Then,*

$$\varphi_{\parallel} \Longrightarrow_{\mathfrak{G}} \varphi_{\rightarrow};\ \varphi_{\rightarrow} \in_{\mathrm{sys}} \Gamma(\varphi_{\rightarrow})$$

Operator $\Longrightarrow_{\mathfrak{G}}$ reads 'implies graphically' and operator $\in_{\mathrm{sys}}$ means 'is a component of the system'. □

Similar definition can be set in concern to circular formulas.

Definition 26 [Graphical Implication between Serialization of a Parallel formula and Circular Serial Formula belonging to the Gestalt] *If $\varphi^{\circlearrowleft}_{\parallel}$ and $\varphi^{\circlearrowleft}_{\rightarrow}$ are parallel and serial formulas according to Definition 22 and Definition 19, respectively. Then,*

$$\varphi^{\circlearrowleft}_{\parallel} \Longrightarrow_{\mathfrak{G}} \varphi^{\circlearrowleft}_{\rightarrow};\ \varphi^{\circlearrowleft}_{\rightarrow} \in_{\mathrm{sys}} \Gamma(\varphi^{\circlearrowleft}_{\rightarrow})$$

Operators $\Longrightarrow_{\mathfrak{G}}$ and $\in_{\mathrm{sys}}$ means have the same meaning as in the preceding definition. □

This completes the discussion on the possibilities of drawing the serial consequences (possible interpretations) from parallel systems. In this respect, a serialization on the basis of parallelism has the role to investigate the details, possible particularities and the like.

6.5 Axiomatic Gestalts

The axiomatic gestalts include all the formulas obtained by the possible replacements of parenthesis pairs in given axioms. In mathematics, axioms function as the given initial and hypothesized theorems and as given rules of inference (method, deduction, induction). It would be extremely interesting to make a look into the background of the axiomatic gestalts and to observe all the possible 'axiomatic' formulas, that is those, proceeding from the given axioms by their 'gestalting'.

Mathematical axioms [7, 8] can be rewritten in an informational form. Let us take the implication axiom set ([7], p. 66) in the form

$$\begin{array}{rl} 1) & \alpha \Longrightarrow (\beta \Longrightarrow \alpha); \\ 2) & (\alpha \Longrightarrow (\alpha \Longrightarrow \beta)) \Longrightarrow (\alpha \Longrightarrow \beta); \\ 3) & (\alpha \Longrightarrow \beta) \Longrightarrow ((\beta \Longrightarrow \gamma) \Longrightarrow (\alpha \Longrightarrow \gamma)) \end{array}$$

For seeing the possibilities in the sense of a gestalt, informational graphs in Fig. 4 can be used. These graphs are uniquely described by the corresponding parallel systems of basic transitions, that is,

$$\begin{array}{rl} 1) & (\alpha \Longrightarrow \beta;\ \beta \Longrightarrow \alpha); \\ 2) & (\alpha \Longrightarrow \alpha;\ \alpha \Longrightarrow \beta;\ \beta \Longrightarrow \alpha); \\ 3) & (\alpha \Longrightarrow \beta;\ \beta \Longrightarrow \beta;\ \beta \Longrightarrow \gamma;\ \gamma \Longrightarrow \alpha; \\ & \alpha \Longrightarrow \gamma) \end{array}$$

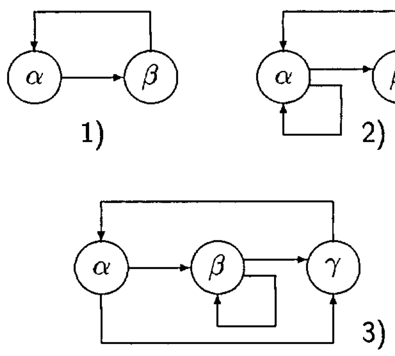


Figure 4: *A graphical interpretation of the implication axioms where graphs give an insight to the corresponding gestalts* $\Gamma(\varphi_1), \Gamma(\varphi_2)$, *and* $\Gamma(\varphi_3)$.

The last system of formulas represents the so-called parallelization of axiom formulas by the most elementary informational-implication transitions of the form $\xi \Longrightarrow \eta$.

In the axiomatic gestalts corresponding to the axiom formulas φ_1, φ_2 and φ_3 in 1), 2), and 3), the number of the possible formulas is $N = \frac{1}{\ell+1}\binom{2\ell}{\ell}$ where ℓ is the number of $\Longrightarrow$-operators in a formula. Thus,

$$\Gamma(\varphi_1) \rightleftharpoons \begin{pmatrix} \alpha \Longrightarrow (\beta \Longrightarrow \alpha); \\ (\alpha \Longrightarrow \beta) \Longrightarrow \alpha \end{pmatrix};$$

$$\ell_1 = 2;\ N_1 = \tfrac{1}{3}\tbinom{4}{2} = 2;$$

$$\Gamma(\varphi_2) \rightleftharpoons$$

$$\begin{pmatrix} \alpha \Longrightarrow (\alpha \Longrightarrow (\beta \Longrightarrow (\alpha \Longrightarrow \beta))); \\ (\alpha \Longrightarrow \alpha) \Longrightarrow (\beta \Longrightarrow (\alpha \Longrightarrow \beta)); \\ \vdots \\ (((\alpha \Longrightarrow \alpha) \Longrightarrow \beta) \Longrightarrow \alpha) \Longrightarrow \beta \end{pmatrix};$$

$$\ell_2 = 4;\ N_2 = \tfrac{1}{5}\tbinom{8}{4} = 14;$$

$$\Gamma(\varphi_3) \rightleftharpoons$$

$$\begin{pmatrix} \alpha \Longrightarrow (\beta \Longrightarrow (\beta \Longrightarrow (\gamma \Longrightarrow (\alpha \Longrightarrow \gamma)))); \\ (\alpha \Longrightarrow \beta) \Longrightarrow (\beta \Longrightarrow (\gamma \Longrightarrow (\alpha \Longrightarrow \gamma))); \\ \vdots \\ ((((\alpha \Longrightarrow \beta) \Longrightarrow \beta) \Longrightarrow \gamma) \Longrightarrow \alpha) \Longrightarrow \gamma \end{pmatrix};$$

$$\ell_3 = 5;\ N_3 = \tfrac{1}{6}\tbinom{10}{5} = 42$$

Novikov ([8], p. 75) replaces axioms 2) and 3) by a single axiom of the form

$$\begin{array}{rl} 4) & (\alpha \Longrightarrow (\beta \Longrightarrow \gamma)) \Longrightarrow \\ & \quad ((\alpha \Longrightarrow \beta) \Longrightarrow (\alpha \Longrightarrow \gamma)); \end{array}$$

$$\ell_4 = 6;\ N_4 = \tfrac{1}{7}\tbinom{12}{6} = 132$$

The graph of this axiom is presented in Fig. 5 and, as one can see, differs from graphs 2) and 3) in Fig. 4, substantially. The basic parallel formula system for this graph is

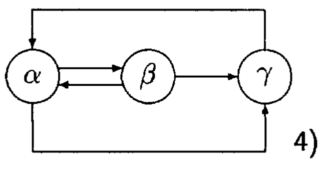


Figure 5: *A graphical interpretation of the implication axiom* 4) *where the graph gives an insight to the corresponding gestalt* $\Gamma(\varphi_4)$.

$$\begin{array}{rl} 4) & (\alpha \Longrightarrow \beta;\ \beta \Longrightarrow \gamma;\ \gamma \Longrightarrow \alpha;\ \beta \Longrightarrow \alpha; \\ & \alpha \Longrightarrow \gamma) \end{array}$$

The gestalt $\Gamma(\varphi_4)$ includes 132 formulas of length $\ell_4 = 6$. Another presentation of the graph in Fig. 5 uses a system of all circular formulas (graph loops), so, system

$$\alpha \Longrightarrow (\beta \Longrightarrow (\gamma \Longrightarrow \alpha)); \\ \alpha \Longrightarrow (\gamma \Longrightarrow \alpha); \alpha \Longrightarrow (\beta \Longrightarrow \alpha)$$

describes the situation in a particular case.

6.6 Serial and Reversely Serial Gestalts

A serial decomposition of an informational entity roots in the causal nature of entities and its informational components. For instance, the analysis of an entity progresses into the direction of discovering its informational details, stepping deeper and deeper into the structure. But, commonsensically, when reaching a deep informational detail, the process can be reversed, so that the analysis proceeds in the opposite direction, for example, verifying the obtained analytical (decompositional) results and accomplishing them informationally. On the level of conscious thought, such a forward and backward informational processing is thoroughly possible.

Let us discuss in short the case of a serial and, simultaneously, reversely serial decomposition of an entity, marked by α_1, as shown in Fig. 6. At the first look, this structure is a multiloop

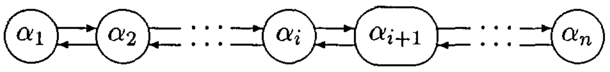


Figure 6: *A graphical interpretation of the serially and reversely serially [$\frac{1}{2}n(n-1)$-tuple-loop] structured decomposed system for entity α_1.*

one. There is, for example, the longest cycle [$\ell_{\max} = 2(n-1)$], in which operands appear in the sequence $\alpha_1, \alpha_2, \ldots, \alpha_i, \alpha_{i+1}, \ldots, \alpha_n, \alpha_{n-1}, \ldots, \alpha_{i+1}, \alpha_i, \ldots, \alpha_2, \alpha_1$, and there are all the possible other shorter cycles [$\ell = 1, 2, \ldots, 2n-3$]. A short analysis shows that the decomposition structure in Fig. 6 has a number of loops (L)

$$L = \frac{1}{2}n(n-1)$$

that is, numerically,

n	1	2	3	4	5	6	7	8	9	10
L	0	1	3	6	10	15	21	28	36	45

A parallel formula system for the graph in Fig. 6 is, certainly,

$$\begin{pmatrix} \alpha_1 \models \alpha_2; & \cdots & \alpha_i \models \alpha_{i+1}; & \cdots & \alpha_{n-1} \models \alpha_n; \\ \alpha_2 \models \alpha_1; & \cdots & \alpha_{i+1} \models \alpha_i; & \cdots & \alpha_n \models \alpha_{n-1} \end{pmatrix}$$

Let us define, precisely, the kinds of different "loops" and their numbers according to the graph, formula, and causal situation.

Definition 27 [A Concept of Graphical, Formula, and Causal Loop for Simultaneously Serial and Reversely Serial Case] *Let $\mathfrak{G}$ be a graph in Fig. 6. Let us distinguish three kinds of loops:*

1. *A* graphical loop *is a loop visible through the circumspection of the graph which, regardless of its circular structure, is considered as graphically different from all the other possible loops in the graph.*
2. *A* formula loop *is a loop which, in any appropriate form (arbitrary displacements of parenthesis pairs and arbitrary choice of the leftmost operand of the loop) corresponds to the graph loop.*
3. *A* causal loop *follows from a formula loop by an arbitrary displacement of the parenthesis pairs.*

The are three different numbers of loops for the graph in Fig. 6:

1. *The number $L_{\mathfrak{G}}$ of all graphical loops is*
$$L_{\mathfrak{G}} = \frac{1}{2}n(n-1)$$
2. *The number L_φ of all formula loops amounts to*
$$L_\varphi = 2n^2$$
3. *The number $L_{\varphi()}$ of all causal loops attains*
$$L_{\varphi()} = \sum_{i=1}^{2n^2} \frac{1}{\ell_i + 1}\binom{2\ell_i}{\ell_i}$$
where ℓ_i marks the length of the corresponding formula loop.

Evidently, $L_{\varphi()}$ corresponds to the number of all the formulas included in all the gestalts corresponding to the graph in Fig. 6. □

6.7 Circular and Reversely Circular Gestalts

Circularly serial decomposition of an informational entity roots in the causal and metaphysicalistic nature of informational entities and its informational components. The reversal circularity brings something new into the discourse of the possible circular structures and its practical implications.

Let us see in short the case of a circularly serial and, simultaneously, circularly reversely serial decomposition of an entity, marked by α_1, as shown in Fig. 7. A parallel formula system for the graph

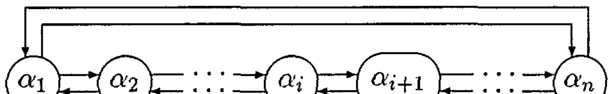


Figure 7: *A graphical interpretation of the circularly serially and circularly reversely serially [$\frac{1}{2}n(n+1)$-tuple-loop] structured decomposed system for entity α_1.*

in Fig. 7 is

$$\begin{pmatrix} \alpha_1 \models \alpha_2; & \alpha_2 \models \alpha_1; \\ \alpha_2 \models \alpha_3; & \alpha_3 \models \alpha_2; \\ \vdots & \vdots \\ \alpha_{n-1} \models \alpha_n; & \alpha_n \models \alpha_{n-1}; \\ \alpha_n \models \alpha_1; & \alpha_1 \models \alpha_n \end{pmatrix}$$

The n longest serial loops and their counterloops are determined as

$$\begin{array}{l} \alpha_i \models (\alpha_{i+1} \models \cdots (\alpha_{n-1} \models (\alpha_n \models \\ \quad (\alpha_1 \models (\alpha_2 \models \cdots (\alpha_{i-1} \models \alpha_i) \cdots)))) \cdots); \\ \alpha_i \models (\alpha_{i-1} \models \cdots (\alpha_2 \models (\alpha_1 \models \\ \quad (\alpha_n \models (\alpha_{n-1} \models \cdots (\alpha_{i+1} \models \alpha_i) \cdots)))) \cdots); \\ i = 1, 2, \ldots, n \end{array}$$

respectively. The length of each of $2n$ circular formulas is n and, thus, the gestalts for these formulas only include, altogether,

$$\frac{2n}{n+1}\binom{2n}{n}$$

formulas (with all the possible displacements of the parenthesis pairs).

Evidently, according to Definition 27, for the circular case, there is

$$L_{\mathfrak{G}}^{\circlearrowleft} = \tfrac{1}{2}n(n+1); \quad L_{\varphi}^{\circlearrowleft} = 2(n+1)^2;$$

$$L_{\varphi()}^{\circlearrowleft} = \sum_{i=1}^{2(n+1)^2} \frac{1}{\ell_i + 1}\binom{2\ell_i}{\ell_i}$$

6.8 Metaphysicalistic and Reversely Metaphysicalistic Gestalts

Metaphysicalistic formulas concern the interior (internal states) of an informing entity and, in this way, replace the reductionistic and rigidly (algorithmically) determined propositions and predicates. In such a context, metaphysicalistic formulas can behave informationally, that is, as circularly and intentionally spontaneous entities. One can imagine how a metaphysicalistic formula—on the level of the conscious informational phenomenalism in the living brain (mind)—models the nervous processes constituting the essential conscious entity. In neural systems, the one-way (direction) propagation of information—from synapses, dendrites, neuronal somata, axons to the synapses, and so forth—is a commonly recognized phenomenon (scientific philosophy). However, on the conscious or artificial (constructionist) level, the direction of information propagation can be reversed in the form of the thought flow or a machine processing[2]. Such a reverse metaphysicalistic process can represent an essentially different interpretation of the original process, particularly in the sense of specifically changed causalism where, in a cycle, causes and effects interchange their roles. One has to remind that the original and the reversed process take place in a circularly interweaved environment where a strict distinction of causes and their effects (consequences) is no longer possible.

A metaphysicalistic direct and reversal structure is graphically presented in Fig. 8. The bidirectional metaphysicalistic system is much more cycled than a standard metaphysicalistic system of an entity, represented in Fig. 10. While the original system has only 6 loops, the additional reversing causes a 30–loop structure, according to the formula $\frac{1}{2}n(n+1)$ and Fig. 8, $\frac{1}{2}n(n+1)+4 = 32$ for $n = 7$. The reader can calculate the entire complexity of the circularly structured me-

[2]See, for instance, the thermodynamic theory of thought processes [11].

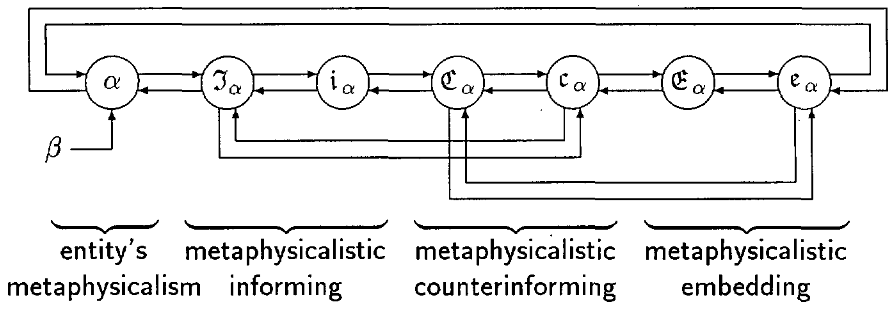

β
entity's metaphysicalism
metaphysicalistic informing
metaphysicalistic counterinforming
metaphysicalistic embedding

Figure 8: *A graphical interpretation of the circularly (32-tuple-loop) structured basic metaphysicalistic system of informational entity α being informationally impacted by the exterior entity β.*

taphysicalism and its reverse in the form of particular informational gestalts by himself/herself.

6.9 Parallel Gestalts

Parallel gestalts are nothing other than gestalts of gestalts. One can imagine in which way such a situation appears when examining parallel systems of arbitrarily structured (e.g., circularly circular) serial formulas.

Parallel gestalts concern systems of serial formulas where each formula has its gestalt and the possibilities of different serial formulas included in different gestalts has to be considered. Evidently, in such a case, the sum of the gestalt possibilities can be taken into account.

For a gestalt of a gestalt as a parallel system and a gestalt of parallel systems of basic transitions, there is evidently,

$$\Gamma(\Gamma(\varphi_{\rightarrow})) \equiv \Gamma(\varphi_{\rightarrow});\ \Gamma(\Gamma(\varphi_{\rightarrow}^{\circlearrowleft})) \equiv \Gamma(\varphi_{\rightarrow}^{\circlearrowleft});$$
$$\Gamma(\varphi_{\parallel}) \equiv \varphi_{\parallel};\ \Gamma(\varphi_{\parallel}^{\circlearrowleft}) \equiv \varphi_{\parallel}^{\circlearrowleft}$$

7 Inference Gestalts

Besides axioms, inference rules build up the skeleton of logical inferetialism, consisting of processes of deriving, deducing, inducing, abducing, etc. in the framework of logical reasoning and theories constructing.

Modi informationis (modes of informational inference) can be used in different informational deduction processes. The basic rules (principles) of inference in the propositional and predicate logic are, for instance, substitution, modus ponens and modus tollens, which perform in the framework of truth and falseness (the principle of tertium non datur). The reader can imagine how these rules can be transferred in the realm of informational entities (a kind of informational logic). Beside these principles, other inference rules can be introduced, known in the common speech as modus agendi, essendi, rectus, operandi, obliquus, vivendi, possibilitatis, etc., as already constituted in the Latin language. *Informational modus*

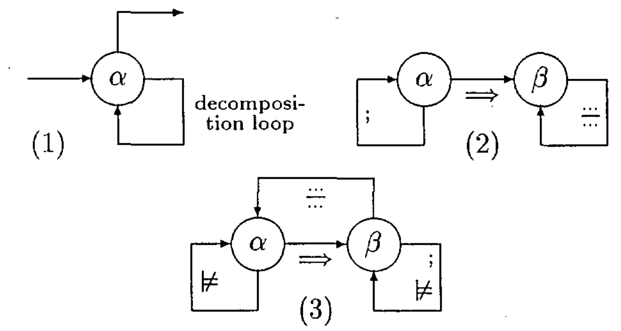


Figure 9: *Informational graphs for informational modus agendi* (1), *modus ponens* (2), *and modus tollens* (3).

agendi (informational phenomenalism) of an entity, marked by operand α, is shown by graph (1) in Fig. 9 and represents the mode in which an informational entity (operand, phenomenon, thing, process) acts or operates (informs and is informed), that is,

$$\alpha \rightleftharpoons \begin{pmatrix} \models \alpha; \\ \alpha \models; \\ \alpha \models \alpha \end{pmatrix} \quad \begin{array}{l} \text{input (internalism)} \\ \text{output (externalism)} \\ \text{interior (metaphysicalism)} \end{array}$$

The 'feedback' arrow (vector) hides the potentiality of α's decomposition (the α's metaphysicalism $\alpha \models \mathring{\alpha}$), the input arrow presents the α's internalism $\models \alpha$, and the output arrow the α's externalism $\alpha \models$.

Informational modus ponens represents the interpretation of the logical (philosophical) modus ponens into informational realm. Similarly, the *informational modus tollens* represents the interpretation of the logical (philosophical) modus tollens into informational realm. The detachment (modus ponendo ponens and modus tollendo tollens) formulas are, respectively,

$$(\mathrm{MP})\ \frac{\alpha;\alpha \Longrightarrow \beta}{\beta} \quad \text{and} \quad (\mathrm{MT})\ \frac{\alpha \Longrightarrow \beta;\beta \not\models \beta}{\alpha \not\models \alpha}$$

Their graphs (2) and (3) are drawn in Fig. 9. The arrows are marked by ';' (informational parallelism), $\Longrightarrow$ (informational implication), $\overset{..}{\underset{...}{-}}$ (informational detachment), and $\not\models$ (particular noninforming). It is to stress that $(\xi \not\models \xi) \Longrightarrow (\xi \not\models; \not\models \xi)$ expresses the modus existendi of a certain noninforming of the entity presented by the informational operand ξ.

Evidently, the gestalts for MP (rule marker ρ_{MP}) and MT (rule marker ρ_{MT}) are, respectively, equal to the rules themselves, that is,

$$\Gamma(\rho_{\mathrm{MP}}) \equiv \rho_{\mathrm{MP}} \quad \text{and} \quad \Gamma(\rho_{\mathrm{MT}}) \equiv \rho_{\mathrm{MT}}$$

because the parallel structured premises $\alpha; \alpha \Longrightarrow \beta$ and $\alpha \Longrightarrow \beta; \beta \not\models \beta$ have to be treated as indivisible entities.

Several other forms of modi informationis can be discussed. The one concerning the intention of an informational entity is the so-called *modus rectus*, by which the detachment of the intention of an informing entity would be possible.

8 Intelligent Gestalts

Intelligent gestalts are a consequence of intelligent informational formulas. Which kind of a formula could be comprehended as informing in an intelligent way? What is intelligent informing?

Definition 28 [Intelligent Informational Formula] *An intelligent informational formula possesses its specific intelligent metaphysicalism [13]. The rough structure of an intelligent formula, not being decomposed to the necessary details yet, is given by Fig. 10 and expressed as intelligence ι concerning α (functionally) by the parallel system of basic transitions.* □

How can the graph in Fig. 10 be described in a form embracing all possible formalistic phenomenalism? The answer is: by a parallel system of basic transition formulas determined by the paths between two entities. So, let us construct the parallel system (Fig. 10) in the form

$$\iota(\alpha) \rightleftharpoons \begin{pmatrix} \alpha \models \iota; & & \\ \iota \models \mathfrak{I}_\iota; & \mathfrak{I}_\iota \models \mathfrak{i}_\iota; & \mathfrak{i}_\iota \models \mathfrak{C}_\iota; \\ \mathfrak{C}_\iota \models \mathfrak{c}_\iota; & \mathfrak{c}_\iota \models \mathfrak{E}_\iota; & \mathfrak{E}_\iota \models \mathfrak{e}_\iota; \\ \mathfrak{i}_\iota \models \mathfrak{I}_\iota; & \mathfrak{c}_\iota \models \mathfrak{C}_\iota; & \mathfrak{e}_\iota \models \mathfrak{E}_\iota; \\ \mathfrak{c}_\iota \models \mathfrak{I}_\iota; & \mathfrak{e}_\iota \models \mathfrak{C}_\iota; & \mathfrak{e}_\iota \models \iota \end{pmatrix}$$

Let us write only one formula for each of the six loops in Fig. 10, considering the input operand α. For the shortest loops to the longest one, there is,

$$\begin{pmatrix} \alpha \models \iota; \\ \mathfrak{I}_\iota \models (\mathfrak{i}_\iota \models \mathfrak{I}_\iota);\ \ \mathfrak{C}_\iota \models (\mathfrak{c}_\iota \models \mathfrak{C}_\iota); \\ \qquad \mathfrak{E}_\iota \models (\mathfrak{e}_\iota \models \mathfrak{E}_\iota); \\ \mathfrak{I}_\iota \models (\mathfrak{i}_\iota \models (\mathfrak{C}_\iota \models (\mathfrak{c}_\iota \models \mathfrak{I}_\iota))); \\ \qquad \mathfrak{C}_\iota \models (\mathfrak{c}_\iota \models (\mathfrak{E}_\iota \models (\mathfrak{e}_\iota \models \mathfrak{C}_\iota))); \\ \iota \models (\mathfrak{I}_\iota \models (\mathfrak{i}_\iota \models \mathfrak{C}_\iota \models (\mathfrak{c}_\iota \models (\mathfrak{E}_\iota \models (\mathfrak{e}_\iota \models \iota))))) \end{pmatrix}$$

This system consisting of the input formula $\beta \models \iota$ and the six circular formulas of lengths $\ell = 2, 4$, and 7, describes only a very small part of the informational graph in Fig. 10. The length ℓ of a formula is determined by the number of binary operators occurring in it. The subgraph corresponding to the system of circular formulas in parallel cannot be drawn in a usual form because the informational graph (without parenthesis pairs) expresses the entire parallelism and, in this respect, informational gestaltism. But, this reduced or specifically particularized system already determines the discussed parallel system which can be constructed out of the reduced system of circular formulas and formula $\beta \models \iota$.

8.1 Interpretive Gestalts

Interpretive gestalts are a consequence of interpretive formulas by which existing formulas are "interpreted" by the introduction of additional parallel formulas or by further serial decomposition of the existing formulas. The process of interpretation is not limited in advance because at each system situation a new interpretive detail concerning the system constituents can be added.

8.2 Understanding Gestalts

The understanding system v in Fig. 11, understanding and producing the meaning of something

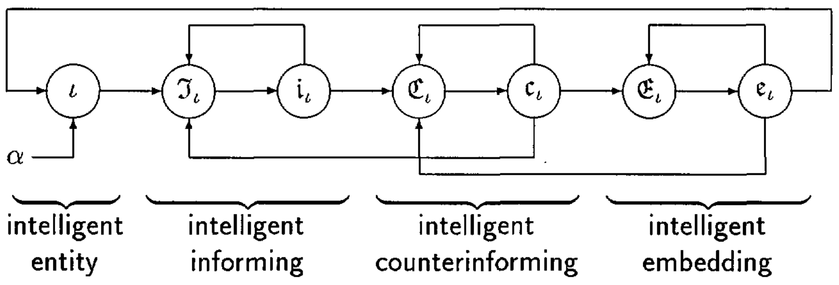


Figure 10: *A graphical interpretation of the circularly (sextuple-loop) structured parallel metaphysicalistically intelligent system of informational entity ι.*

β, is a paragon of a sufficiently complex understanding device, concerning various phenomenalisms in philosophy, psychology, cognitive science, artificial intelligence and, first of all, the general theory of the informational.

Let us introduce the following markers (entities) of understanding (subscribed by υ) and their initial meanings:

υ understanding;	β to be understood;
$\mathfrak{I}_\upsilon$ intending;	$\mathfrak{i}_\upsilon$ intention;
$\mathfrak{S}_\upsilon$ sensing;	$\mathfrak{s}_\upsilon$ sensibility;
$\mathfrak{O}_\upsilon$ observing;	$\mathfrak{o}_\upsilon$ observation;
$\mathfrak{B}_\upsilon$ being conscious;	$\mathfrak{b}_\upsilon$ consciousness;
$\mathfrak{U}_\upsilon$ being unconscious;	$\mathfrak{u}_\upsilon$ unconsciousness;
$\mathfrak{C}_\upsilon$ conceiving;	$\mathfrak{c}_\upsilon$ conception;
$\mathfrak{X}_\upsilon$ signifying;	$\mathfrak{x}_\upsilon$ significance;
$\mathfrak{Y}_\upsilon$ making sense;	$\mathfrak{y}_\upsilon$ reasonableness;
$\mathfrak{P}_\upsilon$ perceiving;	$\mathfrak{p}_\upsilon$ perception;
$\mathfrak{Z}_\upsilon$ concluding;	$\mathfrak{z}_\upsilon$ conclusion;
μ_υ meaning	

The longest loop in Fig. 11 has 22 binary operators, that is, $\frac{22}{23}\binom{44}{22} \approx 201\ 261\ 630\ 000$ possible interpretations alone within its informational gestalt.

9 Gestalts of an Informational Machine

A formal informational machine [19] is a formula system which can handle informational formulas in an informational manner. To set a formal machine conceptually means to make possible the design of a physical informational machine by which informational formulas can be processed informationally. This means that the faculty of informational arising (emerging and appearing of informational formulas) is offered to any informational formula representing an informational entity within the informational machine.

The question which arises is what is the basic formal structure of the informational machine. Which entities must enter into the machine concept to enable the informing of the processed formulas by the machine? Answers to this question are given on some other place [10, 19]. But, for our discussion concerning informational gestalt, it is important to stress that among other features, informational machine must be able to produce and observe the gestalts of appearing and processed informational formulas.

On the other hand, complex circular formulas, being a part of the informing machine itself, are, as any informational entity, observed by the machine also through the gestalt possibilities and being adapted consequently in dependence of the arising circumstances.

10 Star Gestalts

The idea of the star gestalt roots in the formula circularity, although it can be defined for straighforwardly serial formula too.

Definition 29 [A Concept of Star Gestalt of a Serial and Circular Formula] *For the star gestalt, $\Gamma^\star$, of a serial formula, $\varphi_\rightarrow$, that is $\Gamma^\star(\varphi_\rightarrow)$, there is*

$$\Gamma^\star(\varphi_\rightarrow) \rightleftharpoons (\Gamma(\varphi_\rightarrow); \Gamma^<(\varphi_\rightarrow))$$

where $\Gamma^<(\varphi_\rightarrow)$ is a system of all gestalts for formulas $\varphi_{\overset{<}{\rightarrow}}$ (subformulas) obtained from formula $\varphi_\rightarrow$ as possible, in respect to $\varphi_\rightarrow$ shorter formulas. Formula $\varphi_{\overset{<}{\rightarrow}}$ is any formula which follows (is constructed) from the graph of formula $\varphi_\rightarrow$, starting from an arbitrary operand in the direction of

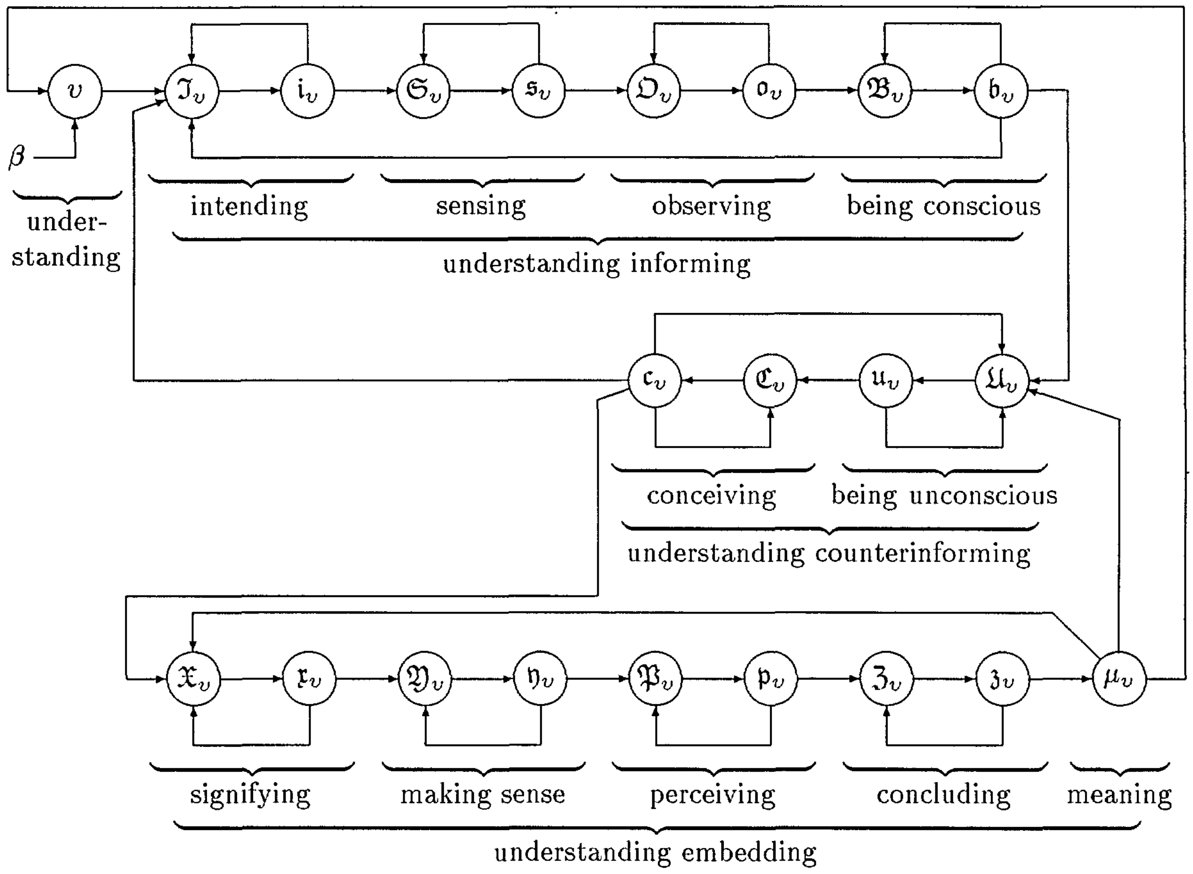


Figure 11: *A graphical interpretation of the circularly (16-tuple-loop) structured (metaphysicalistic) parallel understanding system v of entity β.*

arrows (operators) to an arbitrary point (operand or operator).

The star gestalt of a circular formula, $\varphi_{\rightarrow}^{\circlearrowleft}$, includes an infinite number of formulas obtained in the following manner:

(a) *Let $\mathfrak{G}(\varphi_{\rightarrow}^{\circlearrowleft})$ denote the graph of circular formula $\varphi_{\rightarrow}^{\circlearrowleft}$, that is a circular informational graph.*

(b) *Let the construction of a formula begin at the place of any graph operand (circle, oval).*

(c) *A formula φ, serial or circular, of the star gestalt $\Gamma^{\star}(\varphi_{\rightarrow}^{\circlearrowleft})$, is any formula obtained by moving from a starting circle (operand) along the graph arrows (operators) and ending at the arbitrary circle or arrow.*

□

The example which follows shows how formulas can come to existence within a star gestalt, using an informational graph. Let us show how, according to the given informational graph, an informational formula can arise. In [14], p. 63, Formula (63), the following formal situation of the understanding $\mathfrak{U}$ and metaphysicalism μ_ι within an intelligent ι entity is discussed:

$$\underbrace{(\iota \models (\overbrace{((\mathfrak{U} \models \mu_\iota) \models \mathfrak{U})}^{\mathfrak{U}\text{–loop}} \models \overbrace{((\mu_\iota \models \mathfrak{U}) \models \mu_\iota)}^{\mu_\iota\text{–loop}})) \models \iota}_{\iota\text{–loop}}$$

This formula is a step towards the decomposition of entity ι. The underlying graph for this formula is shown in Fig. 12.

The $\mathfrak{U}$-, μ_ι-, and ι-loop are not meant to be informationally (causally) isolated loops. Markers $\mathfrak{U}$, μ_ι, and ι can appear in other formulas of a system. In this way, they can stay open in respect to any actual environment, being changed by the exterior influences.

The reader can see how different informational formulas can arise from the informational graph

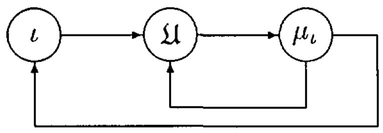


Figure 12: *A graphical interpretation of the circularly (double-loop) structured parallel transitional system* $(\iota \models \mathfrak{I}; \mathfrak{I} \models \mu_\iota; \mu_\iota \models \iota; \mu_\iota \models \mathfrak{I})$.

in Fig. 12. To obtain regular informational formulas, arbitrary transitions, for example, from a beginning entity in the graph, say ι, to an entity are necessary. The following formula system (a part of the star gestalt) shows the emerging of formulas using the graph in Fig. 12:

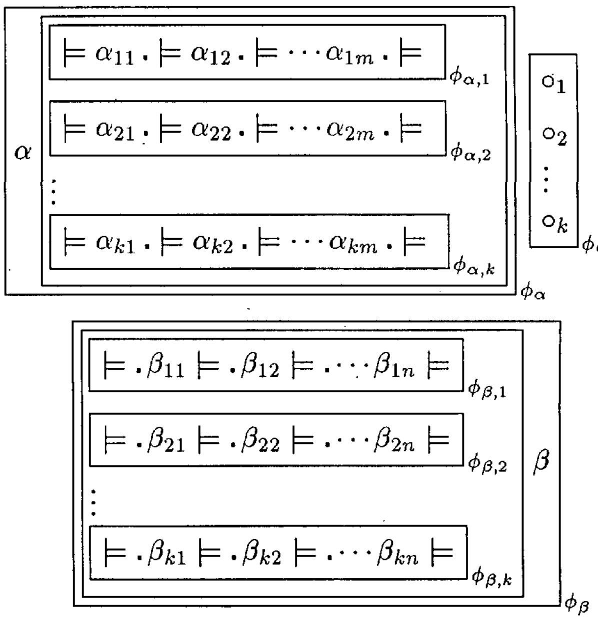
$$\begin{array}{l}
\iota; \\
\iota \models \mathfrak{I}; \\
\iota \models (\mathfrak{I} \models \mu_\iota); \\
\iota \models ((\mathfrak{I} \models \mu_\iota) \models \mathfrak{I}); \\
\iota \models (((\mathfrak{I} \models \mu_\iota) \models \mathfrak{I}) \models \mu_\iota); \\
\iota \models (((\mathfrak{I} \models \mu_\iota) \models \mathfrak{I}) \models (\mu_\iota \models \mathfrak{I})); \\
\iota \models (((\mathfrak{I} \models \mu_\iota) \models \mathfrak{I}) \models ((\mu_\iota \models \mathfrak{I}) \models \mu_\iota); \\
(\iota \models (((\mathfrak{I} \models \mu_\iota) \models \mathfrak{I}) \models ((\mu_\iota \models \mathfrak{I}) \models \mu_\iota))) \models \iota; \\
\vdots
\end{array}$$

In such a transition through the graph, the choosing of parenthesis pairs is spontaneous and, in this respect, different informational formulas can come into existence. Looking causally, each of the listed formula represents a special case and can deliver, for example, different semantics concerning the involved entities of a formula.

11 Conclusion

The introduced informational frame and informational gestalt view have brought new insight into the possibilities of a general informational theory [13, 14, 15, 16, 17, 18, 19, 20].

The informer-observer problem can be exhaustively analyzed through the study of framing of the transition $\alpha \models \circ \models \beta$, for instance, in the form



$\models \alpha_{11} \cdot \models \alpha_{12} \cdot \models \cdots \alpha_{1m} \cdot \models$
$\phi_{\alpha,1}$
$\models \alpha_{21} \cdot \models \alpha_{22} \cdot \models \cdots \alpha_{2m} \cdot \models$
$\phi_{\alpha,2}$
α
$\models \alpha_{k1} \cdot \models \alpha_{k2} \cdot \models \cdots \alpha_{km} \cdot \models$
$\phi_{\alpha,k}$
ϕ_α
$\circ_1$
$\circ_2$
$\circ_k$
$\phi_\circ$
$\models \cdot \beta_{11} \models \cdot \beta_{12} \models \cdots \beta_{1n} \models$
$\phi_{\beta,1}$
$\models \cdot \beta_{21} \models \cdot \beta_{22} \models \cdots \beta_{2n} \models$
$\phi_{\beta,2}$
β
$\models \cdot \beta_{k1} \models \cdot \beta_{k2} \models \cdots \beta_{kn} \models$
$\phi_{\beta,k}$
ϕ_β

What is α's informing and what is β's observing? How does phenomenon α inform its reality and how does phenomenon β observe the α's reality? Is the informing of α nevertheless observingly pre-understood by β and, in this way, is α's reality comprehended in advance by β? The last enframed and formalized formula system offers not only various possibilities for such interpretations, but also much more than it can be said by words.

The reader can observe that both ϕ_α and ϕ_β are harmonious frames, that is well-formed formulas. In this respect, α is an independent informer and β is an independent observer. Phenomenon α informs its reality by k different phenomenalisms, so phenomenon β can observe these k phenomenalisms of α by the k properly chosen observational phenomenalisms, when frames ϕ_α and ϕ_β are concatenated by the separator frame $\phi_\circ$ (performing as a concatenation operator) into the resulting transition of the form $\alpha \models \circ \models \beta$.

Let us suppose that α informs its reality by k different phenomenalities and that β has chosen k adequate phenomenalities for observing of α. The number of different phenomenalities is not limited and, in some way, it depends on the observing capabilities of β, so, in principle, $k \to \infty$.

What are the observing phenomenalisms of β? Let α represent a physical phenomenon in the sense of the contemporary physical sciences. The observing entity (observer, apparatus) β can use different theoretical and experimental concepts, methodologies, and methods for observation of α, for example, mathematical (formalistic, recursive), Euclidean (geometrical), Newtonian (gravitational), Hamiltonian (mechanical), Maxwellian (electromagnetical), Lorenzian (particle-motional), Einsteinian (relativistic), quantum, Hilbertian (axiomatic, spatial), Schröderian, cosmological, Weylian (quantum-gravitational), computational, mind-informational [9], Slechtanian (informationally thermodynamical [10, 11]), etc. All these theories and methods belong to scientifically accepted forms of informationalism (artificialisms, methodologisms, scientisms, cybernetism) and depend on human consciousness, intuition, and 'reality-adequateness'. By development of science, theories and methods will improve, change, and their number will increase. *But, the observer entity will observe only the phenomenalisms for which it is theoretically, methodologically, and experimentally capable* [4]. In this sense, the preunderstanding of α's phenomenalism will be, in a way, preunderstood by the observer.

References

[1] Patricia Smith Churchland: *Neurophilosophy. Towards a Unified Science of the Mind-Brain*. A Bradford Book. The MIT Press, Cambridge, MA, 1986.

[2] J.A. Fodor: *The Language of Thought*. Harvard University Press, Cambridge, MA, 1975.

[3] J.A. Fodor: *Representations*. MIT Press, Cambridge, MA, 1981.

[4] D. Gernert: *What Can We Learn from Internal Observers?* In H. Atmanspacher and G.J. Dalenoort (Eds.): *Inside versus Outside*, Springer Series in Synergetics **63** (1994), 121-133.

[5] M. Heidegger: *Sein und Zeit*. Sechzehnte Auflage. Max Niemeyer Verlag, Tübingen, 1986.

[6] M. Heidegger: *Being and Time*. Translated by J. Macquarrie & E. Robinson. Harper & Row, Publishers, New York, 1962.

[7] D. Hilbert und P. Bernays: *Grundlagen der Mathematik*. Erster Band. Die Grundlagen der mathematischen Wissenschaften in Einzeldarstellungen, Band XL. Verlag von Julius Springer, Berlin, 1934.

[8] П.С. Новиков: *Элементы математической логики*. Государственное издательство физико-математической литературы (Физматгиз), Москва, 1959.

[9] R. Penrose: *The Emperor's New Mind*. Oxford University Press-Vintage, New York, 1990.

[10] J. Slechta: *The Brain as the 'Hot' Cellular Automaton*. Proceedings of the 12th Int. Congress of Cybernetics, pp. 862-869, Namur, Belgium, 1989.

[11] J. Slechta: *On Thermodynamics of the Thought Processes within a Living Body in a Changing Environment*. Proceedings of the 12th Int. Congress of Cybernetics, pp. 878-885, Namur, Belgium, 1989.

[12] A.N. Whitehead and B. Russel: *Principia Mathematica*. Cambridge University Press, Cambridge, 1950 (in three volumes).

[13] A.P. Železnikar: *Metaphysicalism of Informing*. Informatica **17** (1993) No. 1, 65-80.

[14] A.P. Železnikar: *Formal Informational Principles*. Cybernetica **36** (1993) No. 1, 43-64.

[15] A.P. Železnikar: *Verbal and Formal Deduction of Informational Axioms*. Cybernetica **37** (1994) No. 1, 5-32.

[16] A.P. Železnikar: *Informational Being-in*. Informatica **18** (1994) No. 2, 149-173.

[17] A.P. Železnikar: *Informational Being-of*. Informatica **18** (1994) No. 3, 277-298.

[18] A.P. Železnikar: *Principles of a Formal Axiomatic Structure of the Informational*. Informatica **19** (1995) No. 1, 133-158.

[19] A.P. Železnikar: *A Concept of Informational Machine*. Cybernetica **38** (1995) No. 1, 7-36.

[20] A.P. Železnikar: *Organization of Informational Metaphysicalism*. Cybernetica **39** (1996) (to be published).

A Distributed Client/Server Benchmark for Network Performance Measurement

Jozo Dujmović,
Department of Computer Science, San Francisco State University
1600 Holloway Avenue, San Francisco, CA 94132, USA
jozo@cs.sfsu.edu
AND
Robert Kinicki, and Sanjay Subbanna
Department of Computer Science, Worcester Polytechnic Institute
100 Institute Road, Worcester, MA 01609, USA
rek@cs.wpi.edu, sanju@cs.wpi.edu

Keywords: network benchmarking, network traffic generator, LAN performance measurement, WAN performance measurement, client/server pairs

Edited by: Rudi Murn
Received: May 12, 1995 **Revised:** September 9, 1995 **Accepted:** January 15, 1996

We have developed a scalable network traffic generator and a general computer network benchmark for Unix platforms. This benchmark can be used to evaluate performance of user-level applications which interface directly with the transport layer of TCP/IP running on all types of computer networks. The network workload consists of distributed client/server process pairs (DCSP) and is called the DCSP benchmark. It can include any number and any distribution of communicating client/server pairs, yielding a very high level of flexibility and scalability of network traffic. We propose a standard classification of network workloads, define network performance indicators, and introduce performance measurement methods based on various versions of the DCSP benchmark. We also present experimental results generated using DCSP workloads to compare LAN configurations, study network saturation phenomena, and test WAN communications.

1 Introduction and Background

Computer performance evaluation is used for solving a variety of problems related to system design, system comparison and selection, and system tuning/optimization. Traditional performance evaluation techniques include analytic modeling, simulation, and benchmarking (performance measurement using benchmark programs). In this paper we are interested in network benchmarking as a technique for solving system comparison and selection problems.

Benchmarking has been an established technique for evaluating computer system performance since the mid-1960's, but controversies related to benchmarking are still frequently encountered [10, 8]. The most frequent source of different opinions is related to the selection of workload. There are two extreme standpoints: (1) benchmarking is meaningful only when performed with actual natural workload (a mix of user application programs) (e.g. [8], p.48), and (2) benchmarking can generate meaningful results with synthetic workloads[1], including industry standard benchmark suites [6, 7, 13]. In cases where the purpose of benchmarking is to provide inputs for the process of evaluation and selection of computer systems, the disadvantages of natural workloads

[1]The term "synthetic workload" denotes here all general workloads that differ from specific application programs of a given user. For example, such synthetic programs include small kernel programs (e.g. Eratosthenes' sieve), specifically designed synthetic workloads (e.g. Dhrystone), benchmark programs based on frequently used subroutines (e.g. LINPACK), and all other benchmarks based on selected more complex program products, such as compilers (e.g. *gcc*), interpreters, electronic circuits design programs, etc.

are well known and include high cost, low portability, and reduced credibility of natural workloads related to changes of workload caused by changes of equipment [5, 4]. On the other hand, synthetic workloads and standard benchmark suites are suitable because they usually provide low cost and high portability of measurement, the computer industry strongly supports this approach [3, 6, 7, 13, 18], and theoretical analyses show that errors caused by differences between natural and synthetic workloads are automatically reduced in all cases where individual performance indicators of a computer system are positively correlated [4]. Therefore, there are good reasons to believe that synthetic benchmarks can generate valuable performance indicators, useful as inputs for the decision process of evaluation and selection of competitive computer networks.

However, utilizing benchmarks to analyze computer network performance and extending benchmarking techniques into the domain of distributed systems has proven to be a complex problem. The plethora of possible choices for topology, communication configuration, network technology, protocols, file servers, computer platforms and operating systems yield an extensive problem domain.

There have been a large variety of research studies attempting to answer questions related to some aspect of computer network performance. This paper presents results from a new general purpose computer network benchmark based on distributed client-server pairs (DCSP). Before discussing the philosophy of the DCSP benchmark, it is useful to review a few of the efforts from the literature which analyze computer network behavior and identify performance bottlenecks.

One major focus in computer network performance has been to evaluate and analyze techniques for improving the response time of communications mechanisms such as IPC[2] [2, 15] and RPC [16, 17, 1]. For example, Thekkath and Levy [17] explore avenues for achieving low-latency communication by evaluating the low-level performance of a proposed RPC mechanism running on new-generation networks (FDDI and ATM). Using a very specialized testbed consisting of two DECstations communicating over an Ethernet, an FDDI, and a simplistic ATM LAN, they are able to determine that network controllers will play an increasing role in affecting communication latency in high-throughput LANs. Although a very thorough study, their results rely heavily on higher level assumptions about distributed application behavior, and their measurement procedures are too custom-tailored to be broadly useful.

Cabrera et al. [2] present a study of the impact of IPC mechanisms in Berkeley Unix 4.2BSD. Using artificial workloads, they present a detailed timing analysis of the dynamic behavior of TCP/IP and UDP/IP communication protocols. Using user space processes to send messages, network throughput and latency are obtained for Ethernet implementations. They assess the effects of different processors, network hardware interfaces, and host loads on network interprocess communication. A significant result of their study is identifying checksumming and data copying as expensive components of an IPC sent over TCP/IP.

By heavily instrumenting Sun Sparcstations connected over an Ethernet, Papadopoulos and Parulkar [15] studied the performance of SUNOS IPC over TCP/IP. They analyzed the effects of socket buffer size, window size, checksum costs and interactions with the operating system. They conclude that IPC requires a major portion of the CPU cycles to sustain high Ethernet utilization.

Kay and Pasquale [11] instrumented the TCP/IP and UDP/IP protocol stacks in the DEC Ultrix4.2a kernel and connected an HP Logic Analyzer to a link connecting DECstation 5000/200 workstations. They show that non-data touching components such as device driver code, *mbuf* management, and ARP processing can contribute significantly to the TCP overhead in network communications.

These studies are valuable for identifying specific pieces of the network communications mechanism which significantly contribute to overall performance. However, these research efforts are not

[2]Following is the meaning of standard abbreviations: IPC = Inter Process Communication, RPC = Remote Procedure Call, FDDI = Fiber Distributed Data Interface, ATM = Asynchronous Transfer Mode, IP = Internet protocol, TCP = Transmission Control Protocol, UDP = User Datagram Protocol, LAN = Local Area Network, WAN = Wide Area Network, ARP = Address Resolution Protocol, PING = Packet InterNet Groper, ICMP = Internet Control Message Protocol, NFS = Network File System (network file access protocol developed by SUN Microsystems), or (depending on context) Network File Server, FTP = File Transfer Protocol.

intended to provide general tools for evaluating and comparing the performance of specific network configurations. Very little published work is available on generic network benchmarks which utilize distributed applications; this is excluding transaction processing benchmarks (e.g. [13]), and various proprietary benchmarks. We briefly comment on three of the known efforts.

At HICSS-26, Kinicki and Finkel [12] report on measurements made using the WPI Benchmark Suite to compare the performance of Mach 2.5 versus Mach 3.0 supporting client-server interactions between two HP 486 PCs communicating via TCP/IP. The benchmarks allow for multiple clients to interact over an Ethernet with a single server. This study does provide insight into comparing operating system support for client-server activities, but the benchmarks are limited in that the clients are not distributed over all the machines on the network.

LADDIS [18] is a distributed application benchmark which specifically focuses on evaluating NFS performance on a network. Accepted as a SPEC benchmark, LADDIS is designed to be scalable and managed from a single prime client. Using a "standard mix" of NFS operations, LADDIS is a benchmark to be used to compare NFS implementations. Multiple clients per server are supported and a number of tunable control parameters are available. Some LADDIS disadvantages are: it only evaluates NFS behavior, reliance on UDP, and only balanced round-robin resource allocation. The ability to disable UDP checksums is necessary for vendor compatibility, but causes concern when comparing competing vendor platforms.

Netperf [9] is a public domain network benchmark developed by the Networking Performance Team at the Information Network Division of Hewlett-Packard Company. It performs TCP and UDP tests using bulk data transfer from client memory to server memory (i.e. no disk file transfer is involved). The measured performance indicators include elapsed times, throughput, etc. Netperf is documented as a tool, but the directions for its usage and the interpretation of its results are not available.

The DCSP benchmark presented in Section 2 fits in with these three network benchmarks in that it is an applications level network benchmark. It is intended as a general purpose network benchmark to be used to solve two classic benchmarking problems: (1) performance evaluation, comparison and selection of competitive networked computer systems and (2) performance monitoring, tuning, and optimization of existing networks.

2 The Organization of Communicating Client/Server Pairs

The nature of workload in computer networks can be abstracted to a combination of local processing performed by individual nodes, and unidirectional or bidirectional communication between pairs of nodes. Bidirectional communication is the most general model of network workload, because it can include a scalable amount of local processing, as well as file transmit and receive operations. Additional reasons to organize network benchmarks using a cyclic data transfer (i.e. the transfer of data from a client to a server, and then back to the client) are: (1) scalability, (2) flexibility, and (3) elimination of clock synchronization problems. The cyclic data transfer operation can be repeatedly performed for any specific number of pairs of nodes in the network. The total network workload can be scaled and adjusted by selecting both the number and the distribution of communicating client/server pairs per network. This yields high flexibility in workload design and is applicable for all types of networks. An important advantage of this organization is that performance measurements can be accomplished independently on both the client and the server side of each communicating client/server pair, eliminating the problem of clock synchronization in the network (performance indicators are measured independently for each node). Since all processes running on each node collect the performance data, it is possible to obtain a detailed insight into the activity and performance of both the individual nodes and the whole network.

There are a number of other requirements that network benchmarks should satisfy. First, in all cases it is highly desirable that all network performance measurements can be accomplished by a single performance analyst. The workload must be suitable for all sizes of networks (LAN, MAN, WAN), and all major network topologies. The benchmark should use the same hardware and system software resources that are used by actual network users. In addition, if benchmarking is performed in a Unix environment, then benchmarks should be portable across most Unix systems.

The basic structure of our client/server program is presented in Fig. 1. The benchmark program has two entry points corresponding to the client and the server processes. For each data transfer operation there are four combinations for the location of the source and destination of data, and data transfers can be organized (1) from memory to memory, (2) from memory to disk, (3) from disk to memory, and (4) from disk to disk. In this way it is possible to include or exclude the activity of the file management part of the operating system. As in all black-box measurement methods it is possible to subtract elapsed times for measurements performed with and without a selected feature in order to isolate the net effect of the analyzed feature (in this case the Unix file management system).

The main loop consists of three operations: SEND, RECEIVE, and WAIT. The client process starts by requesting connection to the server node. After establishing the connection and performing other initialization operations the client process repeats a number of SEND-RECEIVE-WAIT cycles. In each cycle it sends a data file to the server, receives the same data file back from the server, and then waits a random time interval before starting a new SEND-RECEIVE-WAIT cycle. The transmitted data file consists of M data quanta whose size is Q. The transmitted data file size is MQ, and it can be adjusted by selecting the values of M and Q. The maximum control over the data transmission process is obtained if the quantum size is selected to be the default TCP socket buffer size, and in our experiments we used $Q = 4$ KB (4096 bytes). In cases where the random wait time is uniformly distributed it can be adjusted by selecting its average value $\overline{T}_w$ (of course, the highest traffic intensity can be achieved by selecting $\overline{T}_w = 0$). When the measurement is completed the client process sends an end-of-measurement message to the server.

The server process starts by accepting the connection from the client and then repeats a RECEIVE-WAIT-SEND cycle until it receives the client's message that the measurement is to be terminated. The client and the server activities are integrated in the same program, but they differ both in the start of measurement and the end of measurement process. The server process should be started before the client processes.

The performance measurement consists of a selected number of cycles and at the end of each cycle we log a record consisting of the following data:

- $u[i]$ = accumulated user's CPU time at the beginning of the i^{th} send operation,
- $s[i]$ = accumulated system's CPU time at the beginning of the i^{th} send operation,
- $t_1[i]$ = begin time of the i^{th} send operation,
- $t_2[i]$ = end time of the i^{th} send operation,
- $t_3[i]$ = end time of the i^{th} receive operation.

The recording process starts at the predefined time t_{start} and ends at the predefined time t_{stop}, but the client and server processes will remain active until the time t_{end}. In other words, if the real time t satisfies the condition $t_{start} < t < t_{stop} < t_{end}$, then all client and server processes generate log files containing the following records: $u[i], s[i], t_1[i], t_2[i], t_3[i] \quad , \quad i = 1, \ldots, m$. Here m denotes the number of cycles performed during the recording interval $[t_{start}, t_{stop}]$. The evaluator must start (manually or automatically) all client and server processes sufficiently before the time t_{start}. In such a way the network traffic attains a steady state *before* the beginning of the measurement process. The process of recording measured values in the log file terminates at the time t_{stop} when the traffic is still in steady state. Thus the clocks on various nodes need not be synchronized, because the data transfer will continue until the time t_{end}. At the time t_{end} all client processes send the end-of-measurement message to their corresponding server processes, and both the client and the server processes terminate. Therefore, if the maximum difference between the unsynchronized clocks of nodes in the network is less than $t_{end} - t_{stop}$ then the whole measurement will

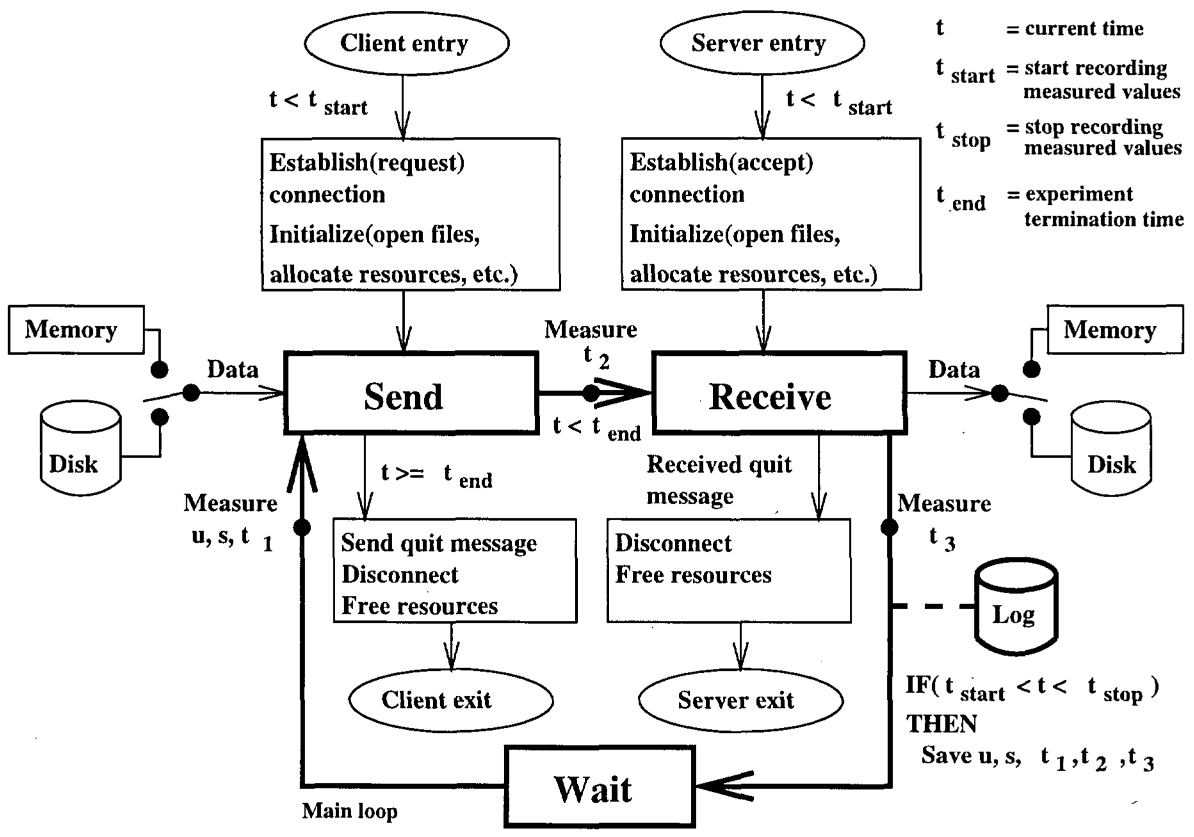

Client entry
Server entry
t = current time
t start = start recording measured values
t stop = stop recording measured values
t end = experiment termination time
t < t start
t < t start
Establish(request) connection
Initialize(open files, allocate resources, etc.)
Establish(accept) connection
Initialize(open files, allocate resources, etc.)
Memory
Disk
Data
Send
Measure t 2
t < t end
Receive
Data
Memory
Disk
t >= t end
Received quit message
Measure u, s, t 1
Send quit message
Disconnect
Free resources
Disconnect
Free resources
Measure t 3
Log
Client exit
Server exit
IF(t start < t < t stop) THEN Save u, s, t 1 ,t 2 ,t 3
Wait
Main loop

Fig. 1: Basic Structure of the Client/Server Program

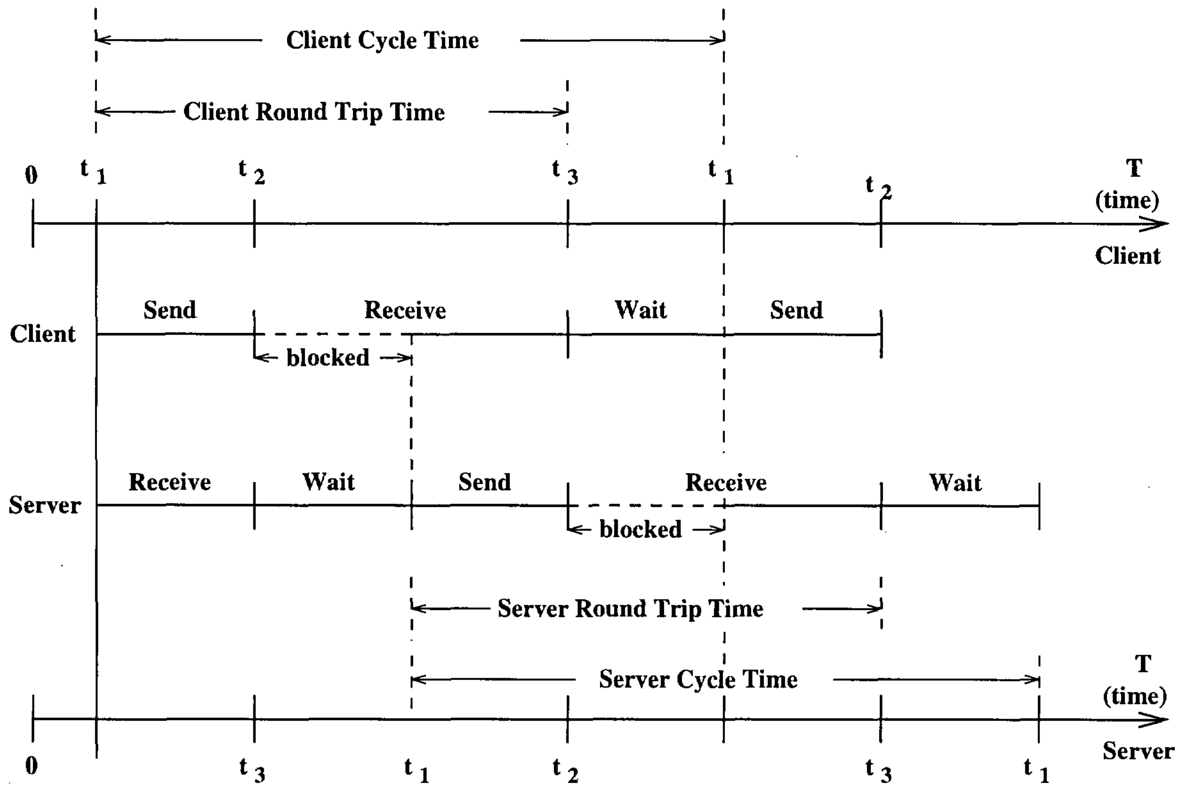


Fig. 2: Timing Diagram for a Client/Server Pair

be performed during the steady state of the network traffic. By selecting sufficiently large values of t_{end} any difference between unsynchronized clocks can be easily compensated. This is important in all cases where we have no control over some (usually remote) node clocks.

Using the recorded times $u[i]$, $s[i]$, $t_1[i]$, $t_2[i]$, $t_3[i]$, $i = 1, \ldots, m$ we can define the following performance indicators of the i^{th} cycle $(i = 1, \ldots, m-1)$:

- Send Time: $T_s[i] = t_2[i] - t_1[i]$
- Receive Time: $T_r[i] = t_3[i] - t_2[i]$
- Wait Time: $T_w[i] = t_1[i+1] - t_3[i]$
- Round Trip Time: $R[i] = T_s[i] + T_r[i] = t_3[i] - t_1[i]$
- Cycle Time: $T_c[i] = t_1[i+1] - t_1[i]$
- User Time: $T_{user}[i] = u[i+1] - u[i]$
- System Time: $T_{sys}[i] = s[i+1] - s[i]$
- CPU Time: $T_p[i] = T_{user}[i] + T_{sys}[i]$
- Processor Utilization per process: $U[i] = T_p[i]/T_c[i]$

These performance indicators are random variables that depend on the workload distribution and parameters Q, M, and $\overline{T}_w$, as well as on the performance of the network hardware and software. Competitive networks can be compared using the parameters of distributions of these random variables, primarily the average values $\overline{T}_s$, $\overline{T}_r$, $\overline{T}_w$, $\overline{R}$, $\overline{T}_c$, $\overline{T}_{user}$, $\overline{T}_{sys}$, $\overline{T}_p$, and $\overline{U}$.

The changes of states and relationships of round-trip cycles for client and server processes are shown in Fig. 2. In cases where all nodes are identical and the network workload is balanced (i.e. each node has the same number of server and client processes) the mean values of client and server cycle times and client and server round-trip times should be the same. In all other cases (unbalanced workload, different routing, etc.) that is not the case.

Network measurements based on a specific form of data transfer can also be realized using any of network-oriented Unix commands such as *ping*, *ftp*, *mail* or *finger*. The simplest Unix-based cyclic data transfer can be realized using the *ping* command. The *ping* mechanism is based on ICMP and limited to repeating round-trips between two nodes with packets having machine dependent size of at least 64 and no more than 4096 bytes. It reports only the packet loss statistics, and the round-trip times. It is primarily designed for manual fault isolation. and its use in system evaluation and selection studies is severely limited. However, an expanded version of *ping*, called *windowed ping*, can generate a controlled volume of network workload and can be used for testing network performance. This approach was successfully implemented by Mathis [14] to test and evaluate the IP performance of specific network products and technologies. In the case of system evaluation and selection studies this method has a limited value, since it is primarily Internet oriented, and has some of limitations of the original *ping* approach.

The proposed cyclic data transfer measurements offer more flexibility than the Unix command approach. The DCSP workload is designed particularly for network evaluation and selection analyses: it has more adjustable parameters than Unix commands, it can be organized to collect more performance indicators, and it is suitable for all types of networks. In addition, the DCSP benchmark is useful for the performance analysis of various file location strategies (memory, local disks, or NFS disks), and for measuring the operating system overhead.

3 The DCSP Network Workload

The communicating client/server pair is the basic building block of our synthetic network workload. The structure and the traffic intensity of the network workload can be determined by defining a suitable distribution of client/server pairs over the nodes of the analyzed network. We assume the general network configuration shown in Fig. 3, where workstations (or more complex computer systems) share a common transport medium with or without network file servers. The communicating client/server pairs can be organized in two ways as shown in Fig. 4. We use either asymmetrical (simplex) or symmetrical (duplex) communication between two nodes. In the case of asymmetrical communication one node has a client process and the other node has the server process. In the symmetrical case the communication is bidirectional: each node includes both client and server processes. If a higher level of network load is needed, then each node will have more than one client and server process.

The DCSP network workload consists of any number, and any distribution of symmetrical and/or asymmetrical client/server pairs. It is obvious that the number of different network workloads that can be built in this way is practically unlimited. It is also clear that any workload can be obtained by using multiple copies of the same basic client/server benchmark program. Therefore, this family of network workloads is extremely flexible, scalable, controllable, and also reliable, because it consists of well tested uniform basic elements. All these properties certainly contribute to our ability to create proper network workloads; they are necessary, but not sufficient to claim that selected DCSP workloads properly represent specific real network workloads and automatically generate meaningful network performance indicators. At this point we can proceed in two directions: (1) to develop basic standard network workloads that use the most common network topologies and generate meaningful general network performance indicators, and (2) to develop a methodology for modeling arbitrary real network workloads using optimum distributions of communicating client server/pairs. Of course, both directions are feasible, but only the first direction will be covered in the rest of this paper.

The basic network and workload parameters of the network benchmarking process using communicating client/server pairs are the following:

- the number of nodes in the network: n ,
- total number of client and server processes per network: N ,
- the average number of processes per node (node load): $L := N/n$,
- the individual node load: $L_j, \; j = 1, \ldots, n$.

A specific organization of network workload can be achieved by adopting a given distribution of client and server processes over the nodes of a network. In the case of solving the problem of comparison and selection of competitive networks we suggest the use of *balanced workload* where $L_1 = L_2 = \ldots = L_n = L = const.$ Of course, the balanced workload is an idealized case, but it is the most important special case of network workload, because load balancing efforts are usually aimed at achieving this specific case.

In cases where the purpose of network performance measurement is to study the sensitivity of the network to nonuniform load distributions, the workload must be unbalanced, and the number of processes per node will vary from node to node. Such workloads can be easily organized by nonuniformly distributing the communicating client/server pairs performing the cyclic data transfer. The unbalanced network workloads can be organized in more different ways than the balanced workloads, and an important special case that models a network file server (NFS) and a number of diskless workstations is exemplified in Fig. 5. This characteristic structure of workload will be called the star structure. The clients take data from local memory, and server processes (all located on the network file server node) save data on the server disk(s). This type of the DCSP synthetic workload simulates the high intensity of NFS disk traffic and provides an adjustable level of data communication with clients.

Balanced network workloads should be organized with varying levels of node load. According to the increasing level of node load we suggest the following classification of balanced network workloads:

- **Pairs Structure:**
  - asymmetrical: $N = n$, $L = 1$
  - symmetrical : $N = 2n$, $L = 2$
- **Ring Structure**
  - unidirectional: $N = 2n,\; L = 2,\; (n > 2)$
  - bidirectional : $N = 4n,\; L = 4,\; (n > 2)$
- **Combinatorial Structure:**
  $N = 2n(n-1)$, $L = 2(n-1)$, $(n > 3)$.

Assuming an even number of nodes the asymmetrical pairs structure of network workload for the case of a typical Ethernet-based LAN is exemplified in Fig. 6. This is the case with the lowest node load, $L = 1$. In the symmetrical case

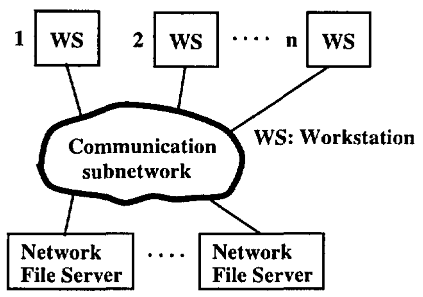


Fig. 3: General Network Configuration

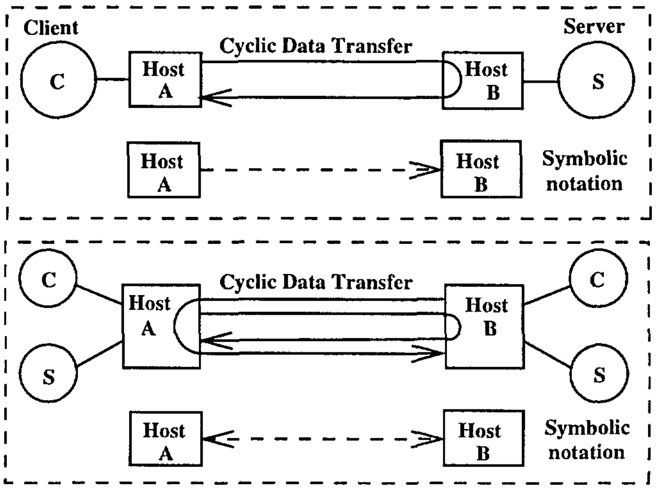


Fig. 4: Symbolic notation of Simplex & Duplex Communication

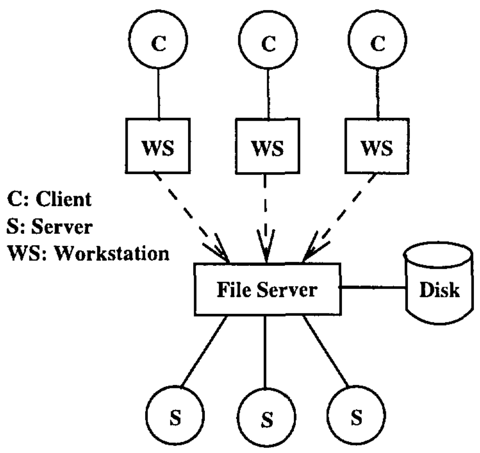


Fig. 5: Unbalanced DCSP workload to model workstations with a file server

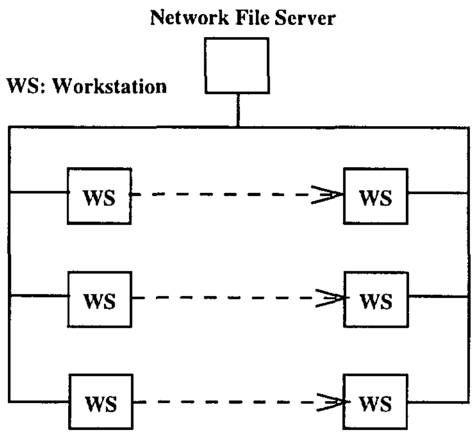


Fig. 6: Pairs Structure of the DCSP Workload

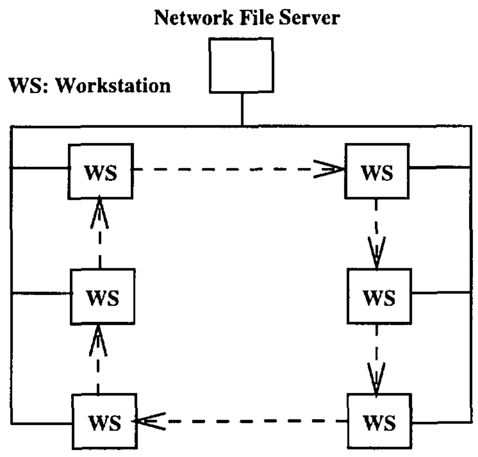


Fig. 7: Ring Structure of the DCSP Workload

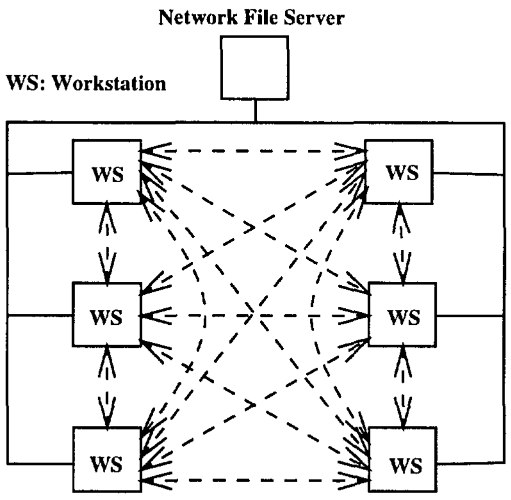


Fig. 8: Combinatorial DCSP Workload

the communicating nodes remain the same but the bidirectional communication is used, yielding $L = 2$. In the pairs structure each node communicates with only one node in the network.

An increased level of communication and node load can be achieved by using the ring structure of network workload where each node always communicates with two adjacent nodes. The unidirectional ring structure is exemplified as a LAN shown in Fig. 7. In such a case each node has a client and a server process, yielding the node load $L = 2$. However, the node load can be easily doubled by using the bidirectional communication. In such a case we have two clients and two servers per each node, and consequently $L = 4$.

In cases where a very high level of load is needed it is convenient to use the combinatorial structure of network workload exemplified in Fig. 8. This workload always uses bidirectional communication. Each node acts as both the client and the server for the remaining $n - 1$ nodes. Consequently, each node must support $L = 2(n - 1)$ processes.

For all presented structures of network workload the network traffic intensity can be adjusted by selecting (1) the number of nodes, (2) the unidirectional/bidirectional communication, (3) the location of data, (4) the average wait time, and (5) the size of transmitted data. These adjustable parameters provide for high flexibility in selecting a suitable network workload. In cases where the node load is low it is possible to use a low or zero wait time. However, in cases of high (and especially combinatorial) node load, the nonzero wait time is the primary factor for adjusting the level of network workload.

The DCSP workload enables the measurement of the file access overhead in the way illustrated in Fig. 9. This figure shows the main hardware and software components of each node and the way the DCSP benchmark controls the location of data used in the cyclic data transfer. If the data are located in the application buffer (which is in the main memory), then the file management part of Unix will not be active, and the system part of the CPU time will mainly reflect the activities of TCP/IP. However, if we store data on local disks, then the corresponding system part of the CPU time increase reflects the file management overhead. In addition, the total round-trip time also increases, due to disk activities performed by both the client node and the server node.

4 A Standard Classification of Network Measurements

Let us consider a network having n identical nodes and optional network file servers connected by a transport medium, as shown in Fig. 3. We assume that individual nodes can be diskless workstations, or workstations or servers having local disk units.

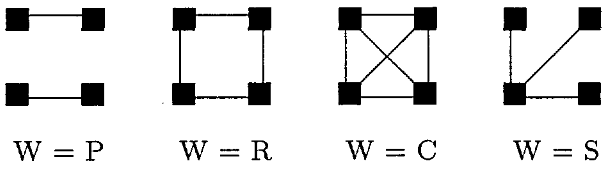
Similar to the classification of queuing models using letter codes separated by slashes, we propose a standard classification of network measurements in the following form:

$$\mathbf{W/D/C/n : M/Q/\overline{T}_w/L}$$

where W denotes the organization of workload, D denotes the location of data, C is the type of communication between nodes, and n denotes the number of active nodes in the network (with or without the NFS nodes). In the case of the DCSP benchmark the part of the description following the colon $(M/Q/\overline{T}_w/L)$ defines more detailed workload parameters (number of quanta, quantum size in kilobytes, the average wait time in seconds, and the average node load), and in some cases (where it has an assumed value) it can be omitted. Following is the list of main options:

1. Workload structure (W) options:
   - **P** = client/server pairs
   - **R** = ring structure
   - **C** = combinatorial structure
   - **S** = star structure
2. Data location options (D):
   - **L** = local disk (workstation)
   - **E** = external disk (network file server)
   - **M** = data generated and stored in the local main memory
3. Communication between pairs of nodes (C):
   - **A** = asymmetrical (unidirectional communication)
   - **S** = symmetrical (bidirectional communication)

In a trivial case all communicating client and server processes can be located on the same node; however, we normally assume $n > 1$. If $n = 2$ then the only interesting workload is $W = P$. Generally, the case $W = P$ assumes that n is an even number, and that the workload is balanced. If $n = 3$ then there is no difference between the cases $W = R$ and $W = C$; these cases differ only if $n > 3$. Therefore, the difference between the basic workloads (P,R,C,S) becomes visible only if $n \geq 4$. For example, if $n = 4$ then the structure of four basic workloads is



The proposed workload classification technique yields a spectrum of well defined network workloads: if $n > 3$ then for any number of active (communicating) nodes there are $4 \cdot 3 \cdot 2 = 24$ different W/D/C types of workload. Users of the DCSP benchmark can easily select and specify a desired type of workload. For example, the notation $R/L/S/6$ denotes that the corresponding network performance measurement is realized using the ring workload structure, data located on local disks of each workstation, and symmetrical communication where each of the six workstations includes two client and two server processes. So, the $R/L/S/6$ case creates a network workload of $nL = 6 \cdot 4 = 24$ processes.

5 Network Performance Measurements Using the DCSP Benchmark

To verify our approach to network workload design and network benchmarking, we performed three groups of experiments. The first group of experiments was performed to verify the use of the DCSP benchmark in performance comparison of three different LAN configurations. These experiments show user-level measurements of selected LAN performance indicators. The measured indicators can be used as inputs in the process of comparing competitive systems and selection of the most convenient option. The second group of experiments investigate the behavior of LAN's when the DCSP workload increases, causing network saturation phenomena. The third group of experiments illustrate the use of the DCSP benchmark in WAN performance measurement.

5.1 LAN performance comparison

In this experiment we used three local area network configurations: (1) DEC network based on DECstation 5000/20, (2) HP network based on HP 9000/834, and (3) SUN network based on SUN SPARCstation IPC workstations. In our experiments the workstations were connected by Ethernet, and the DCSP benchmark was the only activity in the network. The network performance indicators for two elementary pairs and ring structures with zero wait time are presented in Table 1, where all times are expressed in seconds. To create a high intensity of network activity we applied a zero wait time, which causes the equality of the round-trip time and the cycle time. The purpose of the experiment is to exemplify the expected values of performance indicators and to show how the proposed network benchmarking process can contribute to the comparison of various networks.

The most important results shown in Table 1 are the average round-trip time ($\overline{R}$) and the average processor utilization per process ($\overline{U}_1$). The workload is balanced and the obtained value of $\overline{R}$ is approximately the same for all nodes and for all client and server processes in the network. Therefore, we can define the global network throughput, X, as the total number of round-trips that the analyzed network performs in a time unit (usually a second):

$$X = \frac{nL}{2\overline{R}} \ .$$

Similarly, we can define the average processor utilization at each node, $\overline{U}$, as follows:

$$\overline{U} = L\overline{U}_1 \ .$$

For example, in the presented R/L/A/3 case we have $L = 2$, $n = 3$, and the resulting performance indicators for DEC, HP, and SUN networks are

$$\begin{aligned} X &= 1.75 \ sec^{-1} \quad (DEC) \\ &= 0.77 \ sec^{-1} \quad (HP) \\ &= 1.20 \ sec^{-1} \quad (SUN) \ . \end{aligned}$$

These throughputs show differences in performance that can be used for a global comparison of the presented networks. For all three networks the processor utilization at each node is approxi-

Table 1: *Performance Indicators for DEC, HP, and SUN Networks*

NETWORK WORKLOAD **P/L/A/2 : 10/4/0/1**		$\overline{T}_s$	$\overline{T}_r$	$\overline{T}_w$	$\overline{R}$	$\overline{T}_c$	$\overline{T}_{user}$	$\overline{T}_{sys}$	$\overline{T}_p$	$\overline{U}_1$[%]
DECstation 5000/20	Client	0.22	0.90	0	1.12	1.12	0.48	0.10	0.58	51.60
	Server	0.23	0.90	0	1.12	1.12	0.48	0.10	0.58	51.81
HP 9000/834	Client	1.07	1.50	0	2.56	2.57	1.18	0.10	1.28	49.80
	Server	1.06	1.51	0	2.57	2.57	1.19	0.09	1.28	49.70
SUN Sparc IPC	Client	0.68	1.02	0	1.70	1.70	0.80	0.06	0.86	50.27
	Server	0.76	0.94	0	1.70	1.70	0.74	0.06	0.80	47.13

NETWORK WORKLOAD **P/L/S/2 : 10/4/0/2**		$\overline{T}_s$	$\overline{T}_r$	$\overline{T}_w$	$\overline{R}$	$\overline{T}_c$	$\overline{T}_{user}$	$\overline{T}_{sys}$	$\overline{T}_p$	$\overline{U}_1$[%]
DECstation 5000/20	Client	0.34	1.81	0	2.15	2.16	0.48	0.11	0.58	27.07
	Server	0.35	1.81	0	2.15	2.16	0.48	0.10	0.58	26.93
HP 9000/834	Client	1.94	2.99	0	4.93	4.94	1.18	0.11	1.29	26.15
	Server	1.86	3.07	0	4.93	4.94	1.18	0.10	1.28	25.87
SUN Sparc IPC	Client	1.21	1.89	0	3.09	3.10	0.79	0.07	0.86	27.85
	Server	1.36	1.73	0	3.09	3.09	0.75	0.06	0.81	26.24

NETWORK WORKLOAD **P/M/S/2 : 10/4/0/2**		$\overline{T}_s$	$\overline{T}_r$	$\overline{T}_w$	$\overline{R}$	$\overline{T}_c$	$\overline{T}_{user}$	$\overline{T}_{sys}$	$\overline{T}_p$	$\overline{U}_1$[%]
DECstation 5000/20	Client	0.07	0.12	0	0.18	0.19	0.00	0.07	0.07	39.78
	Server	0.07	0.12	0	0.19	0.19	0.00	0.07	0.07	37.15
HP 9000/834	Client	0.20	0.26	0	0.46	0.46	0.00	0.08	0.08	17.82
	Server	0.19	0.27	0	0.46	0.46	0.00	0.08	0.08	17.89
SUN Sparc IPC	Client	0.07	0.12	0	0.19	0.20	0.00	0.05	0.05	29.31
	Server	0.07	0.12	0	0.19	0.20	0.00	0.05	0.05	28.88

NETWORK WORKLOAD **R/L/A/3 : 10/4/0/2**		$\overline{T}_s$	$\overline{T}_r$	$\overline{T}_w$	$\overline{R}$	$\overline{T}_c$	$\overline{T}_{user}$	$\overline{T}_{sys}$	$\overline{T}_p$	$\overline{U}_1$[%]
DECstation 5000/20	Client	0.24	1.48	0	1.72	1.72	0.48	0.11	0.59	34.17
	Server	0.26	1.45	0	1.71	1.72	0.48	0.10	0.58	34.36
HP 9000/834	Client	1.47	2.44	0	3.91	3.92	1.18	0.12	1.30	33.24
	Server	1.26	2.65	0	3.91	3.91	1.18	0.12	1.30	33.28
SUN Sparc IPC	Client	0.95	1.56	0	2.50	2.50	0.80	0.07	0.86	34.53
	Server	1.04	1.46	0	2.50	2.51	0.75	0.07	0.81	32.63

NETWORK WORKLOAD **R/L/S/3 : 10/4/0/4**		$\overline{T}_s$	$\overline{T}_r$	$\overline{T}_w$	$\overline{R}$	$\overline{T}_c$	$\overline{T}_{user}$	$\overline{T}_{sys}$	$\overline{T}_p$	$\overline{U}_1$[%]
DECstation 5000/20	Client	0.35	2.82	0	3.16	3.16	0.48	0.11	0.59	18.78
	Server	0.43	2.73	0	3.16	3.16	0.48	0.11	0.59	19.10
HP 9000/834	Client	2.89	4.76	0	7.65	7.65	1.19	0.13	1.32	17.22
	Server	2.47	5.18	0	7.65	7.65	1.19	0.14	1.33	17.37
SUN Sparc IPC	Client	1.84	2.41	0	4.25	4.25	0.80	0.07	0.86	20.47
	Server	1.29	2.81	0	4.10	4.11	0.75	0.07	0.82	20.65

mately 67%. Other experimental results presented in Table 1 can be summarized as follows:

- Client and server processes behave in a consistent way and generate identical results within the range of inevitable measurement errors.

- In all cases the recorded send time is less than the receive time, reflecting the way in which send and receive calls are implemented in Unix.

- In all cases where the data are located on disks, the measured user's CPU time is substantially larger than the system's CPU time, regardless of the fact that the majority of processing consists of executing system's programs. This reflects the way in which Unix accounts for user and system CPU times.

- In cases where the data is located in memory, the user part of the CPU time becomes negligible, and the round-trip times substantially reduce with respect to the cases which include disk accesses. These results show that the majority of the CPU time is spent supporting disk file access operations. In addition, our other measurements indicate that disk operations are frequently responsible for creating a bottleneck and causing saturation phenomena in multinode networks.

- In the case of paired workload (P/L/A/2 and P/L/S/2), the average processor utilization per single process approximately satisfies the relation $\overline{U}_1 = 1/nL$. The total processor utilization for a node in the network is $\overline{U} = L\overline{U}_1 = 1/n$.

- In the case of the ring workload (R/L/A/3 and R/L/S/3), the average processor utilization per single process approximately satisfies the relation $\overline{U}_1 = 2/nL$. This kind of result can appear if the communicating client and server processes make the corresponding processors busy during the data transfer. Then, at any time, 2 out of n processors are busy, and $n-2$ processors are idle waiting for a chance to communicate. In such cases the average processor utilization for each workstation is $\overline{U} = L\overline{U}_1 = 2/n$.

- Symmetrical workloads in the case of the zero wait time approximately satisfy

$$\overline{U}_{1(sym)} = \overline{U}_{1(asym)}/2\ , \quad \overline{U}_{sym} = \overline{U}_{asym}\ .$$

- The comparison of network performance can be based on average round-trip times. The resulting ratios (the ratio to the best system whose performance is denoted 1) are:

$$DEC : SUN : HP = 1 : 1.5 : 2.3\ (P/L/A/2)$$
$$DEC : SUN : HP = 1 : 1.4 : 2.3\ (P/L/S/2)$$
$$DEC : SUN : HP = 1 : 1.5 : 2.3\ (R/L/A/3)$$
$$DEC : SUN : HP = 1 : 1.3 : 2.4\ (R/L/S/3)$$

  These results are consistent and illustrate the use of the DCSP benchmark for system evaluation and comparison. Of course, in this case we are primarily interested in workloads which include disk activities. The role of disk performance is also visible from the fact that the corresponding ratios in the P/M/S/2 case are DEC : SUN : HP = 1 : 1 : 2.5 .

The goal of the presented analysis was to exemplify the applicability of the proposed DCSP workload, and not to compare competitive products. The availability of DEC, HP, and SUN networks was the primary reason for their comparison. It is easy to note that these networks are not sized as competitive products.

5.2 LAN Performance for Various Numbers of Nodes

This experiment was performed to investigate the use of the DCSP benchmark in the study of LAN saturation phenomena. LAN performance measurements were performed for two local networks of DECstations: the first with 2100's and the second with 5000/20's. We used three workloads differing only in data location: R/E/S/n:10/4/0/4, R/L/S/n:10/4/0/4, and R/M/S/n:10/4/0/4. Since L=4 the global network throughput (expressed in round-trips per second) is $X = 2n/\overline{R}$.

In the case of DECstation 2100 LAN the average round-trip times $\overline{R}(n)$ and the corresponding throughputs $X(n)$ for $2 \leq n \leq 7$ are shown in Figures 10 and 11. The cases which use data on local disks and data on external (NFS) disks differ

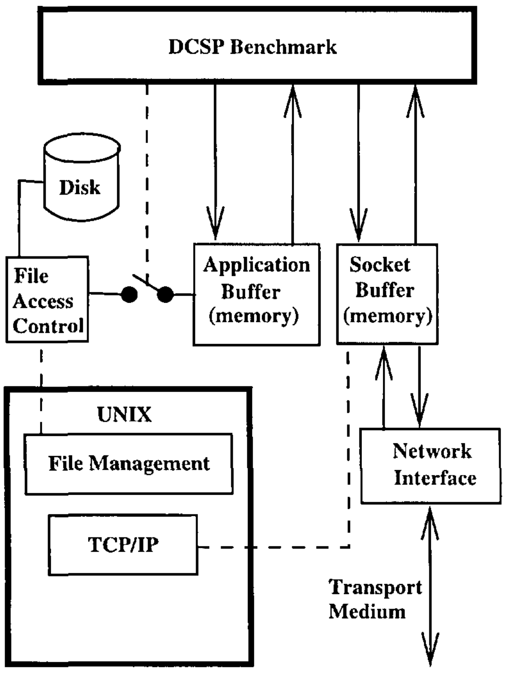


Fig. 9: Measurement of File Access Overhead

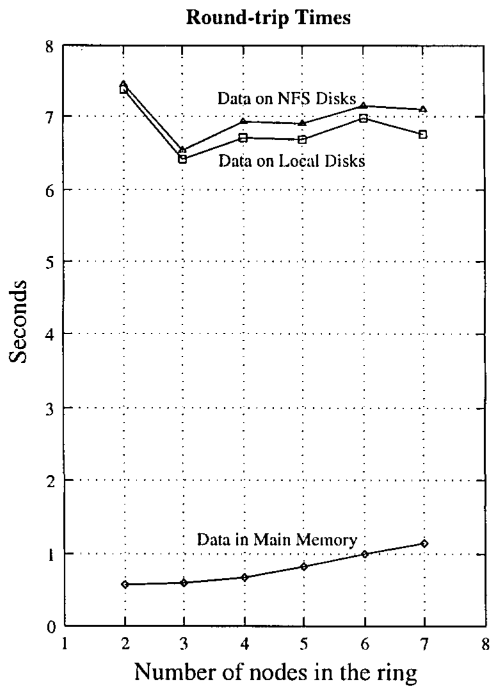


Fig. 10: Average round-trip times for DEC 2100 LAN running DCSP R/E/S/n, R/L/S/n, and R/M/S/n workloads

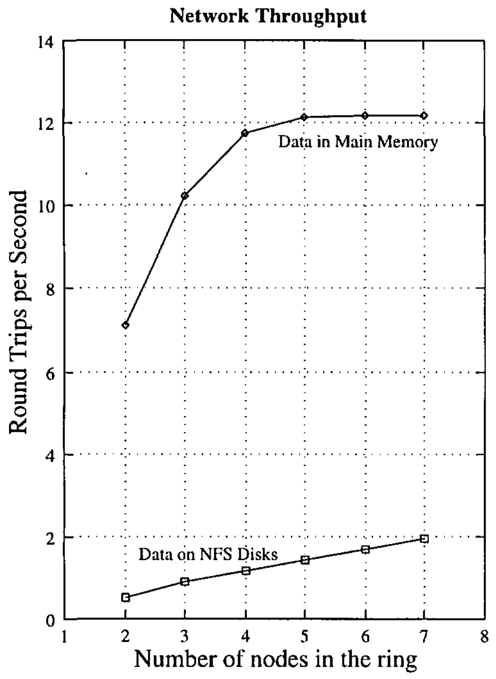


Fig. 11: Global network throughput for DEC 2100 LAN running DCSP R/E/S/n, and R/M/S/n workloads

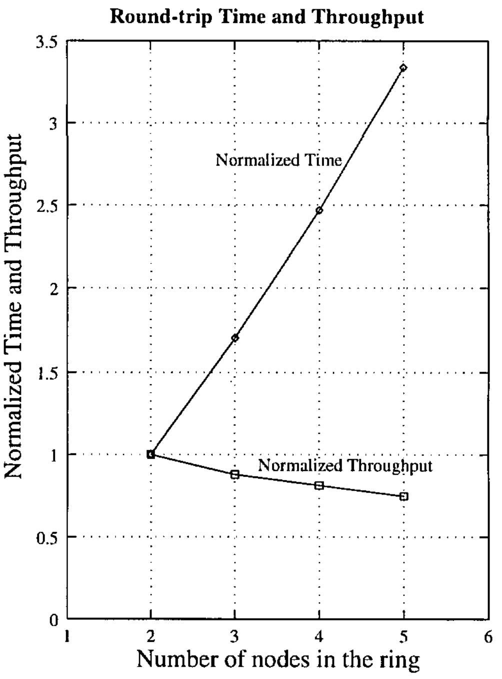


Fig. 12: Normalized round-trip times and throughputs for DEC 5000/20 LAN and DCSP R/E/S/n workload

insignificantly. This may support the contention that NFS caches disk files at local disks, supporting disk file I/O from local disk storage rather than from NFS via the network. In addition, this eliminates disk queuing effects on the NFS disk. Then the relatively slow local disks of these workstations become the bottleneck of the network. Without disk queuing, and far from other possible bottlenecks (e.g. those related to the transport medium), the round-trip times are almost constant, there is a moderate overlap in workstation activities and the global network throughput is an increasing function of n that shows no saturation.

The effects of slow disks become apparent if we disconnect disks and store data in the workstation memory (see Fig. 9). The resulting effect is a dramatic reduction of the round-trip time (more than 10 times for a small number of nodes). Now the network has a different bottleneck and the saturation effects are clearly visible in Fig. 11. The network with 5 nodes is already fully saturated; its maximum global throughput ($X = 12.25$ round-trips per second) can be used as a valuable global performance indicator.

In the case of the LAN with DEC 5000/20 workstations the results of the DCSP benchmark are summarized in Fig. 12. The presented results are normalized so that the unit values correspond to the initial case where $n = 2$. The round-trip time linearly increases, and an increasing overhead causes the global network throughput to be a decreasing function of n. The processor speed of DEC 5000/20 is greater than in the previous case of DEC 2100. Therefore, this type of saturation for R/E/S/n workload could be caused by the NFS disk queuing.

All presented measurements were performed using $T_w = 0$. Using $T_w > 0$ we can create additional opportunities for overlapped (simultaneous) usage of network resources yielding an increase in the number of workstations working without saturation.

5.3 WAN Performance Measurements

The DCSP workload can be used to get a user-level insight into some global aspects of WAN performance. Our experiment was based on the P/E/S/2:10/4/0/2 workload installed on two HP 9000 workstations communicating using the Internet. One workstation was located in Worcester, MA, and the other was in San Francisco, CA. The organization of WAN measurement is similar to LAN measurement based on the DCSP workload. Of course, the measured performance indicators are now substantially affected by transport delays between remote locations. Two typical distributions of the round-trip time, similar to the Poisson distribution, are shown in Fig. 13.

The round-trip central processor times (both the user part and the total time) are distributed similarly to the normal distribution with a very small standard deviation, and with practically constant system CPU time, as shown in Fig. 14. The shape of all presented distributions is similar to the LAN case. However, in the case of the round-trip time both the average value and the standard deviation are greater than in the LAN case.

As expected, the WAN traffic load affects the round-trip time distribution. The distributions presented in Fig. 13 show the difference between two three-hour experiments performed during the day (2-5 p.m. EST) and night (0-3 a.m. EST). The day experiment included 1122 round-trips yielding $\overline{R}_{day} = 8.4$ seconds and standard deviation $\sigma_{day} = 3.7$ seconds. The night experiment included 1817 round-trips yielding shorter round-trip times, $\overline{R}_{night} = 5.9$ seconds, and substantially smaller standard deviation $\sigma_{night} = 1.95$ seconds. The same workstations communicating in isolated LAN environment attain the maximum performance limit $\overline{R} = 4.9$ seconds and negligible standard deviation $\sigma = 0.08$ seconds. Therefore, this measurement shows a rather good performance of Internet communications, including low transport delays, and relatively small performance variations between the periods of high and low load. Of course, it is possible to find examples with different WAN performance results; our goal was not to provide a detailed analysis of WAN performance, but to show that relevant WAN performance indicators can be obtained from user-level measurements based on the DCSP workload.

The presented results can be compared to the ping round-trip times for the same pair of workstations. The ping measurement results (for day time conditions) are shown in Fig. 15. Not surprisingly, the ping times are substantially shorter than the corresponding DCSP times (ping results are limited to the transfer of 4 KB, but even 10

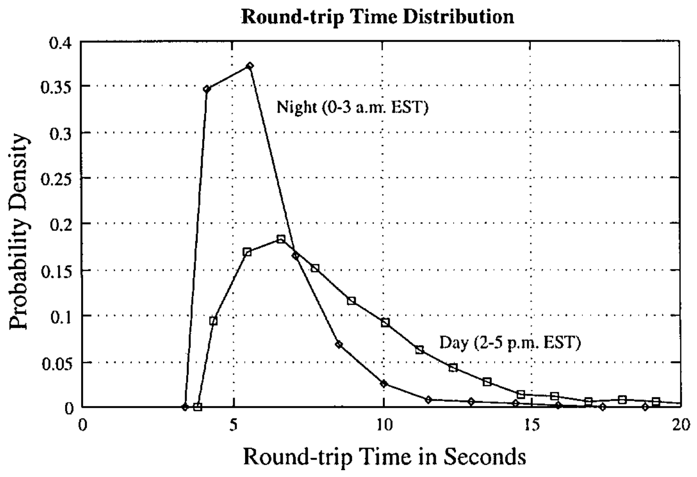


Fig. 13: Typical WAN round-trip time distributions for day and night conditions (DCSP P/E/S/2 workload)

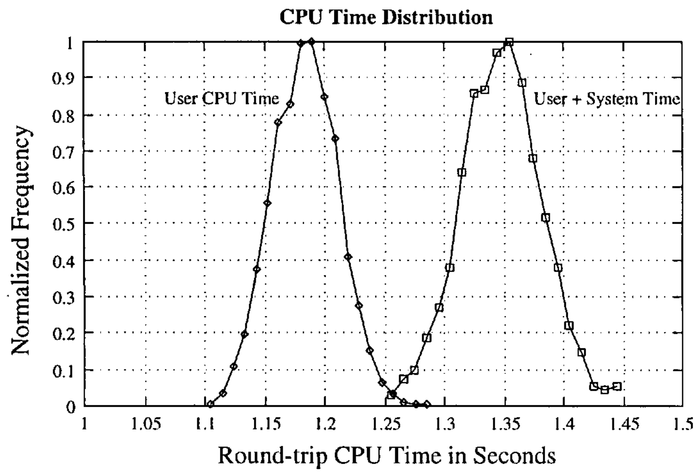


Fig. 14: Typical distributions of WAN round-trip processor time (DCSP P/E/S/2 workload)

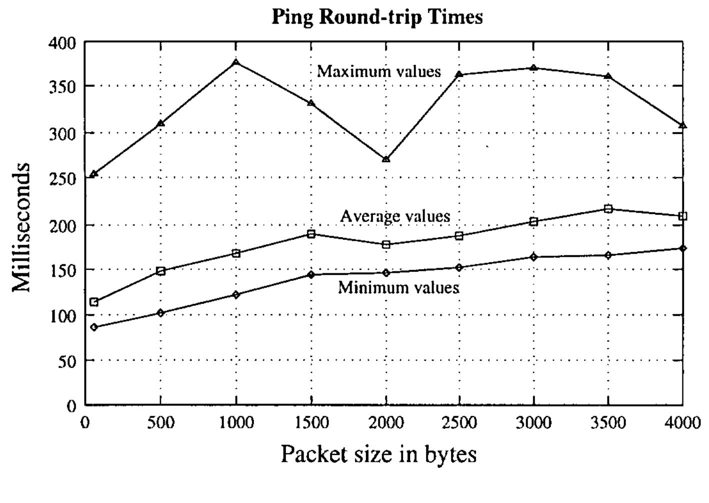


Fig. 15: Ping round-trip times between San Franscisco, CA, and Worcester, MA

such transfers can be completed in less than 2 seconds). This shows that the ping usage of network resources does not reflect and cannot represent the type of user-level workload which is characterized and generated by the DCSP benchmark.

6 Conclusions, Current Work, and Future Work

We have presented a simple black-box approach to benchmarking computer networks with a scalable network workload. Our goal was to provide a methodology for the user-level network performance measurement suitable for evaluation, comparison, and selection of computer networks. In addition, our workload is suitable for measurements of saturation phenomena in local networks, and for user-level WAN measurements.

Using a suitable number and various characteristic distributions of communicating client/server pairs we can organize a number of specific balanced and unbalanced scalable network workloads. These workloads are suitable for measuring, at each node, the following nine performance indicators: (1) send time, (2) receive time, (3) wait time, (4) round-trip time, (5) cycle time, (6) user CPU time, (7) system CPU time, (8) total CPU time, and (9) processor utilization. All indicators are measured using basic facilities available in C and Unix, without specialized performance monitoring tools. All client and server processes run at the user level and all measurements can be performed without support and/or consent of system programmers and administrators. Depending on the objectives of performance analysis, the DCSP workload can be executed either alone, or in addition to a real network workload.

We have also proposed a simple technique for standardized classification of network performance measurements based on specifying the type of network workload, the location of data for cyclic data transfer, the type of communication between nodes, and the number of nodes in the analyzed network. This classification can be modified and/or extended, and it was found very practical for organizing and controlling a large number of results generated by DCSP network benchmarks.

We verified the validity of our approach with measurements on local area network configurations consisting of HP, SUN, and DEC workstations. Our measurements show that the DCSP workload can provide user-oriented and user-level insight into the overall network performance and can be used as a means for comparing alternative network solutions, and making justifiable capacity planning and equipment procurement decisions. In addition, the presented experiments show that the DCSP workload can be used for identifying a variety of different dynamic behaviors of existing networks. In particular, by combining results obtained for various DCSP workloads which use data located in memory and on local and NFS disks, it is possible to identify and control both the overhead generated by Unix, and various LAN saturation phenomena.

In cases of networks with relatively large number of nodes it is necessary to have tools for automatic generation of the DCSP workload. In addition, we sometimes need to measure more detailed and/or more precise performance indicators than those that can be generated at the simple C/Unix user level. These requirements define the basic direction for the future work. This work is directed towards a *network performance measurement environment* which integrates three major components: (1) a DCSP network workload generator, (2) a network performance monitor, and (3) models for performance analysis, performance prediction, and capacity planning.

Our current work is focused on the development of a *control module* for the DCSP workload. This module generates communicating client/server pairs and distributes them on active nodes to create a desired structure of the network workload. In addition, it specifies all parameters of performance measurement, starts the measurement, and supervises the collection of measured data. At the end of measurement, it transfers and integrates all files containing measured results and creates a data base that is structured according to requirements of performance analysis modules.

The network performance monitor is a module specialized for generating specific measured data both for the DCSP workload and for the real network workload that can run concurrently with the DCSP workload. It performs traditional online monitoring of network performance indicators, enabling the analyst to trace dynamic phenomena, to experiment, and to adjust the parameters of the network hardware, software, and workload. In addition, the monitor also generates control data necessary for the control module. For example, the collecting of DCSP data can be conditioned in a variety of ways; in such cases, it should be temporarily suspended if the conditions are not satisfied, and automatically restarted when the conditions are satisfied.

Acknowledgment

The authors would like to thank Mr. Rufus J. Freddie for his participation in LAN measurements.

References

[1] Bershad, B.N., T.E. Anderson, E.D. Lazowska, and H.M. Levy, *Lightweight Remote Procedure Call.* ACM TOCS, 8(1), 1990, pp. 37-55.

[2] Cabrera, L., E. Hunter, M.J. Karels, and D.A. Mosher, *User-Process Communication Performance in Networks of Computers.* IEEE TSE, Vol 14, No. 1 (1988) pp. 38-53.

[3] Dixit, K. and J. Reilly, *SPEC Developing New Component Benchmark Suites.* SPEC Newsletter, Vol. 3, No. 4, pp. 14-17, Dec. 1991.

[4] Dujmović, J.J., *Workload Characterization, Benchmarking, and the Concept of Total Resources Consumption.* Proceedings of the Second International Conference on Computer Capacity Management (H.L. Bording and D.J. Schumacher, Ed.), San Francisco, April 8-10, 1980, pp. 151-163.

[5] Ferrari, D., G. Serazzi, and A. Zeigner, *Measurement and Tuning of Computer Systems.* Prentice-Hall, 1983.

[6] Funk, B.K., *SPEC - The Changing Computer Benchmark Consortium.* SPEC Newsletter Vol. 3, Issue 3, pp. 3-5, September 1991.

[7] Graphics Performance Characterization Committee, *The Effort to Standardize Performance Measurement.* GPC Quarterly Report, Vol. 2, No. 4, 1992, pp. 10-12.

[8] Hennessy, J.L. and D.A. Patterson, *Computer Architecture A Quantitative Approach.* Morgan Kaufmann, 1990.

[9] Hewlett-Packard Company, *Netperf: A Network Performance Benchmark.* Revision 1.7. Information Networks Division, Hewlett-Packard Company, March 5, 1993, pp. 1-19.

[10] Jain, R, *The Art of Computer Systems Performance Analysis.* John Wiley, 1991.

[11] Kay, J., and J. Pasquale, *The Importance of Non-Data Touching Processing Overheads in TCP/IP.* Computer Communication Review Vol. 23, No. 4 (1993) pp. 295-268.

[12] Kinicki, R.E. and D. Finkel, *Comparison of Mach Distributed Performance Using the WPI Benchmark Suite.* Proceedings of the 26th Hawaii International Conference on Systems Sciences, Vol. 2, 1993, pp. 40-49.

[13] Kohler, W., *How to Improve OLTP Performance and Price/Performance.* TPC Quarterly Report, Oct. 15, 1992, pp. 1-9.

[14] Mathis, M, *Windowed ping: an IP layer performance diagnostic.* Computer Networks and ISDN Systems 27 (1994) 449-495.

[15] Papadopoulos, C., and G. M. Parulkar, *Experimental Evaluation of SUNOS IPC and TCP/IP Protocol Implementation.* IEEE/ACM Transactions on Networking, Vol. 1, No. 2 (1993) pp. 199-216.

[16] Van Renesse, R., H. Van Staveren, and A.S. Tanenbaum, *The performance of the Amoeba Distributed Operating System.* Software - Practice and Experience 19(3), 1989, pp. 223-234.

[17] Thekkath, C.A. and H.M. Levy, *Limits to Low-Latency Communication High-Speed Networks.* ACM TOCS, 11(2), 1993, pp. 178-203.

[18] Watson, A. and B. Nelson, *LADDIS: A Multi-Vendor and Vendor-Neutral SPEC NFS Benchmark.* 1992 LISA VI - October 19-23, 1992, Long Beach, CA, pp. 17-32.

Petri Nets and Self-Stabilization of Communication Protocols

Wuxu Peng[1]
Department of Computer Science
Southwest Texas State University
San Marcos, TX 78666, U.S.A.
AND
Kia Makki
Department of Computer Science
University of Nevada Las Vegas
Las Vegas, Nevada 89154-4019
U.S.A.

Keywords: Petri nets, self stabilization, communication protocols

Edited by: Xindong Wu

Received: June 18, 1995 **Revised:** September 19, 1995 **Accepted:** September 29, 1995

Using Petri nets as the concurrent model, we discuss the issue of self-stabilizing communication protocols in this paper. We argue that (1) self-stabilizing extensions of communication protocols should not interfere with their normal operations by giving a new definition of self-stabilizing extension *and a ring net that meets the new definition, and (2) communication protocols are usually delay sensitive and* time-Petri nets *is a more suitable formal model (than ordinary Petri nets) to express self-stabilizing property. We also explore the possibility and suitability of using Petri nets with a new type of transitions – the* max transitions *– to express self-stabilizing communication protocols. Finally, to facilitate practical implementations, two types of interferences –* external *and* internal *interferences – are formally identified and discussed.*

1 Introduction

Communication protocols are an integrated and critical part of distributed systems and computer networks. Concisely specifying and efficiently verifying communication protocols are always major challenges to network designers.

The traditional approach to verifying communication protocols is similar to the well-known *weakest pre-condition* and *strongest post-condition* technique employed in verifying sequential programs. For example, to verify that a protocol is free of deadlocks, we may show that if the protocol starts from a legal state then it can never be in an illegal deadlock state, i.e. a state in which several machines are waiting for messages from one another in order to proceed. This traditional approach has severe problems, however. The distributed nature of distributed systems and computer networks makes such systems vulnerable to events absent in centralized systems. For instance, the communication channels linking machines may break or suffer from outside interference. Such a problem may cause either loss or corruption of messages. A corrupted message that passes the normal error detection mechanism may cause the system to enter an illegal state. A power surge in one machine may cause this machine to mal-function transiently and hence send conflict messages to different machines in the system. We observe that in both these two cases, a little perturbation can cause a system to stray from its legal states and remain there thereafter. To bring the system back to its legal states, drastic measures such as shutting down and rebooting the system have to be taken.

The concept of self-stabilization of computer

[1]The first author is supported in part by a faculty research enhancement grant from Southwest Texas State University.

systems was first proposed by E. W. Dijkstra in his seminal paper [5]. Informally, we say that a computer system is ***self-stabilizing*** if and only if the following two conditions are satisfied:

(1) if the system is in a legal state, it will remain in the set of legal states thereafter provided there is no perturbation;

(2) if some perturbation causes the system to enter an illegal state, then the system can return to a legal state within a *finite* amount of time *without* outside interference.

From the definition, it is clear that a self-stabilizing system is much more robust than a system that is merely logically correct. Self-stabilizing systems are *faults-resilient* in the sense that they can recover from system faults and failures by themselves. For example, when a computer system is rebooted, it is very important to ensure that all system parameters are properly initialized. For a self-stabilizing system, however, we can just turn on the power without concern about its initial state. The self-stabilization property guarantees that the system will eventually enter a legal state within a finite amount of time and remain there thereafter.

Although the concept of self-stabilization was proposed nearly twenty years ago, it did not receive its deserved attention until the middle of 1980s. After Dijkstra's paper [5], Kruijer in [14] considered the issue of self-stabilization in tree-structured systems with distributed control. Bastani, Yen and Chen considered the self-stabilization problem of a class of distributed programs for industrial process-control systems in [1]. Brown, Gouda, and Wu considered the self-stabilization problem for token ring systems in [2]. The behavior of each machine in the ring is characterized by a regular expression and three *non-interfering* and *delay-insensitive* self-stabilizing protocols were presented. Gouda and Multari examined the issue of designing self-stabilizing communication protocols in [9]. The most important conclusion from that paper is the observation that the existence of both timeout and infinite number of states is a necessary condition for a communication protocol to be self-stabilizing. Burns and Pachl showed in [3] a *uniform* self-stabilizing solution (i.e. each machine in the ring executes the same code) for unidirectional rings where the number of machines is *prime*.

Research on the issue of self-stabilization of Petri nets only started in the late of 1980s. Cherkasova, Howell, and Rosier considered the issue of self-stabilizing bounded Petri nets in [4]. Their definition of self-stabilization is based on reachability sets. Ghosh discussed in [8] the issue of self-stabilizing a special class of Petri net models – the marked graphs. His definition is based on sequences of firing transitions and self-stabilization is achieved by using inhibitor arcs.

In this paper we consider the issue of self-stabilization of communication protocols expressed in Petri nets. We first argue that self-stabilizing extension of any communication protocol should not interfere with the normal operation of the protocol. This claim is supported by a new definition of *self-stabilizing extension* and illustrated by a new self-stabilizing extension of a ring net from [8].

Timeout is a fundamental technique used in communication protocols. For instance, in a token ring, timeout is used to detect loss of tokens. In sliding window protocols, timeout is used to retransmit previously sent but not yet acknowledged messages. As pointed out in [9], timeout is a necessary condition for communication protocols to achieve self-stabilization. The need of timeout in communication protocols can be attributed to the distributed nature of distributed systems and computer networks. Unreliable communication channels may lose or corrupt messages. Individual machines can crash due to various hardware and/or software failures. The need of timeout indicates that communication protocols are delay sensitive, not delay insensitive. For example, long delays in communications are normally treated as errors because most users usually cannot tolerate an excessively long delay. Therefore delay insensitive protocols, such as the three elegant protocols in [2], may not be suitable for many practical applications.

One elegant model for delay sensitive communication protocols is *Time-Petri nets* (TPNs), introduced in [15]. Informally, a TPN is just an ordinary Petri net with additional restrictions on the timing of firing of transitions. We argue that TPNs are more suitable (than Petri nets) to express self-stabilizing property of communication protocols. We show that a time-sensitive token ring can be extended to be self-stabilizing using a TPN that is operating at half of the speed of the original net. The convergence rate (the rate at which a disturbed system returns to a legal state) of the extended net can be accurately bounded.

We also explore the possibility of using *max transitions* to express self-stabilizing communication protocols. The notion of max transitions was originally proposed by Ghodsi and Kant in [7] to model the abort semantics in performance measurement. We show that, with a modified definition, max transitions can often give a more compact representation of self-stabilizing protocols. Two examples, one is the ring net and another is the sliding window protocol, are given to illustrate the point.

We strongly believe that practical implementation consideration should be a critical factor in evaluating possible self-stabilizing extensions of communication protocols. A self-stabilizing extension, no matter how elegant theoretically, has little value if it could not be practically implemented. Out of this consideration, we formally identify two types of interferences – *external* and *internal* interferences – which should be avoided in self-stabilizing protocols.

The remainder of this paper is organized as follows. Section 2 provides necessary definitions and concepts. In particular, we define the concept of self-stabilizing extension of a Petri net. We believe that our definition is innovative. Section 3 Illustrates our self-stabilizing definition through examples. Section 4 discusses, out of practical considerations, how to construct self-stabilizing extension of communication protocols through *decentralized control*. In Section 5 we discuss, through two examples, how to use max transitions to express self-stabilization. The modeling power of Petri nets with max transitions is briefly discussed. Section 6 formally defines and discusses two types of interferences – *external* and *internal* interferences. Section 7 provides concluding remarks and future research topics.

2 Basic definitions

In this section we define most of the concepts used in this research, including Petri nets, extended Petri nets, max transitions and generalized Petri nets, self-stabilizing extensions of Petri nets. The definition of TPNs is delayed until Section 4.

Definition 2.1 (Petri nets) A Petri net is a four-tuple (P,T,I,O), where $P = \{p_1,\cdots,p_n\}$ is a finite set of **places**; $T = \{t_1,\cdots,t_m\}$ is a finite set of **transitions** satisfying $P \cap T = \emptyset$; $I : T \rightarrow P^\infty$ is the **input** function; and $O : T \rightarrow P^\infty$ is the **output** function, where P^∞ is the **multinet** over P.

Let **N** denote the set of nonnegative integers. A **marking** μ is an assignment of **tokens** to places of a Petri net, i.e. μ is a function from P to **N**:

$$\mu : P \rightarrow \mathbf{N}.$$

A **marked** Petri net is a five-tuple (P,T,I,O,μ), where $C = (P,T,I,O)$ is a Petri net and μ is a marking of C. A transition t in a marked Petri net with marking μ is **enabled** if $\forall p_i \in I(t)\ \mu(p_i) > 0$. A transition t enabled in a marking μ may **fire** and result in a new marking μ' by first removing one token from each of its input places and then adding one token to each of its output places, i.e. $\mu' = \mu - I(t) + O(t)$.

Definition 4 (*Extended Petri nets*) *An extended Petri net is a five-tuple (P,T,I,O,I'), where (P,T,I,O) is a Petri net, and $I' : P \rightarrow T$ is the set of* **inhibitor arcs**.

Let (P,T,I,O,I',μ) be a marked extended net. A transition $t \in T$ is enabled (with respect to μ) if $\forall p \in I(t)\ \mu(p) > 0$ and $\forall p \in I'(t)\ \mu(p) = 0$. As in the Petri net case, the firing of t will result in a new marking $\mu' = \mu - I(t) + O(t)$. Notice that if $I' = \emptyset$, then the extended net is an ordinary Petri net.

Definition 5 (*Generalized Petri nets*) *A generalized Petri net is a seven-tuple (P,T,I,O,I',M,S), where (P,T,I,O,I') is an extended net, $M \subseteq T$ is the set of* **max transitions**, *and $S : T \rightarrow 2^P$, where $\forall p \in S(t)$ there exists a path leading from place p to transition t, is the* **input set** *of max transitions. The enabling and firing of non-max transitions are the same as in extended net. A max transition t is enabled in a marking μ if $\exists p \in I(t)\ \mu(p) > 0$. An enabled max transition t may fire by first* *removing one token from each place in $I(t)$, then removing all tokens from each place in the input set $S(t)$, then adding one token to each place in $O(t)$.*

Following the notation used in [7], a max transition t will be denoted by a rectangle with an associated copyright sign @. Notice that our definition here is slightly different from the original definition in [7].

For a marked net with a given initial marking, more than one transition may be enabled and thus it is possible for more than one transition to fire simultaneously. Here we adopt the classic interleaving semantics for firing. The set L of all possible firing sequences starting from a given marking forms a formal language over T [17, 16] (i.e. $L \subseteq T^*$, where T^* is the Kleene closure of the set T).

Definition 6 *A net N_2 is a self-stabilizing extension of a net N_1 (with respect to a given marking) if and only if every firing sequence α generated by N_2 from an arbitrary initial marking can be expressed as $\beta\alpha'$ satisfying:*

(1) $\beta \in T_2^*$ and $\alpha' \in T_1^*$;

(2) there exists a finite sequence $\beta' \in T_1^*$ such that $\beta'\alpha'$ is a legal firing sequence in N_1.

Two comments are in order here. First, the above definition does not require the original net N_1 to be live or safe. Assume that net N_1 is specified by a predicate R. Then by above definition, net N_2 is a self-stabilizing extension of N_1 as long as starting from any initial marking N_2 will satisfy R after a finite number of firing steps. Second, the definition implicitly requires that the extension should not interfere with the normal operations of the original protocol. This is reflected in the requirement that $\beta'\alpha'$ must be a legal firing sequence in N_1.

Another implication of the above definition is that every terminating net is trivially self-stabilizing extension of itself (taking $\alpha' = \varepsilon$). Therefore in this paper we consider net modeling non-terminating communication protocols. As every sequence from T^* is finite, for non-terminating protocols Definition 6 has to be changed to use infinite firing sequences over T, notated as T^ω, which is the set of all infinite strings over T (cf. [21] for details).

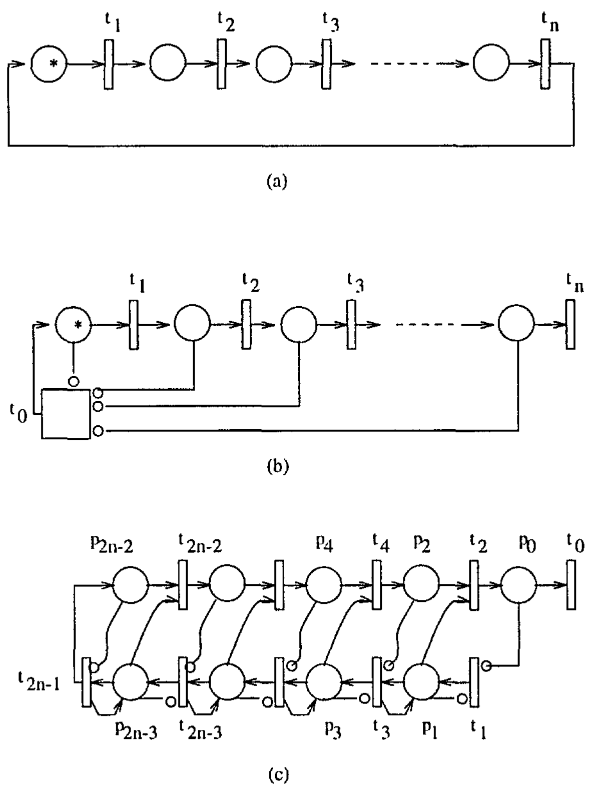


Figure 1: A ring net and its two extensions by Ghosh: (a) the ring net; (b) the first extension; (c) the second extension.

3 Self-stabilizing without interfering normal operations

Ghosh discussed in [8] the issue of self-stabilizing a special class of Petri net models (ring nets and marked graphs). In both cases, inhibitor arcs are used to achieve self-stabilization. Figure 1 shows a ring net and its two self-stabilizing extensions.

One problem with the two self-stabilizing extensions is that, in both cases, the system explicitly interferes with the operation of the net, even in the absence of operation errors. For instance, in the first extension on Figure 1.(b), transition t_n drains all tokens in the net and then the absence of tokens in the whole net causes transition t_0 to be enabled and fire. We believe that in a self-stabilizing extension of any system, the extension should not interfere with the normal operation of the original system and interference will come only after the system enters an illegal state. Incidentally, it is easy to see that both extensions in Figure 1(b) and Figure 1(c) do not satisfy Definition 6. The net in Figure 1(a) can be characterized by the regular expression $(t_1t_2\cdots t_n)^*$.

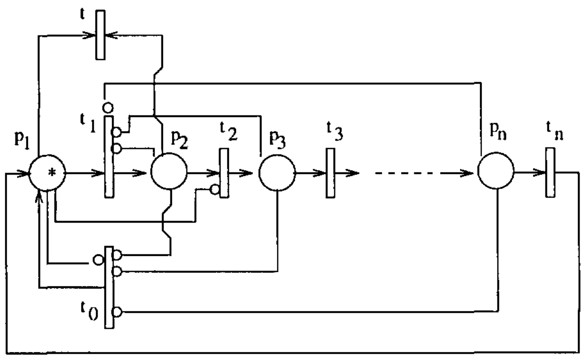


Figure 2: An alternative self-stabilizing extension of the ring net

Every normal firing cycle in both nets in Figure 1(b) and Figure 1(c) includes the transition t_0, which is absent in the original net.

A new self-stabilizing extension of the net in Fig 1(a) is given in Figure 2. As shown by the following theorem, the new extension is guaranteed to operate as usual as long as the system is in legal states and will try to recover from failure as soon as it enters an illegal states.

Theorem 4 *The network in Figure 2 is a self-stabilizing extension of the ring net in Figure 1.(a).*

Proof: Self-stabilization is clearly implied by the following three observations:

Observation 1: if there is no token in the net, then transition t_0 can fire and bring the system to a legal state.

Observation 2: assume that there is at least one token in place p_1. Transition t_1 cannot fire if there are tokens in any other places. Tokens in places $p_3, p_4, \cdots, p_n$ will eventually reach place p_1.

Observation 3: If there are tokens in both places p_1 and p_2, then only transition t is enabled. Every firing of t will drain two tokens from the net. If place p_2 becomes empty after t's firing but p_1 still holds more than one token, firing of t_1 will move one token from p_1 to p_2. So if initially the total number k of tokens in the net is odd, then after $k/2$ firings of transition t only one token will be left in place p_1. If k is even, then after $k/2$ firings of transition t the net will be empty. ■

4 Self-stabilizing through decentralized control

In this section we argue and show that time-Petri Nets, introduced in [15], is more suitable to model the self-stabilizing properties of communication protocols.

The main problem with the ring net in Figure 2 is the difficulty associated with implementing inhibitor arcs that involve non-neighboring places. For instance, checking whether transition t_0 in Figure 2 is enabled practically asks for the verification of the emptiness of all places in the net, while checking if transition t_1 is enabled demands the verification of the emptiness of places $p_2, p_3, \cdots, p_n$. In a ring net, such a verification is equivalent to a round of communications for consensus among all the transitions, which is difficult and expensive to implement (considering the fact that such verifications must be performed before every firing of transition t_0 or t_1).

There are at least two approaches that can be used to cope with the above problem. The first approach is to launch the communications checking for absence of tokens periodically, not before every time t_0 or t_1 is fired. Although this may reduce the cost of implementing self-stabilization, it also compromises the self-stabilizing property. The second approach is to make use of the time sensitive property of most practical communication protocols.

Timeout is a fundamental technique used in communication protocols. For instance, in a token ring, timeout is used to detect loss of tokens. In sliding window protocols, timeout is used to retransmit previously sent but not yet acknowledged messages. As pointed out in [9], timeout is a necessary condition for communication protocols to achieve self-stabilization. The need of timeout in communication protocols can be attributed to the distributed nature of distributed systems and computer networks. Communication channels are not reliable and messages can be lost or corrupted. Individual machines can crash due to various hardware and/or software failures.

The need of timeout can also be justified by the fact that long delays in communications are normally treated as errors because most users usually will not tolerate an excessively long delay. This points out that communication protocols are de-

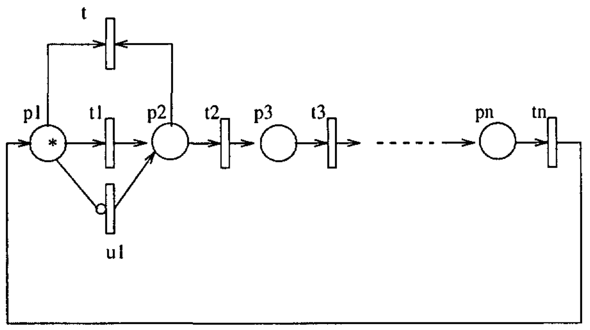


Figure 3: A self-stabilizing extension of the ring net based on TPN

lay sensitive, not delay insensitive. Therefore delay insensitive protocols, such as the three elegant protocols in [2], may not be suitable for many practical applications.

One elegant model for delay sensitive communication protocols is *Time-Petri nets* (TPNs), introduced in [15]. Informally, a TPN is just an ordinary Petri net with additional restrictions on the timing of firing of transitions. Formally we have the following definition.

Definition 7 *(cf. [15]) A TPN is a PN in which each transition t_i is associated with two real numbers t_i^* and t_i^{**} satisfying*[2]

- $t_i^*, t_i^{**} \geq 0$;
- $t_i^* \leq t_i^{**}$.

The firing rule of a TPN is defined as:

- *If t_i has been enabled for a period of time equal to or greater than t_i^*, then t_i* **can** *fire, using the the same rule for token redistribution as in an ordinary PN.*
- *If t_i has been enabled for a period of time equal to t_i^{**}, then t_i* **must** *fire, using the the same rule for token redistribution as in an ordinary PN.*

Intuitively, t_i^* is the minimum time for which transition t_i has been continuously enabled before it can fire, while t_i^{**} is the maximum time for which transition t_i has been enabled continuously before it must fire.

[2]Please note that for historical reasons, the notations t_i^* and t_i^{**} used in this definition are similar to that of the Kleene closure.

To illustrate the usefulness of TPNs in expressing delay-sensitive communication protocols, we give a self-stabilizing extension of the ring net, as shown in Figure 3. In that figure, it is assumed that $\theta = \Sigma_{i=3}^{n} t_i^{**}$, and the time parameters for transitions in Figure 3 satisfy the conditions $t_1^* + t_2^* \geq \theta$, $t_2^* > 0$ and $t^* = t^{**} = 0$, and $t_1'^* \geq \theta + t_1^{**} + t_2^{**}$. The following theorem states that the net in Figure 3 is self-stabilizing.

Theorem 5 *The network in Figure 3 is a self-stabilizing extension of the ring net in Figure 1.(a).*

Proof: It is clear from the structure of the net that if there is no token in the net, transition t_1' will fire after $t_1'^* = \theta + t_1^{**} + t_2^{**}$ units of time.

Assume that there is more than one token in the net. First we observe that any pair of tokens that are already in place p_1 and p_2 respectively will be removed by the firing of transition t. There are still two cases to consider.

Case 1: there are two or more tokens in places $p_3, p_4, \cdots, p_n$ at some time point. Due to the requirement that $t_1^* + t_2^* \geq \theta$, one of these tokens must have been in either place p_1 or p_2 when the other tokens arrive at place p_1. Therefore the transition t will be enabled after at most θ time units and the firing of t will remove two tokens from the net. **Case 2**: there are two or more tokens, one is in place p_1 (or p_2) while the others are in places $p_3, p_4, \cdots, p_n$. If the token in p_1 or p_2 is still in p_1 or p_2 when a token from $p_3, \cdots, p_n$ arrives at p_1, these two tokens will enable t and be removed. If the token in p_1 or p_2 is not in p_1 or p_2 when a token from $p_3, \cdots, p_n$ arrives at p_1, the requirement that $t_1^* + t_2^* > \theta$ ensures that the latter will still be in p_1 or p_2 when the former arrives at p_1.

It should be noted that it is not guaranteed that the t_1' will fire only when the net is empty. However, from the above discussion, it should be clear that even if t_1' accidentally generates an extra token (this may happen if the existing token has been delayed unexpectedly long by one of the transitions), the net will discard one or both tokens. In the formal case, the system immediately returns to a legal state. In the second case, t_1' will regenerate a new token, which again brings the net to a legal state. This completes the proof. ■

The net in Figure 3 has several distinct properties over the net in Figure 2. First, the only inhibitor arc from place p_1 to transition t_1' in Figure 3 connects two neighboring place and transition. This implies that a node need only check the emptiness of its neighboring channel, not all the channels over the net. Second, it shows that self-stabilization of a ring net as shown in Figure 1(a) can be achieved by operating the net at half of the normal operation speed. Third, the convergence rate of of the net in Figure 3 can be easily and accurately estimated. Assume that m tokens were in the net at some time point. It can be easily seen, as from the argument in the proof of above theorem, that after at most $(m/2)\theta$ time units, either all the m tokens are drained from the net or only one token is left. In the late case, the net already returns to a legal state. In the former case, transition t_1' will regenerate a single token after $\theta' = \theta + t_2^{**}$ time units, i.e. the net returns to a legal state within $\theta' + (m/2)\theta$ time units.

It is also interesting to discuss the convergence rates of the two self-stabilizing Petri nets in Figure 1(b) and Figure 1(c) with the one in Figure 3. On the surface, the two nets in Figure 1(b) and (c) converge much fast than the net in Figure 3. However, in practice, this may not necessarily be the case. For instance, to implement the n inhibitor arcs in Figure 1(b), transition t_0 must receive some kind of *token not present* report from each of the n places. This may take a substantial amount of time.

5 Using max transitions

In last section, inhibitor arcs are used to achieve self-stabilization. As pointed by Ghosh, use of inhibitor arcs seems mandatory in order to achieve self-stabilization. In this section we show that using max transitions together with inhibitor arcs could greatly simplify the representation of self-stabilizing communication protocols.

Figure 4 shows a net using both max transitions and inhibitor arcs. Transition t_1 is the only max transition with input set $\{p_1, p_2, \cdots, p_n\}$.

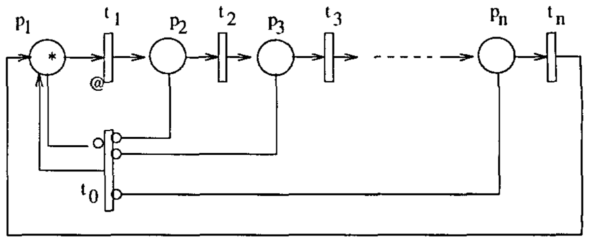


Figure 4: A self-stabilizing extension of the ring net using both max transitions and inhibitor arcs

Theorem 6 *The network in Figure 4 is a self-stabilizing extension of the ring net in Figure 1.(a).*

Proof: Similar to the net in Figure 2, the case where the net is empty is handled by transition t_0. By the definition of max transitions, the firing of transition t_1 will remove all tokens from all its input places. Hence every firing of t_1 will remove all the tokens in places $p_1, p_2, \cdots, p_n$. ■

Our second example, as shown in Figure 5, is a self-stabilizing extension of the well known sliding window protocol. This example further illustrates the usefulness of max transitions in specifying and expressing self-stabilizing property. A good introduction to sliding window protocols and their self-stabilizing versions can be found in [20, 9]. In Figure 5, the input sets $S(t_3) = \{p_1, p_2, p_3, p_5\}$, $S(t_{11}) = \{p_7, p_8, p_4, p_6\}$. The possible error where there are tokens in both places p_1 and p_2 (or p_7 and p_8) signals errors such as invalid values for variables, which may cause several conditions in a guarded command to incorrectly hold simultaneously, for instance. The max transition t_3 (similarly for t_{11}) will remove all tokens in places p_1, p_2, p_3 and p_5 before it fires. This is equivalent to reset the various variables to correct and consistent values and clean up the buffers. For simplicity of presentation, the time parameters for this protocol are omitted. Some constraints on the time parameters are obvious. For instance, t_4^* must be at least equal to t_{11}^{**}.

Theorem 7 *The sliding window protocol in Figure 5 is self-stabilizing.*

Proof: As always, the error of total-loss-of-token is handled by the inhibitor arcs. The error of creation of arbitrary tokens is solved through the

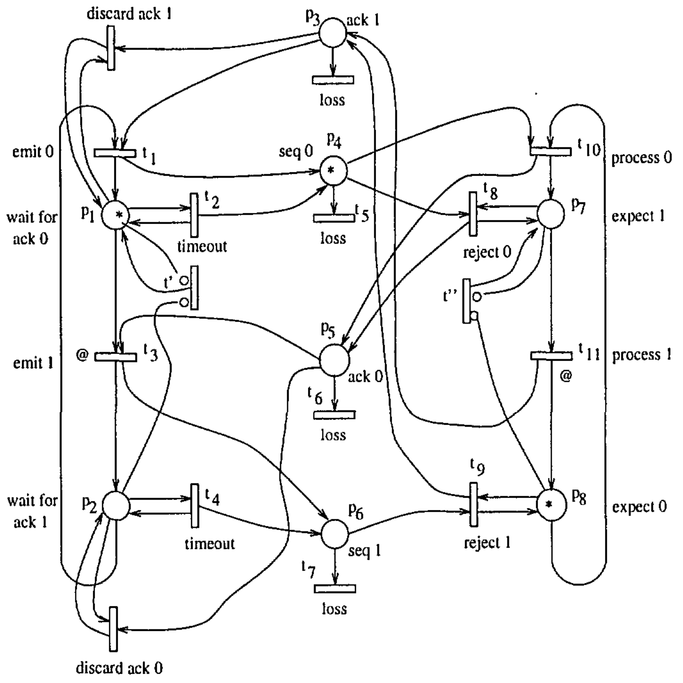


Figure 5: A self-stabilizing extension of sliding window protocol

max transitions t_1 and t_3. If there is more than one token in places $p_1, ...,$ the firing of transition t_1 will remove all tokens in the places. ■

It is interesting to point out that Petri nets with max transitions are strictly less powerful than Petri nets with inhibitor arcs. This is evidenced by the fact that the language L^*, where $L = \{a^n b^n | n > 0\}$, cannot be generated by any Petri nets with max transitions. In fact the class of ordinary Petri nets added with max transitions is not closed under the Kleene closure operation. Details about the modeling power of max transitions are being given in a forthcoming paper in [13].

6 External and internal interferences

As evidenced from the previous sections, indiscriminated use of inhibitor arcs and/or max transitions may cause extreme difficulty in implementing self-stabilizing protocols. The concept of "decentralized control" used in Section 4 is intended to remedy the problem. We formally propose here the concepts of *external* and *internal* interferences, which will allow us to formally quantify the problem.

In general, a component (e.g. a sender, receiver, or a channel) within a communication protocol may be represented by several places and transitions in a Petri net. Let C_1 and C_2 be two communication components that are not physically linked. Informally, an external interference occurs when one place (transition, resp.) belonging to C_1 is forced to connect to a transition (place, resp.) belonging to C_2 by some arc added during the process of self-stabilizing the net. This point has been stated in Section 4, where we argued that the self-stabilizing extension shown in Figure 2 is very difficult and expensive to implement. However, controlled use of inhibitor arcs, i.e. the arcs that don't cause external interference, such the net shown in Figure 3, may minimize the implementation difficulty. Similar arguments hold for the use of max transitions. For instance, although the use of max transitions in Figure 4 gives a more concise representation, implementing the max transition t_1 in that figure is quite difficult and expensive. On the other hand, the two max transitions in Figure 5 are quite easy to implement because they involve only places local to the sender or receive process.

A place or transition in a net is said to be *internal* if it is only connected to transitions or places belonging to the same component. An internal interference occurs when the internal states of one or more components are forced to be connected to the outside during the self-stabilization extension procedure. The first protocol given in [2] is an example of internal interference, where the state change of a process depends on the internal states of one of its two neighbor processes. On the other hand, the sliding window protocol in Figure 4 does not suffer from neither external nor internal interference, because the max transition t_3 (t_{11}, resp.) does not refer to any places or transitions in the receiver (sender, resp.) at all. The communication channels there are modeled by four places and none of them is internal.

Both external and internal transitions will cause difficulty in practical implementation, thus should be avoided.

7 Discussions

Using Petri nets as the concurrent model, we have discussed the issue of self-stabilizing communication protocols. We have argued that (1) self-stabilizing extensions of communication protocols should not interfere with their normal operations, and (2) communication protocols are usually delay sensitive and *time-Petri nets* is a more suitable formal model (than ordinary Petri nets) to express self-stabilizing property. The possibility and suitability of using Petri nets with *max transitions* to express self-stabilizing communication protocols have also been explored.

As pointed by Ghosh in [8], the use of inhibitor arcs to represent self-stabilizing Petri nets seems unavoidable. This may raise the question about the tractability and suitability of Petri nets for self-stabilizing communication protocols. We feel that in-discriminated use of inhibitor arcs will at least cause extreme difficulty in implementing self-stabilizing protocols. This point has been stated in Section 4, where we argued that the self-stabilizing extension shown in Figure 2 is very difficult and expensive to implement. However, controlled use of inhibitor arcs, such as the net shown in Figure 3, may minimize the implementation difficulty. Similar arguments hold for the use of max transitions. For instance, although the use of max transitions in Figure 4 gives a compact representation, implementing the max transition t_1 in that figure is quite difficult and expensive. On the other hand, the two max transitions in Figure 5 are quite easy to implement because they involve only places local to the sender or receive process. The issue of using generalized time-Petri nets to specify and verify communication protocols is worth of further investigation.

It should be pointed out that the notion of *recoverability* and *recoverable communication protocols* proposed by Merlin and Farber in [15] is quite close to (although weaker than) the self-stabilization concept adopted in this paper.

Both the nets in Figure 2 and Figure 3 are asymmetric. This is in agreement with the conclusions in [5]. In particular, it is not difficult to see that there is no symmetric version of self-stabilizing extension of the net shown in Figure 3. Trying to design such an extension will easily reveal the difficulty and impossibility.

The fact that the self-stabilizing extension in Figure 3 can only operate at half of the speed of the original net indicates that self-stabilization is usually expensive to implement. This also implies that self-stabilization may be a property too strong and too expensive for certain applications. The bound on the convergence rate for the self-stabilizing extension in Figure 3 is new and it is not known if this is the optimal bound for this type of delay sensitive rings.

The issue of self-stabilizing *colored Petri nets* is another interesting and important issue. Many complex concurrent phenomena are very difficult, if not impossible, to be modeled by low-level Petri nets. Some others cannot be adequately described by low-level Petri nets at all. Research in the area of *high-level* Petri nets is intended to remedy this problem [12]. High-level Petri nets are in general more expressive than its low-level counter parts and thus often give more succinct and manageable specifications. More importantly, all the formal analysis techniques developed for low-level nets have been easily extended to high-level nets.

The class of *Colored Petri nets* (CPNs) is one of the most influential types of high-level nets. Informally, in a CPN, tokens are differentiable – they are typed (i.e. *colored*). Each place now is associated with a *color set* – the set of types of tokens that can legally be in it. Transitions are now guarded with boolean expressions. A transition can fire provided that (1) it is enabled in the usual sense (for low-level nets) and (2) the associated guard expression is evaluated to true. Each arc is associated with an *arc expression* which regulates conditions under which certain types of tokens can be removed (from input places) or produced (to output places). Detailed definition can be found in [11].

Figure 6 (a) is a CPN that models the one-bit sliding window protocol. The color set associated with each place used is obvious and the explanation is omitted. Compared with Figure 5, the CPN is much smaller and succinct. A self-stabilizing extension of the CPN P is shown in Figure 6 (b), where the input set for the max transitions t_1 and t_6 are $\{p_1, p_3\}$ and $\{p_2, p_4\}$ respectively.

Further research is needed to extend existing methodologies for low-level Petri nets to CPNs and develop new methodologies suitable for CPNs.

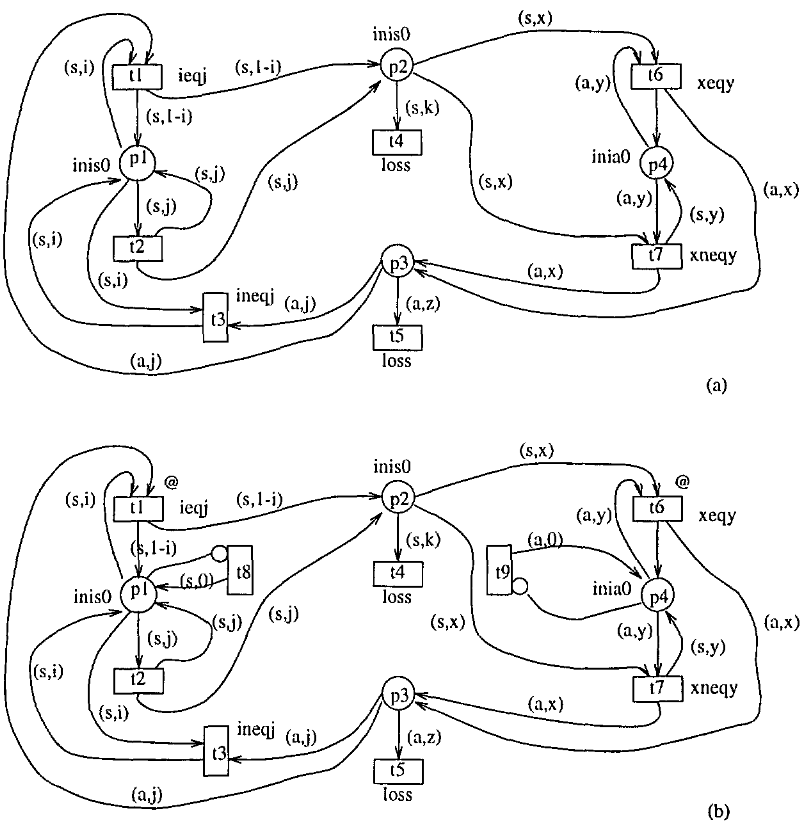


Figure 6: A CPN modeling the one-bit sliding window protocol: (a) the one-bit sliding window protocol; (b) the corresponding self-stabilizing extension

References

[1] F. Bastani, I. Yen, and I. Chen, *Class of inherently fault-tolerant distributed programs.* IEEE Transaction on Software Engineering, Vol. 14(10), Oct. 1989, pp. 1432-1444.

[2] G.M. Brown, M.G. Gouda, and C. Wu, *Token systems that self-stabilize.* IEEE Transaction on Software Engineering, Vol. 14(6), June. 1989, pp. 845-852.

[3] J. Burns and J. Pachl, *Uniform self-stabilizing rings.* ACM Trans. on Programming Lang. and Systems, Vol. 11, No. 2, 1989, pp. 330-344.

[4] L. Cherkasova, R. Howell, and L. Rosier, *Bounded self-stabilizing Petri nets.* Proc. of the 12th Annual Petri Net Conference, Dec. 1991.

[5] E.W. Dijkstra, *Self-stabilizing systems in spite of distributed control.* Communications of Association for Computing Machinery, Vol. 17(11), Nov. 1974, pp. 643-644.

[6] Carlo Ghezzi, Dino Mandrioli, Sandro Morasca, and Mauro Pezze, *A general way to put time in Petri nets.* Proc. of the 5th International Workshop on Software Specification and Design, IEEE Computer Society Press, 1989.

[7] M. Ghodsi and K. Kant, *Well-formed generalized task graphs.* Proc. of 1991 IEEE International Conference on Parallel and Distributed Systems, Dec. 1991.

[8] S. Ghosh, *Stabilizing Petri nets.* Proc. of 1991 IEEE International Conference on Parallel and Distributed Systems, Dec. 1991.

[9] M.G. Gouda and N.J. Multari, *Stabilizing communication protocols.* Technical Report, TR-90-20, Dept. of Comp. Sci, The Univ. of Texas at Austin, June 1990.

[10] M.G. Gouda, R. Howell and L. Rosier, *The instability of self-stabilization.* Acta Informatica, Vol 27, Sept. 1990, pp. 697-724.

[11] K. Jensen, *Coloured Petri nets: a high-level language for system design and analysis.* In Advances in Petri nets 1990. Lecture Notes in Computer Science 483, Springer-Verlag, 1991, pp. 342-416.

[12] K. Jensen and G. Rozenberg (eds.), *High-level petri nets: theory and practice.* Springer-Verlag, 1991.

[13] K. Kant and W. Peng, *On the modeling power of max transitions and high-level Petri nets with max transitions.* Under preparation.

[14] H.S.M. Kruijer, *Self-stabilization (in spite of distributed control) in tree-structured systems.* Information Processing Letter, Feb. 1979, pp. 91-95.

[15] P.M. Merlin and D.J. Farber, *Recoverability of communication protocols - implications of a theoretical study.* IEEE Trans. on Comm., Vol. COM-24, Sept. 1976, pp. 1036-1043.

[16] T. Murata, *Petri Nets: properties, analysis, and applications.* Proc. of IEEE, Apr. 1989, pp. 541-580.

[17] J. Peterson, *Petri nets theory and the modeling of systems.* Prentice-Hall, Englewood Cliffs, NJ, 1981.

[18] Proc. of International Workshop on Timed Petri Nets, Torino, Italy, IEEE Computer Society Press, July 1985.

[19] P.H. Starke, *Some Properties of Timed Nets under the Earliest Firing Rule.* Advances in Petri Nets 1989, (ed.) G. Rozenberg, LNCS 524, Springer-Verlag, 1989.

[20] A.S. Tanenbaum, *Computer networks.* Prentice-Hall, Englewood Cliffs, NJ, 1989.

[21] W. Thomas. Automata on Infinite Objects. In *Handbook of Theoretical Computer Science, Vol B.* (ed.) J. van Leeuwen, pp:133-192, MIT Press, 1990.

[22] W.M. Zuberek, *Timed Petri Nets Definitions, Properties and Applications.* Microelectronics and Reliability, Vol. 31(4), 1991, pp. 627-644.

Interface Methods: a Means for the Integration of Inheritance in a Concurrent OOP Language

Agostino Poggi
Dipartimento di Ingegneria dell'Informazione
University of Parma, Viale delle Scienze, 43100 Parma, Italy
Phone: +39 521 90 5728, Fax: +39 521 90 5723
E-mail: poggi@CE.UniPR.IT

Keywords: concurrent object-oriented programming languages, interference between inheritance and concurrency, software reuse

Edited by: Rudi Murn

Received: September 26, 1995 **Revised:** November 26, 1995 **Accepted:** January 9, 1996

Several researchers have pointed out that inheritance often conflicts with concurrency in object-oriented languages, resulting in inheritance anomalies where redefinitions of inherited methods are necessary in order to maintain the integrity of objects. Several solutions have been proposed for resolving the anomalies; however, some of them are incomplete and do not solve all the possible types of inheritance anomalies. Others make the definition of classes complex. This paper copes with the inheritance anomaly problem presenting a solution that minimizes the redefinition of inherited methods without increasing the complexity to write them. This solution is based on the use of a special set of methods, called interface methods, that divide their body from two sets of synchronization constraints respectively used to enable the execution of the body of this method and to disable the methods that cannot be executed after this method. In particular, the paper presents a number of examples that illustrate how our proposal can solve easily the different types of inheritance anomalies.

1 Introduction

Inheritance and delegation are the means for sharing knowledge in an object-oriented language. They are not alternative, but can coexist in a same language (see, for example, Vulcan - Kahn et al., 1987). There is an ongoing debate on the advantages of inheritance as opposed to those of delegation. The most common opinion is that inheritance is more important than delegation because it provides more power in reusability, even if delegation provides more power and flexibility in sharing because an object can delegate dynamically its messages to different objects, while the knowledge sharing of inheritance is statically fixed.

However, delegation has often been preferred in concurrent object-oriented programming languages (Wegner, 1990; Chin & Chanson, 1991; Wyatt et al., 1992; Agha et al., 1993) because of the problem of the interference of inheritance with concurrency (inheritance anomaly problem) (America, 1987; Kafura & Lee, 1989; Tomlinson & Singh, 1989; Matsuoka & Yonezawa, 1993). In fact, synchronization often interferes with inheritance because a local change of the synchronization constraints in a class may cause the redefinition of synchronization constraints in the superclass. Moreover, the effect of the local change may be not limited to the superclass because the superclass inherits from other classes and its changes are inherited by all its subclasses.

Matsuoka and Yonezawa (1993) identified three main reasons why inheritance anomalies occur. These three reasons are: i) partitioning of acceptable states, ii) history - only sensitiveness of acceptable states, and iii) modification of acceptable states.

i) partitioning of acceptable states. The life of an object can be described on the basis of the set of states it can assume. This set of states can be divided into disjoint subsets, called accept sets, according to the synchronization constraints of the object. When a subclass adds a new method, the set of states might need to be further partitioned as regards the set of states of the superclass because the synchronization constraints of the new method may not be properly accounted for in the partitioning of the superclass. For example, the states of a first-in first-out bounded buffer class, called *bounded_buffer*, offering a method, *get*, that removes an element from the buffer, and a method, *put*, that stores an element into the buffer, can be divided in three accept sets respectively named *empty*, *partial* and *full*. If a subclass of the *bounded_buffer* class, called *get2_buffer*, adds a method, *get2*, that removes a couples of elements, then the accept set *partial* must be partitioned into two accept sets, *get2_partial* and *one_element*, because it is necessary to distinguish the state at which only one element is into the buffer and so *get2* cannot be executed. In this case, the inherited methods must be redefined if their synchronization code involve expressions on accept sets.

ii) history - only sensitiveness of acceptable states. The synchronization constraints of some methods depend on history information. For example, if a subclass of *bounded_buffer*, called *gget_buffer*, adds a method, *gget*, whose behavior is almost identical to that of *get*, with the exception that cannot be accepted immediately after the invocation of *put*, then the state "after executing put" cannot be defined on the basis of the variables inherited from *bounded_buffer*. In this case, it might be necessary to add a new variable, *after_put*, to maintain the state "after executing put" and to redefine the methods, *put* and *get*, because they must set and reset the value of that new variable.

iii) modification of acceptable states. The introduction of a new method modifies the set of states under which the methods inherited from the superclass could be invoked. For example, let us consider a class, called *locker*, that suspends the service of all the messages through the execution of the method, *lock*, until it receives and accepts a request of the method *unlock* and executes it. This class can be used together the *bounded_buffer* class to define a new class, called *lock_buffer*. In this case, it may be necessary to redefine the methods inherited from *bounded_buffer* because the synchronization constraints introduced by *locker* causes the modification of the acceptable states of *get* and *put* since they cannot be executed when their object is locked.

In recent years, research on the inheritance anomaly problem has achieved some important results and some proposals have shown solutions of the inheritance anomaly problem. However, these proposals are based on some synchronization schemes that often make the definition of classes complex.

In this paper, we present an approach to the problem of interference between inheritance and concurrency that solves the inheritance anomaly problem, but uses a simple synchronization scheme that makes the definition of classes easy and natural. The next section discusses the features and the limits of other approaches to synchronization in concurrent object-oriented systems. Section three presents our solution. Finally, section four presents the main contributions of our work, its relationships with the works presented in section two, and our future research directions.

2 Related work

The possibility to solve the problem of interference between inheritance and concurrency depends on the type of message passing control strategy. In fact, Kafura and Lee (1989) noted that only decentralized message control can be combined with inheritance without restriction of reuse and extensibility because centralized message control groups synchronization constraints into a centralized method which cannot be inherited by a subclass if it almost always differs from the method of the superclass in some synchronization constraints[1].

[1]This is the reason why concurrent object-oriented programming languages that use centralized message passing control usually do not support inheritance (Yonezawa, 1990; America, 1987; Agha, 1986).

One of the first approaches to integrate inheritance with concurrency has been introduced by Hybrid (Nierstrasz, 1987). Hybrid associates a delay queue with each method and this queue can be closed or opened by some other object methods. Nevertheless, when a subclass adds a method, then the methods inherited from its superclass must be modified if they must open or close the delay queue of the new method. This disadvantage is not only present in the cases that involve one of the three types of inheritance anomaly, but in all the cases that a new method modifies the value of a variable involved in the synchronization constraints of some inherited methods. For example, a subclass of the *bounded_buffer* class introduced in the previous section, that adds a method, *last*, for removing the last element of the buffer, must redefine both the inherited methods because *get* must close the delay queue of *last* when the buffer becomes empty, and *put* must open the delay queue of *last* when the buffer contains an element.

ACT++ (Kafura & Lee, 1989) integrates the become operation of actor's languages with *behavior abstractions* which denote sets of open methods. Therefore, the current behavior of an object indicates the set of methods that can be accepted and executed. Nevertheless, this approach has two main limits: it is based on the assumption of closed system and so loses the property of openness of the Actor model (Hewitt & de Jong, 1983), and it does not solve the inheritance anomaly problem. For example, the definition of the *get2_buffer* class introduced into the previous section must redefined the *put* method because, while the inherited method can become *full* or *partial*, the new method can become *full*, *get2_partial* or *one_element*, and must redefined the *get* method because, while the inherited method can become *partial* or *empty*, the new method can become *get2_partial*, *one_element* or *empty*.

Rosette (Tomlinson & Singh, 1989) proposes the integration between inheritance and concurrency through the use of *enable-sets*. *Enable-sets* are "natural units of inheritance" associating a set of executable methods with an object state. This approach derives from *behavior abstraction* and presents the following main properties: i) the composition of *enable-sets* for generating new *enable-sets* (a subclass can both inherits *enable-sets* and use them to define new ones), ii) the possibility to pass *enable-sets* to other objects through messages, iii) the parameterization of *enable-sets*, and iv) the scheduling of messages on the basis of a pattern order dividing messages into classes of priority. Nevertheless, it uses the same synchronization mechanism of *behavior abstraction* and so it has the same problems to cope with the inheritance anomaly problem.

Guide (Decouchant et al., 1989) proposes the use of method guards. Each method has an activation condition (guard) which determines the states where the method may be executed. Method guards are a more natural and elegant synchronization scheme. The partitioning of states is not a problem, because they are able to directly judge whether the message is acceptable or not under the current state, but they do not solve both the history - only sensitiveness of acceptable states and the modification of acceptable states anomalies because in these cases it often involves the redefinition of a large set of methods. For example, the definition of the *gget* class introduced into the previous section involves the redefinition of *put* because it must set the variable *after_put*, and of *get* because it must reset the same variable.

Synchronization Actions (Neusius, 1991) are one of the few synchronization approaches that solves all the three types of inheritance anomaly. Synchronization Actions are based on methods divided in four parts: *matching*, *pre*, *action* and *post* that respectively specify the guard, the pre-actions, the body and the post-actions of the method. Synchronization Actions support intra-object concurrency and use behavior abstractions to exclude mutually interfering operations. This approach strengthens object encapsulation-separating the variables used in the synchronization constraints from the other variables and by imposing that they can be managed only by the synchronization parts, that is, the *matching*, *pre* and *post* parts of a method. However, the fact that the value of these variables cannot be modified by the body of a method often involves duplication of code and variables in the body and in the synchronization parts (for example in the *bounded_buffer* example, the variables used to indicate the first and the last full position cannot be used both to access the corresponding element and to check the state of the buffer). Moreover, its parts cannot be incrementally modified or reused in the definition of other methods.

Frolung (1992) solved the problem of the interference between inheritance and concurrency through the separation between synchronization constraints and object methods. He describes a synchronization constraint as restrictions that apply to invocations of object methods. Constraints can be incrementally modified, the restriction applicable to a given method of a superclass can be increased for the same method of its subclasses, and new constraints can be generated by the aggregation of other constraints. However, the fact that a class can only increase the restrictions of its superclass is often a disadvantage. For example, let us reconsider the *locker* class introduced in the previous section, the synchronization constraint that disables the execution of the methods different from *unlock* when the object is locked, has the following definition:

```
(locked) prevents all_except unlock
```

This definition prohibits that a subclass of the *locker* class could include a new method that may be executed when the object is locked. For example, a locking class, introducing a method, *inquire_lock*, that must be executed irrespective of the value of the lock because it inquires and returns the state of the lock, naturally derives from the *locker*, but cannot be defined as its subclass.

Matsuoka and Yonezawa (1993) proposed a synchronization model that allows both guard-based and accept method set based synchronization schemes to coexist and be integrated, so that the best scheme can be chosen to describe the different synchronization constraints. This approach solves the three types of inheritance anomalies and the association between guards and accept method sets may reduce the re-definitions of the methods belonging to the same set. However, on the one hand, the advantage of this association is limited because a guard which is associated with an accept method set can involve method arguments only if all the methods of this set have those method arguments. On the other hand, in the case of partitioning of states anomaly this approach is always less efficient than guard methods because, while the guard methods approach adds only the new methods and their guards, this approach adds the new methods and the new guards and refines some inherited accept method sets.

3 Our approach

In our approach, an object, that we call *c_unit* (concurrent unit), is composed of four main parts, an interface controller, a set of special methods, called interface methods, which serve messages, a set of other methods, which are used by interface methods to perform their tasks, and a set of variables.

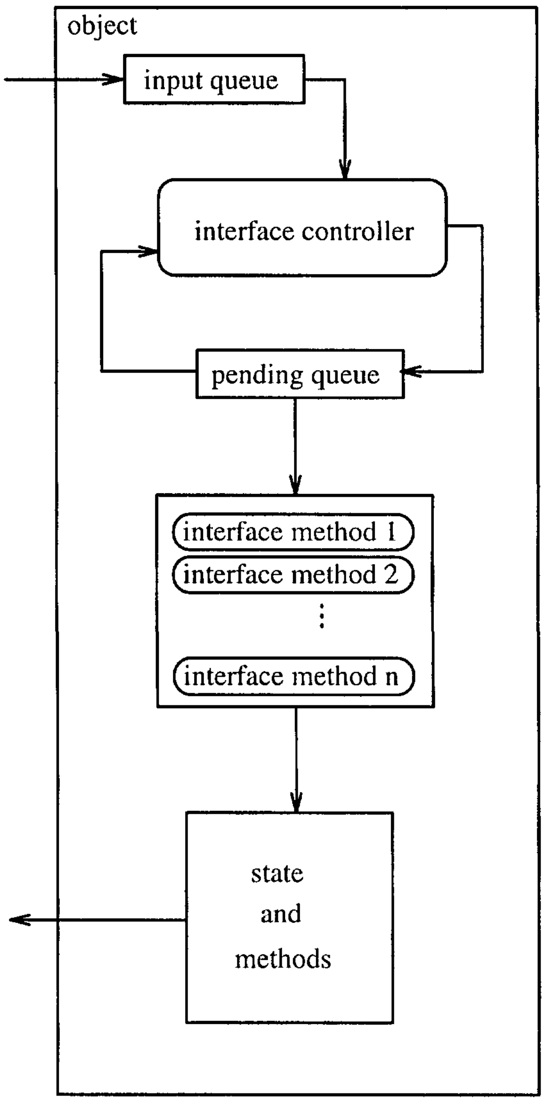


Figure 1: The structure of the object.

The interface controller is an active component that gets the first message from the object input queue and serves it through the appropriate interface method when the synchronization constraints allow the service. If they do not allow the service of the message, then the interface controller puts the message into a pending queue. At the end of a service, the interface controller checks again if the messages in the pending queue may be served and, if so, serves them through the appropriate interface method (figure 1 shows a graphical description of the object model).

An interface method is composed of three parts respectively called *body*, *guard* and *transitions*. The *body* part is the code performing the task of the method. The *guard* and *transitions* parts define the object synchronization constraints. The *guard* is a boolean expression that is based on instance variables, interface method arguments and *guard* parts of other methods, and disables the execution of the *body* of this interface method when it evaluates to false. The *transitions* part is a set of couples, each of them is composed of a boolean expression and a set of interface methods. This boolean expression has the same form of a *guard* with the obvious exception that it cannot contain other *guard* parts, but it can contain other *transitions* parts. When a boolean expression evaluates to true, the corresponding set of interface methods cannot be the next executed method. The *transitions* part is evaluated after the *body* execution and is used to manage synchronization constraints that depend on historic object state information.

3.1 Object definition and specialization

This synchronization mechanism has been used in a C++ implementation of CUBL - Concurrent Unit Based Language (Poggi, 1994). Therefore, the definition and the specialization of an object and of its methods follow C++ rules. However, the three parts of an interface method can be separately specialized.

3.1.1 Example 1: object definition

As example of object definition, let us consider the *bounded_buffer* class introduced into section 1. Its definition contains the variables defining the buffer structure, the class constructor, and the two interface methods, *get* and *put*. Each of these two interface methods is composed of a *body* that performs the method task and a *guard* that defines the state condition where the *body* can be executed (see figure 2).

```
class bounded_buffer c_unit {
 int buf[MAX];
 int in, out;
public:
 bounded_buffer() {
  in = 0;
  out = 0;
 }
 int interface get() {
  guard
   (in != out)
   // true when the buffer
   // is not empty.
  body {
   int res;
   res = buf[out++];
   out % = MAX;
   return res;
  }
 }
 void interface put(int item) {
  guard
   (((in - out) % MAX) < ( MAX - 1))
  // true when the buffer is not full;
  // (MAX - 1) is the  number of
  // elements that the buffer can
  // contain.
  body {
   buf[in++] = item;
   in % = MAX;
  }
 }
}
```

Figure 2: The *bounded_buffer* class.

3.1.2 Examples 2 and 3: object specialization

As specialization example, let us define a bounded buffer that introduces a new interface method for removing the last element put into the buffer. We can obtain this type of buffer by specializing the previous *bounded_buffer* class. This new class, called *extended_buffer*, introduces an interface method, called *last*, that has the same *guard* of the inherited interface method, *get*, and its *body* removes the last element put into the buffer (see figure 3).

```
class extended_buffer : bounded_buffer {
public:
 int interface last() {
  guard
   guard:get()
  // true when the buffer
  // is not empty.
  body {
   in = (--in % MAX);
   return buf[in];
  }
 }
}
```

Figure 3: The *extended_buffer* class. Note that the form guard:get() calls the *guard* part of the *get* interface method inherited from the *bounded_buffer* class.

This approach allows an incremental modification of *guard* synchronization constraints. In fact, let us define a bounded buffer which supports writers priority. We can obtain this type of buffer by specializing the definition of the *bounded_buffer* class. This new class, called *priority_buffer*, adds a synchronization constraint to the *guard* of *get* that checks the presence of *put* requests into the object queue through a variable, called *put_queued*, managed by the interface controller (see figure 4).

```
class priority_buffer : bounded_buffer {
public:
 int interface get() {
  guard
   ((bounded_buffer::guard:get() != 0)
    && ((guard:put() == 0)
        || (put_queued == 0)))
  // true when the buffer is not empty
  // and when the buffer is full or
  // there are not put requests in
  // the object queue.
 }
}
```

Figure 4: The *priority_buffer* class. Note that the form bounded_buffer::guard:get() calls the *guard* part of the *get* interface method inherited from the *bounded_buffer* class and that the form guard:put() calls the *guard* part of the *put* interface method inherited from the *bounded_buffer* class.

3.2 Inheritance anomalies solutions

This approach solves the inheritance anomaly problem reducing the redefinition of method code and eliminating the modification of the main body of the methods when a subclass introduces new synchronization constraints. The following examples show how our approach solves the three types of inheritance anomaly.

3.2.1 Examples 4 and 5: partitioning of states anomaly

As example of the partitioning of states anomaly, let us define the *get2_buffer* class introduced in section 1. This class is a specialization of the previous *bounded_buffer* and its definition introduces an interface method, called *get2*, whose *body* gets couples of elements and whose *guard* evaluates to true when there is more than one element in the buffer (see figure 5).

```
class get2_buffer : bounded_buffer {
public:
 int* interface get2(int* els) {
  guard
   (((in - out) % MAX) > 1)
  // true when the buffer contains
  // more than an element.
  body {
   els[0] = buf[out++];
   els[1] = buf[out++];
   out % = MAX;
   return els;
  }
 }
}
```

Figure 5: The *get2_buffer* class.

This approach also gives a good solution when synchronization constraints depend on method arguments. For example, let us define a bounded buffer allowing the withdrawal of a variable number of elements fixed by the external requests. This bounded buffer is obtained by specializing the previous *bounded_buffer* class. This new class, called *mget_buffer*, introduces an interface method, called *mget*, whose *body* gets *m* elements from the buffer and whose *guard* evaluates to true when there are a number of elements into the buffer equal or greater than *m* (see figure 6).

```
class mget_buffer : bounded_buffer {
public:
 int* interface mget(int* els, int m) {
  guard
   (((in - out) % MAX) >= m)
  // true when the buffer contains a
  // number of elements equal or
  // greater than m.
  body {
   int i;
   for(i = 0; i < m; i++)
    els[i] = buf[out++];
   out % = MAX;
   return els;
  }
 }
}
```

Figure 6: The *mget_buffer* class.

```
class gget_buffer : bounded_buffer {
public:
 void interface put(int item) {
  transitions
   TRUE disable { gget };
  // gget cannot be the
  // next executed method.
 }
 int interface gget()
  guard
   guard:get()
  // true when the buffer
  // is not empty.
  body {
   body:get();
  // calls the body of get.
  }
 }
}
```

Figure 7: The *gget_buffer* class. Note that the forms `guard:get()` and `body:get()` respectively call the *guard* and the *body* of the *get* interface method inherited from the *bounded_buffer* class.

3.2.2 Examples 6 and 7: history - only sensitiveness of acceptable states anomaly

As example of the history - only sensitiveness of acceptable states anomaly, let us define the *gget_buffer* class introduced into section 1. This class is a specialization of the previous *bounded_buffer* and its definition introduces an interface method, called *gget*, that has the same *guard* and *body* parts of the inherited *get* interface method. Moreover, it specializes *put* interface method adding the *transitions* part to disable the *gget* interface method after its execution (see figure 7).

This approach allows an incremental modification of *transitions* synchronization constraints too. In fact, let us specialize the previous *gget_buffer* class introducing an interface method that removes the last element put into the buffer and that, in the same way of *gget*, cannot be accepted immediately after the invocation of *put*. In this case, we define a new class, called *gget_buffer1*, by adding the interface method *last*, and a new transition to the *transitions* of *put* that disables the execution of *last* (see figure 8).

```
class gget_buffer1 : gget_buffer {
public:
 int interface last() {
  guard
   guard:get()
  // true when the buffer is not empty.
  body {
   in = (--in % MAX);
   return buf[in];
  }
 }
 void interface put(int item) {
  transitions
   gget_buffer::transitions:put()
   TRUE disable { last };
  // gget and last cannot be
  // the next executed method.
 }
}
```

Figure 8: The *gget_buffer1* class. Note that the form `guard:get()` calls the *guard* part of the *get* interface method that it is inherited from the *gget_buffer* class and that the form `gget_buffer::transitions:put()` calls the *transitions* part of the *put* interface method that it is inherited from the *gget_buffer* class.

3.2.3 Example 8 and 9: modification of acceptable states anomaly

As example of the modification of acceptable states anomaly, let us define the *lock_buffer* class. The definition of this buffer can be obtained by simply mixing the definition of the *locker* class into the definition of the *bounded_buffer* class. The definition of the *locker* class contains two interface methods, *lock* and *unlock*; the *transitions* part of *lock* disables the execution of all the interface methods different from *unlock*[2]; the *guard* of *unlock* avoids its execution when the object is not *locked* (see figure 9).

Synchronization constraints can be incrementally modified, but can be redefined too. Therefore, problems do not exist in the definition of a subclass of the *locker* class introducing a method, *inquire_lock*, that must be executed irrespective of the value of the lock (see figure 10).

```
class locker c_unit {
 int locked:
public:
 locker() {
  locked = FALSE;
 }
 void interface lock() {
  body {
   locked = TRUE;
  }
  transitions
   TRUE  disable all_except { unlock };
  // only unlock can be executed.
 }
 void interface unlock() {
  guard`
   (locked == TRUE)
  // true when the object is locked.
  body {
   locked = FALSE;
  }
 }
}


class lock_buffer
      : locker, bounded_buffer
{ }
```

Figure 9: The *locker* and *lock_buffer* classes.

[2]Its effect hits all the interface methods of the subclasses of the *locker* class.

```
class inquire_locker : locker {
public:
 int interface inquire_lock () {
  body {
   return locked;
  }
 }
 void interface lock() {
  transitions
   TRUE disable
     all_except { unlock inquire_lock };
  // only unlock and inquire_lock
  // can be executed.
 }
}
```

Figure 10: The *inquire_locker* class.

4 Conclusions

In this paper, we have presented a framework to solve the problem of the interference between inheritance and concurrency in concurrent object-oriented programming languages. This synchronization mechanism has been used into a C++ implementation of CUBL - Concurrent Unit Based Language (Poggi, 1994). CUBL is a concurrent object-oriented programming language, which derives from the Actor model (Agha, 1986) and ABCL (Yonezawa, 1990). This languages includes both passive and active objects, and communication is based on a large set of message exchange primitives which allow broadcasting besides past, now and future messages. CUBL supports physical distribution of the objects in a UNIX network managing communication with external objects via UNIX sockets.

Our synchronization scheme derives from guard methods (Decouchant et al., 1989) adding to them a new synchronization part, that is, the *transitions* part, to solve the two types of inheritance anomalies that guard methods cannot solve, that is, the history - only sensitiveness of acceptable states and the modification of acceptable states anomalies.

Unlike Matsuoka and Yonezawa (1993), we attach guards to single methods and not to accept methods sets because it is simpler and more natural to define a method together with its synchronization constraints than to define together all the synchronization constraints of a class. In fact, when a class contains a large number of accept method sets the addition of a new method may be complex because the programmer must understand what accept method sets must contain this method, and may cause a cumbersome redefinition of the synchronization code when this method must be included into a large part of the inherited accept method sets. Moreover, on the one hand, the *transitions* part may reduce the redefinition of methods in the cases of history - only sensitiveness of acceptable states and modification of acceptable states anomalies. On the other hand, the association of a guard to a single method simplifies and often eliminates re-definitions of inherited methods in the cases of state partitioning anomaly.

Class and methods can be specialized taking advantage of C++ inheritance rules. In particular, the three parts of an interface method can be separately specialized. The use of C++ allows an efficient implementation of interface methods. In fact, its three parts are translated into three inline C++ methods and so the call of an interface method is not made heavy by additional method calls. Moreover, the use of program transformation for the implementation of the *guard* parts reduces the overhead that would be derived from their naive implementation (Matsuoka et al., 1990).

This framework is based on the assumption that an object has at most one active thread and so method executions are serialized. The problem to have several active threads in an object is one of our future research directions and can be simply solved on the basis of the method used by Synchronization Actions (Neusius, 1991). In fact, we can introduce a new synchronization part to be executed after the evaluation of the *guard* part of the method. This new part will have the same form of the *transitions* part and its task will be to disable the methods that cannot be executed in parallel to that method.

Other future research directions are the development of real applications to make experiences on the use of such an object model and the refinement of the object model implementation to increase efficiency.

5 Acknowledgements

The work has been partially supported by the National Research Council through "CNR - Progetto Finalizzato Trasporti" and by the Ministri of University, Scientific Research and Technology through "MURST%40 - Sistemi Intelligenti".

References

[1] G. Agha. *Actors, A Model of Concurrent Computation in Distributed Systems.* The MIT Press, Cambridge, MA, 1986.

[2] G. Agha, P. Wegner, and A. Yonezawa. *Research Directions in Concurrent Object-Oriented Programming.* The MIT Press, Cambridge, MA, 1993.

[3] P. America. Inheritance and subtyping in a parallel object-oriented language. In *Proc. OOPSLA '87*, pages 234–242, Orlando, FL, 1987.

[4] P. America. POOL-T: a parallel object-oriented language. In A. Yonezawa and M. Tokoro, editors, *Object-Oriented Concurrent Programming*, pages 199–220, Cambridge, MA, 1987. MIT Press.

[5] R.S. Chin and S.M. Chanson. Distributed object-based programming systems. *ACM Comput. Surveys*, 23(1):91–124, 1991.

[6] D. Decouchant, S. Krakowiak, M. Meysembourg, M. Riveill, and X. Rousset de Pina. A synchronization mechanism for typed objects in a distributed system. *SIGPLAN Notices*, 24(4):105–107, 1989.

[7] S. Frolung. Inheritance of synchronization constraint in concurrent object-oriented programming languages. In *Proc. ECOOP '92*, pages 185–196, Utrecht, The Netherlands, 1992.

[8] C. Hewitt and P. de Jong. Analyzing the rules of descriptions and actions in Open System. In *Proc. AAAI-83*, pages 162–167, Washington, DC, 1983.

[9] D.G. Kafura and K.H. Lee. Inheritance in actor based concurrent object-oriented languages. *The Computer Journal*, 32(4):297–304, 1989.

[10] K. Kahn, E. Tribble, M. Miller, and D.G. Bobrow. Vulcan: Logical concurrent objects. In B. Shriver and P. Wegner, editors, *Research Directions in Object-Oriented Programming*, pages 75–112, Cambridge, MA, 1987. The MIT Press.

[11] S. Matsuoka, K. Wakita, and A. Yonezawa. Synchronization constraints with inheritance: What is not possible - so what is? Technical Report 10, Department of Information Science, The University of Tokio, Tokio, Japan, 1990.

[12] S. Matsuoka and A. Yonezawa. Analysis of inheritance anomaly in object-oriented concurrent programming languages. In G. Agha, P. Wegner, and A. Yonezawa, editors, *Research directions in Concurrent Object-Oriented Programming*, pages 107–150, Cambridge, MA, 1993. MIT Press.

[13] C. Neusius. Synchronization actions. In *Proc. ECOOP '91*, pages 118–132, Geneva, Switzerland, 1991.

[14] O. Nierstrasz. Active objects in Hybrid. In *Proc. OOPSLA '87*, pages 243–253, Orlando, FL, 1987.

[15] A. Poggi. Agents and resources management with CUBL. In *Proc. HICSS '94*, pages 112–121, Maui, HI, 1994.

[16] C. Tomlinson and V. Singh. Inheritance and synchronization with enable-sets. In *Proc. OOPSLA '89*, pages 103–112, New Orleans, LA, 1989.

[17] P. Wegner. Concepts and paradigms of object-oriented programming. *OOOPS Messenger*, 1(1):7–87, 1990.

[18] B.B. Wyatt, K. Kavi, and S. Hufnagel. Parallelism in object-oriented languages: a survey. *IEEE Software*, 9(6):56–66, 1992.

[19] A. Yonezawa. *ABCL: An Object-Oriented Concurrent System.* The MIT Press, Cambridge, MA, 1990.

The Fringe Bucket-Quadtree, a Robust and Efficient Spatial Data Structure with a High Population Density

R. Maelbrancke and H. Olivié
K.U.Leuven, Department of Computing Science, Celestijnenlaan 200A, B3001 Heverlee, Belgium
Phone: ++32-16-327643, Fax: ++32-16-327996
E-mail: {rudim,olivie}@cs.kuleuven.ac.be

Keywords: quadtree, spatial data structures

Edited by: Matjaž Gams
Received: June 1, 1995 **Revised:** September 20, 1995 **Accepted:** October 10, 1995

In this paper we present three methods for the augmentation of population density in bucket oriented quadtrees. These improvement techniques are incorporated in a new data structure called the Fringe Bucket-quadtree.
Analysis of this structure shows a population density of 77%.

1 Introduction

In a number of application areas, such as geographic information systems (GIS for short) and spatial databases, searches involving several attributes are very common. If such multiple-attribute searches are the rule and the single-attribute searches are the exception, multi-attribute indexes are the best choice. Spatial data structures are best suited to represent multi-attribute indexes. An overview of such data structures is given by Samet [12].
Since the number of data represented in a GIS is huge, indexes for external storage media are recommended. For an optimal use of the external storage capacities, it is recommended that such indexes have a high filling degree or population density. The **population density** is the relationship between the actual number of data items in a bucket and the maximal number of data items the bucket can hold.
In this paper a **fringe search** quadtree is introduced. Fringe search trees have data references in a limited number of nodes at the end of each search path.
The fringe search tree compares to a combination of a node search tree and a leaf search tree. Leaf search trees store their data references only in the leaves. Node search trees store data in the internal nodes.
We call a bucket oriented fringe search quadtree a **Fringe-Bucket quadtree** (FB-quadtree for short). The FB-quadtree has a high population density combined with a low and deterministic number of block acesses.
To argue the way we improved quadtrees, we first give an overview of preceding research results.

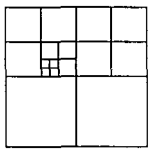


Figure 1: A PR quadtree

1.1 PR quadtrees

In [12], Samet presents *bucket Point Region quadtrees* (PR quadtrees) as an alternative for Finkel's point quadtrees [3].
PR quadtrees differ from quadtrees in three ways (cf. figure 1).

1. when a region is split it is devided into four equal sized squares
2. the PR quadtree is a leaf search tree; every time a node is split the data are distributed among the four sons
3. the leaves can contain more than one information item; the number depends on the size of the data buckets the leaves are referring to

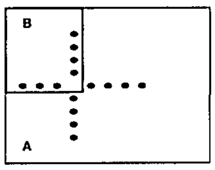


Figure 2: A holey brick (A)

Since PR quadtrees have a data structure comparable to point quadtrees, search algorithms can be borrowed [14]. However, the population density is only 46% on average [1].

1.2 Holey Bricks

Spatial data structures have often been compared with their linear ancestors. Especially when bucket trees were introduced, the average population density of 69% in B-trees became a main goal. Nievergelt was one of the first researchers to mention the population density for spatial data structures in [8]. He conducted simulation studies showing a 70% population density. However, problems such as the splitting of buckets other than the overflowing bucket and the dead space problem due to the lack of merging candidates, may seriously degrade the population density. The kdB-tree [11], a spatial B-tree, reached a population density of only 60% for uniformly distributed data. The BD tree [9], also termed BANG file [4], is another major example. However, all these structures are very dependent on the data distribution.

In 1990, Lomet [7] introduced the holey brick tree (hB-tree for short). It opened a new direction in the quest for the 69%.

The main problem in reaching this goal is the way the splitting of the space is performed.

When the points are distributed in the form of a cross (cf. figure 2), the region cannot be split up into two regions in such a way that there is a good distribution over the two parts.

Therefore Lomet has introduced holey bricks. A region is divided into equal shaped parts if there is no cross distribution problem, otherwise a small part is cut out (cf. figure 2).

However, hB-trees have a fairly complex structure which is hard to maintain.

To avoid a complex data structure when using holey bricks, Ouksel has introduced the Nested Interpolation Based Grid File (NIBGF for short) [10]. The main idea is that on a same level regions are split into two if a $\frac{1}{3}$ to $\frac{2}{3}$ distribution over the two regions can be reached, otherwise a small part is pushed down to a lower level in order to obtain a holey brick. Since the NIBGF uses B-trees to store information, it is a very efficient and robust data structure. A main drawback is the complexity of the algorithms, especially for the deletion.

1.3 Joining Buddy Partitions

Seeger [13] could reach a 74.5% population density [5] by joining neighbour partitions into one bucket in his implementation of the BUDDY+ structure. The fact that the BUDDY+ structure is static restricts its applications.

1.4 Conclusion

We know from Samet that quadtrees are easy to maintain and commonly used structures. It has been argued that the alternatives mentioned above are all very complex and difficult to maintain.

In section 2, methods to improve the population density of PR quadtrees are introduced.

A data structure, incorporating this population density improvements, the Fringe Bucket-quadtree is defined in section 3 and empirically analyzed in section 4.

2 Augmenting the Population Density

Before we give three methods for the augmentation of the population density of PR quadtrees, we give a short description of the node structure.

2.1 The PR Quadtree Node Structure

The node structure consists of two parts, the discriminators and the keys. The discriminators are kept in main memory, while the keys are stored on block structured external storage media.

We represent a node in a PR quadtree as five pointers. Four pointers are used for the North-West (NW), North-East (NE), South-West (SW) and South-East (SE) sons. In case of a leaf node the

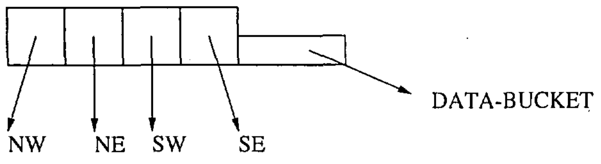


Figure 3: A node of a PR quadtree

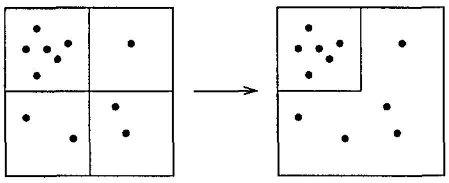


Figure 4: A possible combination of squares in a quadtree

fifth pointer points to the external data bucket containing the keys and the data references.
In figure 3 a schematic representation of the node of a PR quadtree is given.
As a PR quadtree always splits in the same way, the coordinates of the square in the quadtree partition can be calculated dynamically. Therefore the coordinates of the represented square must not be kept in the nodes.

2.2 Method 1: The Combination of Nodes

In PR quadtrees, regions are always split into four equal sized squares. This may lead to underpopulated nodes.
Method 1 combines nodes in such a way that a higher population density is reached. If the nodes are chosen carefully, a fairly good distribution of the points is obtained.
Figure 4 illustrates a combination of squares.
The resulting structure can still be traversed in the same way as a PR quadtree. If a node represents a combination of squares, it has more than one incoming edge.
If a node that represents a combination of squares flows over it is split in such a way that the best population density is obtained. Once a node represents only one square, when it flows over it will give birth to new combined nodes.
The empirical analysis of the implementation of this method (cf. section 4) shows a population density of 66%.

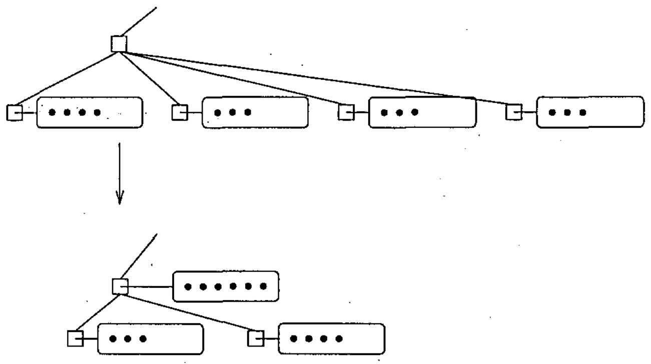


Figure 5: The transformation a leaf search quadtree into a node search quadtree

2.3 Method 2: Transforming the PR Quadtree into a Node Search Tree

The population density can be augmented if the PR quadtree is transformed into a node search tree.
In case of an overflow, the contents of the node do not have to be distributed over the son nodes anymore. On the contrary, the father node may keep the data, while the sons are ready to accept new data.
If we adapt method 1 to node search trees, an overflowing node gives birth to only one son node which is the combination of four squares. Hence, this combined son node represents a space that has the same size as it fathers one. In that way, the 50% minimum bound is obtained if the data are distributed over the father and the son.
This method also solves the cross-distribution problem and the repeated splitting problem (in case of points that are located extermely close to each other). Data are indeed distributed over the levels instead of over the nodes on one level.
In figure 5 an example is given.

Let us analyze the node occupancy mathematically. The lowest level of an original PR quadtree contains at most 3 times the number of nodes in the rest of the tree. We know that the filling degree in the lowest level of a PR quadtree improved using method 1 is 66%. If only insertions are performed, the higher levels of a PR quadtree that is improved with methods 1 and 2, have a population density of 100%, leading to

$$\frac{2}{3} \cdot 66 + \frac{1}{3} \cdot 100 = 77$$

This is more than what Kriegel [5] has obtained with the BUDDY+ structure.
Empirical analysis of method 1 combined with method 2 show a population degree of 77.7%.

2.4 Method 3: Restricting Method 2 to the Lowest Two Levels

As separating discriminators and data is essential for good range search performance [6], method 2 is not ideal.
If we restrict method 2 to the lowest two nodes on each search path of the tree,we only weaken a little the separation of discrimators and data.
If we define this weakening strictly, we require that on each path towards a key, at most two data buckets should be visited. Since nowadays memory caches are much larger than data blocks on external media, these two levels will not remarkably reduce performance.
The empirical analysis of method 3 shows a slightly higher population density than the one obtained with method 2, namely 78%.

3 Fringe Bucket-Quadtrees

3.1 Definition

The multi-attribute keys of the data are represented by n coordinates of an n-dimensional space S.

Preliminary Definitions

- S is split repeatedly in n-squares
- a *simple node* is a node that represents an n-square
- a *compound node* is a node representing a combination of n-squares
- a compound node that contains 2^n n-squares is called *complete.*
- an *inner node* is a node that has no data references
- a node containing a data reference is called a *data node*

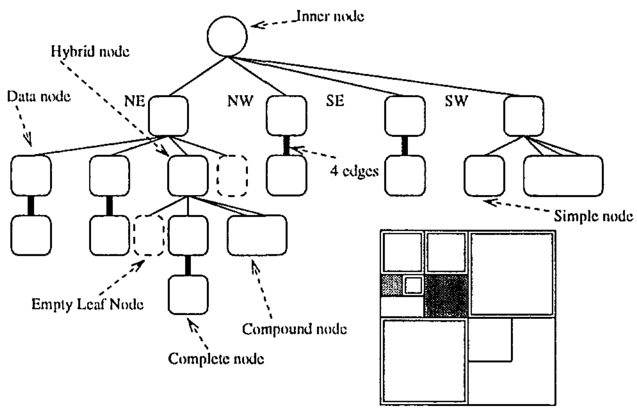


Figure 6: The structure of an FB-quadtree

- a data node that does not represent data for one or more n-squares of the region it represents, is called a *hybrid node* [1]
- a data node with no sons is called a *leaf node*

Definition 1 A hierarchical spatial data structure (tree for short) is called a **Fringe Bucket-Quadtree, FB-quadtree** for short, iff

1. exactly one node, called the *root* has no incoming edges; the root represents S
2. each inner node ν has 2^n outgoing edges; each edge represents an n-square of the partition into 2^n of the subspace S_ν represented by ν; each edge points to the node (either a simple or a compound node) that contains the represented n-square
3. each son node of ν may have at most 2^n incoming edges from ν
4. a compound node does not have sons
5. on each path from the root to a leaf, at most the last two nodes in the path can refer to the data buckets . Such a bucket can contain at most b keys.

We denote by b the bucketsize of an FB-quadtree. In figure 6 a scheme of the FB-quadtree is shown.

[1]This case occurs when a search path is crossing the data node while still more than one data node is following this one

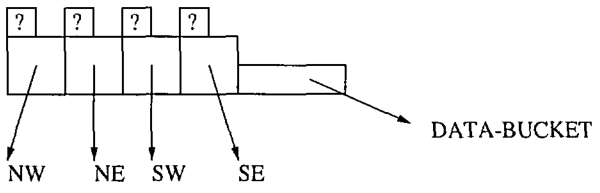


Figure 7: A node of an FB-quadtree

3.2 The Node Structure

The node structure for FB-quadtrees is the data structure of the PR quadtree, augmented with a tag for each outgoing edge. A true edge tag determines the presence of data for the n-square represented by that edge in the referred data bucket. Figure 7 gives a schematic view on the node structure for $n = 2$.

3.3 The Point Search Algorithm

Search for the key k in an FB-quadtree τ

```
BEGIN
    start from the root;
    determine in which $n$-square $k$ can be placed;
    let $E$ be the edge associated with that $n$-square;
    IF the current node is an inner node
    THEN continue the search on the lower level in the node reached along $E$;
    ELSE
        IF the current node is a hybrid node and the $E$-tag is false
        THEN continue the search on the lower level in the node reached along $E$;
        ELSE
            BEGIN
                search in the data bucket of the current node for $k$;
                IF $k$ is found THEN stop
                ELSE continue the search on the lower level in the node reached along $E$;
            END
END
```

3.4 The Insertion Algorithm

The insertion algorithm is described top down.

3.4.1 Insert the Key k

Initially an FB-quadtree consists of one simple leaf node.
Insert the key k.

```
BEGIN
    perform the point search algorithm ($\beta$ denotes the node where the search ends);
    IF the father of $\beta$ denoted by $\phi(\beta)$ has vacant positions
    THEN add $k$ to $\phi(\beta)$
    ELSE CASE $\beta$
        has **vacant** positions: add $k$ to $\beta$; break;
        is a **simple leaf node**: CALL **double** $\beta$ and add $k$ to $\beta$; break;
        **otherwise** CALL **split or reduce** $\beta$ and add $k$ to $\beta$
        END
END
```

3.4.2 Double the Data Node

Double β

```
BEGIN
    create a complete compound node on the level below $\beta$;
    move the keys in $\phi(\beta)$ that correspond to the $n$-square represented by $\beta$ to the son of $\beta$
    set the tag in $\phi(\beta)$ which corresponds to the incominge edge of $\beta$ to false;
    IF the son of $\beta$ contains less than 50% of keys
    THEN move keys from $\beta$ to its son until the 50% is reached
END
```

Figure 8 and figure 9 illustrate this case.

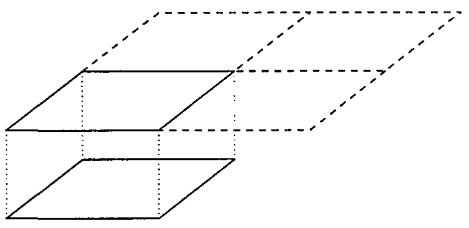


Figure 8: Doubling a data node

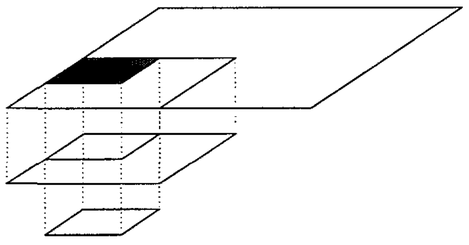


Figure 9: The creation of a hybrid node

3.4.3 Split or Reduce the Data Node

Split or reduce β.

```
BEGIN
    IF all keys are located in one $n$-square of
       the region
    THEN $\beta$ is reduced to a simple data node
       that represents the involved $n$-square
       and is doubled
    ELSE split $\beta$ in a simple data node and ei-
       ther a compound data node or simple
       data node depending on the arity of $\beta$.
END
```

The cases are respectively illustrated in figures 10 and 11.

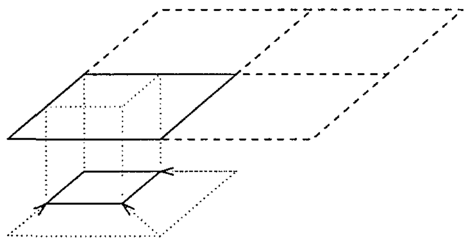


Figure 10: Reducing a bucket

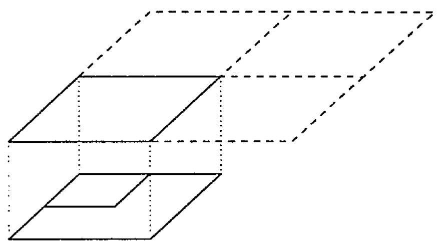


Figure 11: Splitting a bucket

3.5 The Deletion Algorithm

The deletion algorithm can be derived directly from the insertion algorithm:
Delete key k.

```
BEGIN
    perform the point search algorithm ($\beta$ de-
       notes the node where the search ends);
    remove $k$ from $\beta$;
    IF the population in $\beta$ is less than 50%
    THEN
        BEGIN
            IF $\phi(\beta)$ contains more than 50%
               of keys
            THEN move a key from $\phi(\beta)$ to $\beta$
            ELSE
            IF a brother compound node of $\beta$
               contains less than 50% of keys
            THEN join $\beta$ with this brother
            ELSE
            IF $\beta$ is a complete compound node
            THEN
                BEGIN
                 join $\beta$ with $\phi(\beta)$;
                 IF $\phi(\phi(\beta))$ is a hybrid node
                 THEN set to false $\phi(\phi(\beta))$'s
                 edge tag pointing to $\beta$
                END
        END
    IF $\phi(\beta)$ is a hybrid bucket and $\beta$ has no sons
       anymore
    THEN set to false $\phi(\beta)$'s edge tag pointing
       to $\beta$
END
```

Distribution	$b = 16$	$b = 100$
Uniform	77.33 % [77%, 79%]	78.22 % [75%, 80%]
Gauss	78.00 % [77%, 79%]	76.90 % [76%, 78%]
Diagonal Gauss	77.82 % [77%, 78%]	74.73 % [73%, 75%]
Vertical Gauss	77.82 % [77%, 78%]	76.51 % [75%, 77%]
Gauss Clusters	77.60 % [77%, 78%]	75.86 % [74%, 77%]
Average	77.71 %	76.45 %

Table 1: Mean population density and running mean population density bounds of FB-quadtrees

4 Empirical Analysis

4.1 Input Sets

To test the robustness and behaviour of the FB-quadtree, several input sets were generated. All sets consist of one million points taken from [0..8192[×[0..8192[. We have chosen real values to avoid an upper bound on the height of the tree.
Five kinds of distributions have been generated, namely:

1. a uniform distribution
2. a Gaussian distribution
3. a diagonal Gaussian distribution, along the diagonal a small band of gaussian distributed points
4. a vertical line distribution, as in 3, but along a vertical axis
5. ten clusters of Gaussian distributions

4.2 Results

In the following tables(1,2) results of empirical analysis are given. To calculate the mean population density, the running average is calculated from the 10000th insertion on. The result given is the running average while performing the 1000000th insertion. The lower and upper bounds of the running average are given too ([minimum %, maximum %]).
To show the influence of the bucket size, the experiment is redone for small b's and shows an even better filling degree. Remarkable is the high filling for $b = 2$, which means that more than half of the buckets are filled. These results are displayed in table 3.

Distr. / Level	b	[1,2,3,4,5,6,7,8,9,10,11,12,13]
Uniform	16	[0,0,0,0,0,0,97,73,58,0,0,0,0]
	100	[0,0,0,0,100,100,56,0,0,0,0,0,0]
Gauss	16	[0,0,89,85,85,85,85,86,72,61,0,0,0]
	100	[0,98,82,87,86,85,85,61,0,0,0,0,0]
Diagonal Gauss	16	[0,0,0,0,0,0,59,78,85,90,73,59,0]
	100	[0,0,0,0,0,53,84,72,90,59,0,0,0]
Vertical Gauss	16	[0,0,0,0,0,0,97,96,83,90,72,59,0]
	100	[0,0,0,0,0,0,47,88,88,61,0,0,0]
Gauss Clusters	16	[0,7,5,21,25,45,77,86,85,86,68,58,0]
	100	[5,4,12,23,40,69,82,85,82,61,0,0,0]

Table 2: The distribution of the population densities (in %) among the levels

Distribution	$b = 2$	$b = 4$	$b = 8$
Uniform	85.26 %	81.02 %	79.15 %
Gauss	85.19 %	81.00 %	79.04 %

Table 3: The population density under several bucket sizes (100000 points)

4.3 Discussion of the Results

The empirical analysis gives us a good indication of the improvements that have been made applying the introduced adaptations to PR quadtrees.
The population density of FB quadtrees is higher than more complicated structures which were invented to obtain a higher population density (e.g. hB-trees, NIBGF, BUDDY, ...).
In comparison with the results obtained by Ang and Samet [1] for uniformly distributed data the gain is about 72%: 47% → 81%. This comparison is shown in table 4.

5 Conclusion and Further Research

We have introduced three methods for the augmentation of the population density in PR-

b	PR quadtree	FB-quadtree
2	46.0 %	85.2 %
4	46.3 %	81.0 %
8	47.5 %	79.0 %
Avg.	46.6 %	81.7 %

Table 4: Population comparison with PR quadtrees

quadtrees. We gained about 70%, resulting in a 77% population density, by combining parts and allowing the two lowest nodes in a path to contain data.
These methods have resulted in a new structure called the Fringe Bucket-quadtree. It is very useful for database implementations.
Our further work will consist in the elaboration and the analysis of a generalization of the FB-quadtree for higher dimensions.
Current analysis of the octtree case show comparable results.

References

[1] Ang C. & Samet H. (1989) Node distribution in a PR quadtree. In *Design and Implementation of Large Spatial Databases*, p. 233–252, Berlin Heidelberg New York: Springer Verlag.

[2] Bayer R. & McCreight E. (1972) Organisation and maintenance of large ordered indexes. *Acta Informatica*, 1, 3, p. 173–189.

[3] Finkel R. A. & Bentley J. L. (1974) Quad trees: a data structure for retrieval on composite keys. *Acta Informatica*, 4 ,1, p. 1–9.

[4] Freeston P. (1987) The BANG file. In *Proceedings of ACM SIGMOD*, p. 260–269.

[5] Kriegel H. P. & Schiwietz .M et al (1989) Performance comparison of point and spatial access methods. In *Design and Implementation of Large Spatial Databases*, p. 89–114, Berlin Heidelberg New York: Springer Verlag.

[6] Lomet D. B. (1991) Grow and post index trees: Role, techniques and future potential. In *Advances in Spatial Databases*, p. 183–206, Berlin Heidelberg New York: Springer Verlag.

[7] Lomet D. B. & Salzberg B. (1990) The hB-tree: A multiattribute indexing method with good guaranteed performance. *ACM Transactions on Database Systems*, 15 ,4, p. 625–658.

[8] Nievergelt J. & Hinterberger H. & Sevcik K. C. (1984) The grid file: an adaptable symmetric multykey file structure. *ACM Transactions on Database Systems*, 9, 1, p. 38–71.

[9] Ohsawa J. & Sakauchi M. (1983) The BD-tree: a new n-dimensional data structure with high efficient dynamic characteristics. *Information Processing*, 83, p. 539–544.

[10] Ouksel M. A. & Mayer O. (1992) A robust and efficient spatial data structure. *Acta Informatica*, 29, p. 335–373.

[11] Robinson J. T. (1984) The K-D-B-tree: A search structure for large multidimensional dynamic indexes. In *Proceedings of ACM SIGMOD Conference on Management of Data.*

[12] Samet H. (1990) *The Design and Analysis of Spatial Data Structures.* Addison-Wesley Series on Computer Science and Information Processing. Addison-Wesley Publishing Company.

[13] Seeger B. & Kriegel H. P. (1990) The buddy-tree: an efficient and robust access method for spatial data base systems. In *Proceedings of the 16th International Conference on Very Large Data Bases*, p. 590–601.

[14] Shafferd C. A. & Samet H. & Nelson R. C. (1990) Quilt: a geographic information system based on quadtrees. *Int. J. Geographical Information Systems*, 4, 2, p. 103–131.

THE NEW YORK ACADEMY OF SCIENCES (NYAS)

Oh Plato, Oh Socrates, you who walked and talked in the grove of Academe, join us in our walks and talks—and take pride in the published records of the new and the true that our Academy gives the world.

—John A. Wheeler
Astrophysicist, University of Texas [1]

In this column, *Informatica* will bring periodically, news and relevant information concerning the largest worldwide academy of sciences—The New York Academy of Sciences. Reports will include the initiatives and activities of the NYAS members related to the editorship of *Informatica* and the support offered by NYAS to scientific events and communities outside NYAS.

1 The World-wide Relevance of NYAS

NYAS is a proper cosmopolitan academy, according to its membership spread over the globe, its international organization experience, scientific excellence regarding members and publications, and a long—almost 180 years lasting—tradition and democracy. This position is wide before other, nationally conceptualized academies of sciences, in which certain national(ist) preferences are dominating, for example, political impact, national appurtenance, privileged status of local academy members (e.g., their permanent monthly financial rewarding), ethnic (nepotistic) selection, arrangement of rules, etc.

2 A Short Presentation of NYAS

2.1 Introduction

A new member of the Academy [1] becomes part of the oldest and most distinguished scientific societies in America, if not in the world—a society which since 1817 has numbered among its members Thomas Jefferson, John James Audubon and, in this century, more than 40 Nobel Laureates and many other eminent scientists.

The international membership embraces over 47,000 distinguished scientists from every state in the U.S.A. and some 160 countries all over the world [2]. Through individual membership one's knowledge will be enriched and broaden when communicating with colleagues in one's own and related fields, but it will also bring the member a remarkable range of tangible benefits of substantial value from other subjects.

2.2 Board of Governors

The board of the governors in the period July 1995–June 1996 is the following:

- CHAIRMAN OF THE BOARD
  Joshua Lederberg, Nobel Laureate in Physiology or Medicine, Geneticist, The Rockfeller University
- PRESIDENT
  Henry M. Greenberg, Columbia University
- PRESIDENT-ELECT
  Martin L. Leibowitz, Managing Director, Salomon Brothers Inc
- TREASURER
  Henry A. Lichstein, Vice President, Citibank, N.A.
- GOVERNORS

  Eleanor Baum, Dean, School of Engineering, The Cooper Union

  Barry R. Bloom, Weinstock Professor, Department of Microbiology and Immunology, Albert Einstein College of Medicine

  D. Allan Bromley, Sterling Professor of the Sciences & Dean of Eng., Yale University

  Edward Cohen, Chairman & CEO, Ammann & Whitney, Inc

  Susanna Cunningham-Rundles, Associate Professor of Immunology, Cornell University Medical College

  Bill Grenn, Director, N.Y.C. Housing Development Corporation

  Sandra Panem, Vice President, Oppenheimer Management Corporation

  Richard A. Rifkind, Chairman, Sloan-Kettering Institute

  Dominick Salvatore, Professor and Director of Graduate Program, Department of Economics, Fordham University

  David E. Shaw, Managing General Partner, D.E. Shaw & Co

William C. Steere, Jr., Chairman & CEO, Pfizer Inc

Shmuel Winograd, Director, Mathematical Sciences Department, T.J. Watson Research Center, IBM

- PAST CHAIRMAN
  Cyril M. Harris, Batchelor Professor Emeritus of Electrical Engineering & Professor Emeritus of Architecture, Columbia University
- HONORARY LIFE GOVERNOR
  William T. Golden, Chaiman Emeritus, American Museum of Natual History
- LEGAL COUNSEL: *Helene L. Kaplan,* Skadden, Arps, Slate, Meagher & Flom

2.3 Senior Staff

CHIEF EXECUTIVE OFFICER *Rodney W. Nichols*

EXECUTIVE EDITOR, THE ANNALS *Bill M. Boland*

CONTROLLER *Vernon A. Bramble*

EDITOR-IN-CHIEF, THE SCIENCES *Peter G. Brown*

DIRECTOR, COMMUNICATIONS *Ann E. Collins*

DIRECTOR, MARKETING & MEMBERSHIP *Katherine T. Goldring*

MANAGER, SECTIONS AND MEETINGS *Matthew M. Katz*

DIRECTOR, HUMAN RESOURCES *Leslie A. Mooney*

DIRECTOR, POLICY STAFF *Susan U. Raymond*

DIRECTOR, EDUCATION *Lori D. Skopp*

DIRECTOR, HUMAN RIGHTS OF SCIENTISTS *Svetlana Stone*

DIRECTOR, CONFERENCES *Renée Wilkerson (acting)*

2.4 Section Chairs and Vice-Chairs

ANTHROPOLOGY CO-CHAIRS
Linda Basch and R. Brian Ferguson
CO-VICE-CHAIRS
Antonio Lauria-Pericelli and Johanna Lessinger

ATMOSPHERIC SCIENCES CHAIR
Stanley Gedzelman
VICE-CHAIR *Mark Kramer*

BIOCHEMISTRY CHAIR
Laurence P. Wennogle
VICE-CHAIR *Jo Anne M. Saye*

BIOLOGICAL SCIENCES CHAIR
Gertrude Pfaffenbach
VICE-CHAIR *Lorraine M. Moran*

BIOMEDICAL SCIENCES CHAIR
Victoria N. Luine
VICE-CHAIR *Christina R. McKittrick*

CHEMICAL SCIENCES CHAIR
A.M. Venkatesan
VICE-CHAIR *Kang Zhao*

COMPUTER AND INFORMATION SCIENCES CHAIR
D. Frank Hsu
VICE-CHAIR *Pauline M. Rothstein*

ECONOMICS CHAIR
Laurence R. Klein
CO-VICE-CHAIRS *Dominick Salvatore and Edmund Phelps*

ENGINEERING CHAIR
Joel J. Kirman
VICE-CHAIR *James Cohen*

ENVIRONMENTAL SCIENCES CHAIR
David C. Locke
VICE-CHAIR *John L. Cusack*

GEOLOGICAL SCIENCES CHAIR
Gerald M. Friedman
VICE-CHAIR *Daniel Habib*

HISTORY AND PHILOSOPHY OF SCIENCE CHAIR
David Cassidy
VICE-CHAIR *Bruce Chandler*

INORGANIC CHEMISTRY AND CATALYTIC SCIENCE CHAIR
Lynn Francesconi
VICE-CHAIR *Robert Beer*

LINGUISTICS CHAIR
Avraham Schweiger
VICE-CHAIR *F. Frank LeFever*

MATHEMATICS CHAIR
Helen Strassberg
VICE-CHAIR *David Seppala-Holtzman*

MICROBIOLOGY CHAIR
Zigmund C. Kaminski
VICE-CHAIR *Stephen Morse*

NEUROSCIENCE CHAIR
Alcmene Chalazonitis
VICE-CHAIR *Marlene Schwanzel-Fukuda*

PHYSICS AND ASTRONOMY CHAIR
Harry Sticker
VICE-CHAIR *David W. Kraft*

POLYMER SCIENCE CHAIR
Kalle Levon
VICE-CHAIR *Abraham Ulman*

PSYCHOLOGY CHAIR
Joan Gay Snodgrass
VICE-CHAIR *Barbara A. Mowder*

SCIENCE AND PUBLIC POLICY CHAIR
Frank Fischer
VICE-CHAIR *Maarten de Kadt*

SCIENCE EDUCATION CHAIR
John W. Fedors
VICE-CHAIR *Ellen Goldstein*

WOMEN IN SCIENCE CHAIR
Alice Deutsch
VICE-CHAIR *Frances Stern*

Chairs and vice-chairs data are from [3].

2.5 Scientific Conferences

Each year the Academy sponsors 15–20 conferences where internationally recognized authorities from different specialities gather to discuss some of the most significant research being conducted today. The hallmark of an Academy conference is an interdisciplinary approach of the topics of common interest.

Some Academy conferences are held in Europe (e.g., Czech Republic, Hungary, Italy, etc.). It depends upon the initiative and organization possibilities of the members outside U.S.A.

Every country grapples with the goal of ensuring that science and technology are powerful engines of economic growth. The Academy has joined with the Federal Reserve Bank of New York to present a conference on Technology and Economic Development. The insights will be important worldwide.

2.6 Human Rights

The Academy has an active human rights program, including identification of and protest against human rights abuses of scientists around the world. Andrei Sakharov and Fang Lizhi, who both made their first U.S. appearances at the Academy, are among the scientists who have credited the Academy with coordinating efforts that led to their release from exile.

In 1995, NYAS Human Rights of Scientists Award was imparted to Chinese Academics Xu Liangying and Ding Zilin in recognition of their roles in calling for the establishment of democratic law in China, and for their compassionate support of survivors and other dissidents involved in the Tiananmen Square incident. Xu, who drafted the petition, is a noted physicist and translator of Albert Einstein's works. Ding, a philosopher at the Chinese People's University in Beijing, whose son was killed at Tiananmen Square, is compiling a book of names of those who were killed or injured in the incident. She has endured numerous house arrests and death threats by the Chinese government.

The Academy has long been a champion in the field of human rights. In May, the Academy teamed with Columbia University's Center for the Study of Society and Medicine to present a workshop for health professionals.

3 A Short Analysis of the NYAS Membership

For Spring 1995 a picture of the Academy's worldwide membership was provided [5]. With membership growing rapidly during the past three years, the Academy turned to Roper Strach Worldwide, a market research firm, to learn more about members and their views. The study confirms recent changes in the membership and examines members' views of current and prospective Academy programs and services.

Since 1989 the number of members outside the U.S. has risen significantly and now makes up about half the membership. The Academy's approximately 47000 members in some 160 countries around the world are distributed regionally as follows:

Eastern Europe	7	percent
Western Europe	22	percent
Asia and the Pacific	12	percent
Africa and the Middle East	4	percent
South and Central America	4	percent
North America	51	percent

In addition, to the U.S., 15 countries, each, are home to 500 or more Academy members. Those countries in descending order are Japan, Germany, Russia, France, Canada, and Italy—with more than 1,000 members each—and Australia, the United Kingdom, Spain, Switzerland, Sweden, Brazil, Argentina, Israel, and Belgium.

The sample used for the survey was stratified into cells that reflect the regional distribution of the members, and it consisted of 3,500 members in 36 countries in each of which there are 70 or more members.

The avarage age of the Academy members is 55 years, with the mean age of members in the U.S. being somewhat higher than that in other parts of the world. The Asia-Pacific region, with the mean age 48, is the only area where the mean age is below 50. Overall, 15 percent of the members are female, but that segment ranges from 17 percent in the U.S. to only 3 percent in Western Europe.

According to the NYAS survey, nine of ten members are employed, and three out of four are employed full-time. Among those employed, the distribution in terms of setting is as follows:

College or university	34	percent
Industry or private business	19	percent
Medical school	13	percent
Hospital or clinic	10	percent
Government	9	percent
Private practice	7	percent
Independent consultant	6	percent
Nonprofit organization	5	percent
Other	2	percent

On the average, academy members belong to 4.5 science organizations in addition to the academy. The mean is highest in the U.S., at 4.8, and lowest in Eastern Europe, at 3.3. Overall, only 6 percent of the Academy members do not belong to any other science organization, with the range from only 3 percent in U.S. to 16 percent in Eastern Europe.

4 A Detailed Presentation of NYAS Publications

The NYAS regular publications are the Annals, the Sciences Magazine, and Focus—the member newsletter.

The proceedings of the Academy's conferences and selected other meetings are published in the Annals series. Each of the approximately 30 volumes published annually provides a sharply focused picture of current knowledge and advanced work. Out of the thousands of scientific publications worldwide, the Annals are ranked in the top two percent of those publications most cited by scientists.

Each book of the Annals is typically priced at $100 or more (some of them at $190), but members may *select one volume each year at no charge, plus up to five additional volumes as only $15 each.* The Annals make invaluable additions to a personal library at savings that total many times the low annual dues ($105).

The Sciences is a unique bimonthly magazine of all the sciences which has twice won the National Magazine Award for General Excellence. Its highly readable articles have covered such fascinating topics as *The Mother Tongue* from which all languages derive, *No Simple Slumber,* exploring the enigma of sleep, *Terminal Viruses,* that attack computers, *How Many People Can the Earth Support?*, and so much more.

In the November/December issue (1995, pp. 36–39), the paper *Ghost in the Machine* is attractive. The cast of characters on the Internet now includes robotic impersonators as well as the multiple personalities of its human users. Like dreams and beasts, the computer stands on the margins of human life. It is a mind that is not yet a mind. It is an object, ultimately mechanism, but it acts, interacts and seems, in a certain sense, to know. After all, people also act, interact and seem to know, yet ultimately they are made of matter and programmed DNA. We think we can think. But can *it* think? Could it ever be said to be alive?

Each issue is beautifully illustrated with fine art that serves as an enlightening companion to the text. Individuals and libraries worldwide subscribe to *The Sciences,* but for the NYAS member the subscription is included in the membership.

Focus is the Academy's bimonthly member newsletter. Focus keeps members informed about upcoming Conferences and Annals, reports on Academy activities and programs, news about fellow Academy members and about stimulating events open to members. *Focus* helps to link a member to the Academy and to more than 47000 other members around the world—in 160 countries.

5 NYAS E-Mail and Internet Services

Currently, the ordinary e-mail to and from Academy is possible. Some of the topics are the following:

Conferences	`conference@nyas.org`
Corporate Membership	`corpmem@nyas.org`

Darwin Associates	darwin@nyas.org
Development	development@nyas.org
Education	education@nyas.org
Focus	focus@nyas.org
Human Rights	humanrights@nyas.org
Junior Academy	jracademy@nyas.org
Lyceum Club	lyceum@nyas.org
Membership	membership@nyas.org
Policy Programs	policy@nyas.org
Publications	publications@nyas.org
Public Relations	pubrelations@nyas.org
Sections	sections@nyas.org

The Academy has announced its presence on the Internet. Data are taken from [4].

6 The NYAS 1996 Membership Directory

The NYAS announced the publication of New York Academy of Sciences 1966 Membership Directory. The 1996 edition will completely update the 1994 edition [2]. It will be an important library-quality reference volume for over 45,000 Academy members and will enable members to share information and ideas with their colleagues all over the world.

A complete member listings will include name, academic information, areas of speciality, job title, employer name, address and phone, fax and electronic mail numbers plus residence address and phone number. In addition, NYAS membership year and special Academy honors will also be included.

The directory will include three customized cross-reference sections to help someone to locate fellow members with ease:

- The alphabetical section—members listed with a complete biographical profile as described above.
- The Geographical Section—members listed alphabetically under country, city and state.
- The Discipline/Specialty Section—members listed alphabetically under primary specialty categories.

In addition, the Directory will include helpful information about the Academy.

7 Possibilities Offered to Scientific Events and Communities

The NYAS publishes proceedings of conferences held under sponsorship other than its own in the *Annals.* If someone is organizing a multidisciplinary conference on a topic of current or impending scientific interest and wish to have the proceedings considered for publication in the *Annals,* he/she can send the proposal to the Executive Editor. The NYAS seeks conferences broad enough so that one topic may be examined under the lens of several disciplines, capturing a picture of a field at a critical point in its development. The *Annals* are offered to more than 47,000 members and distributed to 750 libraries and academic institutions worldwide. For further information contact *Bill Boland,* Executive Editor, the *Annals,* New York Academy of Sciences, 2 East 63rd Street, New York, NY 10021.

8 A Middle-European View of NYAS Importance

Middle Europe has its Western (capitalist) and Eastern (postcommunist) Part. Especially for the last, it is important to catch the train of scientific and technological development. Why can NYAS play in this respect a significant role particularly in the individual development of scientists and engineers?

As mentioned before, the national academies in the Eastern Europe are elitist and financially privileged communities for the members and they lack national initiative and action for the development of science and technology (S&T), irrespective whether they have an international system of scientists' communication, annals and journals for S&T advancement, education of young and future scientists, a system for protecting the human rights of scientists, academies' international conferences embracing the most relevant fields of development, or not, and, last but not least, the foreign members of those academies do not have the same rights and privileges as the national members. In these academies, virtually, the nationalistic and privileged status of members is in the foreground.

NYAS offers an opposite alternative to national academies. As the reader can see from this report, it is an internationally wide open Academy, where the initiative of each of its members is welcome and appreciated. It is led by distinguished internationally recognized personalities and boards and accompanied by international conferences and scientific publications. NYAS offers a service for human rights, youth education program, and world-wide communication among scientists.

9 Local Clubs of NYAS

Currently, in Slovenia, there are 34 members of NYAS (November, 1995 [2]). In November, 1995, the initiative group for founding a Club was established. The program of the Club is expected to be regularly international, with foreign members and impact.

Members could meet once a month in a domestic and social environment. At the meetings, after a common dinner, the following topics could be discussed:

- Presentation of a member's professional work, his/her scientific (research, engineering) activity and publications, special individual problems.
- Scientific and company research activities in the Club country: national science politics, academies of sciences, concourses, projects, scientists, researchers.
- International scientific activity: meetings with guest researchers (professors), academics, directors from U.S.A. and other parts of the globe.
- Programs and activities for scientific education in basic and middle schools.
- Publications of the Club.
- The possibilities for organizing international conferences, meetings, institutions (American university, foundation) in the Club country.

10 The Active NYAS Members among the Editors, Referees, and Readers of Informatica

Because of the fast growing of the Academy's membership in the last year, it is not possible to get the data about the active members among the editors, referees, and editors of Informatica. The new Academy Membership Directory will be out of print at the end of May, 1996, and will include only membership data up to December, 1995. Although data should not become public in general, according to the agreement of members, an innerly identified group of the active Academy's members could agree on some common initiative concerning the Academy possibilities on one side, and the benefits of Informatica, its prestige, its citing by authors in other journals, and its distribution over the globe, on the other side.

So far, three authors of Informatica decided to express their NYAS membership publicly. The Editor-in-Chief hopes and recommends that other authors follow them. On the other hand, the Editor will appreciate if editors, referees, and readers of *Informatica* can transmit information to his e-mail box (`anton.p.zeleznikar@ijs.si`).

References

[1] Invitation of the Board of Governors of the New York Academy of Sciences to Membership in the Academy, NYAS, New York 1995.

[2] Directory of Members 1994, NYAS, New York 1994.

[3] The Sciences **35** (1995) Number 5 (September/October, p. 4.

[4] Focus August/September 1995, p. 5.

[5] Focus October/November 1995, p. 1–2.

[6] Catalog of the NYAS Annals August/September 1995, p. 5.

Compiled by *J. Šlechta* and *A.P. Železnikar*

Marquis Who's Who in Science and Engineering 1996–1997

Introduction

Who's Who in America was founded by A.N. Marquis in 1899.

Academics, scientists, engineers, and researchers in every field are advancing the knowledge each day. As the pace and scope of these advancements continue to grow, communication between scientists of all specializations becomes increasingly necessary. *Who's Who in Science and Engineering* is designed to facilitate such communication. It provides scientists and engineers, as well as others, the ability to identify researchers and achievers throughout the scientific community.

The Contents of Who's Who

Biographies of 24,000 such individuals will appear in the 3rd Edition of *Who's Who in Science and Engineering.* While the majority of listees come from the United States, many come from over 115 other nations.

The published biography becomes part of the permanent library collections of many of the world's most distinguished corporations, organizations, and institutions. Among those that have requested copies of the 3rd Edition immediately upon publication are: American Association for the Advancement of Science; Arec Associates, Inc.; Centers for Disease Control; Compuware Corporation; MacArthur Foundation; Lawrence Livermore Laboratory; Massachusetts Institute of Technology (M.I.T.); U.S. Department of Commerce, etc.

Many "vanity publishers" and "membership organizations" include biographical information in their publications regardless of reference-worthiness. In this respect, Marquis Who's Who is not to confuse with other publishers who charge a fee for publishing a biography, or who require to purchase a book.

Each edition of *Who's Who in Science and Engineering* provides the most current and authoritative biographical information available on prominent individuals in the fields of science and engineering. The Geographic, Professional, and Awards indexes are additional features within this publication. The Geographic and Professional indexes group Biographees by country, state, city, as well as by field and specialty. Containing 16 distinct fields and over 100 specializations, these indexes provide quick and easy reference to all the biographies in the book. The new and expanded Awards Index lists over 300 award-granting agencies and over 500 awards bestowed upon approximately 1,800 scientists and engineers. This unique index enhances the reference value of Marquis *Who's Who in Science and Engineering,* and attests to Marquis' commitment to the principle of reference value within all Marquis publications.

Board of Advisors

The 3rd Edition was compiled with the assistance of the following distinguished individuals:

- *Hojjat Adeli,* Professor of Civil Engineering, The Ohio State University;
- *William C. Anderson,* Executive Director, American Academy of Environmental Engineers;
- *Robert F. Barnes,* Executive Vice President, American Society of Agronomy;
- *Bruno A. Boley,* Professor of Civil Engineering, Columbia University;
- *William L. Duax,* Executive Vice President of Research, Medical Foundation of Buffalo, Inc.;
- *Herman Feshbach,* Professor Emeritus, Massachusetts Institute of Technology;
- *Walter Freiberger,* Chairman; University Committee on Statistical Science, Brown University;

- *Y.C. Fung,* Professor of Bioengineering and Applied Mechanics, University of California, San Diego;
- *Pierre Galletti,* Professor of Medical Science, Brown University;
- *Roland D. Glen,* Former Director, Association of Consulting Chemists and Chemical Engineers, Inc.;
- *Bryan Gregor,* Former Secretary, The Geochemical Society;
- *Michael Guillen,* Science Editor, ABC-TV, Instructor, Core Curriculum Program, Harvard University;
- *Leonard C. Klein,* Secretary, American Microchemical Society;
- *Robert G. McKinnell,* President, International Society of Differentiation, Inc.;
- *Frederick W. Oehme,* Professor of Toxicology, Kansas State University;
- *Jeremiah P. Ostriker,* Director, Princeton University Observatory;
- *Linda I. Resnik,* Executive Director, White House Conference on Library and Information Science; and
- *Charles Wert,* Former Head, Department of Metallurgy and Mining Engineering, University of Illinois.

Biographees

According to the reference value of the Biographee's outstanding achievements, Marquis Who's Who selects the biographical profile for inclusion in the forthcoming Edition. At this occasion, a Certificate of Recognition is commissioned. This certificate is *absolutely free,* the way Marquis is thanking for biographee's participation. Biographees may reserve their copy of *Who's Who in Science and Engineering* at a special savings ($60.00) available only to them. This pre-publication reservation is offered solely as a courtesy, without obligation.

Publication Date and Address

Schedulded publication date of the 3rd Edition is April, 1996.

The production order of the new 3rd Edition of Marquis *Who's Who in Science and Engineering* will be carefully estimated to fill the requirements of the institutions that rely on Marquis for accurate biographical reference materials—schools, libraries, and corporations, to name a few. Once the print run is established, the edition will be closed and *will not be reprinted.*

The address is:

Marquis Who's Who
121 Chanlon Road
New Providence, NJ 07974

A Request and Advice to the Editors and Authors of *Informatica*

As the Editor-in-Chief of *Informatica* is informed, the biographical profiles of some of the Editors of *Informatica* have been selected for inclusion in the forthcoming 3rd Edition of Marquis *Who's Who in Science and Engineering* (April 1996) and 14th Edition of Marquis *Who's Who in the World* (December 1996). The Editor-in-Chief of *Informatica* expresses his deep respect with congratulations for such an achievement.

Further, the Editor invites the Members of the Editorial Board and Authors of *Informatica* to inform him (submitting a note via the e-mail) upon such nominations, selections, international academy and society memberships, professional and other awards, and other important achievements, for the sake of reference which is given to the local authorities in the competition for the *Informatica* financial support, and for the sake of the international recognition of individuals and the journal *Informatica.*

At the end of the year, Informatica will publish together with the Contents of the Volume the achievements of *Informatica*'s Editors and Authors.

Compiled by *A.P. Železnikar*

Press Release

European Design & Test Conference and Exhibition 1996

The World's Favourite CAE Conference - With a Major International Exhibition

By the number of submissions, ED&TC 96 is the largest event of this type in the world! ED&TC has received a total of 396 regular submissions, from which an international review panel has selected 86 papers for inclusion in the IEEE Proceedings (27% acceptance rate) and 48 User's Forum papers.

The number of submissions has allowed to extend the traditional 4- track structure with a 5th track during a major part of the Conference resulting in 2 tracks on CAD, 1 on Design, 1 on Test and 1 for the User Forum.

This year, there have been two major new trends:

1. The number of papers on mixed/analog digital systems has gone up significantly. Such analog signals within a system may represent, for example, antenna signals of portable phones or sensor and actuator signal in automobile electronics. Thus, design automation for complete systems has to take analog signals into account. It is interesting to know that analog and mixed-signal designs have found special attention in Europe and at the ED&TC Conference.

2. This year, for the very first time, ED&TC covers the design and design tools for microsystems. CAD tools for the design of microelectromechanical systems are emerging and papers will discuss how existing microelectronics CAD tools are to be extended and complemented to allow the design of these miniaturised inter- disciplinary components. The research on testing these components will also be addressed, as well as the boost to the market by foundries. Outstanding papers include contributions on CAD by TIMA, Grenoble in cooperation with T.U. Budapest and T.H. Darmstadt, and on fault modeling by Fh-G, Dresden.

The full programme is available at: `http://ls12-www.informatik.uni-dortmund.de/edtc/96/edtc.html`

In addition, the Conference will include a number of innovative contributions to just recently introduced areas:

1. Hardware/software codesign is now moving from a rather broad overall view of the issues towards a more detailed view of particular solutions. For example, solutions on where software code should be stored in an embedded system in order to optimize the performance will be presented by Tomiyama and Yasuura from Kyushu University, Japan.

2. Behavioural synthesis techniques (to a major extent originating from Europe) are now being introduced into industry. ED&TC will include a highly-ranked paper by R. Bergamashi et al. (IBM) on how to verify the results of these techniques.

3. Low-power design continues to reduce the power consumption of microelectronics. New approaches for replacing standard-sized transistors by transistors optimized for low power consumption will be introduced by Coudert (Synopsys Inc, USA).

4. IDDQ-testing for CMOS devices is becoming a more mature topic. The Conference will include a highly-rated overview by T. Williams et al. (IBM) of what to expect in this area in the next ten years.

5. With the trend towards more and more complex systems, built-in self test (BIST) techniques are increasingly used. In this area, papers on gate-delay testing and MCM testing have received high scores.

The Technical Programme will be introduced by 3 keynote speakers: M. Jacobs from SIEMENS, G. House from MENTOR GRAPHICS and W. Maly from Carnegie-Mellon University. They will discuss a worldwide perspective on the 3 main themes of the Conference. A European dimension will be brought by G. Metakides, Director of IT Programme at the European Commission, by chairing this keynote speakers session.

ED&TC will feature this year hot topic sessions on system aspects. Each session will include an invited talk and a round table. The topics selected for ED&TC 96 are: microprocessor design, broadband telecom and IDDQ testing in practice.

Once again, ED&TC will feature a User Forum in addition to the four tracks of "hot topics", CAD, Design and Test. The User Forum will bring the latest in industrial and academic experience in circuit design which, together with the Poster session, will allow for intensive discussion with authors. In addition, IEEE Design and Test is co- organising the Panel session on "Mixed-Signal Design & Test: Reality or Illusion-Do We Have the Right Tools?". There are two other panel sessions on the subjects "Submicron Design Tools: What Problems Can We Expect, and Who Will Supply the Tools?" and "DFT is not the Issue - Bad Design and Bad CAD is!". Tutorials will also be available on hot issues in design and test of ASICs and Systems.

Alongside the Conference is a full-scale, international exhibition, featuring the world's leading suppliers of CAE, ASICs, FPGAs PLDs, and Test program development software. A visit to the exhibition is a chance to meet technical specialists from the companies driving design technology. Inside the exhibition area will be a theatre running technical presentations from exhibiting companies, which will be free of charge to all visitors to the exhibition.

A last, but not least, touch is a gala dinner and a visit to the new Grand Louvre including explanations by the Museum guides.

ED&TC 96 is sponsored by EDAA, the IEEE Computer Society DATC and TTTC, and ACM/SIGDA, in cooperation with other European Societies.
More information:

C. Lopez-Barrio
General Chair ED&TC 96
Tel.: +34 1 337 4310
Fax: +34 1 337 4212

P. Marwedel
Programme Chair ED&TC 96
Tel.: +49 231 755 6111
Fax: +49 231 755 6116

L. Eggermont
Vice General Chair ED&TC 96
Tel.: +31 40 784 961
Fax: +31 40 786 422

B. Courtois
EDAA Chair
Tel.: +33 76 57 46 15
Fax: +33 76 47 38 14

J. Kenyon
Exhibition Chair
Tel.: +44 171 404 0564
Fax: +44 171 831 2057

Machine Learning List

The Machine Learning List is moderated. Contributions should be relevant to the scientific study of machine learning. Mail contributions to `ml@ics.uci.edu`. Mail requests to be added or deleted to `ml-request@ics.uci.edu`. Back issues may be FTP'd from `ics.uci.edu` in `pub/ml-list/V<X>/<N>` or N.Z where X and N are the volume and number of the issue; ID: anonymous PASSWORD: <your mail address> URL- `http://www.ics.uci.edu/AI/-ML/Machine-Learning.html`

Errata

Due to an error, the name of James Geller slipped from the list of referees in Informatica Vol. 19 No. 4 at page 424 in Acknowledgements. We sincerely apologise.

ICCCN'96 CALL FOR PAPERS FIFTH INTERNATIONAL CONFERENCE ON COMPUTER COMMUNICATIONS AND NETWORKS

October 16 – 19, 1996
Double Tree Hotel, Rockville, Washington D. C., USA

Sponsored by DataTech and NASA. In cooperation with NSF, NIST, USL, USC Communication, IEEE Computer Society, ACM, and IEEE Communication Society.

The objective of this conference is to provide an effective forum for original and fundamental advances in Computer Communications and Networks and to foster communication among researchers and practitioners working in a wide variety of scientific areas with a common interest in improving Computer Communications and Networks.

Scope

The primary focus of the conference is on new and original research results in the areas of design, implementation and applications of Computer Communications and Networks. We solicit the submission of papers that address novel, challenging and innovative results. Therefore the topics that will be addressed include, but are not limited to:

ATM Networking, Internet Services/Applications, Distributed Multimedia Applications, Real Time Communications, Quality of Services (QoS) Issues, LAN/WAN Internetworking, Interoperability, Personal Communication Services, Network Management, Wireless Networks, Intelligent Networks, Multicast Protocols, Network Security, Optical Networks, Reliable Networks, High Speed Network OAM/Protocols, Video-on-Demand, Traffic Management, Multimedia Human-Machine Interface, Performance Modeling/Analysis, Communication Software, Protocol Verification/Validation/Testing

Submission

Authors are invited to submit complete and original papers. Papers that may be submitted for consideration include those that have not previously been published in another forum, or are not currently being published or reviewed by another journal or conference. All submitted papers will be refereed for quality, correctness, originality and relevance. The program committee reserves the right to accept a submission as long, short or poster presentation. Of particular interest are papers which address experiences with concrete Computer Communications and applications. All accepted papers will be published in the conference proceedings. Authors will be interested to know that special issues of journals containing outstanding papers from the conference are being planned.

Manuscripts should include an abstract and be limited to 5000 words. Submissions should include the title, author(s), author's affiliation, e-mail address, fax number and postal address. In case of multiple authors, an indication of which author is responsible for correspondence and preparing the camera ready paper for the proceedings should also be included. Six copies of the manuscript should be submitted by Friday, March 22, 1996 to Program Co-Chair:
Dr. David Lee
AT&T Bell Laboratories
600 Mountain Avenue, RM 2C-423
Murray Hill, New Jersey 07974, USA
`lee@research.att.com`
Tel: (908) 582-5872; Fax:(908) 582-5857

For more information about the conference (as opposed to paper submissions) please send e-mail to `ic3n@cacs.usl.edu`.

Important Dates

Paper submission deadline: March 22, 1996
Notification of acceptance: May 30, 1996
Camera ready papers due: July 1, 1996

Workshops and Tutorials

Proposals are solicited for tutorials and workshops. Please send your proposal by March 22, 1996 to Professor M. Singhal, the Tutorial Chair, Department of Computer and Information Science, The Ohio State University, 2015 Neil Ave, Columbus, OH 43210, USA, singhal@cis.ohio-state.edu, Tel: (614) 292-5839, Fax: (614) 292-2911. To Professor K. Efe, the Workshop Chair, The Center for Advanced Computer Studies, The University of SW Louisiana, Lafayette, LA 70504, efe@cacs.usl.edu, Tel: (702) 482-6876, Fax: (702) 482-5791.

Steering Committee

I. Chlamtac-U. Mass, V. Li (chair) - USC, K. Makki - UNLV, J.S. Meditch - U. Wash, E. K. Park - USNA, R. Pickholtz - GWU, N. Pissinou - CACS/ USL, T. Suda - UC Irvine, J.W. Wong - U. Waterloo

General Co-Chairs

K. Makki - UNLV and N. Pissinou - CACS/USL

Program Committee

S. Aggarwal-SUNY/Binghamton, C. Alaettinoglu-USC/ISI, M. H. Ammar-GA/Tech, H. Arabnia-U. Georgia, M. Bayoumi-USL, K. Bhat-AT&T, A. Bush-NSF, K. Calvert-GA/Tech, S. T. Chanson-UBC, D. Cohen-Bellcore, T. S. Dillon-LaTrobe U., T. Y. Feng-NSF, O. Frieder (Co-Chair)-GMU, E. Geraniotis-DEEI, F. Golshani-ASU, M. G. Gouda-UT/Austin, Z. Haas-Cornell, M. Halem-NASA, J. Harms-U. Alberta, E. Hahne-AT&T, C.-S. Kang-Hannam U., I. Khan-Qualcomm, J. B. Kim-CSUN, D. Lee (Co-Chair)-AT&T, Y.-H. Lee-U. Florida, B. J. Leon-CACS/USL, F. J. Lin-Bellcore, M. T. Liu-OSU, W. Liu-BellSouth, W. M. Moh-SJSU, R. Miller-UMD, M. S. Obaidat-CUNY, S. Olariu-ODU, M. T. Ozsu- U. Alberta, W. Peng-SWTSU, S. Sahni-U. Florida, H. Saito-NTT, M.-C. Shan-HP, U. Shankar-UMD, H. Sharif-UNL, A. Silberschatz-AT&T, V. Srinivasan-IBM, D. Su-NIST, Y. Zhang-Hughes

Conference Coordinators

C. Fayet-INT/France, A. Gaivoronski-ITALTEL/Italy, C.-S. Kang-Hannam U./Korea, U. Krieger-DBP/Germany, H. Saito-NTT/Japan

Publicity Co-Chairs

S. Bhattacharya-Honeywell, Mallikarjun Tatipamula-BNR

Local Arrangements

D. Su-NIST

Treasurer

E. K. Park-USNA

AIMSA'96 7th International Conference on Artificial Intelligence: Methodology, Systems, Applications Sozopol, Bulgaria, September 18 - 20, 1996 Call for Papers and Participation

General information

The AIMSA conference series has provided a biennial forum for presentation of AI research and development since 1984. The conferences, which are held in Bulgaria, cover the full range of AI, and are noted for their well established tradition of international scientific exchange between Central and Eastern Europe and the rest of the world.

AIMSA'96 is sponsored by ECCAI, European Coordinating Committee for Artificial Intelligence.

Topics of interest (including, but not limited to): Automated Reasoning, Cognitive Modeling, Genetic Algorithms, AI Languages and Architectures, AI Applications, Intelligent Decision Support Systems, Knowledge Acquisition, Knowledge-Based Systems, Knowledge Representation, Machine Learning, Natural Language Processing, Neural Networks, Robotics, Vision.

Submission of papers

Papers should be written in English and be printed on A4 paper (double-sided printing is encouraged), using high quality printers. Format should be single -spaced, 12 point type with no more than 50 lines per page and maximum 5000 words (10 pages). The first page of each copy of a submitted paper should contain title of the paper, names and addresses of all authors, an abstract (100-150 words) and a list of keywords. Electronic and fax submissions will not be considered.

The Proceedings of AIMSA-96 will be published by IOS Press and will be available at the conference.

Four copies of submitted papers must be received before 1 April 1996 by the Program Chair:

Allan Ramsay - AIMSA'96 Program Chair, Department of Language and Linguistics, UMIST, PO BOX 88, Manchester M60 1QD, United Kingdom, Email: `allan@ccl.umist.ac.uk`, Phone: (44) (0) 161 200 3108 (direct), (44) (0) 161 2036 3311 (switch), Fax: (44) (0) 161 200 3099

Selection of papers

Each paper will be reviewed by at least two members of the program committee. Selection criteria include accuracy and originality of ideas, clarity and significance of results, quality of presentation.

At least one of the authors of an accepted paper is expected to present the paper at the conference.

The program committee will select one paper of exceptional quality among the accepted papers. The authors will receive the best paper award during the conference.

Program committee

Benedict du Boulay (UK), Pedro Meseguer (Spain), Peter Braspenning (The Netherlands), Ewa Orlowska (Poland), Ivan Bratko (Slovenia), Ivan Plander (Slovakia), Peter Brusilovsky (Russia), Allan Ramsay (UK, programme chair), Christo Dichev (Bulgaria), Ronan Reilly (Ireland), Danail Dochev (Bulgaria), Vassil Sgurev (Bulgaria), Altay Guvenir (Turkey), Mark Stickel (USA), Steffen Hoelldobler (Germany), Dan Tufis (Romania), Philippe Jorrand (France), Dimitar Vakarelov (Bulgaria), Vladimir Marik (Czech Republic), Tibor Vamos (Hungary), Alberto Martelli (Italy), David Young (UK)

Important dates

Submission deadline: 1 April, 1996
Notification of acceptance: 20 May, 1996
Deadline for final papers: 10 June, 1996
Conference: September 18 - 20, 1996

Organization

- Bulgarian Artificial Intelligence Association
- Institute for Information Technologies
- Union of Bulgarian Mathematicians

with the support of the Bulgarian Society for Cognitive Science and of the Bulgarian Association for Pattern Recognition.

Local arrangements

Christo Dichev - AIMSA'96, Institute for Information Technologies, Bl. 29A, Acad. G. Bonchev St., 1113 SOFIA - BULGARIA, Email: `cdichev@iinf.bg`, Phone: (359) (2) 707 586, Fax: (359) (2) 720 497
`http://www.aubg.bg`
`/faculty/cs/nikolaev/aimsa96.html`

Language

The official language of the conference is English.

Joint event

Second Joint Conference on Knowledge-Based Software Engineering (JCKBSE'96), Sozopol, Bulgaria, 21-22 September, 1996.

The aim of JCKBSE'96 is to provide a forum for scientific interchange among scientists in Knowledge-Based Software Engineering mainly from Japan, the CIS countries and Bulgaria on a broad spectrum of academic research, implementation technologies and application systems.

Conference venue

Sozopol is a small ancient and picturesque town on the Black sea cost, famous with its architecture and art exhibitions. It is a favorite place for rest and entertainment to most Bulgarians and many foreigners. AIMSA-96 will take place in a holiday home near the beach and in a walking distance from downtown Sozopol.

Travel information

Sozopol is located 30 km south to Bourgas. Bourgas has an international and domestic airport with frequent flights from and to Sofia (the capital city of Bulgaria). There are also direct flights connecting the major European cities with Sofia and Bourgas. Transportation between the conference location and the Bourgas airport will be provided.

THE MINISTRY OF SCIENCE AND TECHNOLOGY OF THE REPUBLIC OF SLOVENIA

Address: Slovenska 50, 61000 Ljubljana, Tel.: +386 61 1311 107, Fax: +386 61 1324 140.
WWW:http://www.mzt.si
Minister: Prof. Rado Bohinc, Ph.D.
State Secretary for Int. Coop.: Rado Genorio, Ph.D.
State Secretary for Sci. and Tech.: Ciril Baškovič
Secretary General: Franc Hudej, Ph.D.

The Ministry also includes:
The Standards and Metrology Institute of the Republic of Slovenia
Address: Kotnikova 6, 61000 Ljubljana, Tel.: +386 61 1312 322, Fax: +386 61 314 882., and
The Industrial Property Protection Office of the Republic of Slovenia
Address: Kotnikova 6, 61000 Ljubljana, Tel.: +386 61 1312 322, Fax: +386 61 318 983.

Scientific Research and Development Potential. The statistical data for 1993 showed that there were 180 research and development institutions in Slovenia. Altogether, they employed 10,400 people, of whom 4,900 were researchers and 3,900 expert or technical staff.

In the past ten years, the number of researchers has almost doubled: the number of Ph.D. graduates increased from 1,100 to 1,565, while the number of M.Sc.'s rose from 650 to 1,029. The "Young Researchers" (i.e. postgraduate students) program has greatly helped towards revitalizing research. The average age of researchers has been brought down to 40, with one-fifth of them being younger than 29.

The table below shows the distribution of researchers according to educational level and sectors (in 1993):

Sector	Ph.D.	M.Sc.
Business enterprises	51	196
Government	482	395
Private non-profit organizations	10	12
Higher education organizations	1022	426
Total	1,565	1,029

Financing Research and Development. Statistical estimates indicate that US$ 185 million (1,4% of GDP) was spent on research and development in Slovenia in 1993. More than half of this comes from public expenditure, mainly the state budget. In the last three years, R&D expenditure by business organizations has stagnated, a result of the current economic transition. This transition has led to the financial decline and increased insolvency of firms and companies. These cannot be replaced by the growing number of mainly small businesses. The shortfall was addressed by increased public-sector spending: its share of GDP nearly doubled from the mid-seventies to 0,86% in 1993.

Income of R&D organizations spent on R&D activities in 1993 (in million US$):

Sector	Total	Basic res.	App. res.	Exp. dev.
Business ent.	83,9	4,7	32,6	46,6
Government	58,4	16,1	21,5	20,8
Private non-p.	1,3	0,2	0,6	0,5
Higher edu.	40,9	24,2	8,7	8
Total	184,5	45,2	63,4	75,9

The policy of the Slovene Government is to increase the percentage intended for R&D in its budget. The Science and Technology Council of the Republic of Slovenia is preparing the draft of a national research program (NRP). The government will harmonize the NRP with its general development policy, and submit it first to the parliamentary Committee for Science, Technology and Development and after that to the parliament. The parliament approves the NRP each year, thus setting the basis for deciding the level of public support for R&D.

The Ministry of Science and Technology is mainly a government institution responsible for controlling expenditure of the R&D budget, in compliance with the NRP and the criteria provided by the Law on Research Activities. The Ministry finances research or co- finances development projects through public bidding, partially finances infrastructure research institutions (national institutes), while it directly finances management and top-level science.

The focal points of R&D policy in Slovenia are:

- maintaining the high level and quality of research activities,
- stimulating collaboration between research and industrial institutions,
- (co)financing and tax assistance for companies engaged in technical development and other applied research projects,
- research training and professional development of leading experts,
- close involvement in international research and development projects,
- establishing and operating facilities for the transfer of technology and experience.

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 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 700 staff, has 500 researchers, about 250 of whom are postgraduates, over 200 of whom have doctorates (Ph.D.), and around 150 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 Slovenia (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.

In the last year on the site of 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.

At the present time, part of the Institute is being 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 is being developed at a particularly historical moment, characterized by the process of state reorganisation, privatisation and private initiative. The national Technology Park will take the form of a shareholding company and will host an independent venture-capital institution.

The promoters and operational entities of the project are the Republic of Slovenia, Ministry of 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 Economic Relations and Development, the National Chamber of Economy and the City of Ljubljana.

Jožef Stefan Institute
Jamova 39, 61000 Ljubljana, Slovenia
Tel.:+386 61 1773 900, Fax.:+386 61 219 385
Tlx.:31 296 JOSTIN SI
WWW: http://www.ijs.si
E-mail: matjaz.gams@ijs.si
Contact person for the Park: Iztok Lesjak, M.Sc.
Public relations: Natalija Polenec

INFORMATICA
AN INTERNATIONAL JOURNAL OF COMPUTING AND INFORMATICS
INVITATION, COOPERATION

Submissions and Refereeing

Please submit three copies of the manuscript with good copies of the figures and photographs to one of the editors from the Editorial Board or to the Contact Person. At least two referees outside the author's country will examine it, and they are invited to make as many remarks as possible directly on the manuscript, from typing errors to global philosophical disagreements. The chosen editor will send the author copies with remarks. If the paper is accepted, the editor will also send copies to the Contact Person. The Executive Board will inform the author that the paper has been accepted, in which case it will be published within one year of receipt of e-mails with the text in Informatica LaTeX format and figures in .eps format. The original figures can also be sent on separate sheets. Style and examples of papers can be obtained by e-mail from the Contact Person or from FTP or WWW (see the last page of Informatica).

Opinions, news, calls for conferences, calls for papers, etc. should be sent directly to the Contact Person.

QUESTIONNAIRE

☐ Send Informatica free of charge

☐ Yes, we subscribe

Please, complete the order form and send it to Dr. Rudi Murn, Informatica, Institut Jožef Stefan, Jamova 39, 61111 Ljubljana, Slovenia.

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 two 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 the European computer science and informatics 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: ....

Referees:

Witold Abramowicz, David Abramson, Kenneth Aizawa, Alan Aliu, John Anderson, Catriel Beeri, Fevzi Belli, Istvan Berkeley, Azer Bestavros, Balaji Bharadwaj, Jacek Blazewicz, Laszlo Boeszoermenyi, Ivan Bratko, Jerzy Brzezinski, Marian Bubak, Leslie Burkholder, Frada Burstein, Wojciech Buszkowski, Ryszard Choras, Jason Ceddia, Wojciech Chybowski, Andrzej Ciepielewski, Vic Ciesielski, David Cliff, Travis Craig, Tadeusz Czachorski, Sait Dogru, Georg Dorfner, Maciej Drozdowski, Marek Druzdzel, Hesham El-Rewini, Pierre Flener, Terrence Forgarty, Hugo de Garis, Eugeniusz Gatnar, James Geller, Janusz Gorski, Georg Gottlob, David Green, Herbert Groiss, Inman Harvey, Elke Hochmueller, Rod Howell, Tomáš Hruška, Ryszard Jakubowski, Piotr Jedrzejowicz, Eric Johnson, Li-Shan Kang, Roland Kaschek, Jan Kniat, Stavros Kokkotos, Kevin Korb, Gilad Koren, Henryk Krawczyk, Ben Kroese, Zbyszko Krolikowski, Benjamin Kuipers, Matjaž Kukar, Aarre Laakso, Phil Laplante, Bud Lawson, Ulrike Leopold-Wildburger, Joseph Y-T. Leung, Raymond Lister, Doug Locke, Jason Lowder, Andrzej Marciniak, Witold Marciszewski, Vladimir Marik, Jacek Martinek, Tomasz Maruszewski, Florian Matthes, Timothy Menzies, Dieter Merkl, Zbigniew Michalewicz, Roland Mittermeir, Madhav Moganti, Tadeusz Morzy, Daniel Mossé, John Mueller, Hari Narayanan, Jerzy Nogieć, Stefano Nolfi, Tadeusz Pankowski, Warren Persons, Niki Pissinou, Gustav Pomberger, James Pomykalski, Gary Preckshot, Ke Qiu, Michael Quinn, Gerald Quirchmayer, Luc de Raedt, Ewaryst Rafajlowicz, Wolf Rauch, Peter Rechenberg, Felix Redmill, David Robertson, Marko Robnik, A.S.M. Sajeev, Bo Sanden, Iztok Savnik, Wolfgang Schreiner, Guenter Schmidt, Heinz Schmidt, William Spears, Przemysław Stpiczyński, Maciej Stroinski, Tomasz Szmuc, Zahir Tari, Jurij Tasič, Piotr Teczynski, Ken Tindell, A Min Tjoa, Wieslaw Traczyk, Marek Tudruj, Andrzej Urbanski, Kanonkluk Vanapipat, Zyunt Vetulani, Olivier de Vel, John Weckert, Gerhard Widmer, Stefan Wrobel, Janusz Zalewski, Yanchun Zhang a Knowledge Archives (A Plan 1992; see also Železnikar 1993) and solving the problem of Automatic Knowledge Acquisition (Wah et al. 1993). The third task is the creation of the Electronic Library and Librarian; an Electronic Librarian should be able to analyze the contents of arbitrary discourses (from text-books, scientific papers, legal sources, etc.) in order to satisfy diverse requests of end users from U.S.A. and many other countries (Wah et al. 1993).

These tasks[1] reflecting the gradual transition to the informational orientation in computer science pose the problem of creating natural-language-processing systems (NLPSs) capable to analyze practically arbitrary texts pertaining to very many application domains. Widely-applicable NLPSs will be highly complicated computer systems. For effective designing complex technical systems of many other kinds, one uses special mathematical theories, e.g. airdynamics in case of airplanes design and hydrodynamics for constructing ships. Analogically, we need to work up a mathematical theory of natural language (NL) use, or mathematical theory of NL-communication, or mathematical linguocybernetics (MLC) for speeding-up the progress in designing NLPSs. Such a theory should be a collection of interrelated mathematical models destined for helping creators of NLPSs to overcome difficulties of logical character encountered in their work (Fomitchov 1983, 1984; Fomichov 1988, 1992, 1993a).

During more than twenty years passing after publishing the known pioneer works of R. Montague on formal semantics of NL, a number of other approaches to the mathematical study of NL-semantics has been suggested. The set of these approaches includes, in particular, Theory of Generalized Quantifiers, Situation Semantics, Discourse Representation Theory (DRT), Dynamic Montague Grammar, Dynamic Predicate Logic, and the works of M.J. Cresswell and G. Chierchia on structured meanings(SMs) of sentences. It is shown in (Fomichov 1993a, 1993b) that these approaches don't provide a sound initial basis for developing MLC. The main reasons for this conclusion are that these approaches don't study the problem of describing conceptual structures of arbitrary real NL-discourses pertaining to science, business, medicine, law, etc. and don't investigate sufficiently deeply the role of knowledge about the world in NL use.

[1]This paper is a private author's work and no part of it may be used, reproduced or translated in any manner whatsoever without written permission of the author except in the case of brief quotations embodied in critical articles.

In particular, it is difficult not to agree with the opinion of Ahrenberg (1992, p. 7) that "in spite of its name, DRT can basically be described as formal semantics for short sentence sequences rather than as a theory of discourse". This opinion seems to be true also with respect to the content of the monograph (Kamp & Reyle 1993).

Like mentioned above approaches, Situation Theory (Aczel et al. 1993; Barwise 1989, 1992, 1993; Barwise & Cooper 1993; Barwise & Etchemendy 1990; Cooper 1991; Devlin 1991), Theory of Situation Schemata (Fenstad, Langholm, Vestre 1992), Categorial Semantics (see van Benthem 1992) study only some separate aspects of NL-semantics and don't raise the problem of investigating SMs of complicated real discourses and NL use as a whole.

On the contrary, Integral Formal Semantics (IFS) represented, in particular, (in Fomichov 1981, 1982; Fomitchov 1983, 1984; Fomichov 1988, 1992, 1993a, 1993b, 1994) from the very beginning was aimed at studying general regularities of NL use. One of the main principles of IFS is as follows: the basis of researches aimed at constructing a mathematical theory of NL use is to be a problem-independent formal model reflecting many peculiarities of semantic (or conceptual) structures of sentences and discourses of arbitrary great length and providing a description of some class of formal languages convenient for building semantic representations (SRs) of NL-texts in a large spectrum of applications and on different levels of representation. Let's call such a model a widely-applicable *Metagrammar of Conceptual Structures*, or a widely-applicable *Conceptual Metagrammar* (CM).

A widely-applicable CM should enable us to build formal semantic analogues of sentences and discourses; hence the expressive power of formal languages determined by the model may be very close to the expressive power of NL (if we take into account the surface semantic structure of NL-texts). Besides, a CM is to be convenient for describing various knowledge about the world (Fo-

EDITORIAL BOARDS, PUBLISHING COUNCIL

Informatica is a journal primarily covering the European computer science and informatics 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 Refereeing Board. Each paper bears the name of the editor who appointed the referees. Each editor can propose new members for the Editorial Board or Board of Referees. Editors and referees inactive for a longer period can be automatically replaced. Changes in the Editorial Board and Board of Referees 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 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.

Volume 20 Number 1 February 1996 ISSN 0350-5596

Informatica

An International Journal of Computing and Informatics

Contents: