Volume 18 Number 2 June 1994 ISSN 0350-5596

Informatica

An International Journal of Computing and Informatics

Profile: Branko Souček
Brain Windows
Dynamic Conceptual Mapping
Informational Being-In
Neural Networks
Machine Learning

The Slovene Society Informatika, Ljubljana, Slovenia

Informatica 18 (1994) Number 2, pp. 107–254

Informatica

An International Journal of Computing and Informatics

Basic info about Informatica and back issues may be FTP'd 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: ftp://ftp.arnes.si/magazines/informatica

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 1994 (Volume 18) is
– DEM 50 (US$ 34) for institutions,
– DEM 25 (US$ 17) for individuals, and
– DEM 10 (US$ 4) for students
plus the mail charge DEM 10 (US$ 4).

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, Department for Computer Science, Jožef Stefan Institute: Tel (+386) 61 1259 199, 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)
Authomatic Control Society of Slovenia (Borut Zupančič)
Slovenian Association of Technical and Natural Sciences (Janez Peklenik)

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

PROFILES

The readers can observe substantial differences existing between the contents concerning concrete profiles of editors of Informatica. In some cases the scientific achievements of the profiled editor are in the foreground, in other cases, the brilliant career seems to play the decisive role. Commonly, both achievement and career give the reason for the decision to put someone into the Informatica's profile collection. However, this does not mean that a critical amount of the scientific achievement and career must not be present.

The work of Professor Branko Souček fulfills both conditions. As a challenge to the Japanese fifth generation computer systems, he launched the term of the six generation computer systems, although at first through his editor function of a dedicated Wiley book series. In this series, he is not only the editor but also the leading author as it can be recognized from the list of Wiley's books which follows.

Dr. Souček is one of the earliest editor of Informatica. Today, his research is in

1. the brain windows evoked potentials and the holographic network for neurological diagnoses;

2. the holographic fuzzy learning for credit scoring; and in

3. IRIS-FEED, Integrated Reasoning and Informing Service—Federated ExEcutive Database.

The reader will get an impression of his research work on the next pages of Informatica, in the paper entitled *Neurological Diagnoses Based on Evoked Brain Windows and on Holographic Learning* which is a result of years of the interdisciplinary research.

By these introductory notes and the short curriculum vitae which follows many other matters of primary and secondary importance cannot be entirely embraced and recognized by those who do not know the style of work and life of Professor Souček. It is a lot more which could be reflected into detail of his innovative research, technological design, teaching experience editorial work and academic career.

Branko Souček

Branko Souček is a professor of electronic computer engineering and holds a B.Sc. and Ph.D. in electrical engineering (University of Zagreb, in 1955 and 1963, respectively). He has divided his time between research and teaching at several institutions as:

- Institute "Rudjer Bošković", Zagreb, Croatia;
- Brookhaven National Laboratory, Upton, N.Y., U.S.A.
- Department of Mathematics and Department of Electrical Engineering, University of Zagreb, Croatia;
- State University of New York, Stony Brook, N.Y., U.S.A.;
- Department of Electrical and Computer Engineering, University of Arizona, Tuscon, Arizona, U.S.A.; and
- IRIS International Center, Bari, Italy.

Professor Souček has been the manager in large information technology projects and lecturer/consultant for multinational corporations, including:

- IBM, Boeblingen, Germany;
- Siemens, Munich, Germany;
- Schering, New Jersey, U.S.A.;
- Lockheed, Palo Alto, Ca., U.S.A.; and
- NASA, Newport News, U.S.A.

Dr. Souček experience ranges from the world's first Associative, million channel, real-time data acquisition system, to the original Brain-window theory. He has published more than 100 scientific/technical papers, and has served as invited and tutorial speaker on many informational conferences.

Dr. Souček has served as lecturer/consultant for

- Integrated Computer Systems, Los Angeles, Ca., U.S.A.;

- Software Research Corporation, St. Louis, U.S.A.; and
- Infotech, London, U.K.

He serves as an expert in United Nations agencies IAEA and UNIDO. His books on mini, micro, neural, and real-time computing have been published in U.S.A., Canada, Europe, Russia, China and Japan in over 100,000 copies.

Souček's Book Series Concerning the Sixth Generation Computer Technology

Presently, Professor Souček is an editor of Series of books in "The Sixth Generation Computer Technology", published by John Wiley, New York, U.S.A. The books published during 1988–1994 are the following:

1. B. Souček and Marina Souček: *Neural and Massively Parallel Computers,* J. Wiley, New York, p. 450, 1988;
2. B. Souček: *Neural and Concurrent Real-Time Systems, The Sixth Generation,* J. Wiley, New York, p. 387, 1989;
3. B. Souček and the IRIS Group: *Neural and Intelligent Systems Integration: Fifth and Sixth Generation Integrated Reasoning Information Systems,* J. Wiley, New York, p. 650, 1991;
4. B. Souček and the IRIS Group: *Fuzzy, Holographic, and Parallel Intelligence,* J. Wiley, New York, p. 350, 1991;
5. B. Souček and the IRIS Group: *Dynamic, Genetic and Chaotic Programming,* J. Wiley, New York, p. 550, 1992;
6. B. Souček and the IRIS Group: *Fast Learning and Invariant Object Recognition: The Sixth Generation Breakthrough,* J. Wiley, New York, p. 270, 1992.
7. T. Hrycej: *Modular Learning in Neural Networks: A Modularized Approach to Neural Network Classification,* J. Wiley;
8. R. Sun: *Integrating Rules and Connectionism for Robust Commonsense Reasoning,* J. Wiley; and
9. V.L. Plantamura, B. Souček, G. Visaggio: *Frontier Decision Support Concepts,* J. Wiley.

Current Activities

Professor Souček offers consulting, teaching, projects and studies in:

- **Help Desk.** Supports the person in complex but routine decision making, customer service and support, maintenance and troubleshooting, software development and use, information retrieval, product support.
- **Media Decision Support.** Supports the information systems and software packages in their work. This includes intelligent data bases, pattern base, deductive hypermedia, association maps, decision support in EDI.
- **Process Decision Support.** Supports real-time process control systems. It offers the features of production surveillance, object and pattern recognition, product evaluation, adaptation and learning, robot support.
- Integration of Reasoning, Informing and Serving, IRIS
- The Business Process Reengeneering, BPR
- Intelligent Business Networks, IBN
- Frontier Decision Support Concepts, Wiley, New York, 1994, ISBN 0-471-592560-0

Address:

Prof. Branko Souček
STAR Service; IRIS
Via Amendola 162/1
70126 Bari, Italy
fax: 3980-5484556
phone: 3980-5484555.

Edited by *A.P. Železnikar*

NEUROLOGICAL DIAGNOSES BASED ON EVOKED BRAIN WINDOWS AND ON HOLOGRAPHIC LEARNING

Branko Souček
STAR SERVICE S.p.A., Via Amandola 162/1, 70126 Bari, Italy

Phone: 3980.5484555
Fax: 3980.5484556

Keywords: Brain-windows, evoked potentials, holographic learning, diagnoses, neurology

Edited by: Rudi Murn
Received: November 20, 1993 **Revised:** April 20, 1994 **Accepted:** May 25, 1994

The evoked potentials have been generated in response to auditory stimuli to a person, and light stimuli to insects, resulting in two datasets, HUMAN and INSECT. In both datasets the responses are composed of several peaks with variable latencies. The brain-window logic is used to explain the evoked responses. Brain-windows are generated through mutual coupling of biological oscillators, and modulated by the memory that stores the past history and the present behavior. Latencies of the peaks provide necessary information to discriminate between normal subject and pathological states resulting from injury, tumor or multiple sclerosis. The holographic neural network classifies the subjects, based on the peak latencies. Combining brain-window theory with the holographic learning opens new possibilities for neurological diagnoses, as well as for a new kind of fuzzy neural networks.

1 Introduction

Evoked potentials are frequently used in the brain research. Souček and Carlson [1,2] have found that insect brain generates special kind of evoked time sequence: brain-windows. The brain-window theory is used here to explain the human evoked potentials. Two datasets have been used, called INSECT and HUMAN.

INSECT. A firefly flash is a brilliant burst of light which serves as a signal in a dynamic courtship communication system between males and females. Because it is possible to observe and recored firefly flashes from a distance and to communicate with firefly using artifical flashes, these animals provide ideal material for the analysis of insect brain functions.

The fireflies Photuris versicolor were courted using artificial flashes provided by a flashlight. The flashlight was driven with a relay controlled stimulator. The duration of the flashes varied between 0.1 and 0.2 seconds. The artificial flashes and female responses were recorded using a hand-held photomultiplier, the output of which was fed into a tape recorder. For details see [1,2].

HUMAN. Brainstem Auditory Evoked Potentials (BSAEPs) are generated in response to a brief auditory stimuli with seven peaks appearing within 10 ms following the stimulus in normal subjects. Pathological states resulting from head injury, acoustic tumors and multiple scleroses give rise to delays in the transmission of electrical signals and consequently the peaks are abnormally located. The BSAEPs were obtained from the Vertex-left mastoid, Vertex-right mastoid electrode locations on the scalp employing a Nivolet Pathfinder II system. Details can be found in [3,4,5].

2 The Brain Windows

The theory that explaines the HUMAN and INSECT datasets is based on fuzzy, adjustable logic called "brain windows". The logic is supported by a network of coupled nonlinear oscillators. Upon receiving stimulus, the brain generates a sequence

of time windows of different widths. Receiving and sending windows are interleaved in the sequence. Each receive window recognizes a particular subgroup of stimulus intervals. Each sending window determines the latency of the response from the brain. The windows are arrayed in priorities and controlled by the memory. Memory stores the past history. Brain windows are generated through mutual coupling of the primary oscillator, answer oscillator, and window generator, see Figure 1. Hence, the brain windows are directly related to the inherent biological oscillators and to the memory. The oscillator generates a primary waveform $P(t)$ with a period T_1. Upon receiving a stimulus, the memory M_1 is charged and slowly discharges back toward zero (Figure 1a). In this way, M_1 modulates the primary waveform (Fig. 1b). Hence, $M_1(t)$ is equivalent to the phase-response curve (PRC). The positive and negative phases of the primary waveform designate receive and send wondows. Receive windows, defined by the positive phase of $P(t)$, are periods during which a second stimulus can command an answer. Send windows, during which a response can actually be generated by the brain are defined by negative phases of $P(t)$. A second memory, M_2, recalls the past history of stimulation. Depending on the past history, M_2 can take any value in the range $-1 < M_2 < 1$. The intersection of the memory M_2 and the primary waveform $P(t)$ defines the sequence of the receive-send brain windows. Figures $1c, d, e$ show three receive-send windows sequences for the memory values M_2', M_2'', M_2''', respectively. The basic carrier of information is the interval I between two stimuli. The second stimulus is matched against the train of receive windows. Each receive window recognizes a particular group of intervals. In this way, the brain receives and analyzes the stimulation interval I. This interval can be considered as a question in the communication. The logic of the brain generates the answer to the received question. The answer information is coded in the latency L of the response. The latency is matched against the train of send windows. Each send window defines a particular group of latencies as a group of legal answers. Hence, the receive interval I (question) will produce the answer with the latency L only if I matches one of the receive windows and L matches one of the send windows. The brain windows operate with external, as well as with internal stimuli and responses.

3 Holographic Network for Neurological Diagnoses

Holographic networks are a new brand of neural networks, which have been developed by Sutherland [6,7]. This type of networks significantly differs from the conventional back-propagation layered type. The main difference is that a holographic neuron is much more powerful than a conventional one, so that it is functionally equivalent to a whole conventional network. Therefore there is no need to build massive networks of holographic neurons; for most applications one or few neurons are sufficient. In a holographic neurons there exist only one input channel and one output channel, but they carry whole vectors of complex numbers. An input vector S is called a stimulus and it has the form:

$$S = [\lambda_1 e^{i\theta_1}, \lambda_2 e^{i\theta_2}, \cdots, \lambda_n e^{i\theta_n}].$$

An output vector R is called a response and its form is:

$$R = [\gamma_1 e^{i\varphi_1}, \gamma_2 e^{i\varphi_2}, \cdots, \gamma_n e^{i\varphi_m}].$$

All complex numbers above are written in polar notation, so that modules are interpreted as confidence levels of data, and phase angles serve as actual values of data. The neuron internally holds a complex $n \times m$ matrix X, which enables memorizing stimulus-response associations. Learning one association between a stimulus S and a desired response R reduces to the (noniterative) matrix operation

$$X+ = S^T R.$$

Note that all associations are enfolded onto the same matrix X. The response R^* to a stimulus

$$S^* = [\lambda_1^* e^{i\theta_1^*}, \lambda_2^* e^{i\theta_2^*}, \cdots \lambda_n^* e^{i\theta_n^*}]$$

is computed through the following matrix operation:

$$R^* = \frac{1}{c^*} S^* X.$$

Here c^* denotes a normalization coefficient which is given by $c^* = \sum_{k=1}^{n} \lambda_k^*$.

The response R^* to a stimulus S^* can be interpreted as a vector (i.e. a complex number)

TABLE 1

	Normal	Abnormal (Multiple Sclerosis)	Total
Training Set	30	23	53
Testing Set	46	34	80

composed of many components. Each component corresponds to one of the learned responses. If S^* is equal to one of the learned stimuli S, then the corresponding response R occurs in R^* as a component with a great confidence level (≈ 1). The remaining components have small confidence levels ($\ll 1$) and they produce a "noise" (error).

It is believed that the BSAEP latencies provide necessary and sufficient information to discriminate between normal and pathological states. The experiments have shown that the 2nd, 3rd and the 4th peak latencies are the optimal features for classification. Ho et al [5] have collected a total of 133 BSAEP patterns from patients of which 53 are used for training and the rest were used for testing. Table 1 shows the number of normal and abnormal BSAEP patterns in the training and testing sets.

Holographic Neural Technology [6,7,8] is a relatively new artificial neural system paradigm that resembles to a class of mathematics found within optical holograms. An element of information within the holographic neural paradigm is represented by a complex number operating within the phase and magnitude. The input BSAEP patterns $S = \{s(0), s(1), s(2)\}$ (the 2nd, 3rd and 4th peak latencies) and output class $R = \{r\}$ ($r = -ve$ (Normal) and $r = +ve$ (Abnormal)) were converted from real values to the complex representations in the neural system by sigmoidal preprocessing operation

$$s(k) \rightarrow \lambda_k e^{i\theta_k}$$

$$\theta_k \rightarrow 2\pi(1 + e^{\frac{\mu - s(k)}{\sigma}})^{-1}$$

where μ is the mean of distribution over $S, k = 0, 1, 2, \sigma$ is the variance of distribution, and λ_k is the assigned confidence level. The above transformation maps the above input BSAEP patterns to corresponding sets of complex values.

The experiments with the holographic network show that the number of higher order terms determines not only the learning time, but also the classification accuracy [5]. The optimal number of the higher order product terms are 50. The results of using holographic network in classification of BSAEPs after the first learning trial are presented in Table 2.

TABLE 2

	Ref. diagn. Abnormal	Ref. diagn. Normal
Classif. res.: Class A	31	0
Classif. res.: Class N	3	46
Overall Performance: 96,25 %	Number of Ephocs: 1	Training Time: 1 sec

4 Results

The brain window concept has been used to explain the evoked patterns in HUMAN and INSECT datasets. The experimentaly observed evoked potential and behavior waveforms follow the theoreticaly predicted primary oscillator waveform, as presented in Figure 1. In other words, the waveforms are generated through mutual coupling of biological oscillators and modulated by the memory that stores the past history and the present behavior. Holographic network classifies BSAEP peak latencies and discriminate between normal and pathological states, with 96% accuracy. Experiments show that holographic learning of HUMAN dataset takes 1 second, while backpropagation learning takes 20 seconds.

5 Conclusion

The brain window concept has been used to explain the evoked patterns in HUMAN and INSECT datasets. The experimentaly observed evoked potential and behavior waveforms follow the theoreticaly predicted primary oscillator waveform, as presented in Figure 1. In other words, the waveforms are generated through mutual coupling of biological oscillators and modulated by the memory that stores the past history and the present behavior.

The memory stores the internal context. Memory adjusts the sequence of receive-send windows.

The stimulus interval I presenting the sensory pattern is matched against the train of receive windows. The response latency L, presenting the answer or decision, is matched against the train of send windows. The language is directly related to the physiological findings: biological oscillators and pulses (flashes).

Brain-window language is in excellent agreement with experimental data measured on Photuris versicolor female fireflies stimulated by artificial flashes. Computer analysis of a large volume of experimental data reveals the fact that data points are clustered into islands of dialogues.

INSECT dataset is explained with one level of oscillator-pulse interaction. This concept can be extended to the hierarchy of oscillator-pulse levels. Each level has its sequence of receive-send windows, and its memory (context). The sensory patterns and the decisions or commands pass through the hiearchy in opposite directions.

The HUMAN data set keeps the BSAEPs generated in response to a brief auditory stimuli with seven peaks appearing within 10 ms following the stimulus in normal subjects. The latencies of the initial five peaks of BSAEPs are highly stable in healthy normal subjects under a wide variety of physioloigical conditions such as sleep, wakefulness and anesthesia. However, in pathological states resulting from head injury, acoustic tumors,autistic disorders and multiple sclerosis the peaks are abnormally located. One explanation is the change of delays in transmission of electrical signals.

According to Chiappa [10], there is no strong primary evidence in humans to define the presumed generator sources of BSAEP waveforms. The suggested generator sources are as follows: wave I-distal eighth nerve; wave II-proximal eighth nerve or cochlear nucleus; wave III-lower pons (possibly the superior olivary complex); wave IV-mid or upper pons (possibly the lateral lemniscus tracts and nuclei); wave V-upper pons or inferior colliculus.

It is not known whether BSAEPs are being generated at synapses in gray matter nuclei or by volleys in white matter tracts, or by the result of summation of electrical activity from more than one nucleus.

Because exact generator sources of BSAEP waveforms are not known (synaptic versus tract potentials or both), the pathophysiology of abnormalities is also speculative. In experimental animals, unilateral and bilateral focal brain stem cooling produced BSAEP amplitude and latency abnormalities, respectively. However, the variety of diseases in which BSAEP abnormalities are found suggests that multiple factors can be involved, presumably including segmental demyelination and axonal and neuronal loss, and modification of the brain-window hierarchy.

The brain-window hierarchy is modulated by the memory that stores the past history and the present behavior. Hence in pathological states, the memory disturbance dictates the abnormal location of the response peaks. If this is so, than the subject treatment could be also oriented in the direction of the memory and its contents. The diagnoses might include the stimulation of the subject with the pairs or trains of auditory, light or electrical stimuli. The pattern of the train should coincide with the brain-window sequence. Further experiments, both with simple animals and with human, are needed to clear out remaining questions. The experiments include the use of anesthetic on human, and of ethanol on firefly luciferase.

Combining Brain-windows with holographic learning opens new possibilities: a) explanation of other brain codes, languages and signals; b) design of a new class of fuzzy neural networks; c) new kind of neurological diagnoses; d) adaptation of holographic learning for large training and testing sets found in pattern recognition, speech and vision; e) Brain-window concept for natural language processing, reasoning and language-knowledge archives; f) interaction among the fuzzy Brain-windows representing the symbol, word, schema, frame, association map or cognitive map; g) diagnoses: Parkinson, Huntington, Wilson, Infarcations, Ischemia, Hemorrhages, Coma, Epilepsy, Hysteria, Meningitis, Surgery, etc.

References

[1] B. Souček, A.D. Carlson, Brain Windows in Firefly Communciation, Journal Theoretical Biology, 119, 47-65, 1986

[2] B. Souček, A.D. Carlson, Brain Window Language in Fireflies, Journal Theoretical Biol-

ogy, 125, 93-103,1987

[3] R. Ho, A Neural Network System for Recognition of Evoked Potentials, M. Sc. Thesis, Dept. Computer Science and Systems, McMaster University, Hamilton, Ontario, Canada, 1990.

[4] M.V. Kamath, S.N. Reddy, A.R.M. Upton, D.N. Ghista, M.E. Jernigan, Statistical Pattern Classification of Clinical Brainstem Auditory Evoked Potentials, Int. J. Biomed. Comput., 21, 9-28, 1988

[5] R. Ho, J.G. Sutherland, I. Bruha, Neurological Fuzzy Diagnoses: Holographic vs Statistical vs Neural Method, in Ref. 9

[6] J. Sutherland, A Holographic Model of Memory, Learning and Expression, Int. Journal of Neural Systems 1,3,259-267, 1990

[7] J. Sutherland, The Holographic Neural Method, in Ref. 8

[8] B. Souček and IRIS Group, Fuzzy, Holographic and Parallel Intelligence, J. Wiley, New York, 1992

[9] V.L. Plantamura, B. Souček, G. Visaggio, Frontier Decision Support Concepts: Help Desk, Learning, Fuzzy Diagnoses, Quality Evaluation, Prediction, Evolution, J. Wiley, New York, 1994.

[10] K.H. Chiappa, Evoked Potentials in Clinical Medicine; Chapter 5; Raven Press Ltd, New York, 1989.

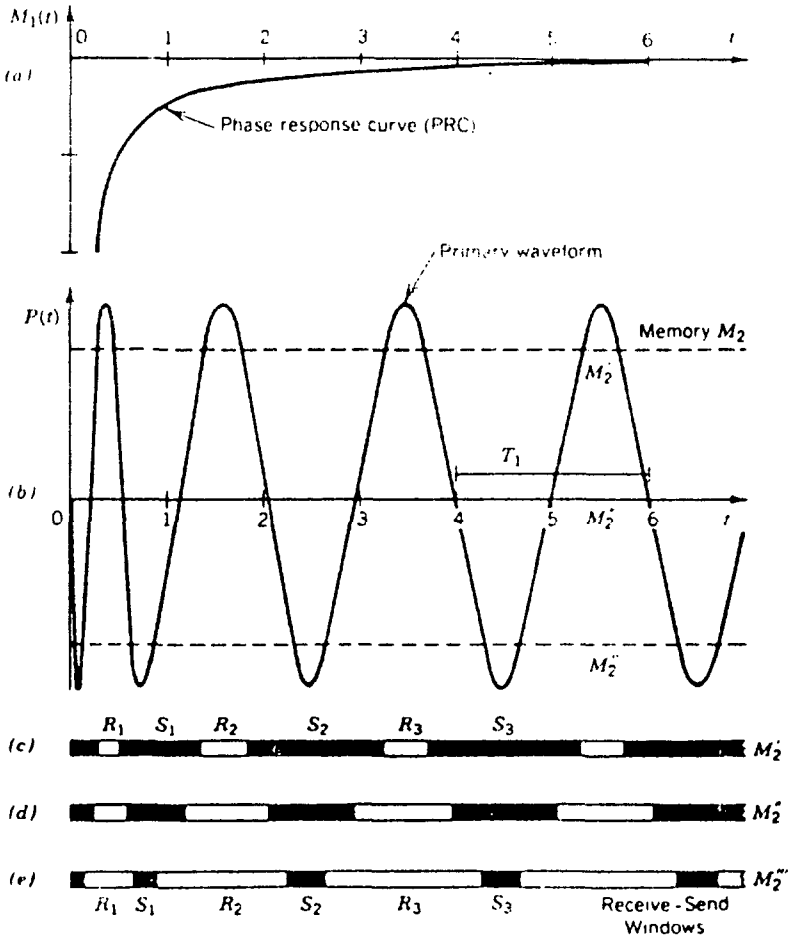


Figure 1: Brain-windows in evoked potentials and time sequences. (a) Phase-response curve (PRC). The simulus flash charges the memory (M_1) which decays with time, producing the PRC. (b) Primary waveform defined by the PRC. As the PRC declines toward zero, the period of the primary waveform increases towards the resting value. Memory levels caused by previous flashes (M_2) are shown as horizontal lines intersecting the primary waveform. These memory levels define the receive-send windows. (c,d,e) Receive-send window periods defined by memory levels (M_2): (c) highly positive memory level (M_2); receive windows narrow, send windows wide; (d) memory at zero level (M_2); (e) highly negative memory level (M_2); receive windows wide, send windows narrow.

Approximating Knowledge in a Multi-Agent System

Miroslav Kubat* and Simon Parsons†‡

*Ludwig-Boltzmann Institute of Medical Informatics and Neuroinformatics,
Department of Medical Informatics, Institute of Biomedical Engineering
Graz University of Technology, Brockmanngasse 41, A-8010 Graz, Austria
email mirek@dpmi.tu-graz.ac.at

†Advanced Computation Laboratory, Imperial Cancer Research Fund,
P.O. Box 123, Lincoln's Inn Fields, London WC2A 3PX, United Kingdom.
email sp@acl.lif.icnet.uk

‡Department of Electronic Engineering, Queen Mary and Westfield College,
Mile End Road, London E1 4NS, United Kingdom.

Keywords: Artificial Intelligence, abstraction, granularity of knowledge, dl-cut, rough concepts

Edited by: Matjaž Gams
Received: August 26, 1993 **Revised:** May 4, 1994 **Accepted:** June 9, 1994

This paper is concerned with establishing a common language that can be used to communicate between the different members of a multi-agent system. We suggest that this may be done by successively approximating the concepts that each agent in the system deals with, and the paper gives algorithms which make this possible. Along the way we introduce the notion of a description language cut, or dl-cut, which is an abstraction to which a rich class of languages may be mapped. The idea of a dl-cut is then used to introduce rough concepts— rough descriptions of the concepts used by the agents. Finally we discuss the way in which rough concepts can be logically combined and used in deductive reasoning, also debating the scope of the validity of inferences using the concepts.

1 Introduction

Over the last twenty years, techniques from artificial intelligence have been successfully applied to problems ranging from factory scheduling [23], to process control [15], and the diagnosis of faults in complex systems [12]. Expert systems have been developed which can replicate or exceed the accuracy of human experts [3], and which have bodies of knowledge that make them as knowledgeable as the best informed human expert [16]. With the increasing power and sophistication of these systems have come a number of well-documented problems — in general intelligent systems do not scale up easily, they tend to be brittle so that their performance breaks down as they leave their domain of expertise rather than degrading slowly as that of a human exp ert would, and it is difficult to ensure that they are consistent.

A number of solutions have been suggested to remedy these ills, each stemming from a major research effort. One is to try to ensure consistency by constructing intelligent systems in a more rigorous way, structuring the knowledge that they contain and ap plying techniques from software engineering. This work is typified by the KADS project [22] and has led to interesting developments in areas such as the formal specification of knowledge-based systems [5].

A second approach is to reduce brittleness by building intelligent systems around a vast body of commonsense knowledge that approximates the kind of knowledge that people use in their interactions with everyday situations. This, in the-

ory, will allow such intelligent systems to fall back on more general ideas when their specific expert knowledge fails. For instance, when a medical diagnosis system is queried about the ailments suffered by someone's car it should be able to transfer some of its basic knowl edge about diagnosis and use this with commonsnese knowledge of how cars work to attempt an answer. The CYC project [14] which aims to do precisely this has recently released the first version of its knowledge base, and it will be int eresting to observe whether the claims made for it are justified.

A third approach, and the one that we will consider in this paper, is to build communities of small, and therefore more manageable, systems. Because the individual systems contribute different skills and knowledge, together they are capable of handling pr oblems that are beyond the scope of a single system. This is the approach of the ARCHON project [11] which has proved itself in the area of industrial process control in general, and in the construction of a co-operative system for elect ricity distribution management [4] in particular.

Now, one of the most interesting things about the ARCHON system is that it provides a framework for combining together existing systems. The motivation for this is the promotion of code and resource reuse, which is clearly a worthy aim, but in doing so raises a new and difficult problem. Different systems developed at different times may use different languages for knowledge representation. If this is the case, how should they be combined? Work on ARCHON understandably stopped short of providing an answer to this question, and it seems that little has yet been published on general solutions to the problem, though there has been some work on translating between different uncertainty handling formalisms in this context [19, 25].

However, some preliminary work has been published on related subjects. Huhns et al. [9] describe ways of integrating different information models of businesses, that is they discuss the problems of relating such models and resolving incompatibilities between them. To do this they make use of the CYC ontology, which is postulated to be of sufficient extent to encompass any notion in any business model, and integrate different models by integrating them into CYC. The result of this work is a system called Carnot, which provides an architecture and tools for integrating the information models of large businesses. Neches et al. [17] make a similar suggestion but from the more general perspective of integrating knowledge representation systems irrespective of domain or interpretation. To do this they suggest the idea of an "interlingua" which is a general language for knowledge interchange. At first blush, such a language certainly seems to be a good idea, but, as Ginsberg [6] argues, there are reasons why the definition of a standard interlingua seem premature.

This work on interlinguae assumes that there will be some underlying ontology, some basic set of concepts and their inter-relations, that is understood by all systems. Now, while such ontologies exist in some domains [17], they are far from universal, so there are domains in which the approaches discussed above will founder. In this paper we present some initial ideas about the way in which a model that is common to a number of intelligent systems that do not have a known common ontology might be constructed automatically from the models of individual agents within the group.

2 Basic concepts

Our inspiration to develop methods to obtain a common interpretation of knowledge from multiple sources stems from the domain of *multistrategy learning*, first proposed by Brazdil [1]. The ability of this principle to improve performance was demonstrated by Brazdil and Torgo [2], and by Torgo and Kubat [24]. In the particular case we will consider here, the problem involves several agents and a central system.

The agents possess knowledge which is expressed in a particular language and the task of the central system is to combine the knowledge into a general structure. The problem in doing this is that the language used by any agent may be different to the languages used by any other agent, and, if this is the case, the central system will need to translate the information obtained from the individual agents into some common basis which we will refer to as the central language *CL*.

Figure 1 illustrates the situtation we will con-

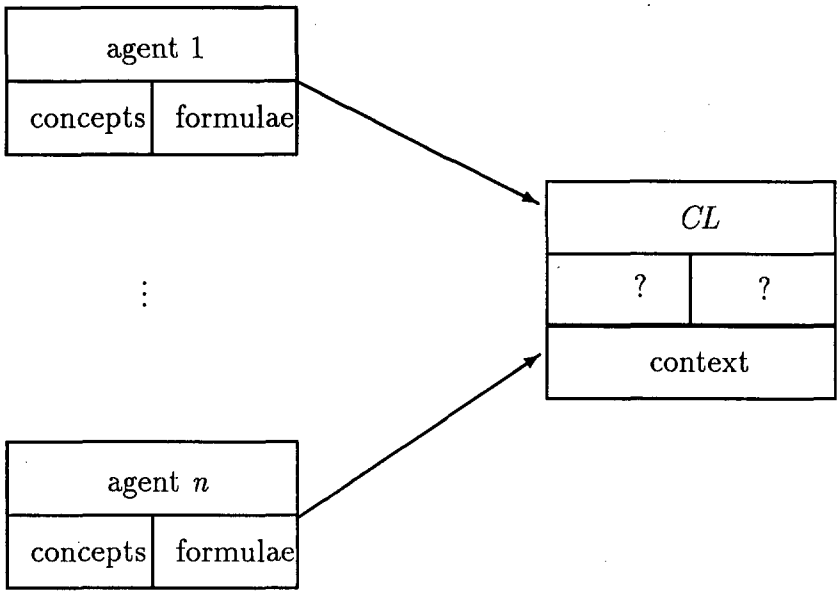
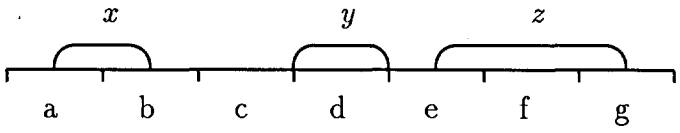


Figure 1: The system under consideration



$$x \subset \{a, b\}, y = \{d\}, \{f\} \subset z \subset \{e, f, g\}$$

Figure 2: Roughly described concepts x, y and z

sider. Each agent has a language which expresses a set of concepts and a set of logical formulae concerning those concepts. The central system contains the *context* of the overall system which plays an important role in any application of the knowledge of the multi-agent system since the kind of concepts with which the system will deal have connotations which vary widely according to the context. Thus, for instance, the concepts 'fertile land' and 'warm day', have widely different meanings in Central Africa and in Sweden.

Note also that in this case, unlike the agents, the central system has no direct access to the environment, however there is no theoretical reason why the central system should not have access, nor, for that matter why there should be a "central" system with a distilled common language. Instead the common language could be replicated in each agent, producing a group with no central focus, but whose members could all understand each other.

Now, when the system of agents is initially set up, the central system has no understanding of the languages used by the various agents. However, it is possible that it can establish a common language by approximation. That is, it is possible for each agent to describe its knowledge to the central system in terms of its set of concepts. Depending upon the wealth of concepts available to the agent this may be a very precise or a very coarse description of its knowledge so that there is no guarantee as to the precision of the translation that is possible between the agent and the central system.

When all the agents do this the central system will end up with a language which can deal to some degree with all of the knowledge dealt with by all the agents, and so is broader than that of any single agent. In addition, due to the intersection between concepts, it may be more detailed

than that of any agent.

What we propose in this paper are some thoughts as to how this might be achieved within the framework of rough concepts which we have developed [13] from ideas on rough sets [21].

A few informal definitions are needed to clarify some of the notions that we will operate with. A language, often called a *description language*, is understood as a set of well formed formulae (*wff*). We will only deal with languages with a finite number of *wffs*. Each *wff* represents a *concept* captured by the language. Each concept, in turn, is interpreted as a set of relevant objects assigned to it by its context. Thus, when speaking about students, we usually do not mean all students in the world. Rather, we implici tly constrain ourselves to the students of our university, students of Computer Engineering, students from the secondary school in the neighbourhood, and the like. The context represents additional information common to all concepts, and in this case we a ssume that the context is common to all agents.

Figure 2 presents a graphical representation of possible relationships between CL and concepts known by an agent. Each segment a through g stands for one *wff* of a simple CL, and consists of objects that cannot be further discerned in CL. Each *wff* is true for one or more objects. The lozenges x, y and z are concepts known by an agent. Note that, due to the different languages used by the agent and the central system, the boundaries of x do not coincide with those of the *wffs* of CL so that x is not described as precisely by CL as by the language of the agent. However, without any additional information, the classification of the objects from the segment b as positive or negative instances of x is completely unknown and cannot be quantified by a probability or a fuzzy degree of membership, so that this imprecise classification is still very useful. In addition, as concepts y and z illustrate, CL may be able to precisely distinguish the concepts of an agent, or even some subsets of some the concepts understood by an agent.

This issue is closely related to work on granularity such as that by Hobbs [8] who defined an *indistinguishability* relation for unary predicates and Imielinski [10] who extended Hobbs work so that the idea can be applied to approxima te reasoning. Our proposal also has notions in common with Carnot [9] which uses the idea of finding the best generalisation of a given concept, and with Ginsberg's [6] discussion of KIFs in which he proposes discarding details s uch as probabilities in order to facilitate interchange between agents that quantify their knowledge and those that do not.

For simplicity, we assume that the information possessed by the agents is noise-free and relevant. Readers interested in a method for pruning out noisy and irrelevant knowledge can find details in work by Brazdil and Torgo [2]. Thus the task that we will address is defined as follows:

Given:

A definition of the central language CL;

The general context expressed either as a set of constraints on the set of objects with which the multi-agent system will deal (such as the set of all types of car manufactured in Europe), or as a list of possible objects (such as Rover Metro, Nissan Micra, Ford Escort, ...);

For each agent, the descriptions of concepts and formulae in the agent's language which allow the agent to classify objects in terms of the concepts;

Find:

The description of all concepts in CL;

The scope of validity of the old as well as newly inferred propositions in CL.

The essence of this translation process is *abstraction*, a phenomenon that has been widely studied in artificial intelligence. A comprehensive analysis is provided by Giunchiglia and Walsh [7] where three types of abstraction are defined, depending on the ability of the source language L_1 and the object language L_2 to distinguish objects. Informally, an abstraction is *constant* if both languages discern the same objects; an abstraction is *decreasing* if L_1 is able to distinguish the same objects as L_2 and possibly some more; an abstraction is *increasing* if L_2 is able to distinguish the same objects as L_1 and possibly some more. The preceeding discussion makes it clear that, depending upon the exact concepts available, our method may give any of these types of abstraction, and indeed may give a mixture of different types for different agents in the same system.

3 Translating into *CL*

In this paper, no strong requirement is made on the syntax of the well-formed formulae— we use a logic-like notation to describe the attributes of the objects which exemplify the concepts the various agents deal with. However, this notation is used purely for convenience since it allows us to write down ideas such as "the shape of a certain class of object is either a cube or a pyramid" in a concise way as, for instance:

$$\Big(shape(x) = cube\Big) \vee \Big(shape(x) = pyramid\Big)$$

or:

$$shape(x, cube) \vee shape(x, pyramid)$$

and it should not be seen as a fundamental limitation on the approach— the results presented in the paper hold whatever form the *wffs* are written in.

To get an idea of the motivation for the dl-cut and rough concepts, consider a simple example.

Example 1.
Let a set of toy blocks be described by their shape: cube, pyramid, ball, and prism. The *CL*-language capable of describing the shape by means of these terms decomposes the set into four disjoint subsets, each of which is represented by at least one object. Suppose the concept to be translated into *CL* is 'stable in an earthquake.' Cubes and pyramids are stable, balls are not stable, and the stability of a prism depends on the ratio of its base area to its height. Since no distinction is made between the different types of prism, *CL* cannot discern short, fat prisms (stable) from tall, thin prisms (unstable). If no additional information is available, the concept 'stable in an earthquake' can only be approximated by its lower bound (sufficient condition) and upper bound (necessary condition):

lower bound:

$\forall x$ *stable(x)* $\equiv$
shape(x, cube) $\vee$ *shape(x, pyramid)*

upper bound:

$\forall x$ *stable(x)* $\equiv$
shape(x, cube) $\vee$ *shape(x, pyramid)*
$\vee$ *shape(x, prism)*

□

The lower-bound description (*core*) is true for cubes and pyramids, whereas the upper-bound description (*envelope*) is true for cubes, pyramids, and prisms. Obviously, the 'distance' between the core and envelope depends on the language *CL*. Co ncepts expressed by the pair [core, envelope] are called *rough concepts.*

The notion of a *dl-cut*, defined below, will facilitate the formalization of the approach that we have just outlined. In the following definition, the universe U is the set of all objects seen by the central system.

Definition 3.1 (dl-cut) *Denote by Dl the set of all* wffs *of a language. A subset dl* $\subseteq$ *Dl is called a* dl-cut *iff it decomposes U into a system of pairwise disjoint sets such that each set is assigned precisely one wff f* $\in$ *dl that is true for all elements of this set.*

Thus in the simple system from Example 1, the universe of all blocks can be decomposed into four disjoint sets each of which is assigned one of the following predicates:

shape(x, cube), shape(x, pyramid),
shape(x, ball), shape(x, prism)

Each predicate is a *wff* and the set of four predicates is a dl-cut. In general there is not a unique dl-cut for a given universe. In this case, an alternative dl-cut is made up of the following three *wffs*:

shape(x, cube) $\vee$ *shape(x, prism),*
shape(x, pyramid), shape(x, ball)

The elements (*wffs*) of a dl-cut are *description items* or *generic concepts* which may be distinguished by the central system, and may be used by it to approximate the concepts handled by the various agents with which it communicates. Knowing th at any disjunction of description items can be considered to be a concept, we can discern, by means of the dl-cut, 2^N different concepts, where N is the number of *wffs* in the dl-cut. The notion of a dl-cut facilitates a mapping of a rich class of languages onto easy-to-handle boolean expressions (for a deeper analysis, see [13]).

Basically, there are two possible approaches to the construction of a dl-cut from the underlying language— a naïve approach, and a concept-oriented approach. We only cover the naïve approach in detail, contenting ourselves with hinting

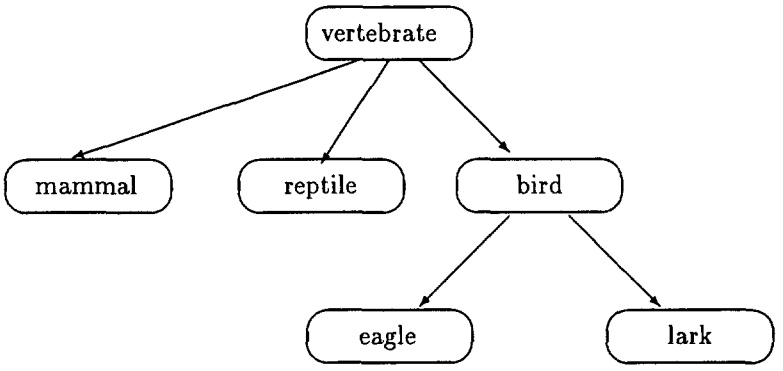


Figure 3: Example of a generalization tree

at how the concept-oriented approach can be employed.

The naïve method uses a hill-climbing search technique. The initial state is the empty set of *wffs*, the final state is a set of *wffs* that form a dl-cut, and the search mechanism is to augment the current set of *wffs* with a *wff* that does not ov erlap with any previous *wff*. Obviously, an exhaustive search would reveal many different dl-cuts whose capacity to model concepts varies. Hence, the search must be made heuristic by adding some criterion for selecting which *wff* to add to the dl-cut. To be useful, this criterion should reflect the ability of the dl-cut that results from the addition the *wff* to model concepts. This approach leads us to propose Algorithm 1.

The prerequisite for this algorithm is the availability of a *generality criterion* which gives some idea of the quality of a dl-cut, and of a mechanism for a *subsumption test* to establish if one *wff* can replace another. The *generality criterion* in a system such as ours which is based upon the manipulation of attributes naturally reflects the number of literals in an expression since each of these corresponds to an attribute of the domain. Thus A is more general than $A \wedge B$ and $A \vee B$ is more general than A. The *subsumption test* can be based on knowledge of the domain, so that 'bird' subsumes 'eagle' as in Figure 3 or on the explicit listing of the concepts represented by a *wff*. Thus if all objects representing concept A also represent concept B, then B subsumes A. By convention, we write $p \subseteq q$ if q subsumes p, and $p \supseteq q$ if p subsumes q.

A similar notion to subsumption is that of overlapping. Two *wffs* l_1 and l_2 are said to *overlap* if $l_1 = l_{11} \vee l_{12} \vee \ldots \vee l_{1n}$, $l_2 = l_{21} \vee l_{22} \vee \ldots \vee l_{2m}$, and $l_{1i} = l_{2j}$ for some i and j. Finally, the language CL is assumed to be sufficiently rich that at least some of its *wffs* l_i must be subsumed by some concepts s_j, that is $l_i \subseteq s_j$, and for each object $u_k \in U$, a *wff* l_p can be found such that l_p is true for u_k.

The input to the algorithm is the set S of all concepts, the context which defines the extent of the universe U, and CL, which when the first dl-cut is constructed will be empty, but for other dl-cuts will be a set of *wffs*. Further, let $L^{(CL)} = \{l_1, \ldots, l_n\}$ be the list of *wffs* from CL, in descending order according to some generality criterion so that for $l_i, l_j \in L^{(CL)}$, if $i < j$, l_j is not more general than l_i.

The algorithm terminates when S is empty or when the ability of $dl^{(CL)}$ to discern concepts cannot be increased without backtracking from the current state.

Algorithm 1

1. Let $dl^{(CL)} = \emptyset$;
2. Starting with l_1, search for the first $l_i \in L^{(CL)}$ such that an $s_j \in S$ can be found for which $l_j \subseteq s_j$. If no such l_i can be found, go to 5;
3. Let $dl^{(CL)} = dl^{(CL)} \cup \{l_i\}$. Discard all $l_j \in L^{(CL)}$ overlapping with l_i;
4. Delete from S all concepts s_i such that $s_i \subseteq d_j$ where d_j is any disjunction of *wffs* from $dl^{(CL)}$. If S is not empty, go to 2;
5. Find a *wff* that is true for the rest of the universe U, add it to $dl^{(CL)}$ and *stop*.

Of course, many different procedures using more powerful heuristics and search techniques can be proposed, and their detailed analysis is an open research topic. The algorithm we have presented can serve as a guideline.

Before we proceed to the illustration of the algorithm by a simple example, a few comments are necessary. Firstly, an additional requirement in step 2 can demand that the concept s_j subsuming l_i is not allowed to subsume any other concept s_k. This requirement makes sense if agents are able to order their concepts by subsumption.

Secondly, since the explicit storing of $L^{(CL)}$ would limit the utility of the procedure to very small languages, the list is intended to be implicit. Thus in the language based on conjunctions of unary predicates, the algorithm would begin with single predicates, then, when the ability of the predicates to describe concepts has been exhausted, proceed to conjunctions of pairs of predicates selected by a suitable heuristics.

It is also possible to derive an alternative algorithm driven by the concepts s_I instead of the *wffs*. This is the 'concept-oriented' approach hinted at above.

Finally, it should be noted that, in general, subsumption checking is NP-hard for first-order logic and must be assisted, in realistic applications, by background classification information based on notions of generality of concept, such as that depicted in Figure 3. A similar problem can arise with the discarding of overlapping *wffs* in step 3.

Example 2.

Consider once again the blocks world of Example 1 which is extended so that the blocks can be described in terms of the material from which they are made as well as their shape— all cubes, prisms, and pyramids are metallic while balls are wooden. The agents understand the concepts 'stable' and 'belongs to Tom', and are able to classify the objects in U as positive and negative examples of the concepts. In the first step, the central system picks the unary predicates one by one until one of them turns out to be subsumed by any of the two concepts.

Suppose that the concept 'stable' subsumes the predicates *shape(x, cube)* and *shape(x, pyramid)* while 'belongs to Tom' subsumes *shape(x, pyramid)*. The system selects *shape(x, pyramid)* as the first *wff* of $dl^{(CL)}$. From now on, all predicates overlapping with *shape(x, pyramid)* will be discarded so that only the predicates *shape(x, cube)*, *shape(x, prism)*, *shape(x, ball)*, *material(x, wood)* and their conjunctions are allowed to appear in any of the future *wffs*. Thus we might end up with $dl^{(CL)}$ being:

$$shape(x, pyramid), shape(x, cube),$$
$$shape(x, ball) \wedge material(x, wood),$$
$$shape(x, prism)$$

□

Now, we can define the important notion of a *rough concept.*

Definition 3.2 (rough concept) *Let* $x^R(dl) = [x^C(dl), x^E(dl)]$ *be a rough set. A* rough concept *is the pair* $[des(x^C), des(x^E)]$, *where* $des(x^C)$ *is the description of the core of* x *in* Dl_{dl} *and* $des(x^E)$ *is the description of the envelope of* x *in* Dl_{dl}.

Note that the core description does not apply to any negative instance of x, the envelope description applies to all positive instances of x, and the complement of the envelope applies only to negative instances of x. Beware, however, that any pair [core,envelope] pertains to a particular dl-cut. Different dl-cuts tend to imply different rough concepts since the core and envelope are *wffs* from $dl^{(CL)}$. In this respect, the idea of rough concepts departs from Pawlak's rough sets [21]. Even though a *wff* can be understood as a set of objects for which it is true, the symbolic interpretation of an approximation of a concept is dominant.

The next algorithm translates the concepts from the agent's language into *CL*. The input is formed by $dl^{(CL)}$ and by the concepts to be translated into *CL*. As output, the algorithm produces rough concepts in *CL*.

Algorithm 2

For each concept C of an agent, and for any $d_{c_i}, d_{e_j} \in dl^{(CL)}$:

1. If d_{c_i} is subsumed by C, then d_{c_i} belongs to the core;
2. If d_{e_j} overlaps with C, then d_{e_j} belongs to the envelope;

3. The core (respectively, the envelope) is the union of all items d_{c_i} (respectively, d_{e_j}), so that $C^C = \bigcup_i d_{c_i}$, (respectively, $C^E = \bigcup_j d_{e_j}$).

Example 3.
Thus in our running blocks world example, we can write down the rough description of the concepts "stable" and "belongs to Tom". Applying Algorithm 2 we find that for the dl-cut described in Example 2:

$$stable(x)^C = \{shape(x, cube), shape(x, pyramid)\}$$

$$stable(x)^E = \{shape(x, cube), shape(x, pyramid), shape(x, prism)\}$$

Thus:

$$stable(x)^R = \Big[\{shape(x, cube), shape(x, pyramid)\}, \{shape(x, cube), shape(x, pyramid), shape(x, prism)\}\Big]$$

Similarly since the only objects that are known not to belong to Tom are prisms, we have:

$$belongs\_to\_Tom(x)^C = \{shape(x, ball) \wedge material(x, wood), shape(x, pyramid)\}$$

$$belongs\_to\_Tom(x)^E = \{shape(x, ball) \wedge material(x, wood), shape(x, pyramid), shape(x, cube)\}$$

Thus:

$$belongs\_to\_Tom(x)^R = \Big[\{shape(x, ball) \wedge material(x, wood), shape(x, pyramid)\}, \{shape(x, ball) \wedge material(x, wood), shape(x, pyramid), shape(x, cube)\}\Big]$$

□

4 Reasoning with Rough Concepts

The work presented in previous sections makes it possible translate concepts from the languages of various agents into *CL*. If we consider that the central system that uses *CL* will need to reason with these concepts, a natural question arises—how can one logically manipulate rough concepts?

Well, if we take x and y as concepts roughly defined in $dl^{(CL)}$ by means of cores and envelopes it can be easily shown that for the cores and envelopes of their disjunction, conjunction, and negation, the following relations hold where the dl in the parentheses is a shorthand for $dl^{(CL)}$, and V is the set of all *wffs* in the language *CL*:

$$\begin{array}{rcl} (x \vee y)^E(dl) & = & x^E(dl) \cup y^E(dl) \\ (x \vee y)^C(dl) & \supseteq & x^C(dl) \cup y^C(dl) \\ (x \wedge y)^E(dl) & \subseteq & x^E(dl) \cap y^E(dl) \\ (x \wedge y)^C(dl) & = & x^C(dl) \cap y^C(dl) \\ (\neg x)^E(dl) & = & V - (x^C)(dl) \\ (\neg x)^C(dl) & = & V - (x^E)(dl) \end{array}$$

For instance, the envelope of a disjunction of two concepts is equal to the union of the envelopes of the individual concepts. The envelope is understood as a subset of V and is subject to unions, intersections, and subtractions; the concepts themselves are subject to disjunctions, conjunctions, and negations.

The relation of *implication* can easily be defined by means of the partial ordering $\leq$ imposed on the space of all *wffs* such that for *wffs* l_i, l_j and l_k: $l_i < l_j$ iff $l_j = l_i \cup l_k$ and $l_i \leq l_j$ iff $l_i < l_j$ or $l_i = l_j$.

Definition 4.1 (implication)
Let $x^R = (x^C, x^E)$ *and* $y^R = (y^C, y^E)$ *be rough concepts. The operation of* implication *is defined as follows:*

$$(x^R \rightarrow y^R) \Leftrightarrow (x^R \leq y^R) \Leftrightarrow [(x^C \leq y^C) \wedge (x^E \leq y^E)]$$

From the well-known properties of partially ordered sets it follows that $(x^R \rightarrow y^R) \Leftrightarrow (\neg x^R \vee y^R) \Leftrightarrow [\neg x^E \cup y^C, \neg x^C \cup y^E]$.

We can consider the elements of *CL*, which are the rough concepts translated into *CL* by the algorithms in Section 3, as the set of language elements that form the basis of a formal logic. These may be propositional constants or predicate symbols. Denoting this set of language elements as $\mathcal{P}$ we can then consider the set of well formed formulae of this logic, denoting this set as $\mathcal{L}(\mathcal{P})$ [18, 20]. For any $p \in \mathcal{L}(\mathcal{P})$ we define the *rough measure* $R(p)$ of p which is the rough description of the concept or combination of concepts that

$R(p)$	$[\emptyset,\emptyset]$	$[\emptyset,X]$	$[\emptyset,V]$	$[X,V]$	$[V,V]$
$RV(p)$	false	roughly false	unknown	roughly true	true

Table 1: Rough truth values ($\emptyset \subset X \subset V$)

correspond to p. More precisely:

$$R(p) = \left[p^{\underline{\leq} C}, p^{\underline{\geq} E}\right]$$

where $p^{\underline{\leq} C}$ is the lower bound on the core of p and $p^{\underline{\geq} E}$ is the upper bound on the envelope.

Example 4.
To delve a little further into our blocks world example,consider the compound concept p which represents $stable(x) \vee belongs\_to\_Tom(x)$. Now:

$$stable(x)^R = \left[\{shape(x,cube), shape(x,pyramid)\}, \{shape(x,cube), shape(x,pyramid), shape(x,prism)\}\right]$$

$$belongs\_to\_Tom(x)^R = \left[\{shape(x,ball) \wedge material(x,wood), shape(x,pyramid)\}, \{shape(x,ball) \wedge material(x,wood), shape(x,pyramid), shape(x,cube)\}\right]$$

so that:

$$R(p) = \left[\{shape(x,ball) \wedge material(x,wood), shape(x,cube), shape(x,pyramid)\}, \{shape(x,ball) \wedge material(x,wood), shape(x,cube), shape(x,prism), shape(x,pyramid)\}\right]$$

□

One way of interpreting the rough measure of an element $p \in \mathcal{L}(\mathcal{P})$ is the degree to which p is true in the universe of rough concepts. That is how universally it is true amongst the rough concepts. Obviously, for $p^C = p^E = \mathcal{L}(\mathcal{P})$, the proposition is always true and we define the *rough truth measure* $RV(p) = true(t)$. For $p^C = p^E = \emptyset$ the proposition is always false. Three other important truth values of $R(p)$ may be posited, and these are summarized in Table 1 (for more detailed discussion, see Parsons et al. [20]).

The symbolic values in Table 1 indicate to what degree a proposition is true in V. However, the pair $R(p) = [p^C, p^E]$ can also be understood as a more general measure since it explicitly determines in what part of the universe of rough concepts the proposition is true, roughly true and so on. More specifically, the expression $R(p) = [p^C, p^E]$ says that p is true in p^C and roughly true in p^E. Similar considerations enable us to define a truth-ordering on the set of propositions.

Definition 4.2 (truth ordering) *Let p_1 and p_2 be propositions. We say that p_1 is* more true *than p_2 iff $p_1^C \supseteq p_2^C$ and $p_1^E \supseteq p_2^E$.*

By now this section has introduced a rough measure of truth, a set of basic symbolic truth values, the scope of validity of a proposition, and the ordering of propositions based on their truth value. With this background, we can study what happens with the truth measure if we subject the formulae of $\mathcal{L}(\mathcal{P})$ to rules of inference. This is expressed by Theorem 4.1 which is proved in the Appendix:

Theorem 4.1 *For formuale p, q and $r \in \mathcal{L}(\mathcal{P})$, variable x and a constant symbol a, modus ponens (a), modus tollens (b), resolution (c), syllogism (d), and universal instantiation (e) have the following effect on rough descriptions:*

$$a)\quad \frac{R(p \rightarrow q) = [\alpha,\beta] \quad R(p) = [\gamma,\delta]}{R(q) = [\alpha \cap \gamma, \beta]}$$

$$b)\quad \frac{R(p \rightarrow q) = [\alpha,\beta] \quad R(\neg q) = [\gamma,\delta]}{R(\neg p) = [\alpha \cap \gamma, \beta]}$$

$$c)\quad \frac{R(p \vee q) = [\alpha,\beta] \quad R(\neg p \vee r) = [\gamma,\delta]}{R(q \vee r) = [\alpha \cap \gamma, \beta \cup \delta]}$$

$$d)\quad \frac{R(p \rightarrow q) = [\alpha,\beta] \quad R(q \rightarrow r) = [\gamma,\delta]}{R(p \rightarrow r) = [\alpha \cap \gamma, \beta \cup \delta]}$$

$$e)\quad \frac{R(\forall x P(x)) = [\alpha,\beta]}{R(P(a)) = [\alpha, V]}$$

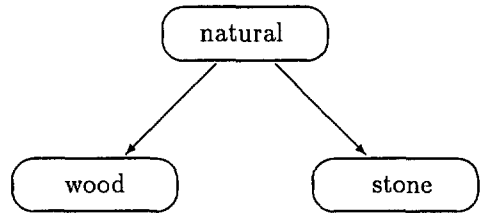


Figure 4: A classification tree

For instance, for modus ponens the theorem reads as follows:

> If the implication $p \rightarrow q$ is true in $\alpha \subseteq V$ and roughly true in $\beta \subseteq V$, and if the formula p is true in $\gamma \subseteq V$ and roughly true in $\delta \subseteq V$, then q is true in $\alpha \cap \gamma$ and roughly true in β.

In this respect, the theorem gives truth-preserving inferential rules for automated reasoning and says in what part of the universe described by CL the rules are really deductive.

Example 5.
Suppose that the dl-cut obtained from CL by Algorithm 1 consists of the following four predicates: $material(x, wood)$, $material(x, stone)$, $material(x, metal)$,
and $material(x, plastic)$. Furthermore, let the background knowledge contain the decision tree from Figure 4 (see overleaf). After simplification with respect to an agent that understands the concepts *interesting* and *tedious* and whose knowledge is summarised by Figure 5 (see overleaf), the dl-cut becomes $material(x, natural)$, $material(x, metal)$, and $material(x, plastic)$, because the background knowledge says that natural material in our universe is either wood or stone and because this simplification does not interfere with the system's ability to discern the concepts *interesting* and *tedious*. The concepts *interesting* and *tedious* are then translated into CL as follows:

$$interesting(x)^R(dl) = \Big[\{material(x, natural)\}, \{material(x, natural), material(x, metal)\}\Big]$$

$$tedious(x)^R(dl) = \Big[\{material(x, metal)\}, \{material(x, metal), material(x, plastic)\}\Big]$$

Thus:

$$\neg interesting(x)^R(dl) = \Big[\{material(x, plastic)\}, \{material(x, plastic), material(x, metal)\}\Big]$$

and:

$$(\neg interesting(x) \rightarrow tedious(x))^R(dl) = \Big[\{material(x, natural), material(x, metal)\}, V\Big]$$

If some piece of knowledge says that, with the exception of metal objects, it is always the case that $interesting(x) \rightarrow foobar(x)$ (where $foobar(x)$ is a concept unknown to the agents) so that $(interesting(x) \rightarrow foobar(x))^R(dl) = [\{material(x, natural), material(x, plastic)\}, material(x, natural), material(x, plastic)\}]$, then it is possible to conclude, using Theorem 3.1 (a), that

$$foobar(x)^R(dl) = \Big[\{material(x, natural)\}, \{material(x, natural), material(x, plastic)\}\Big]$$

□

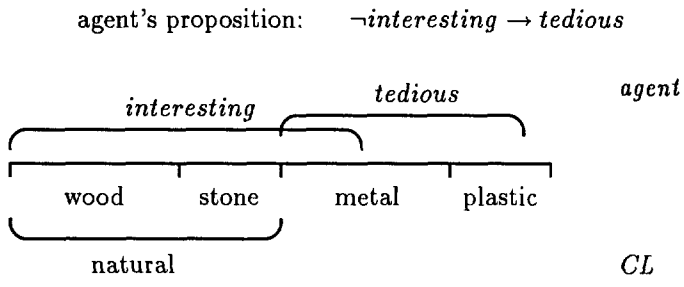


Figure 5: Translation of *interesting* and *tedious* into *CL*

5 A larger example

In this section we give a more extensive example than any so far in order to bring together all the ideas introduced in the paper.

We will consider the construction of a common language from the knowledge of two simple agents. In doing so description language cuts are built from both sets of concepts and their exemplary objects. We will then use the overall language *CL* which understands the concepts known to both agents to build rough descriptions of all concepts, and show how these might be combined in the central system to learn things that were not apparent to the individual systems. The example is kept simple to make it easy to follow, and aims to elucidate new features of our work rather than duplicate previous examples.

Since both Huhns et al. [9] and Pawlak [21] have used similar examples, it seems entirely appropriate that the agents we consider should be concerned with motor vehicles. The first agent understands three concepts, $small\_car(x)$, $fast\_car(x)$, and $slow_car(x)$which are described in terms of the objects in Table 2 (see overleaf) where the table should be read as saying, for instance, that a Rover Metro is an example of both a small car and a slow car. The second agent also knows about slow cars, but also understands the concepts $family\_car(x)$ and $van(x)$, defining these with the examples in Table 3 (see overleaf).

The context of this example is precisely the set of vehicles known by both agents, so that between them they know of every object in U. The set of concepts and objects gives us a deliberately simple dl-cut with which to construct *CL* in the hope of making the example reasonably transparent. Applying Algorithm 1 to the knowledge possessed by the first agent, we initially have $dl^{(CL)} = \emptyset$. Then, one by one we add *wffs*, each of which in this case is a simple term such as $rover\_metro(x)$. Since each *wff* is this simple, $dl^{(CL)}$ increases with each iteration:

iteration 0
$dl^{(CL)} = \emptyset$

iteration 1
$dl^{(CL)} = \{rover\_metro(x)\}$

iteration 2
$dl^{(CL)} = \{rover\_metro(x), austin\_mini(x)\}$

$\vdots$

iteration 6
$dl^{(CL)} = \{rover\_metro(x), austin\_mini(x),$
$ford\_escort(x), vw\_golf(x),$
$reliant\_robin(x), lotus\_eclat(x)\}$

This is a dl-cut that is suitable for describing all the concepts known to the first agent. We then apply the same algorithm to the *wffs* that are formed by the objects known to the second agent. All this second application of the algorithm does is to extend $dl^{(CL)}$ at every iteration, with the exception of the time the *wff* $ford\_escort(x) \vee ford\_cortina(x)$ is considered since this subsumes and thus replaces $ford\_escort(x)$, and the time that $rover\_metro(x)$ is considered since it is already in $dl^{(CL)}$. Thus we have:

iteration 7
$dl^{(CL)} = \{rover\_metro(x), austin\_mini(x),$
$ford\_escort(x) \vee ford\_cortina(x),$
$lotus\_eclat(x), reliant\_robin(x),$
$vw\_golf(x)\}$

$\vdots$

	$small\_car(x)$	$fast\_car(x)$	$slow\_car(x)$
$rover\_metro(x)$	$\star$		$\star$
$austin\_mini(x)$	$\star$		$\star$
$ford\_escort(x)$	$\star$		
$lotus\_eclat(x)$		$\star$	
$reliant\_robin(x)$			$\star$
$vw\_golf(x)$		$\star$	

Table 2: The concepts known by the first agent

	$family\_car(x)$	$van(x)$	$slow\_car(x)$
$ford\_escort(x) \vee ford\_cortina(x)$	$\star$		
$toyota\_corrola(x)$	$\star$		
$ford\_transit(x)$		$\star$	
$vauxhall\_astra(x)$		$\star$	
$rover\_metro(x)$			$\star$
$nissan\_micra(x)$			$\star$

Table 3: The concepts known by the second agent

iteration 15

$$dl^{(CL)} = \{rover\_metro(x), austin\_mini(x), ford\_escort(x) \vee ford\_cortina(x), lotus\_eclat(x), reliant\_robin(x), vw\_golf(x), toyota\_corrolla(x), ford\_transit(x), vauxhall\_astra(x), nissan\_micra(x)\}$$

Given this dl-cut, we can then use Algorithm 2 to build rough descriptions in CL of the concepts known by the two agents. In this case the concepts are quite precisely known, and we have:

$$slow\_car(x)^R = \Big[\{rover\_metro(x), austin\_mini(x), reliant\_robin(x), nissan\_micra(x)\}, \{rover\_metro(x), austin\_mini(x), reliant\_robin(x), nissan\_micra(x)\}\Big]$$

$$fast\_car(x)^R = \Big[\{lotus\_eclat(x), vw\_golf(x)\}, \{lotus\_eclat(x), vw\_golf(x)\}\Big]$$

$$van(x)^R = \Big[\{ford\_transit(x), vauxhall\_astra(x)\}, \{ford\_transit(x), vauxhall\_astra(x)\}\Big]$$

$$family\_car(x)^R = \Big[\{ford\_escort(x) \vee ford\_cortina(x), toyota\_corrolla(x)\}, \{ford\_escort(x) \vee ford\_cortina(x), toyota\_corrolla(x)\}\Big]$$

$$small\_car(x)^R = \Big[\{rover\_metro(x), austin\_mini(x)\}, \{rover\_metro(x), austin\_mini(x), ford\_escort(x) \vee ford\_cortina(x)\}\Big]$$

which neatly illustrates the point made in Section 2 about the nature of the abstraction attainable by the use of rough sets. CL contains knowledge of more concepts than either of the agents, it can define most of them as precisely as either agent, though it has a coarser knowledge of what constitutes a small car. However, it has more precise knowledge of the kinds of slow car than either agent.

Having established CL we can of course use the results of Section 4 to manipulate the rough concepts. For instance, we can evaluate the validity of the idea that all cars known to the system are either fast or slow. That is we can find the rough measure $R(fast\_car(x) \vee slow\_car(x))$, which is:

$$R(fast\_car(x) \vee slow\_car(x)) = \Big[\{rover\_metro(x), reliant\_robin(x), lotus\_eclat(x), austin\_mini(x), vw\_golf(x)\}, \{rover\_metro(x), austin\_mini(x),$$

$$lotus\_eclat(x), reliant\_robin(x), vw\_golf(x)\}\Big]$$

so that the proposition is completely true in that part of the universe known to the central system that does not include vans or family cars, since the rough description of $fast\_car(x) \vee slow\_car(x)$ does not overlap with the rough description of $family\_car(x) \vee van(x)$. However, there is an overlap between the rough descriptions of $fast\_car(x) \vee slow\_car(x)$ and $small\_car(x)$. Indeed, we can test the hypothesis that $slow\_car(x) \leftrightarrow small\_car(x)$, taking this to mean that:

$$slow\_car(x) \rightarrow small\_car(x) \wedge$$
$$small\_car(x) \rightarrow slow\_car(x)$$

Now,

$$(slow\_car(x) \rightarrow small\_car(x))^R =$$
$$\Big[\{rover\_metro(x), austin\_mini(x),$$
$$ford\_escort(x) \vee ford\_cortina(x),$$
$$lotus\_eclat(x), vw\_golf(x),$$
$$toyota\_corrolla(x), ford\_transit(x),$$
$$vauxhall\_astra(x)\},$$
$$\{rover\_metro(x), austin\_mini(x),$$
$$ford\_escort(x) \vee ford\_cortina(x),$$
$$lotus\_eclat(x), vw\_golf(x),$$
$$toyota\_corrolla(x), ford\_transit(x),$$
$$vauxhall\_astra(x)\}\Big]$$

and:

$$(small\_car(x) \rightarrow slow\_car(x))^R =$$
$$\Big[\{rover\_metro(x), austin\_mini(x),$$
$$lotus\_eclat(x), reliant\_robin(x),$$
$$vw\_golf(x), toyota\_corrolla(x),$$
$$ford\_transit(x), vauxhall\_astra(x),$$
$$nissan\_micra(x)\},$$
$$\{rover\_metro(x), austin\_mini(x),$$
$$ford\_escort(x) \vee ford\_cortina(x),$$
$$lotus\_eclat(x), reliant\_robin(x),$$
$$vw\_golf(x), toyota\_corrolla(x),$$
$$ford\_transit(x), vauxhall\_astra(x),$$
$$nissan\_micra(x)\}\Big]$$

so that:

$$(slow\_car(x) \leftrightarrow small\_car(x))^R =$$
$$\Big[\{rover\_metro(x), austin\_mini(x),$$
$$lotus\_eclat(x), vw\_golf(x),$$
$$toyota\_corrolla(x), ford\_transit(x),$$
$$vauxhall\_astra(x)\},$$
$$\{rover\_metro(x), austin\_mini(x),$$
$$ford\_escort(x) \vee ford\_cortina(x),$$
$$lotus\_eclat(x), vw\_golf(x),$$
$$toyota\_corrolla(x), ford\_transit(x),$$
$$vauxhall\_astra(x)\}\Big]$$

which implies that the proposition is less than roughly true, and less true than the proposition that being a small car implies being a slow car.

6 Conclusion

The primary goal of this work was to develop the basis of a method of translating concepts and propositions from different languages used by a set of agents. This goal was achieved with the introduction of the notions of a dl-cut and a rough concept which allow the common language to be established as a dl-cut of all the different languages used by the various agents, and this dl-cut to be used to specify the rough concepts that may be used to express the concepts manipulated by the agents. Two simple algorithms have been provided that make it possible to establish dl-cuts and rough concepts. We also addressed the problem of reasoning with the dl-cuts once they were established, giving results describing how logical reasoning may be performed with the concepts.

Future research in this direction should concentrate on the heuristics needed for the development of more efficient algorithms constructing dl-cuts from description languages. The main issues are, in this respect, the need to minimize the amount of information lost in the process of translation and the complexity of the subsumption checks and tests for overlapping *wffs*.

References

[1] Brazdil, P. B. (1989) Transfer of Knowledge Between Systems: Use of Metaknowledge

in Debugging. In Kodratoff,Y. and Hutchinson, A. (ed.): *Machine and Human Learning.* Michael Horwood, London.

[2] Brazdil, P. B. and Torgo, L. (1990) Knowledge Acquisition via Knowledge Integration. In: Wielinga,B. et al. (eds.): *Current Trends in Knowledge Acquisition*, IOS Press, Amsterdam.

[3] Buchanan, B. G. and Shortliffe, E. H. (1984) *Rule-based expert systems: the* MYCIN *experiments of the Stanford heuristic programming project*, Addison-Wesley, Reading, Mass.

[4] Cockburn, D, Varga, L. Z. and Jennings, N. (1992) Cooperating INtelligent Systems for Electricity Distribution, *Proceedings of the 12th Annual Conference of the British Computer Society Specialist Group on Expert Systems*, Cambridge, UK.

[5] Fensel, D. and van Harmelen, F. (1994) A comparison of languages which operationalize and formalise KADS models of expertise, *The Knowledge Engineering Review*, **9**, (to appear).

[6] Ginsberg, M. L. (1991) Knowledge Interchange Format: The KIF of Death. *AI Magazine*, Fall.

[7] Giunchiglia, F. and Walsh, T. (1992) A Theory of Abstraction. *Artificial Intelligence* **57**, 323–389.

[8] Hobbs, J. (1985) Granularity. *Proceedings of the International Joint Conference on Artificial Intelligence*, Los Angeles, CA.

[9] Huhns, M. N., Jacobs, N., Ksiezyk, T., Shen, W-M, Singh, M. P, and Cannata, P. E. (1993) Integrating enterprise models in Carnot, *Proceedings of the International Conference on Intelligent and Cooperative Information Systems*, Rotterdam.

[10] Imielinski, T. (1987) Domain Abstraction and Limited Reasoning. *Proceedings of the International Joint Conference on Artificial Intelligence*, Milan, Italy, 997–1003.

[11] Jennings, N. R. Wittig, T. Archon, theory and practice, in *Distributed Artificial Intelligence; Theory and Praxis*, (N. M. Avouris and L. Gasser eds.), Kluwer Academic Press, (1992).

[12] de Kleer, J. and Williams, B. C. (1987) Diagnosing multiple faults, *Artificial Intelligence*, **32**, 97–130.

[13] Kubat, M. and Parsons, S. (1993) Reasoning with Roughly Described Concepts. Technical Report 356, Institutes for Information Processing, Graz.

[14] Lenat, D. B. and Guha, R. V. (1990) *Building Large Knowledge-based Systems: Representation and Reasoning in the Cyc Project*, Addison-Wesley, Reading, Mass.

[15] Mamdani, E. H. and Assilian, S. (1975) An experiment in linguistic synthesis with a fuzzy logic controller, *International Journal of Man-Machine Studies*, **7**, 1–13.

[16] McDermott, J. (1982) R1: a rule-based configurer of computer systems, *Artificial Intelligence*, **19**, 39–88.

[17] Neches, R. Fikes, R. Finin, T. Gruber, T. Patil, R. Senator, T. and Swartout, W. R. (1991) Enabling technology for knowledge sharing, *AI Magazine*, Fall.

[18] Parsons, S. and Kubat, M. (1994) A first order logic for reasoning under uncertainty using rough sets, *Journal of Intelligent Manufacturing* (to appear).

[19] Parsons, S. and Saffiotti, A. (1993) Integrating uncertainty handling formalisms in distributed artificial intelligence, *Proceedings of the European Conference on Symbolic and Quantitative Approaches to Reasoning and Uncertainty*, Granada.

[20] Parsons, S., Kubat, M., and Dohnal, M. (1994) A Rough Set Approach to Reasoning under Uncertainty. *Journal of Experimental and Theoretical Artificial Intelligence* (to appear)

[21] Pawlak, Z. (1982) Rough Sets. *International Journal of Computer and Information Sciences 11*, 341–356

[22] Schrieber, G., Wielinga, B., and Breuka, J. (1993) KADS*: A Principled Approach*

to Knowledge-Based System Development, Knowledge-Based Systems, Volume 11, Academic Press, London.

[23] Smith, S. F., Fox, M. S. and Ow, P. S. (1986) Construction and maintaining detailed production plans: investigations into the development of knowledge-based factory scheduling systems, *AI Magazine*, **7**, 4.

[24] Torgo, L. and Kubat, M. (1991) Knowledge Integration and Forgetting. *Proceedings of the Czechoslovak Conference on Artificial Intelligence*, Prague, Czechoslovakia.

[25] Zhang, C. (1992) Cooperation under uncertainty in distributed expert systems, *Artificial Intelligence*, **56**, 21–69.

Appendix

Proof of Theorem 5.1

a) Modus Ponens

$R(q) \supseteq R(p \cap q) = R((\neg p \vee q) \wedge p) = [\alpha \cap \gamma, \beta \cap \delta]$, so the lower bound on the core is $\alpha \cap \gamma$. In addition, $[\alpha, \beta] = R(\neg p \vee q) \supseteq R(q)$, so the upper bound on the envelope is β. Hence $R(q) = [\alpha \cap \gamma, \beta]$. □

b) Modus Tollens

$R(\neg p) \supseteq R(\neg p \cap \neg q) = R((\neg p \vee q) \wedge \neg q) = [\alpha \cap \gamma, \beta \cap \delta]$, so the lower bound on the core is $\alpha \cap \gamma$. In addition, $[\alpha, \beta] = R(\neg p \vee q) \supseteq R(\neg p)$, so the upper bound on the envelope is β. Hence $R(\neg p) = [\alpha \cap \gamma, \beta]$. □

c) Resolution

We know that $R(q \vee r) = R((p \vee q \vee r) \wedge (\neg p \vee q \vee r)) = [((p \vee q \vee r)^C \cap (\neg p \vee q \vee r)^C), ((p \vee q \vee r)^E \cap (\neg p \vee q \vee r)^E)]$. Now, $(p \vee q \vee r)^C \supseteq (p \vee q)^C \cup r^C = \alpha \cup r^C$ and $(\neg p \vee q \vee r)^C \supseteq (\neg p \vee r)^C \cup q^C = \gamma \cup q^C$, so the lower limit on their intersection is $\alpha \cap \gamma$. Similarly, $(p \vee q \vee r)^E = (p \vee q)^E \cup r^E = \beta \cup r^E$ and $(\neg p \vee q \vee r)^E = (\neg p \vee r)^E \cup q^E = \gamma \cup q^E$. Now, the upper limits on r^E and q^E are δ and β, respectively, so the maximum size of the envelope is $\beta \cup \delta$. □

d) Syllogism

This follows immediately from the resolution result. We have $R(\neg p \vee q) = [\alpha, \beta]$ and $R(\neg q \vee r) = [\gamma, \delta]$. Resolving these together gives $R(\neg p \vee r) = [\alpha \cap \gamma, \beta \cup \delta]$ and the result follows. □

e) Universal Instantiation

$R(\forall x_i P(x_i)) = R(P(a)) \wedge R(P(b)) \wedge \ldots \wedge R(P(n)) = [P(a)^C \cap P(b)^C \cap \ldots \cap P(n)^C, P(a)^E \cap P(b)^E \cap \ldots P(n)^E]$. Thus $P(a)^C \supseteq (\forall x_i P(x_i))^C$ and $P(a)^E \supseteq (\forall x_i P(x_i))^E$, so $R(P(a)) = [\alpha, U]$. □

THE THEORY OF DYNAMIC CONCEPTUAL MAPPINGS AND ITS SIGNIFICANCE FOR EDUCATION, COGNITIVE SCIENCE, AND ARTIFICIAL INTELLIGENCE

Vladimir A. Fomichov
Moscow State Institute of Electronics and Mathematics (Technical University)
Moscow State University, Bolshoj Vuzovsky pereulok, 3/12
109028 Moscow, Russia
root@onti.miem.msk.su (to V. Fomichov)

AND

Olga S. Fomichova
Moscow Children and Teenagers Palace for Creative Work
Russia

Keywords: artificial intelligence, cognitive science, theory of teaching, dynamic conceptual mapping, teaching children foreign languages, emotionally-imaginative teaching English, intelligent tutoring system, knowledge archives

Edited by: Anton P. Železnikar
Received: February 1, 1994 **Revised:** March 20, 1994 **Accepted:** March 23, 1994

Important advancements in the theory of teaching and practical teaching young children foreign languages are described and discussed in a broad context of finding the most effective ways of conveying information in various areas of human activity.
It is reported about the development of a new theory of teaching called the theory of dynamic conceptual mappings (the DCM-theory). The basic principles of the DCM-theory are set forth, and its composition is shortly described. The DCM-theory may be characterized as a theory of bridging gaps between conceptual systems of a teacher and a learner. It became the ground for creating new, highly effective methods of teaching young children and teenagers to read and communicate in English. These methods are called the methods of emotionally-imaginative teaching (the EIT-methods) and are based on ideas of artificial intelligence and cognitive science.

The new methods permitted to make a decisive breakthrough in solving the actual problem of diminishing the starting age for teaching children to read and communicate in a foreign language. The effectiveness of the DCM-theory and EIT-methods is proved, first of all, by considerable diminishing the starting age for teaching Russian-speaking young children to read and communicate in English. The new methods enabled to teach 5- and 6-year-old children to read and to discuss fluently complicated texts in English written in Simple Present Tense or Simple Past Tense. It is reported that the EIT-methods have been successfully tested in teaching approximately two hundred children of the age from 4 to 16 years. A number of ideas important for achieving this success is described.

The opportunities of applying obtained results to constructing intelligent tutoring systems with friendly interfaces are pointed out. The significance of results for education, cognitive psychology, cognitive linguistics, artificial intelligence, and applied epistemology is analyzed. In particular, a concrete recommendation concerning the realization of the Knowledge Archives Project is formulated.

1 Introduction

The stormy progress[1] in constructing computers has contributed very much to the stunning extension of possibilities to convey and to receive information of various kinds.

The most striking illustration of these new possibilities is the grandiose project of the Knowledge Archives launched in 1992 by three Japanese research institutes. The intention consists in developing a very large-scale knowledge informational system which is to become the most universal of all application systems. This future Knowledge Archives is to acquire, to store, and to unite diverse knowledge about sciences, technologies, cultures. Thus this system should become the most universal expert system (Knowledge, 1992; Zeleznikar, 1993a).

The Knowledge Archives Project is one of the most bright indications of the gradual transition in studies on artificial intelligence (AI) to a new orientation which may be called, according to Zeleznikar, A. P. (1993b), informational orientation.

The goal of conveying information is its effective perception by some recipients. That is why the transition to the informational orientation in studies on AI should imply the increase of attention to investigating the mechanisms of mastering new information by intelligent systems in order to find the most effective ways to convey information.

On the one hand, one feel the need of studies with the use of formal means aimed at modelling general regularities of conveying and perceiving information by intelligent systems (Zeleznikar, A. P., 1988, 1992, 1993c).

On the other hand, it is important to investigate experimentally the regularities of perceiving information by people.

Teaching may be considered as systematic conveying information from one person (persons) to other persons or person (learners or learner) with the aim to enable learners (a learner) to carry out some activities. These activities, in particular, may consist in further conveying information as in case of preparing future teachers of primary and secondary schools.

Hence a noticeable advancement in some specific area of education may cause an essential progress in solving diverse tasks of effective conveying information, if this advancement is achieved due to a more deep penetration into complicated cognitive mechanisms of human thinking.

It seems that just such a point of view is the most appropriate for considering the present article. Reporting about the development of a new cognitive theory of teaching and, as a consequence, of new, highly effective methods of teaching very young children foreign languages, this article expresses the ideas going far beyond the limits of school education and important for increasing the effectiveness of conveying information in diverse spheres of human activity.

As it is well known, for the success of teaching the learners should have some initial positive motivation. If this condition is satisfied, then the success of teaching strongly depends on the lack or the availability of difficulties in understanding teaching materials.

When a learner (especially a child) masters some new information easily, his or her interest in studies grows, and this contributes to achieving new successes in learning. On the contrary, great difficulties tied with the learning of some new information are usually the cause of the decrease (and even the disappearance) of initial positive motivation.

Numerous difficulties in studying school disciplines felt by many pupils are the sad reality of education in varied countries. Such difficulties strongly influence the psychology of pupils, diminish their self-respect, and contribute usually to the emergence of the psychology of a failure (Glasser, W., 1991; MacIntyre, P. D., & Gardner, R. C., 1991).

That's why the problem of finding methods making easier the learning of diverse disciplines has highly great social significance and attaches the attention of many researchers.

Obviously, fundamental problems of teaching

Correspondence to: V. A. Fomichov, Division "Math. Provision of Information-Processing and Control Systems", Moscow State Institute of Electronics and Mathematics, Bolshoy Vuzovsky pereulok, 3/12, 109028 Moscow, Russia. Tel.: +7(095) 930 98 97. E-mail: root@onti.miem.msk.su (to Prof. V. Fomichov). Fax: +7(095) 227 28 07 (to Prof. V. Fomichov).

are to be solved with respect to the progress in cognitive science. Pitrat, J. (1992) argues the experience of using the results of the artificial intelligence theory for carrying out studies in cognitive science. He supposes that the study of AI will help to look at the natural intelligence from a new angle.

Continuing this thought, one can hope that taking into account the achievements of the AI-theory may contribute to the progress in the theory of teaching.

A similar opinion is expressed also by Schank, R. and Slade, S. (1991, p. 106) believing that "the study of learning within the framework of artificial intelligence will have a significant impact on science, technology and education".

Our experience confirms the fruitfulness of this idea. Achievements of the theory and practice of natural-language-processing systems (NLPSs) together with the data of cognitive psychology and cognitive linguistics allowed us to look from a new side at the problems of teaching. As a result, a new theory of teaching has been developed called the Theory of Dynamic Conceptual Mappings (the DCM-theory). Many basic ideas of this theory are stated in Fomichov, V. A., & Fomichova, O. S. (1993).

In Section 2 the interrelations of teaching and of regularities of natural language understanding are analyzed. Section 3 shows the actuality of diminishing the starting age for studying foreign languages by children. In Section 4 main principles and composition of the DCM-theory are set forth. In Section 5 the use of dynamic conceptual mappings for explaining the rules of reading English letters to very young children is illustrated. Section 6 describes the successful results of using the methods of emotionally-imaginative teaching elaborated on the basis of the DCM-theory at lessons of English destined for children. First of all, the elaborated methods permitted to teach 5- and 6-year-old children to read and to discuss complicated texts in English in Simple Present Tense or Simple Past Tense. In the most countries and in the most schools children are able to do this being 12 or 13 years old.

In Section 7 the significance of the DCM-theory for education is analyzed. Section 8 is devoted to describing the significance of obtained results for cognitive science, artificial intelligence, and applied epistemology. In particular, large possibilities of designing new intelligent tutoring systems are pointed out.

2 Teaching and Regularities of Natural Language Understanding

2.1 The role of natural language in teaching

It is clear that a good carpenter is able to teach somebody his handicraft using a few of words or even without words.

However, natural language (NL) is the chief instrument of teaching diverse theoretical disciplines. Although the set of teaching materials may include photographs, diagrams, objects like patterns of rocks, artificial constructions like models of molecules, etc., the destination and meaning of these materials, their interconnections are explained by means of NL.

It is widely accepted that a text-book may be written by means of a "good language" or a "bad language". In the first case we mean that the materials of the text-book may be easily understood, and in the second case it is difficult for learners to master the information provided by the text-book.

That's why it seems that theories of teaching should take into account the knowledge about the regularities of natural language understanding given by the modern science, and that a more deep attention to such regularities may contribute to the progress in education.

2.2 The role of knowledge about reality in natural language understanding

The regularities of NL-understanding were studied intensively during the seventies and the eighties by specialists on constructing natural-language-processing systems (NLPSs), on cognitive psychology and cognitive linguistics.

As concerns NL-understanding by people, a highly important role in building mental representations of NL-texts is played by diverse cognitive models accumulated by people during the life: semantic frames, explanations of meanings

of notions, prototypical scenarios, social stereotypes, representations of general regularities and area-specific regularities, and other knowledge of varied sorts determining, in particular, the use of metaphors and the creation of commitments for mutual orientation of dialogue participants (Johnson-Laird, P., 1983; Fauconnier, G., 1985; Fillmore, C., 1985; Seuren, P.A.M., 1985; Winograd, T. & Flores, F., 1986; Johnson, M., 1987; Lakoff, G., 1987; Chernov, G.V., 1987; Fomichova, O.S., 1989, 1990).

The experience of constructing NLPSs allows us to draw a similar conclusion about the role of knowledge about reality in NL-understanding. Nowadays it is widely accepted that a NLPS should contain a knowledge base and use diverse knowledge about reality for the processing of NL-texts. This may be necessary for finding the referents of personal and possessive pronouns in discourses, for reconstructing the meaning of incomplete (elliptical) phrases being fragments of discourses, for picking out one of several meanings of a word, for understanding metaphors and syntactically irregular phrases, etc. The role of knowledge about the world is especially high in interpreting the oral speech, since such a knowledge helps to divide the utterances into the fragments corresponding to lexical items.

The mentioned ideas were the starting point for creating the DCM-theory and methods of emotionally-imaginative teaching. Besides, an important role was played by the approach of Pavilionis, R.J. (1985) to interpreting the mechanisms of NL-understanding. This approach belongs to the philosophy of language.

The central place in the approach of Pavilionis occupies the notion of a conceptual system (CS) of a person. This is a system of interrelated information fragments reflecting the cognitive experience of an individual on different levels (including preverbal and non-verbal ones) and as concerns different aspects of cognition. The most abstract concepts of such systems are tied continuously with the concepts reflecting our every-day experience. Each subject of cognition has a CS including the knowledge and beliefs about real and possible states of affairs in the world.

Every language expression is built in the framework of some CS. Analyzing texts, a recipient interprets them forming the conceptual pictures and using the available information. It should be taken into account that a recipient is able to assign meanings only to such texts which can be "inscribed" into his or her CS.

It should be added that the approach of Pavilionis develops many ideas of the theory of speech acts proposed in the fifties–sixties by J. L. Austin, P. F. Strawson, and J. R. Searle.

3 The Problem of Diminishing the Starting Age for Teaching Children Foreign Languages

In the modern world, the knowledge of foreign languages (FLs) is one of the important preconditions of the successful work in many areas of human activity. Nevertheless, studying FLs in schools is difficult for many pupils in diverse countries. That's why one of the actual problems of the theory of teaching is the development of methods making easier learning FLs by children.

As it is noted in Rixon, S. (1992), fifteen years ago and until recently the dominant approach to teaching children FLs in schools was as follows: FLs were studied in secondary schools beginning with the age 11 or more years.

There are data showing the benefits of learning a FL in primary schools (see, in particular, Dabène, L. (1991)).

First, children extend the time for attaining fluency and competence in a FL or FLs. Second, children have less disciplines to study and are able (if it is interesting for them) to give more efforts and time to learning a FL.

That is why the problem of "early" language learning has been emerged in recent years and is being considered as a highly actual one in many countries of Europe, in U.S.A. and Canada. This problem is treated in different manners in diverse countries.

In the context of the French educational structure, "early" language learning is starting to study a language one or two years before the beginning of secondary education (Dabène, L., 1991). In several other countries of Western Europe, during a number of years the initial ages for learning English have been as follows: The Netherlands – 10 years (since 1985), Denmark – 9 years, Austria – 8 years (Rixon, S., 1992).

In such countries as Spain, France, Italy, Hungary, Czechia, Slovakia, Poland, Bulgaria, experimental projects have been realized aimed at teaching all children of primary age a FL. The starting age in Israel is 10 years and the possibility exists to begin learning a FL being 9 years old in many schools (Rixon, S., 1992).

In Russia in the most schools, pupils learn a FL beginning with the 6th grade being 11 or more years old. In big cities there are also the so called specialized schools, where the starting age for learning a FL is 7–8 years.

One can observe a considerable interest to diminishing the initial age of learning a FL in U.S.A. In 1990, the programmes on studying a FL were offered by 22 % of elementary schools in U.S.A. (Met, M., & Rhodes, N., 1990; Arendt, J. D., & Warriner-Burke, H. P., 1990).

However, the problem of teaching all children a FL in primary schools is not easy. It is felt the lack of adequate text-books, curricula, and well-trained teachers. Especially difficult is to teach a child to master the rules of reading and the grammar of a FL because a child doesn't know sufficiently well even the grammar of the native language.

Likely, due to these main reasons the starting age for teaching English in Germany in 1991 stayed 10–11 years, and the starting grade was the first grade of secondary education (Brusch, W., 1991).

Thus, in many countries of Europe, in U.S.A., and in some other countries the problem of creating new methods of teaching FLs enabling to diminish the starting age for learning FLs and making easier and more interesting learning by children a FL is being considered as a highly actual one.

4 The Theory of Dynamic Conceptual Mappings: Main Principles and Composition

4.1 General characterization

The Theory of Dynamic Conceptual Mappings (the DCM-theory) is a new theory of teaching and may be shortly characterized as a theory of bridging gaps between the conceptual systems (CSs) of a teacher and a learner.

The central notion of the new theory of teaching is the notion of a *dynamic conceptual mapping* (DCM). We'll say about a mapping of the kind when a useful (from the standpoint of goals of teaching) correspondence is established during a lesson or in the course of reading a text-book between components of some fragment F1 and components of some other fragment F2 of the inner (or mental) world's picture of a learner.

The following three types of mappings are considered in the theory as playing chief roles in teaching.

First, mappings transforming some fragment of a material to be studied into more general mental representation (MR).

Second, mappings transforming some fragment of the world's picture of a learner into more general MR.

Third, isomorphic correspondences between generalized MR of a fragment of the inner world's picture of a learner and a generalized MR of a knowledge portion to be learned at a lesson.

Thus, the DCM-theory pays a peculiar attention to the use of analogy in teaching.

The central component of the DCM-theory is the description of a number of conceptual mappings realized for learning English.

The following distinctive features of the DCM-theory should be underlined: (a) a great number of invented useful analogies; (b) the high complexity and originality of many invented analogies; (c) the unusual small age of learners successfully understanding these analogies: the children are 5 or 6 (and even in many cases 4.5) years old.

4.2 Main principles

The DCM-theory is based on the following chief ideas:

1. On condition that the learners have some initial positive motivation to study a discipline, in developing a stable, strong positive motivation of learners a decisive role is played by regular repeating the feeling of success.

2. The collection of methods used by a specialist to teach some discipline may be interpreted as an algorithm of enriching conceptual systems, or inner world's pictures, of people by means of establishing dynamic cor-

respondences between fragments of the inner world's pictures of a teacher and a learner.

3. The opportunity to establish such dynamic correspondences is afforded by the existence of some common part of teacher's and learner's CSs.

   This common part is composed by mental representations of every-day situations, pictures of the nature, etc. The size of such a common part of knowledge may considerably vary and this greatly influences the results of teaching. Huge gaps between a CS of a teacher and a CS of a learner are the chief reason of difficulties in studying diverse disciplines.

4. If differences between CSs of a teacher and a learner are rather great (for instance, in the situation of studying the grammar of a foreign language by a child), new fragments of information should be introduced through special "channels" being the most close to the life experience and knowledge of a learner in order to be effectively understood (i.e., "inscribed" into a CS of a learner).

   The number of such special "channels" for introducing new difficult information may considerably vary for learners of diverse ages, professions, and cultures.

   In particular, young children have usually only the following channels: (a) fairy-tales and tales based on well known postulates of behavior; (b) situations of every-day life (including games).

5. Let's agree that the term "a teacher" may designate a school teacher, or a professor at a university, or an author of a text-book, etc. Then the process of entering new portions of information through special "channels" may be explained as follows.

   In such situations when the existence of a considerable gap between CSs of a teacher and a learner is obvious as concerns introducing new information, a teacher is to go beyond the set of notions used traditionally for explaining the information of the kind and is to invent (preferably, preliminarily) some new ways of explaining the material to be studied.

   Searching these new ways, a teacher is to take into account the knowledge about the *full* conceptual system of learner.

   A teacher must try to find in the learner's conceptual picture of the world such fragments which reflect situation similar (or isomorphic) in some generalized sense to situations taking place in teaching materials to be mastered by the learner.

   As a rule, this is possible to do. If such similar situations are discovered, a teacher must invent the ways understandable for the learner (the learners) and enabling to establish the correspondences between each such generalized situation and the situation expressed by the teaching material.

   Then a teacher should select such an analogy which seems to be an optimal one from the standpoint of introducing a new fragment of information pertaining to the studied discipline.

   The realization of this principle is explained in detail in the next section.

6. The inner content of the teaching process as a systematic directed realization of dynamic conceptual mappings determines the form of teaching from the point of view of people who are present at lessons.

   This form may be characterized as emotionally-imaginative teaching (this applies both to children and to adults).

   The main distinguished feature of this form is creating the feeling of success due to the use of numerous comparisons of studied theoretical situations with such situations which are well known to the learners.

   E.g., for children such a role is played by fairy-tales and games, for programmers—by algorithms and structured diagrams.

7. The students of higher educational establishments intending to work in the future as teachers are to master the methods of inventing effective mappings between CSs of a teacher and a learner. Besides the deep knowledge of special disciplines, this is the chief prerequisite of the successful work of a teacher.

That's why it appears to be expedient to use for teaching such students special expressive means permitting to represent visually the fragments of conceptual systems of a teacher and a learner and correspondences between fragments of CSs.

In cases when the lack of such means is felt, it is expedient to develop new effective means of visual representing the fragments of conceptual systems and the correspondences between such fragments.

4.3 The composition of the DCM-theory

The DCM-theory includes the following chief components:

1. Main principles.
2. Theoretical basis of highly effective teaching very young children (4–6 years old) the rules of reading English words.
3. Theoretical basis of highly effective teaching very young children (4–6 years old) chief constructions of English grammar (Simple Present Tense, Simple Past Tense, general and special questions).
4. Theoretical basis of improving the skill of young children and teenagers to read and communicate in English (see, in particular, Fomichov, V. A., & Fomichova, O. S., 1993). This basis is mainly transportable as concerns teaching other FLs.
5. Theoretical basis of teaching Russian-speaking students and post-graduates to find implicit assessments of facts and hypotheses in papers in English on science and technology while translating or abstracting these papers (Fomichova, O. S., 1989, 1990; Fomichov, V. A., & Fomichova, O. S., 1993). This basis is transportable as concerns the other pairs of languages besides the English and the Russian.
6. New ways of visual representing diverse cognitive structures in order to teach students of higher institutions of learning to establish effective dynamic mappings between CSs of a teacher and a learner.

   The mentioned cognitive structures may be semantic representations of sentences and discourses (if it is needed, generalized representations), formal descriptions of notions, and other fragments of knowledge.

The ground for these new ways is provided by Integral Formal Semantics (IFS)—a new, very powerful and flexible approach to the formalization of NL-semantics and NL-pragmatics. The basic ideas of IFS are described, in particular, in Fomitchov, V. A. (1984), Fomichov, V. (1988, 1992, 1993a, 1993b, 1994).

The main new visual means of the DCM-theory as concerns describing conceptual structures and conceptual mappings are based on the theory of K-calculuses, algebraic systems of conceptual syntax, and K-languages (see Fomichov, V., 1992) being the central constituent of IFS. These means are the strings of standard K-languages and oriented labeled graphs (K-graphs) equivalent to such strings. K-graphs provide a number of important advantages from the standpoint of representing semantic structure of sentences and discourses and representing knowledge blocks in comparison with traditional semantic nets and with conceptual graphs of Sowa, J. F. (1984). It should be noted that the expressive possibilities of standard K-languages are close to the expressive possibilities of NL.

The task of describing these new ways of visual representing cognitive structures goes far beyond the scope of the present paper and will be the subject of future research work.

5 About Dynamic Conceptual Mappings for Teaching Very Young Children the Rules of Reading English Words

Teaching very young, non-English-speaking children (4–6 years old) to read English words is a complicated problem. The principal difficulty consists in explaining why some letters or combinations of letters correspond to different sounds in some situations. Besides, most often the children of this age do not read quite well in a native language.

The impossibility to overcome the mentioned difficulty is one of the main reasons of the situa-

tion when all text-books on English published by 1992 in United Kingdom and destined for non-English-speaking children under seven years are oriented to the prereading stage of learning English (Rixon, S., 1992).

Let's illustrate the approach of the DCM-theory to solving this problem in teaching Russian-speaking young children. Consider the way of explaining the rule of reading the letter "Y" proposed by the DCM-theory. The discussed difficulty consists in explaining in a manner understandable for 4–6-year-old children why different letters "I" and "Y" denote the same sound in the words "time", "ice", "cry", "fly", etc. (suppose that children know already the rule of reading the letter "I").

Taking into account the age of learners, to find an appropriate approach to this problem is not easily.

The DCM-theory suggests the following solution proved to be highly effective.

A possible mental representation (MR) of the knowledge portion which is to be understood by children may be depicted visually by the semantic net on the Figure 1. Let's mark this MR by the expression 1-T, where the symbol "T" corresponds to the word "a teacher". The MR 1-T is a fragment of the conceptual system (CS) of a teacher.

One can formulate the following generalization of the discussed situation: "The letters X1 and X2 are not alike, but on some conditions the letter X2 may sound as the letter X1."

Hence it will be quite natural for a teacher to transform the MR 1-T into a more general MR corresponding to the meaning "The entities X1 and X2 are not alike, but on some conditions the entity X2 possesses a property coinciding with some property of the entity X1". When such a transformation is realized, we start to recall the fairy-tales and games where the same generalized situation may be discovered. We map the fairy-tale and game situations into diverse generalized MRs.

The fairy-tale "The Wolf and the Seven Little Kids" is widely known. That's why we can discover quickly that this fairy-tale includes a situation similar to the generalized MR of a situation relating to the pronunciation of letters "I" and "Y".

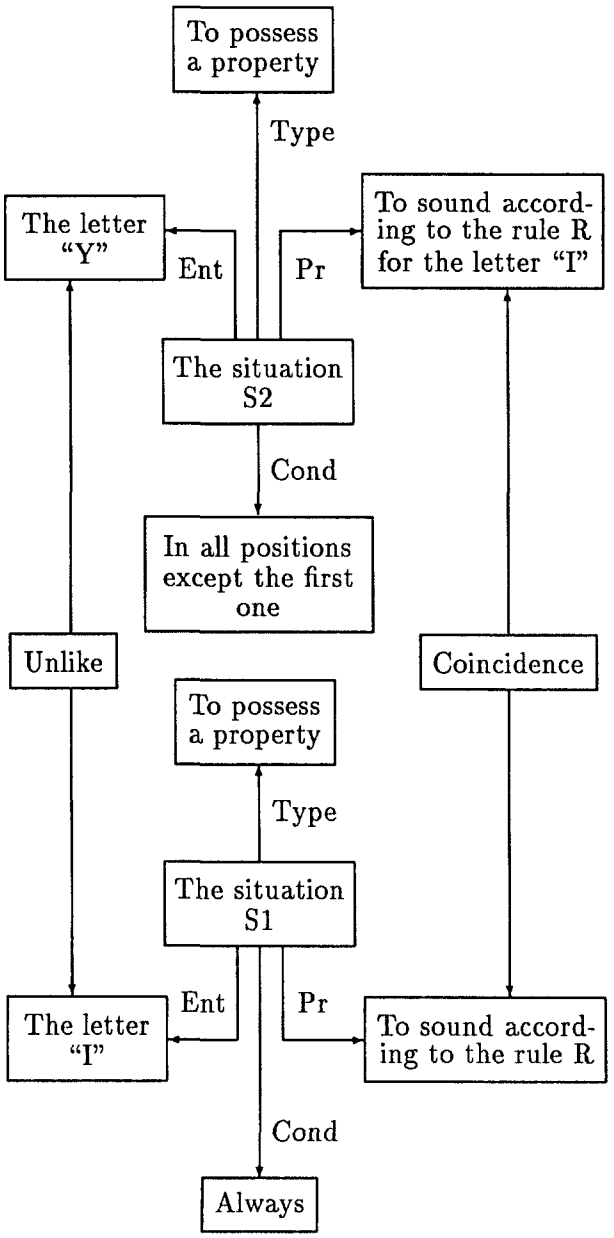


Figure 1: The semantic net depicting a fragment 1-T of the conceptual system of a teacher; "Ent" means "Entity", "Pr" means "Property", and "Cond" means "Condition".

The Figure 2 depicts visually in the form of a semantic net a possible MR of one of situations described in the considered fairy-tale. Recalling this fairy-tale is a success: it is clear for us that an isomorphic correspondence can be established between MRs 1-T and 2-T.

The ground for establishing such a useful correspondence is given by the mapping assigning to the Wolf-Deceiver the letter "Y" and to the Mummy Goat the letter "I". Let's say that the mappings of the kind are analogy-forming conceptual mappings.

The mentioned mapping determines a more large conceptual mapping assigning to the elements of the fragment 2-T the elements of the fragment 1-T. A a consequence, a correspondence is established between MRs 2-T and 1-T.

A teacher knows that the CS of a child includes a fragment 2-L identical to the fragment 2-T of the teacher's CS. Hence a teacher should contribute to creating in the CS of each learner a fragment 1-L identical to the fragment 1-T . In order to do this, a teacher invents some thrilling form (preferably, a fairy-tale-like form) adequate to the CS of a learner.

As a result, a correspondence between fragments 2-L and 1-L will be established similar to the correspondence between fragments 2-T and 1-T. Due to this correspondence, a child will be able to understand and remember the rule of reading the letter "Y".

For example, a teacher may tell the following story:

The letter "Y" wanted to be alike the letter "I". I suppose that the letter "Y" liked the cap of the letter "I" very much. But "Y" quite understood that she wasn't alike "I".

However, she remembered a fairy-tale "The Wolf and the Seven Little Kids". It was clear for the Wolf that he didn't resemble the Mummy Goat and the only way out was to change his voice. So he did, and Seven Little Kids confused him with their Mummy Goat.

So did the letter "Y". She started to sound like the letter "I" in all positions except one. In the beginning of the word the letter "Y" sounds like herself [j], but her voice begins to sound not so lovely because she always tries to resemble "I" and forgets about her own voice.

Be careful, children, try not to resemble some-

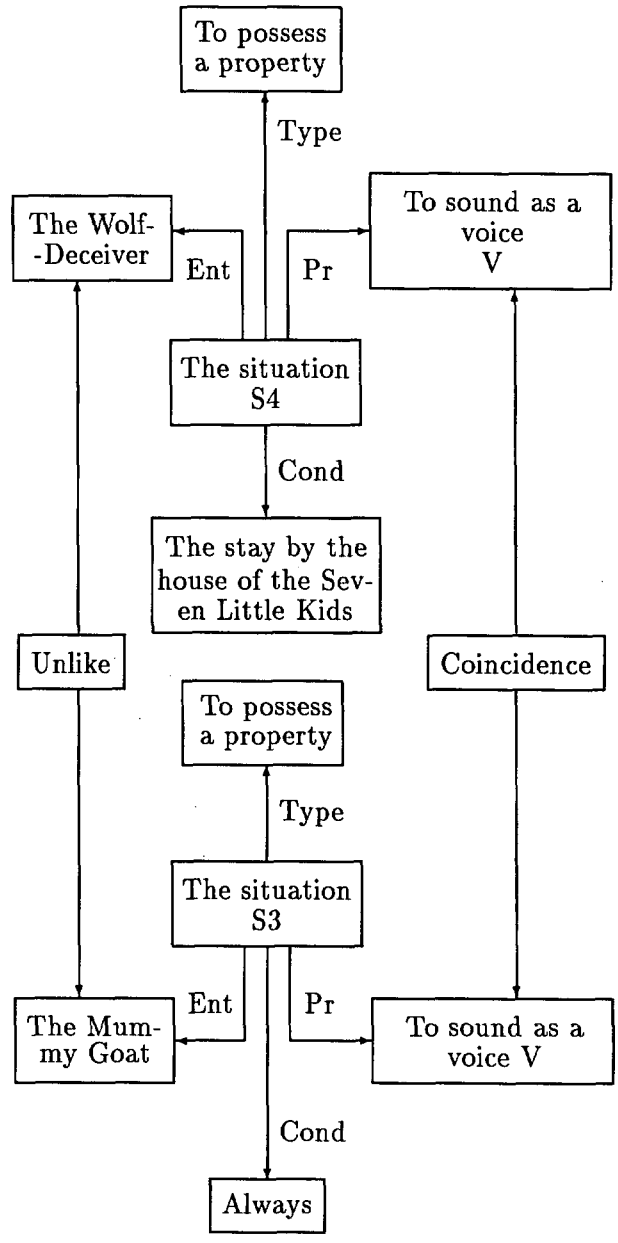


Figure 2: The semantic net visually representing a fragment 2-T of the conceptual system of a teacher; "Ent" means "Entity", "Pr" means "Property", and "Cond" means "Condition".

body but yourself. And in this situation I'm sure your voice will be the most beautiful.

The experience shows (see the next section) that children listen to this story with great delight and discuss it cheerfully. They identify the letters "I" and "Y" with the Mummy Goat and the Wolf-Deceiver, respectively. As a consequence, they remember easily the rule of reading the letter "Y".

A number of other dynamic conceptual mappings is invented for effective teaching young children to read English words.

6 The Methods of Emotionally-imaginative Teaching English

The ideas of the DCM-theory enabled us, in particular, to develop new, highly effective methods of teaching English called the methods of emotionally-imaginative teaching (the EIT-methods). These methods have been successfully tested during four years in teaching approximately two hundred young children and teenagers (4–16 years old) in experimental groups in the Moscow Children and Teenagers Palace for Creative Work on Vorob'yovy Hills by Doctor of Philology Olga Fomichova. Naturally, the EIT-methods are differentiated in accordance with the age of children. Shortly the results may be described as follows.

The starting age for studying English is five, or six, or in many cases four-and-half years. Children of one group may be of different ages.

After the first year of studying English in experimental groups 5- or 6-year-old children know theoretically Simple Present Tense and Simple Past Tense, can use these tenses in dialogues, know special and general questions.

For instance, children are able to read and to discuss quite well the fairy-tale about Cinderella in Simple Past Tense containing approximately 300 words (see the Appendix 1).

Children also like to compose their own fairy-tales (see the Appendix 2).

The EIT-methods have permitted to make a discovery in teaching young children foreign languages: it has been turned out that the prereading stage is not necessary for children under 7 years (despite of the widely accepted opinion expressed in Rixon, S., 1992). Really at lessons of O. S. Fomichova children begin to read after four lessons being acquainted with all English letters.

Children who have been studying English during two years compose their own scripts of the Christmas party in English.

During the first three months of the third year of studies children read and discuss unadapted edition of "Alice's Adventures in Wonderland", speak fluently on topics of every-day life.

Thus, new methods of teaching give a growing from year to year educational success. One of interesting results is as follows: the children begin to read in native (Russian) language.

It should be added that: (1) children don't visit English-speaking countries; (2) the duration of lessons twice a week is only 45 minutes (each group consists of twelve children); (3) children of the first year don't attend schools and have no experience of the work at lessons.

The EIT-methods provide the possibility not only to teach effectively English but also to influence positively the development of the personality of a learner. The new methods develop creative capabilities of children, the feeling of success, the capability to master actively the knowledge (in particular, to work independently with a text-book and with other teaching materials), the confidence of own forces. All children of very different ages (from 4 to 16 years) go to lessons of English with a great joy.

After 5 months of the third year of studies children are able to describe pictures of the nature, their feelings relating to the nature, and, as a consequence, to describe landscapes. They can not only penetrate the vision of artists but also describe their feelings in a very poetical way.

Consider the examples of such descriptions.

Example 1. "I see the beautiful winter landscape. The sky is deepest blue, as blue, as the water of the sea. The snow like carpets covers the ground and is gleaming in sunshine. Far away deep woods are lucent and serene. The forest stream dreams under the ice, which gleams with silver. I see a birch in the middle of the glade near the road leading down to the river. The air is frosty and lucent." (Andrey Todua, the 3rd year of studies, 9 years old.)

Example 2. "Stars twinkle and glimmer; they

flame in the sky, The moon gets yellow in the heaven's high. The wood is soft-murmuring in the gloom of the night, And everything dreams in the magic light." (Natasha Loseva, the 3rd year of studies, 8 years old.)

Example 3. "The sky is aglow with the stars And I'm sitting on the grass. The moon is yellow as brass And cats are singing in the cars." (Katya Safonova, the 3rd year of studies, 11 years old.)

It was their home task, but children are able to speak about landscapes and portraits just on the spot, without a preliminary work. They can not only feel but also express their feelings looking at the picture.

An original way of achieving such a successful result of teaching English is explained in Fomichov, V. A., & Fomichova, O. S. (1993). This way is based on realizing in the course of teaching DCMs of kinds not considered in the present paper.

Thus, lessons of English permit not only to learn a FL but also to bring up a child to love the nature.

A part of EIT-methods is destined for working with teenagers being 12–16 years old. The elaborated methods enable not only to teach English but also to teach teenagers to understand other people (in particular, to understand their parents), to understand how other people see them; to develop the feeling of beauty, the feeling of success, the feeling of a leader, and the interest to studies.

As experience shows convincingly, all this soothes considerably the difficulties of the transitional age, positively influences the interrelations between teenagers and their parents, between one teenager and the other teenagers, contributes to the future successes in studies.

The successes of young children have enabled the second author to teach them during the 4th year of studies not only English, but also German as a second FL. For teaching German two languages are being used: English (as the main language) and Russian. The progress of children (8–10-year-old) in German is very quick because they have a good linguistic basis.

7 The Significance of the Theory of Dynamic Conceptual Mappings for Education

7.1 Teaching very young children foreign languages

The obtained results show that a decisive breakthrough is made in diminishing the starting age for teaching children to read and communicate in English. In the most countries and in the most schools the starting age for learning a foreign language (FL) is 11 years. Children studying in experimental groups read quite well and discuss complicated texts in Simple Present Tense or Simple Past Tense being five or six years old.

Hence the EIT-methods have afforded the opportunity to gain at least 5 years or even 6 years in learning English.

The same methods may be used, for example, for teaching French-, German-, Dutch-, Greek-, Italian-, Spanish-speaking children to read and communicate in English.

The developed new effective methods may be used not only for teaching English, but also, after some adaptation and transformation, for teaching a number of other languages.

7.2 Teaching English-speaking young children to read fluently in native language

There are weighty grounds to believe that the elaborated EIT-methods may be successfully applied to the speeded-up (in comparison with traditional methods of primary school) teaching English-speaking children (the age from 4 to 6 years) to read fluently in native language and to express their thoughts.

The accumulated data allow us to conjecture that this will positively influence the development of the personality of a child and create excellent preconditions for the successful study in primary school.

7.3 Teaching children mathematics and other disciplines

The ideas of the DCM-theory are helpful not only for teaching FLs or native language. On

the basis of the DCM-theory, one of the authors of this paper developed methods of emotionally-imaginative teaching young children mathematics and tested these methods in two experimental groups of a primary school. One group consisted of 6-year-old children, the other—of 7-year-old children.

Due to invented dynamic conceptual mappings, children mastered easily, with joy, very quickly the mathematical notions relating to arithmetics, algebra, and set theory. According to traditional methods, one learn some of these notions being much older (in particular, 4 or 5 years older).

It seems that the successful experience of applying the ideas of the DCM-theory to teaching such distinct disciplines as foreign language and mathematics indicates large opportunities of using the DCM-theory in teaching diverse disciplines.

7.4 The significance of obtained results for education as a whole

The EIT-methods elaborated in Moscow have permitted to achieve an extremely great progress in solving the actual problem of diminishing the starting age of learning English. The basis for this progress is provided by the DCM-theory.

Numerous invented conceptual mappings realized by means of both known and specially composed fairy-tales have permitted to find a key to the rich inner world of children, to "switch on" their vivid fantasy in order to create the preconditions for understanding complicated rules of reading and English grammar.

It should be mentioned that the complexity of DCMs invented for explaining the use of Simple Present Tense and Simple Past Tense is higher than in case of explaining the rule of reading the letter "Y". Nevertheless, 5- and 6-year-old children master quite easily the use of these tenses due to the thrilling form of explaining grammatical constructions.

The significance of the DCM-theory and EIT-methods goes far beyond the limits of teaching children FLs (although this direction of applying results is highly actual).

In fact, it is discovered that the educational potential of children is very high, much higher than it is widely accepted to suppose. Children, and even very young children, may master complicated teaching materials quickly and (what is the most important) with joy, without excessive straining their forces, without the damage for own health but with great benefit for the development of the own personality.

That's why it appears that the DCM-theory and EIT-methods create the prerequisites for the emergence of a new wave of investigations in education with the following aims: (a) to find and distinguish such fragments of knowledge described in school text-books and pertaining to diverse disciplines which are the most difficult to understand for pupils; (b) taking into account peculiarities of conceptual systems of children from varied age groups and countries, to invent special DCMs for overcoming such difficulties in learning school disciplines; (c) to write new experimental text-books and to develop new syllabuses for using in the course of teaching special DCMs making easier studying various disciplines; (d) to apply these new experimental text-books and syllabuses to teaching children in schools, to correct (if it is necessary) these new text-books and syllabuses, and to elaborate text-books and syllabuses for the wide use.

Our results allow us to hope that very many pupils and, as a consequence, their parents in varied countries will benefit from investigations of the kind.

7.5 The visual means of the Artificial Intelligence Theory and teachers preparation

Basic ideas of the artificial intelligence (AI) theory stimulated the birth of the DCM-theory. The example in Section 5 shows that the use of semantic nets provides the possibility to represent highly visually the correspondences between fragments of people's conceptual systems interpreted as DCMs.

That's why one can draw the conclusion that it will be worth while to acquaint students intending to work in the future as teachers with basic ideas of the AI-theory and with the visual means of representing semantic structures of NL-texts, structures of conceptual memory, knowledge blocks (modules).

As it is widely known, the inventory of such means includes, in particular, semantic nets, frame-like knowledge representation languages, conceptual graphs of Sowa, J. F. (1984).

Besides, very large opportunities to build semantic representations of sentences and discourses and to describe knowledge blocks are provided by strings of standard K-languages (see Fomichov, V., 1992, Sections 6–9) and by oriented labeled graphs (K-graphs) equivalent to such strings.

The study of visual means of the AI-theory by the future teachers should be subordinated to the task of learning how one can invent DCMs making essentially easier the understanding of difficult knowledge portions by children. It may be presumed that the use of visual means of the AI-theory for representing diverse cognitive structures will be of considerable benefit from the standpoint of teaching how to invent effective DCMs.

8 The Significance of Obtained Results for Cognitive Science, Artificial Intelligence, and Applied Epistemology

8.1 The significance for Cognitive Psychology

The success achieved in teaching very young children to read and communicate in English on the basis of using numerous fairy-tale-like analogies shows that children have a great cognitive potential, which has been often underestimated before. Hence the obtained results in teaching children English may be interpreted as raising a new voluminous task before cognitive psychologists: the task of investigating (jointly with teachers) the manners and effectiveness of using diverse dynamic conceptual mappings for teaching children (possessing diverse capabilities and belonging to diverse age groups) various disciplines.

It seems that such investigations may considerably contribute to solving the problem of elaborating new, unified theories of cognition posed by Newell, A. (1990).

8.2 The significance for Cognitive Linguistics

During the second half of the eighties in linguistics the formation of a new branch was completed called cognitive linguistics. Cognitive linguists consider the language "as an instrument for organizing, processing, and conveying information... The formal structures of languages are studied not as if they were autonomous, but as reflections of general conceptual organization, categorization principles, processing mechanisms, and experiential and environmental influences" (Geeraerts, D., 1990, p. 1).

The successful results of emotionally-imaginative teaching young children English give a new powerful confirmation of the truth of at least one basic idea of cognitive linguistics.

This is the idea that there exists no syntax as an autonomous subsystem of language system. Syntax should depend on descriptions of cognitive structures, on semantics of NL.

NL-understanding by people doesn't include the phase of constructing the pure syntactic representations of texts. The transition from a NL-text to its mental model is carried out on the basis of various knowledge and is of integral character (Johnson-Laird, P., 1983; Seuren, P. A. M., 1985; Lakoff, G., 1987; Chernov, G. V., 1987; Caron, J., 1989; Langacker, R. W., 1990).

Children taught according to the EIT-methods do not study the syntactic structures of English phrases, the use of Simple Present Tense and Simple Past tense in a manner widely accepted in schools (most often, in secondary schools). These rules are successfully introduced due to invented DCMs based on specially composed fairy-tales and on every-day experience of children. This is the main reason of unusual effectiveness of the DCM-theory and EIT-methods as concerns teaching children English as a FL.

There are grounds to believe that a more detailed analysis of the DCM-theory and EIT-methods may allow us to establish a number of additional interrelations with the ideas of cognitive linguistics.

With respect to obtained results, it may be supposed that cognitive linguists are able to contribute greatly to making easier the study of foreign and native languages at schools.

8.3 Education, the Artificial Intelligence Theory, and Applied Epistemology

Taking into account our experience, we believe that the specialists on AI and on applied episte-

mology may make an important contribution to solving a number of actual problems of education. Let's restrict now to the following two aspects.

First, they may acquaint students—future teachers with the existing visual means for representing diverse cognitive structures (see also Subsection 7.5).

Second, such specialists dealing during many years with semantic representations of NL-texts, structures of conceptual memory, knowledge blocks, etc. may use their professional knowledge for inventing effective dynamic conceptual mappings destined for studying diverse disciplines.

8.4 The design of new Intelligent Tutoring Systems

In the end of the eighties the principles of a new paradigm for constructing intelligent tutoring systems (ITSs) were formulated by Self, J. (1988) and were realized in the works of a number of researchers.

The distinguished features of the new paradigm are the help to a learner in carrying out an independent research work, in enriching the knowledge and the skill as concerns studied disciplines, and in developing general creative capabilities of learners. For this, amicable (friendly) interfaces of ITSs are to be built.

Johnson, W. L. (1991, p. 129) notes that "a key concern of current tutoring systems is how to relate the expert knowledge that the student will acquire to the commonsense knowledge that he or she already knows".

That's why the new paradigm for building ITSs excellently harmonizes with the key ideas of the DCM-theory and elaborated EIT-methods. These methods are based on (a) numerous invented conceptual mappings permitting to compare studied fragments of a discipline with situations well known to learners, (b) special kinds of tasks developing the creative potential of learners.

The accumulated rich teaching materials can be successfully used for constructing ITSs destined to master (a) English as a FL by children or by adults, (b) the rules of reading and writing in English by English-speaking young children.

In Fomichov, V. A., & Fomichova, O. S. (1993) the idea is stated about working out some special sorts of ITSs for learning FLs. Such ITSs are to have an amicable NL-interface aiding to study the language means for describing the nature, the feelings evoked by nature. The means of the kind are necessary both for carrying out various activities on the nature and for understanding lectures on art, discussing landscapes.

Additionally, the DCM-theory and EIT-methods may stimulate the elaboration of ITSs in accordance with the modern paradigm for learning other FLs, besides English, and other disciplines, besides foreign language (for example, mathematics).

The obtained experience of highly effective teaching very young children to read and communicate in English as a FL enables us to formulate a simple, but important recommendation concerning such an unsimilar, at first sight, scientific problem as the realization of the Knowledge Archives project.

As it is mentioned in the introduction, one of the tasks to be solved by the Knowledge Archives is to unite the knowledge about diverse cultures.

The difference between two cultures may be very considerable, not less than the difference between conceptual systems of a young child and of adults belonging to one culture.

That's why we propose to develop and to include into the Knowledge Archives special intelligent interfaces (they may be called culture interfaces) enabling the carriers of one culture to perceive effectively the peculiarities of studied another culture, to overcome gaps between the native and studied cultures.

For instance, some special electronic course describing the every-day life and even the meaning of diverse gestures in Russia of the 19th century might make easier for a Japanese or an Indian the understanding of the Russian literature of this period. On the contrary, practical ignoring the existence of huge gaps between many pairs of cultures will highly diminish the effectiveness of interchanging information between such cultures.

9 Conclusions

The methods of emotionally-imaginative teaching based on the theory of dynamic conceptual mappings have permitted to discover that the educational potential of very young children (4–6-year-old) is very high, considerably higher than it is widely accepted to believe. The results de-

scribed above open many new important possibilities for the joint effective research work of teachers, university specialists on education, cognitive psychologists, cognitive linguists, specialists on AI and applied epistemology. As a consequence, this promises to enlarge essentially the opportunities of children to learn with joy.

Besides, we expect that the ideas expressed above will be of considerable use for building more effective Intelligent Tutoring Systems destined for adults and for the realization of the Knowledge Archives project.

APPENDIX 1

The full text of the fairy-tale which is fluently read and discussed by all 5- and 6-year-old Russian children after the first year of studying English in experimental groups of O. Fomichova in the Moscow Children and Teenagers Palace for Creative Work on Vorob'yovy Hills.

CINDERELLA

Long ago, in a land far away, there lived a man. He had a kind and beautiful daughter. His wife died and he married again. The stepmother was cruel and unkind. Each night the girl sat among the cinders by the hearth, staring sadly into the fire. Her stepsisters noticed this and called her Cinderella. In the day time she washed dishes, swept the floor, kept the house clean, worked in the kitchen-garden, watered the flowers, cooked the meal.

One day the King's son made up his mind to hold a ball. Everyone went to the ball except Cinderella.

But suddenly a bright light lit up the dark kitchen. It was a fairy. The fairy went into the garden and waved her wand over a large pumpkin and it immediately turned into a beautiful golden coach. Then she turned six white mice into six fine horses. A fat rat became a coachman and two lizards became fine footmen. She gave Cinderella a beautiful gown and shining glass slippers.

In the Palace Cinderella was the most beautiful girl. She danced much and was happy. But suddenly she remembered about fairy's warning. At twelve o'clock Cinderella ran off through the garden. The Prince ran after her and found a glass slipper.

When Cinderella reached home, the golden coach had disappeared and instead of her beautiful gown she wore her old dress again.

Next morning the Prince announced that he was in love with the mysterious girl and would marry the girl who could wear the tiny slipper.

The Prince's servants travailed through the town and asked each girl to try the slipper on. Cinderella tried the slipper too. It fitted perfectly. Stepmother was outraged. Cinderella became a Prince's bride.

They were married just a few days later and the whole kingdom was happy.

APPENDIX 2

Here there is an example of the tale composed by Anya Orlova in the end of the first year of studies. She is five years old.

THE FAIRY-TALE ABOUT TWO KITTENS

Long ago, in a land far away, there lived two cats. They had two kittens. One of them was white and one of them was black. Their names were Murka and Barsik. Every night the kittens sat among the toys staring sadly into the window, because they wanted to go to their grandmother. In the day time they helped their mother: watered the flowers, kept the house clean, worked in the kitchen-garden, swept the floor, cooked the meal, washed dishes.

One day the mother made up her mind to send the kittens to the grandmother. Everyone went except the father. But suddenly the mother decided to send the kittens alone. The kittens went alone. While the kittens went through the forest, they picked the flowers. The grandmother was a fairy. She waved her wand over the flowers and they immediately turned into two little puppies sleeping in the basket.

Next day the kittens and the puppies went to the cinema. The film was very interesting. It was about Mutroskin cat. The kittens and the puppies were happy. But suddenly they remembered about grandmother's warning: not to be late at home. The kittens and the puppies ran off through the street.

The grandmother cooked their favorite dish. It

was an apple-pie. They ate it with pleasure and thanked the grandmother.

Next day they returned home. The mother and the father were happy. They presented to them five interesting books and promised them to send kittens to the grandmother next summer.

It should be added that twenty from twenty four 5- and 6-year-old Russian children studying in experimental groups of O. S. Fomichova in Moscow during one year (eight months) can compose the tales of the kind. The other four children are able to compose simpler tales. Besides, if the children listen for the first time to a tale of the same volume composed by a girl or a boy from their group, they remember quite well this tale and can retell it actively just on the spot.

This proves the high effectiveness of elaborated methods of emotionally-imaginative teaching children English and positive influence of these methods on the development of the personality of a child.

References

[1] Arendt, J. D., Warriner-Burke, H. P., 1990, *Teaching all students: reaching and teaching students of varying abilities,* Foreign Language Annals (New York), Vol. 23, No. 5, 445–452.

[2] Brusch, W., 1991, *The role of reading in foreign language acquisition: designing an experimental project,* ELT Journal, Vol. 45, No. 2, 156–163.

[3] Caron, J., 1989, *Précis de Psycholinguistique,* Paris, Presses Universitaires de France.

[4] Chernov, G. V., 1987, *The Foundations of Synchronous Interpretation,* Moscow, Vys'shaya Shkola.

[5] Dabène, L., 1991, *Enseignement précoce d'une langue ou éveil au langage?* Francais dans le Monde (Paris), No. 8/9, 57–64.

[6] Fauconnier, G., 1985, *Mental Spaces,* Cambridge (MA), Bradley Books.

[7] Fillmore, C., 1985, *Frames and the semantics of understanding,* Quaderni di Semantica, Vol. 6, 222–253.

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

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

[10] Fomichov, V., 1992, *Mathematical models of natural-language-processing systems as cybernetic models of a new kind,* Cybernetica (Belgium), Vol. XXXV, No. 1, 63–91.

[11] Fomichov, V. A., 1993a, *Towards a mathematical theory of natural-language communication,* Informatica. An Int. J. of Computing and Informatics (Slovenia), Vol. 17, No. 1, 21–34.

[12] Fomichov, V. A., 1993b, *K-calculuses and K-languages as powerful formal means to design intelligent systems processing medical texts,* Cybernetica (Belgium), Vol. XXXVI, No. 2, 161–182.

[13] Fomichov, V. A., *Integral Formal Semantics and the design of legal full-text databases,* Cybernetica, 1994 (In press).

[14] Fomichov, V. A., & Fomichova, O. S., 1993, *The role of the artificial intelligence theory in developing new, highly effective methods of foreign languages teaching,* in PEG93: Proc. of the Seventh Intern. PEG Conf. AI Tools and the Classroom: Theory into Practice. Moray House Institute of Education, Heriot-Watt University, Edinburgh, July 2–4, 1993, 319–329.

[15] Fomichova, O. S., 1989, *The structure of an informational component of a scientific paper and the ways of explicating implicit assessments in a scientific text.* Paper deposited in the Institute of Information on Social Sciences (INION) Ac. Sc. USSR 30/06/89, No.

Dep. 38650, 28 pp. (in Russian).—Abstract in the Bibliographic Directory of INION "New Soviet Literature on Social Sciences", Series "Linguistics", 1989, No. 12.

[16] Fomichova, O. S., 1990, *The problem of expressing assessments of factuality on the basis of analysis of semantic structure of a scientific text (the problem of abstracting and translating scientific texts from English into Russian).* Synopsis of Ph.D. thesis. Moscow Linguistic University (former Moscow State Institute of Foreign Languages), 1990 (in Russian).

[17] Geeraerts, D., 1990, *Editorial Statement,* Cognitive Linguistics, Vol. 1, No. 1, 1–3.

[18] Glasser, W., 1991, *Schools without Failure,* Moscow, Progress (translation into Russian).

[19] Johnson, M., 1987, *The Body in the Mind: The Bodily Basis of Meaning, Imagination, and Reason,* Chicago, The University of Chicago Press.

[20] Johnson, W. L., 1991, *Book Review (J. Psottka, L. D. Massey and S. A. Mutter, eds., Intelligent Tutoring Systems: Lessons Learned),* Artif. Intelligence, Vol. 48, No. 1, 125–134.

[21] Johnson-Laird, P., 1983, *Mental Models,* Cambridge (MA), Harvard University Press.

[22] Knowledge, 1992: *A Plan for the Knowledge Archives Project,* The Economic Research Institute, Japan Society for the Promotion of Machine Industry, and Systems Research & Development Institute of Japan, Tokyo, March, 1992.

[23] Lakoff, G., 1987, *Women, Fire and Dangerous Things: What Categories Reveal about the Mind,* Chicago, The University of Chicago Press.

[24] Langacker, R. W., 1990, *Concept, Image, and Symbol.* The Cognitive Basis of Grammar, Berlin, New York, Mouton de Gruyter.

[25] MacIntyre, P. D., & Gardner, R. C., 1991, *Investigating language class anxiety using the focused essay technique,* Modern Lang. Journal (Madison, Wisconsin), Vol. 75, No. 3, 296–304.

Met, M., & Rhodes, N., 1990, *Elementary school foreign language instruction: priorities for the 1990s,* Foreign Language Annals (New York), Vol. 23, No. 5, 433–443.

[26] Newell, A., 1990, *Unified Theories of Cognition,* Cambridge (MA), Harvard University Press.

[27] Pavilionis, R. J., 1985, *On meaning and understanding,* Acta Philosophica Fennica, Vol. 38.

[28] Pitrat, J., 1992, *La symbiose entre l'intelligence artificielle et la science cognitive,* Technique et Science Informatiques, Vol. 11, No. 2, 9–24.

[29] Rixon, S., 1992, *English and other languages for younger children: practice and theory in a rapidly changing world (State of the art article),* Lang. Teaching, Vol. 25, No. 2, 73–93.

[30] Schank, R. C., & Slade, S. B., 1991, *The future of Artificial Intelligence: learning from experience,* Applied Artificial Intelligence, Vol. 5, No. 1, 97–107.

Self, J., 1988, *Bypassing the intractable problem of student modelling,* in C. Frasson (Ed.), ITS-88. Proc. Intelligent Tutoring Systems, Montreal, June 1–3, 1988.

[31] Seuren, P. A. M., 1985, *Discourse Semantics,* Oxford, Basil Blackwell.

[32] Sowa, J. F., 1984, *Conceptual Structures—Information Processing in Mind and Machine,* Addison-Wesley, Reading (MA).

[33] Winograd, T., & Flores, F., 1986, *Understanding Computers and Cognition. A new foundation for design,* Reading (MA), Addison-Wesley Publishing Company, Inc.

[34] Železnikar, A. P., 1988, *Principles of Information,* Cybernetica (Belgium), Vol. XXXI, No. 2, 99–122.

[35] Železnikar, A. P., 1992, *Towards an Informational Language,* Cybernetica, Vol. XXXV, No. 2, 139–158.

[36] Železnikar, A. P., 1993a, *Mission and Research Reports,* Informatica (Slovenia), Vol. 17, No. 1, 81–100.

[37] Železnikar, A. P., 1993b, *Towards an informational orientation,* Informatica, Vol. 17, No. 1, p. 1.

[38] Železnikar, A. P., 1993c, *Metaphysicalism of informing,* Informatica, Vol. 17, No. 1, 65–80.

INFORMATIONAL BEING-IN

Anton P. Železnikar
Volaričeva ulica 8, 61111 Ljubljana, Slovenia
anton.p.zeleznikar@ijs.si

Keywords: abduction, Being-in, Being-in-the-world, circularity, decomposition, deduction, externalism, informational includedness (involvement, embedding), induction, inference, informational modi (ponens, tollens, rectus, obliquus), internalism, metaphysicalism, parallelism, phenomenalism, reasoning, serialism

Edited by: Oliver B. Popov
Received: February 8, 1994 **Revised:** April 18, 1994 **Accepted:** April 28, 1994

In this paper the phenomenon of informational Being-in, that is, includedness is studied in a formally recursive (informational) way, dealing with basic definitions of includedness (informational in-volvement, em-bedding) and their consequences. It seems that the informational includedness is a phenomenon of informational entities, which involves them in a perplexedly recursive way and offers the richness of the informationally spontaneous parallelism, serialism, and circularity. In this respect, together with its informational openness and recursiveness, informational Being-in can come semantically as close as possible to its philosophical notion (concept) [2, 1]. Some includable structured phenomena of inference or reasoning (deduction, induction, abduction, modus ponens, tollens, rectus, and obliquus) are shown in a formal manner. The disposed formal apparatus enables an unbounded and even deepened philosophical investigation of the phenomenon of Being-in and its consequences. So, a formalistic investigation of informational Being-in can enrich its philosophical understanding.

1 Introduction

Being-in is the original term coined by *Heidegger* (in German, In-Sein or In-sein, in English, inhood) [2]. Informational Being-in[1] belongs to the most significant existentiales of the informational. "When someone calls our attention to the fact that 'in' also has an existential sense which expresses *involvement*, ...we tend to think of this as a metaphorical deviation from physical inclusion." (Dreyfus in [1], p. 41.) Informational 'in' means informationally involved, distributed and, for example, being dual in the sense of energy and information (Šlechta in [5], the Hamiltonians H_{EI} and H_{IE} in equations 14 and 15, respectively). "Grimm goes on to argue that the preposition 'in' is derived from the verb, rather than the verb from the preposition." ([2], p. 80.) This conclusion is essential, for Being-in in the informational has an active (verb-like, operational) role.

Being-in belongs to primordial situations concerning informational entities. It is a consequence of informing of entities and vice versa. In this paper, the basic informational properties of the phenomenon, state, or process termed Being-in will be studied. Instead of the philosophical term Being-in, the term *includedness* (or informational includedness) will be frequently used, which comes closer to the formal terminology in traditional mathematics (e.g., the notion of a subset), but is essentially different in its existential (informingly arising) nature if compared with the categorical (reductionistic) relation of inclusion. *Informational includedness* is a new term, determined in an informationally recursive way (circularly) and, in this respect, extending the structure of informational includedness boundlessly in an includable way.

Being-in is an informational existentiale, a for-

[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.

mal existential expression, which concerns Being-in-the-world (informational realm of the exterior) as its essential state. An informational entity (thing, matter) informs as the entity in the world (its environment, informational region, within itself). The world is the synonym of that in which an entity informs, that is, the informational entity embracing informational realm. The question is, how can this general view (an entity's informational openness, interweavement, or connectedness) be considered in its informational entirety.

When an informational formula occurs as part of a larger formula, it is said to be in *included position* (see [13], included ppl. a.); otherwise it is said to be in absolute position and to constitute an informational entity. Some morphemes occur in included position, either partial or complete. In some sentences there are devices that signal the inclusion of two or more separate sentences. The included position is that had by a word, phrase or other linguistic form when it is part of a larger form. All of these forms of includedness are classical and do not embrace conceptualism of the informational includedness as, for instance, a distributed involvement of an entity within an informational realm.

The task of the present study is to develop and formalize a general concept of the Being-in of an informational entity, where this concept can be particularized according to the specific informational needs and occurring circumstances.

2 Being-in qua Informing

A philosophy of informational includedness roots in the philosophical notion of Being-in (for example, [2], ¶12) as an informational (philosophical) phenomenon which concerns the entity's Being. As we shall see, Being-in of something can meaningly never be exhausted, it simply does not come to an end because of its recursively open informational nature. So we have to present this informational virtue, faculty, or property of something in a strict formal way, that is, by systems of informational formulas describing the phenomenon of Being-in. The informational includedness means something essentially different in respect to the set-theoretical inclusion in mathematics, although the symbols $\subset$ and $\supset$ (the alternative to $\subset$) are used to mark both phenomena.

How does Being-in, that is, informational includedness inform? As a property of something which informs in a broader realm, it must be expressed as includedness, that is, as something concerning the informational operator (for example, $\models_{\text{in}}$, $\models_{\text{include}}$ or, simply, $\subset$, which read 'is in', 'is included in' or 'is an informational part of', respectively). Informational includedness means functional involvement of an entity into the informing of the other entity and itself.

The Being-in as such always concerns an entity, that is, something, marked by α. As a phenomenon, the Being-in is involved in something in an informational way. According to [9], we introduce the following four modes of the informational existentiale concerning something α in an includable way:

$\alpha \subset$	reads as: *α informs includingly;* *α's externalism* of including; *α is/are included* (in);
$\subset \alpha$	reads as: *α is informed includingly;* *α's internalism* of including; *α include(s);*
$\alpha \subset \alpha$	reads as: *α informs includingly itself* and *is informed includingly by itself;* *α's metaphysicalism* of including; *α includes and is included in itself;*
$\begin{pmatrix} \alpha \subset; \\ \subset \alpha \end{pmatrix}$	reads as: *α informs includingly* and *is informed includingly;* *α's phenomenalism* of including; *α includes and is included*

To fulfill the existential criteria of includedness, evidently, entity α informationally includes (involves) some informing entities and is informationally included in (involved by) some informing entities.

In this point of the study, the question arises, what could the meaning (interpretation) of the formalized forms of informational includedness of something be? The formalized externalistic interpretation of includedness could be the following:

$$(\alpha \subset) \rightleftharpoons \begin{pmatrix} \alpha \models; \\ \models \alpha; \\ \Xi(\alpha \subset) \end{pmatrix}$$

where $\Xi(\alpha \subset)$ is an element of the informational power set (symbol $\mathcal{P}$) with 16 elements (including the empty set $\emptyset$), that is,

$$\Xi(\alpha \subset) \in \mathcal{P}\left(\left\{\begin{array}{l}(\models \alpha) \subset, \\ (\alpha \models) \subset, \\ (\models \alpha) \subset \alpha, \\ (\alpha \models) \subset \alpha\end{array}\right\}\right)$$

In general, an informational set $\{\alpha_1, \alpha_2, \cdots, \alpha_n\}$ is interpreted as the parallel system (array) of entities, that is,

$$\{\alpha_1, \alpha_2, \cdots, \alpha_n\} \rightleftharpoons (\alpha_1; \alpha_2; \cdots; \alpha_n)$$

The formalized internalistic interpretation of includedness is, for example,

$$(\subset \alpha) \rightleftharpoons \begin{pmatrix}\alpha \models; \\ \models \alpha; \\ \Xi(\subset \alpha)\end{pmatrix}$$

where $\Xi(\subset \alpha)$ is an element of the power set, that is,

$$\Xi(\subset \alpha) \in \mathcal{P}\left(\left\{\begin{array}{l}(\alpha \models) \subset \alpha, \\ (\models \alpha) \subset \alpha, \\ (\alpha \models) \subset, \\ (\models \alpha) \subset\end{array}\right\}\right)$$

The metaphysicalistic case of includedness interpretation could be

$$(\alpha \subset \alpha) \rightleftharpoons \begin{pmatrix}\alpha \models \alpha; \\ \Xi(\alpha \subset \alpha)\end{pmatrix}$$

where

$$\Xi(\alpha \subset \alpha) \in \mathcal{P}(\{\alpha \models \alpha\})$$

and the phenomenalistic case

$$(\alpha \subset; \subset \alpha) \rightleftharpoons \begin{pmatrix}\alpha \models; \\ \models \alpha; \\ \Xi(\alpha \subset); \\ \Xi(\subset \alpha)\end{pmatrix}$$

These are initial cases of includedness and each of them speaks in its own way, so various interpretations are possible.

3 A Definition and Consequences of Informational Includedness

Let us introduce the basic definition of informational includedness which will cover also the four previously shown examples (includable externalism, internalism, metaphysicalism, and phenomenalism).

Definition 1 [Informational Includedness] *Let entity α inform within entity β, that is, $\alpha \subset \beta$. This expression reads:* α informs within (is an informational component or constituent of) β. *Let the following parallel system of includedness (Being-in) be defined recursively:*

$$(\alpha \subset \beta) \rightleftharpoons_{\mathrm{Def}} \begin{pmatrix}\beta \models \alpha; \\ \alpha \models \beta; \\ \Xi(\alpha \subset \beta)\end{pmatrix}$$

where for the extensional part $\Xi(\alpha \subset \beta)$ of the includedness $\alpha \subset \beta$, there is,

$$\Xi(\alpha \subset \beta) \in \mathcal{P}\left(\left\{\begin{array}{l}(\beta \models \alpha) \subset \beta, \\ (\alpha \models \beta) \subset \beta, \\ (\beta \models \alpha) \subset \alpha, \\ (\alpha \models \beta) \subset \alpha\end{array}\right\}\right)$$

The most complex element of this power set is denoted by

$$\Xi^{\beta,\alpha}_{\beta,\alpha}(\alpha \subset \beta) \rightleftharpoons \begin{pmatrix}(\beta \models \alpha) \subset \beta, \alpha; \\ (\alpha \models \beta) \subset \beta, \alpha\end{pmatrix}$$

Cases, where $\Xi(\alpha \subset \beta) \rightleftharpoons \emptyset$ and $\emptyset$ denotes an empty entity (informational nothing), are exceptional (reductionistic). □

Consequence 1 [An Extension of Informational Includedness] *Let us introduce the following markers:*

$$\begin{aligned}\theta(\xi) &= \begin{pmatrix}\xi \models (\beta \models \alpha); \\ (\beta \models \alpha) \models \xi\end{pmatrix}; \\ \vartheta(\xi) &= \begin{pmatrix}\xi \models (\alpha \models \beta); \\ (\alpha \models \beta) \models \xi\end{pmatrix}; \\ \xi &\in \{\beta, \alpha\}\end{aligned}$$

Then, the cases of includedness within Ξ-elements in Definition 1 induce, evidently,

$$((\beta \models \alpha) \subset \xi) \rightleftharpoons \begin{pmatrix} \theta(\xi); \\ \Xi((\beta \models \alpha) \subset \xi) \end{pmatrix};$$

$$((\alpha \models \beta) \subset \xi) \rightleftharpoons \begin{pmatrix} \vartheta(\xi); \\ \Xi((\alpha \models \beta) \subset \xi) \end{pmatrix};$$

$$((\beta \models \alpha) \subset \beta, \alpha) \rightleftharpoons \begin{pmatrix} \theta(\beta); \\ \theta(\alpha); \\ \Xi((\beta \models \alpha) \subset \beta, \alpha) \end{pmatrix};$$

$$((\alpha \models \beta) \subset \beta, \alpha) \rightleftharpoons \begin{pmatrix} \vartheta(\beta); \\ \vartheta(\alpha); \\ \Xi((\alpha \models \beta) \subset \beta, \alpha) \end{pmatrix}$$

where

$$\Xi((\beta \models \alpha) \subset \beta, \alpha) \rightleftharpoons \begin{pmatrix} \Xi((\beta \models \alpha) \subset \beta); \\ \Xi((\beta \models \alpha) \subset \alpha) \end{pmatrix};$$

$$\Xi((\alpha \models \beta) \subset \beta, \alpha) \rightleftharpoons \begin{pmatrix} \Xi((\alpha \models \beta) \subset \beta); \\ \Xi((\beta \models \alpha) \subset \alpha) \end{pmatrix}$$

□

Instead of proving this consequence, let us extend recursively Definition 1 one step deeper.

Consequence 2 [A Further Extension of Informational Includedness] *According to the basic definition of includedness, there is recursively,*

$$(\alpha \subset \beta) \rightleftharpoons_{\text{Def}} \left(\begin{array}{l} \beta \models \alpha; \\ \alpha \models \beta; \\ \Xi^{\beta,\alpha}_{\beta,\alpha}(\alpha \subset \beta) \rightleftharpoons \\ \left(\begin{array}{l} \beta \models (\beta \models \alpha); \\ (\beta \models \alpha) \models \beta; \\ \Xi((\beta \models \alpha) \subset \beta); \\ \beta \models (\alpha \models \beta); \\ (\alpha \models \beta) \models \beta; \\ \Xi((\alpha \models \beta) \subset \beta); \\ \alpha \models (\beta \models \alpha); \\ (\beta \models \alpha) \models \alpha; \\ \Xi((\beta \models \alpha) \subset \alpha); \\ \alpha \models (\alpha \models \beta); \\ (\alpha \models \beta) \models \alpha; \\ \Xi((\alpha \models \beta) \subset \alpha) \end{array} \right) \end{array} \right)$$

where, for instance, the first extensional part is

$$\Xi((\beta \models \alpha) \subset \beta) \in \mathcal{P} \left(\left\{ \begin{array}{l} (\beta \models (\beta \models \alpha)) \subset \beta, \\ ((\beta \models \alpha) \models \beta) \subset \beta, \\ (\beta \models (\beta \models \alpha)) \subset \alpha, \\ ((\beta \models \alpha) \models \beta) \subset \alpha \end{array} \right\} \right)$$

the second one

$$\Xi((\alpha \models \beta) \subset \beta) \in \mathcal{P} \left(\left\{ \begin{array}{l} (\beta \models (\alpha \models \beta)) \subset \beta, \\ ((\alpha \models \beta) \models \beta) \subset \beta, \\ (\beta \models (\alpha \models \beta)) \subset \alpha, \\ ((\alpha \models \beta) \models \beta) \subset \alpha \end{array} \right\} \right)$$

the third one

$$\Xi((\beta \models \alpha) \subset \alpha) \in \mathcal{P} \left(\left\{ \begin{array}{l} (\alpha \models (\beta \models \alpha)) \subset \beta, \\ ((\beta \models \alpha) \models \alpha) \subset \beta, \\ (\alpha \models (\beta \models \alpha)) \subset \alpha, \\ ((\beta \models \alpha) \models \alpha) \subset \alpha \end{array} \right\} \right)$$

and the fourth one

$$\Xi((\alpha \models \beta) \subset \alpha) \in \mathcal{P} \left(\left\{ \begin{array}{l} (\alpha \models (\alpha \models \beta)) \subset \beta, \\ ((\alpha \models \beta) \models \alpha) \subset \beta, \\ (\alpha \models (\alpha \models \beta)) \subset \alpha, \\ ((\alpha \models \beta) \models \alpha) \subset \alpha \end{array} \right\} \right)$$

□

Within this consequence, the circular structures of the form

$$((\beta \models \alpha) \models \beta);$$
$$(\beta \models (\alpha \models \beta));$$
$$((\alpha \models \beta) \models \alpha);$$
$$(\alpha \models (\beta \models \alpha))$$

belonging to the first, second, third, and fourth extension will become significant in the context of entity metaphysicalism.

Let us explain in short the meaning of informational operators $\rightleftharpoons$, $\rightleftharpoons_{\text{Def}}$, $\subset$, $\in$, $=$ and, through this explanation, point out the difference regarding the equally marked mathematical operators and relations. Let have the following interpretation:

$\rightleftharpoons$	mean(s), informs meaningly;
$\rightleftharpoons_{\text{Def}}$	mean(s) by definition;
$\subset$	informs within (includingly);
$\in$	is an element of informational set of entities, informational lumps;
$=$	is a marker for, is the same as;

The meaning of an informational operator correlates with the meaning of the meaningly adequate verbal phrase which expresses an informational activity, happening, occurring, state, position, attitude, etc. The meaning of a mathematical operator concerns solely the mathematically well-defined abstract objects.

4 Informational Consequences of Includedness

The most characteristic consequences of informational includedness are circular forms of parallelism and serialism.

4.1 Parallelism and Serialism of Includedness

The includedness of an informational entity in concern to an informational entity induces a certain phenomenon of parallelism and serialism. More precisely, includedness generates informational circularity in the form of parallel and serial cycles which are of essential significance in emerging of the so-called metaphysicalism. Metaphysicalism shapes the background for the arising of cognitive and intelligent information, by mixing of intelligent informational lumps and composing them by intelligent selection into informational structures performing understanding, generating meaning, that is, cognizing.

The parallelism of $\alpha \subset \beta$ is already observed in Definition 1, where $\alpha \subset \beta$ is a parallel structure of transitions $\beta \models \alpha$, $\alpha \models \beta$, and extension $\Xi(\alpha \subset \beta)$. This structure is circularly-parallel in components $\beta \models \alpha; \alpha \models \beta$ in respect to β via (implicitly) α, that is, in a parallel transitive way. On the other hand, the extensional part of informational includedness $\Xi(\alpha \subset \beta)$ in Definition 1, can be chosen, for instance, as

$$\Xi^{\beta}_{\beta}(\alpha \subset \beta) \rightleftharpoons \begin{pmatrix} (\beta \models \alpha) \subset \beta; \\ (\alpha \models \beta) \subset \beta \end{pmatrix}$$

This structure is circularly serial in respect to β via α, etc. [e.g., $\Xi^{\beta}_{\alpha}(\alpha \subset \beta)$ is circularly serial also in respect to α via β].

4.2 Parallelism of Includedness

In some respect, the parallelism of includedness is straightforward, that is, informationally transparent. As we shall see, the parallel includedness can be defined in a common way (a mathematical fashion), moving from one 'relation' of includedness to the other.

Consequence 3 [Transitivity of Parallel Informational Includedness] *Let for informational entities α_i, α_j, and α_k be $\alpha_i \subset \alpha_j; \alpha_j \subset \alpha_k$. Then, the implication pertaining to the includable transitivity,*

$$(\alpha_i \subset \alpha_j; \alpha_j \subset \alpha_k) \Longrightarrow (\alpha_i \subset \alpha_k)$$

is informationally righteous. □

Proof. The intuitive proof of the consequence belongs to the semantics of a language (speech). If α_i informationally involves α_j and if α_j informationally involves α_k, then, in a language-logical sense, α_i involves α_k informationally via entity α_j.

Another, more formalistic proof of the consequence follows from the axiomatic concept of the informing of entities. Informational operator $\subset$ has to be comprehended as a particular case of operator $\models$. In case of parallel informing $\alpha \models \beta; \beta \models \gamma$, there follows $\alpha \models \gamma$. Q.E.D.

Consequence 4 [Parallelism of Informational Includedness] *Let for informational entities $\alpha_1, \alpha_2, \cdots, \alpha_n$ be*

$$\alpha_i \subset \alpha_{i+1};\ i = 1, 2, \cdots, n-1\,.$$

This formula depicts a parallel system of includedness, that is,

$$\begin{aligned} &\alpha_1 \subset \alpha_2; \\ &\alpha_2 \subset \alpha_3; \\ &\vdots \\ &\alpha_{n-1} \subset \alpha_n \end{aligned}$$

which is transitively includable and parallel straightforward. There is,

$$\left.\begin{aligned} &\alpha_{i+1} \models \alpha_i; \\ &\alpha_i \models \alpha_{i+1}; \\ &\Xi(\alpha_i \subset \alpha_{i+1}) \end{aligned}\right\}\ i = 1, 2, \cdots, n-1$$

This parallel system implies the parallelism of different elementary informational forms of entities $\alpha_1, \alpha_2, \cdots, \alpha_n$ in a parallel descending, ascending, and also circular order regarding index i and the system's structural dependence on the includable extensions $\Xi(\alpha_i \subset \alpha_{i+1})$ $(i = 1, 2, \cdots, n-1)$. □

Proof. Let us look the parallel elementary structures, that is, parallel components of the parallel included system. There is

$$\begin{pmatrix} \alpha_1 \subset \alpha_2; \\ \alpha_2 \subset \alpha_3; \\ \vdots \\ \alpha_{n-1} \subset \alpha_n \end{pmatrix} \rightleftharpoons$$

$$\begin{pmatrix} \alpha_n \models \alpha_{n-1}; & \alpha_1 \models \alpha_2; & \Xi(\alpha_1 \subset \alpha_2); \\ \vdots & \alpha_2 \models \alpha_3; & \Xi(\alpha_2 \subset \alpha_3); \\ \alpha_3 \models \alpha_2; & \vdots & \vdots \\ \alpha_2 \models \alpha_1; & \alpha_{n-1} \models \alpha_n; & \Xi(\alpha_{n-1} \subset \alpha_n) \end{pmatrix}$$

Different forms of Ξ's are possible, conditioning the nature of the elementary circularity between α-entities. Different types of circularity will be shown in the subsequent consequences. Q.E.D.

Consequence 5 [Manifold Parallelism of Informational Includedness] *The implication concerning the manifoldness of parallel structured includedness follows directly from Consequence 3 in the form*

$$(\alpha_i \subset \alpha_{i+1}; i = 1, 2, \cdots, n-1) \Longrightarrow$$

$$\begin{pmatrix} \alpha_j \subset \alpha_k; j < k; \\ j \in \{1, 2, \cdots, n-1\}; k \in \{2, 3, \cdots, n\} \end{pmatrix}$$

The parallel manifoldness of includedness will become the basis for the manifoldness of the circular parallelism. □

The last consequence means that there are parallel groups of parallel includable cases of length ℓ $(2 \leq \ell \leq n-1)$ of the form

$$\left.\begin{matrix} \alpha_{i_1} \subset \alpha_{i_2}; \\ \alpha_{i_2} \subset \alpha_{i_3} \end{matrix}\right\} \; 1 \leq i_1 \leq i_2 \leq i_3 \leq n; \ell = 2;$$

$$\left.\begin{matrix} \alpha_{i_1} \subset \alpha_{i_2}; \\ \alpha_{i_2} \subset \alpha_{i_3}; \\ \alpha_{i_3} \subset \alpha_{i_4} \end{matrix}\right\} \; 1 \leq i_1 \leq i_2 \leq i_3 \leq i_4 \leq n; \ell = 3;$$

$$\vdots$$

$$\left.\begin{matrix} \alpha_1 \subset \alpha_2; \\ \alpha_2 \subset \alpha_3; \\ \vdots \\ \alpha_{n-1} \subset \alpha_n \end{matrix}\right\} \; \ell = n-1$$

According to Consequence 4, we can construct the ascending and descending sequences of parallel formulas $\alpha_{i_j} \models \alpha_{i_k}$ in regard to subscripts i_j and i_k.

4.3 Circular Parallelism of Includedness

Let us define a complete form of circular parallelism in the following form.

Definition 2 [Circular Parallelism of an Informational System]. *An informational system of entities $\alpha_1, \alpha_2, \cdots, \alpha_n$ is called circularly parallel, if*

$$\alpha_i \models \alpha_{i+1}; \; i = 1, 2, \cdots, n-1;$$
$$\alpha_n \models \alpha_1$$

where $n = 2, 3, \cdots$. □

Definition 3 [Completely Circular Parallelism of an Entity System]. *A system of (parallel) informational entities $\alpha_1, \alpha_2, \cdots, \alpha_n$ is called completely circularly parallel, if*

$$\alpha_i \models \alpha_j; \; i, j = 1, 2, \cdots, n$$

The circular completeness of parallelism enables the occurrence of all possible cycles of formulas $\alpha_i \models \alpha_j$ in which operands $\alpha_1, \alpha_2, \cdots, \alpha_n$ appear, in a recursive way. □

The circular parallelism follows from the transitive parallelism of informational includedness (Consequence 3), if to the parallel sequence $\alpha_1 \subset \alpha_2; \alpha_2 \subset \alpha_2; \cdots; \alpha_{n-1} \subset \alpha_n$ formula $\alpha_n \subset \alpha_1$ is added.

Consequence 6 [Circular Parallelism of Informational Includedness] *Let for informational entities $\alpha_1, \alpha_2, \cdots, \alpha_n$ be*

$$\alpha_i \subset \alpha_{i+1}; \; i = 1, 2, \cdots, n-1;$$
$$\alpha_n \subset \alpha_1$$

This system causes a circularly complete informational system of entities $\alpha_1, \alpha_2, \cdots, \alpha_n$ (Definition 3). □

Proof. According to the transitivity of informational includedness (operator $\subset$) (Consequence 3) there is, evidently,

$$\begin{pmatrix} \alpha_1 \subset \alpha_2; \\ \alpha_2 \subset \alpha_3; \\ \vdots \\ \alpha_{n-1} \subset \alpha_n; \\ \alpha_n \subset \alpha_1 \end{pmatrix} \Longrightarrow \begin{pmatrix} \alpha_n \subset \alpha_{n-1}; \\ \alpha_{n-1} \subset \alpha_{n-2}; \\ \vdots \\ \alpha_2 \subset \alpha_1; \\ \alpha_1 \subset \alpha_n \end{pmatrix}$$

The left sequence of formulas appears in an cyclic ascending order while the right sequence is cyclically descending (an opposite direction of the cycle). This obviously yields

$$\begin{pmatrix} \alpha_i \subset \alpha_{i+1}; \\ i = 1,2,\cdots,n-1; \\ \alpha_n \subset \alpha_1 \end{pmatrix} \rightleftharpoons \begin{pmatrix} \alpha_i \subset \alpha_j; \\ i,j = 1,2,\cdots,n \end{pmatrix}$$

The last formula shows the power of includable circularity, where

$$\begin{pmatrix} \alpha_i \subset \alpha_j; \\ i,j = 1,2,\cdots,n \end{pmatrix} \Longrightarrow \begin{pmatrix} \alpha_i \models \alpha_j; \\ i,j = 1,2,\cdots,n \end{pmatrix}$$

The power of includable circularity (operator $\subset$) is stronger than that of the direct circularity of informing (operator $\models$), because still the Ξ-extensional cases have to be considered. Q.E.D.

4.4 Serialism of Includedness

Let us study the possible cases of serialism of informational includedness and their consequences in regard to informing among entities. In the first step we study a straightforward serialism, and in the second step a circular one.

Definition 4 [Serialism of Informational Includedness] *Let us introduce the subscripted forms $\Phi_i^{\subset}$ of serialism concerning the informational includedness of entities $\alpha_1, \alpha_2, \cdots, \alpha_n$ and mark them in the following way:*

$$\Phi^{\subset}(\alpha_1,\alpha_2,\cdots,\alpha_n) \Longrightarrow$$

$$\begin{pmatrix} \Phi_1^{\subset}(\alpha_1,\alpha_2,\cdots,\alpha_n) \rightleftharpoons \\ (\alpha_1 \subset^* (\alpha_2 \subset (\alpha_3 \subset (\cdots(\alpha_{n-1} \subset \alpha_n)\cdots)))); \\ \Phi_2^{\subset}(\alpha_1,\alpha_2,\cdots,\alpha_n) \rightleftharpoons \\ ((\alpha_1 \subset \alpha_2) \subset^* (\alpha_3 \subset (\cdots(\alpha_{n-1} \subset \alpha_n)\cdots))); \\ \vdots \\ \Phi_{n-1}^{\subset}(\alpha_1,\alpha_2,\cdots,\alpha_n) \rightleftharpoons \\ (((\cdots((\alpha_1 \subset \alpha_2) \subset \alpha_3)\cdots) \subset \alpha_{n-1}) \subset^* \alpha_n) \end{pmatrix}$$

By the asterisk marked operators $\subset$, that is, $\subset^*$, the main separators between the informer and the observer part of the expression are meant. Implicational operator $\Longrightarrow$ marks only specific includable serial cases on its right side (but not all possible cases). □

Example [Includedness of Includedness] Let us see what does the example of includedness of includedness, for example,

$$(\alpha \subset \beta) \subset \gamma$$

mean? Let us extendingly interpret this formula in a regular informational manner, when,

$$\left(\left(\begin{pmatrix} \beta \models \alpha; \\ \alpha \models \beta; \\ \Xi(\alpha \subset \beta) \end{pmatrix} \subset \gamma\right) \rightleftharpoons \begin{pmatrix} \gamma \models \begin{pmatrix} \beta \models \alpha; \\ \alpha \models \beta; \\ \Xi(\alpha \subset \beta) \end{pmatrix}; \\ \begin{pmatrix} \beta \models \alpha; \\ \alpha \models \beta; \\ \Xi(\alpha \subset \beta) \end{pmatrix} \models \gamma; \\ \Xi\left(\left(\begin{pmatrix} \beta \models \alpha; \\ \alpha \models \beta; \\ \Xi(\alpha \subset \beta) \end{pmatrix} \subset \gamma\right)\right) \end{pmatrix}\right)$$

The right part of the formula can be linearly decomposed (informationally multiplied), e.g.,

$$((\alpha \subset \beta) \subset \gamma) \rightleftharpoons$$

$$\begin{pmatrix} \gamma \models (\beta \models \alpha); \gamma \models (\alpha \models \beta); \gamma \models \Xi(\alpha \subset \beta); \\ (\beta \models \alpha) \models \gamma; (\alpha \models \beta) \models \gamma; \Xi(\alpha \subset \beta) \models \gamma; \\ \Xi((\beta \models \alpha) \subset \gamma); \Xi((\alpha \models \beta) \subset \gamma); \\ \Xi((\Xi(\alpha \subset \beta) \subset \gamma) \end{pmatrix}$$

where for a maximal case of informationally (mutually) involved entities α, β, and γ, there is,

$$\Xi_{\beta,\alpha}^{\beta,\alpha}(\alpha \subset \beta) \rightleftharpoons \begin{pmatrix} (\beta \models \alpha) \subset \beta, \alpha; \\ (\alpha \models \beta) \subset \beta, \alpha \end{pmatrix};$$

$$\Xi_{\gamma,(\beta\models\alpha)}^{\gamma,(\beta\models\alpha)}((\beta \models \alpha) \subset \gamma) \rightleftharpoons$$

$$\begin{pmatrix} (\gamma \models (\beta \models \alpha)) \subset \gamma, (\beta \models \alpha); \\ ((\beta \models \alpha) \models \gamma) \subset \gamma, (\beta \models \alpha) \end{pmatrix};$$

$$\Xi_{\gamma,(\alpha\models\beta)}^{\gamma,(\alpha\models\beta)}((\alpha \models \beta) \subset \gamma) \rightleftharpoons$$

$$\begin{pmatrix} (\gamma \models (\alpha \models \beta)) \subset \gamma, (\alpha \models \beta); \\ ((\alpha \models \beta) \models \gamma) \subset \gamma, (\alpha \models \beta) \end{pmatrix};$$

$$\Xi_{\gamma,\Xi(\alpha\subset\beta)}^{\gamma,\Xi(\alpha\subset\beta)}(\Xi_{\beta,\alpha}^{\beta,\alpha}(\alpha \subset \beta) \subset \gamma) \rightleftharpoons$$

$$\begin{pmatrix} (\gamma \models \Xi_{\beta,\alpha}^{\beta,\alpha}(\alpha \subset \beta)) \subset \gamma, \Xi_{\beta,\alpha}^{\beta,\alpha}(\alpha \subset \beta); \\ (\Xi_{\beta,\alpha}^{\beta,\alpha}(\alpha \subset \beta) \models \gamma) \subset \gamma, \Xi_{\beta,\alpha}^{\beta,\alpha}(\alpha \subset \beta) \end{pmatrix}$$

We see how the complexity of informational includedness rapidly grows by the number of involved entities.

That which has to be clearly kept in mind is

$$((\alpha \subset \beta) \subset \gamma) \not\Rightarrow (\alpha, \beta \subset \gamma)$$

for only the process $\alpha \subset \beta$ is included in γ, but not α and β (a property of *non*-transitivity in case of informational includedness).

Another informational property which follows from extensions $\Xi^{\gamma,(\beta\models\alpha)}_{\gamma,(\beta\models\alpha)}((\beta \models \alpha) \subset \gamma)$ and $\Xi^{\gamma,(\alpha\models\beta)}_{\gamma,(\alpha\models\beta)}((\alpha \models \beta) \subset \gamma)$ is a consequent descending and ascending circularity in respect to γ, that is,

$$(\gamma \models (\beta \models \alpha)) \models \gamma;$$
$$\gamma \models ((\beta \models \alpha) \models \gamma);$$
$$(\gamma \models (\alpha \models \beta)) \models \gamma;$$
$$\gamma \models ((\alpha \models \beta) \models \gamma)$$

respectively. Other, mixed cycles, are also evident. □

Consequence 7 [A Consequence of Serialism of Informational Includedness Concerning the Informing] *A consequence of Definition 4 is simply the following:*

$$\Phi^{\subset}(\alpha_1, \alpha_2, \cdots, \alpha_n) \Longrightarrow \begin{pmatrix} \Phi^{\models}(\alpha_1, \alpha_2, \cdots, \alpha_n); \\ \Phi^{\models}(\alpha_n, \alpha_{n-1}, \cdots, \alpha_1) \end{pmatrix}$$

where for a number $n \geq 2$ of involved entities

$$\Phi^{\models}(\alpha_1, \alpha_2, \cdots, \alpha_n) \rightleftharpoons$$

$$\begin{pmatrix} \Phi^{\models}_1(\alpha_1, \alpha_2, \cdots, \alpha_n) \rightleftharpoons \\ (\alpha_1 \models^* (\alpha_2 \models (\alpha_3 \models (\cdots(\alpha_{n-1} \models \\ \qquad \alpha_n)\cdots))); \\ \\ \Phi^{\models}_2(\alpha_1, \alpha_2, \cdots, \alpha_n) \rightleftharpoons \\ ((\alpha_1 \models \alpha_2) \models^* (\alpha_3 \models (\cdots(\alpha_{n-1} \models \\ \qquad \alpha_n)\cdots))); \\ \vdots \\ \Phi^{\models}_{n-1}(\alpha_1, \alpha_2, \cdots, \alpha_n) \rightleftharpoons \\ (((\cdots((\alpha_1 \models \alpha_2) \models \alpha_3)\cdots) \models \alpha_{n-1}) \models^* \\ \qquad \alpha_n) \end{pmatrix}$$

and adequately

$$\Phi^{\models}(\alpha_n, \alpha_{n-1}, \cdots, \alpha_1) \rightleftharpoons$$

$$\begin{pmatrix} \Phi^{\models}_1(\alpha_n, \alpha_{n-1}, \cdots, \alpha_1) \rightleftharpoons \\ (\alpha_n \models^* (\alpha_{n-1} \models (\alpha_{n-2} \models (\cdots(\alpha_2 \models \\ \qquad \alpha_1)\cdots))); \\ \\ \Phi^{\models}_2(\alpha_n, \alpha_{n-1}, \cdots, \alpha_1) \rightleftharpoons \\ ((\alpha_n \models \alpha_{n-1}) \models^* (\alpha_{n-2} \models (\cdots(\alpha_2 \models \\ \qquad \alpha_1)\cdots))); \\ \vdots \\ \Phi^{\models}_{n-1}(\alpha_n, \alpha_{n-1}, \cdots, \alpha_1) \rightleftharpoons \\ (((\cdots((\alpha_n \models \alpha_{n-1}) \models \alpha_{n-2})\cdots) \models \alpha_2) \models^* \\ \qquad \alpha_1) \end{pmatrix}$$

is the ascending and descending (counterascending) serial sequence of informing in respect to the greatest subscript n. □

Proof. The last consequence considers only the ascending and descending sequences of entities $\alpha_1, \alpha_2, \cdots, \alpha_n$ in respect to the subscript n. Informational entities $\Phi^{\models}(\alpha_1, \alpha_2, \cdots, \alpha_n)$ and $\Phi^{\models}(\alpha_n, \alpha_{n-1}, \cdots, \alpha_1)$ are evident consequences of entity $\Phi^{\subset}(\alpha_1, \alpha_2, \cdots, \alpha_n)$. There exists even a stronger consequence of Definition 4, as presented by the next consequence. Q.E.D.

Consequence 8 [A General Consequence of an Ordered Serialism of Informational Includedness] *The following implication represents the most general system of the ordered serial includedness:*

$$\Phi^{\subset}(\alpha_1, \alpha_2, \cdots, \alpha_n) \Longrightarrow$$

$$\begin{pmatrix} \Phi^{\models}_{ij}(\alpha_i, \alpha_{i+1}, \cdots, \alpha_j); \\ \Phi^{\models}_{ij}(\alpha_j, \alpha_{j-1}, \cdots, \alpha_i); \\ 1 \leq i;\ i < j;\ j \leq n; \\ i = 1, 2, \cdots, n-1; \\ j = 2, 3, \cdots, n \end{pmatrix}$$

where $(i, i+1, \cdots, j)$ and $(j, j-1, \cdots, i)$ is an ascending and descending interval (of natural numbers), respectively, and the length $\ell(\Phi^{\models}_{ij}(\alpha_i, \alpha_{i+1}, \cdots, \alpha_j))$ is between 2 and n. □

Proof. Except by the consequence determined serial sequences, there exist other, 'non-ordered' sequences as can be easily recognized from the previous example. That is, besides the (alphabetically, numerically) 'ordered' sequences, proceeding from $\Phi^{\subset}(\alpha, \beta, \gamma)$ for $\ell = 3$, that is,

$$(\alpha \models \beta) \models \gamma;\ \alpha \models (\beta \models \gamma);$$
$$\gamma \models (\beta \models \alpha);\ (\gamma \models \beta) \models \alpha$$

there are 'non-ordered' sequences

$$(\beta \models \gamma) \models \alpha;\ \beta \models (\gamma \models \alpha);$$
$$(\beta \models \alpha) \models \gamma;\ \beta \models (\alpha \models \gamma);$$
$$(\gamma \models \alpha) \models \beta;\ \gamma \models (\alpha \models \beta)$$

etc. and infinitely many others, recursively arising serial sequences. In this respect, $\Phi^{\subset}(\alpha_1, \alpha_2, \cdots, \alpha_n)$ symbolizes the possible appearance of all $\models$–serial formulas concerning operands α_1, $\alpha_2, \cdots$, α_n. Therefore, symbol $\Longrightarrow$ is used instead of $\rightleftharpoons$ in the last definition and consequences. Q.E.D.

4.5 Circular Serialism of Includedness

Circularity belongs to the most significant faculties of informational serialism. By circular informational formulas the most complex and various phenomena concerning cognitive, intelligent, and understanding processes and entities can be not only conceptualized, but brought into positions and attitudes of informational arising (informational autopoiesis, self-reference, consciousness, etc.). This level of circularity, caused by cyclic informational includedness, reaches its highest point within the circular metaphysicalism.

Definition 5 [Circular Serialism of Informational Includedness] *Let us introduce the markers $\Phi^{\subset}_{\circlearrowleft}$ of circular serialism concerning the informational includedness of entities $\alpha_1, \alpha_2, \cdots, \alpha_n$ in cyclic respect to entity α_1 and mark them as follows:*

$$\Phi^{\subset}_{\circlearrowleft}(\alpha_1, \alpha_2, \cdots, \alpha_n, \alpha_1) \Longrightarrow$$

$$\left(\begin{array}{l}
\Phi^{\subset}_{\circlearrowleft_1}(\alpha_1, \alpha_2, \cdots, \alpha_n, \alpha_1) \rightleftharpoons \\
(\alpha_1 \subset^* (\alpha_2 \subset (\alpha_3 \subset (\cdots(\alpha_n \subset \alpha_1)\cdots))));\\
\\
\Phi^{\subset}_{\circlearrowleft_2}(\alpha_1, \alpha_2, \cdots, \alpha_n, \alpha_1) \rightleftharpoons \\
((\alpha_1 \subset \alpha_2) \subset^* (\alpha_3 \subset (\cdots(\alpha_n \subset \alpha_1)\cdots)));\\
\vdots\\
\Phi^{\subset}_{\circlearrowleft_n}(\alpha_1, \alpha_2, \cdots, \alpha_n, \alpha_1) \rightleftharpoons \\
(((\cdots((\alpha_1 \subset \alpha_2) \subset \alpha_3)\cdots) \subset \alpha_n) \subset^* \alpha_1)
\end{array}\right)$$

By the asterisk marked operators of includedness ($\subset^*$) the main separators between the informing (cyclic informer) and the observing (cyclic observer) part of cyclically structured expression are meant. □

Consequence 9 [A Consequence of Circular Serialism of Informational Includedness Concerning the Informing] *A consequence of Definition 5 is simply the following*

$$\Phi^{\subset}_{\circlearrowleft}(\alpha_1, \alpha_2, \cdots, \alpha_n, \alpha_1) \Longrightarrow$$
$$\left(\begin{array}{l}
\Phi^{\models}_{\circlearrowleft}(\alpha_1, \alpha_2, \cdots, \alpha_n, \alpha_1);\\
\Phi^{\models}_{\circlearrowleft}(\alpha_1, \alpha_n, \alpha_{n-1}, \cdots, \alpha_1)
\end{array}\right)$$

where for $n \geq 2$

$$\Phi^{\models}_{\circlearrowleft}(\alpha_1, \alpha_2, \cdots, \alpha_n, \alpha_1) \rightleftharpoons$$

$$\left(\begin{array}{l}
\Phi^{\models}_{\circlearrowleft_1}(\alpha_1, \alpha_2, \cdots, \alpha_n, \alpha_1) \rightleftharpoons \\
(\alpha_1 \models^* (\alpha_2 \models (\alpha_3 \models (\cdots(\alpha_{n-1} \models \\
\qquad (\alpha_n \models \alpha_1))\cdots))));\\
\\
\Phi^{\models}_{\circlearrowleft_2}(\alpha_1, \alpha_2, \cdots, \alpha_n, \alpha_1) \rightleftharpoons \\
((\alpha_1 \models \alpha_2) \models^* (\alpha_3 \models (\cdots(\alpha_{n-1} \models \\
\qquad (\alpha_n \models \alpha_1))\cdots)));\\
\vdots\\
\Phi^{\models}_{\circlearrowleft_n}(\alpha_1, \alpha_2, \cdots, \alpha_n, \alpha_1) \rightleftharpoons \\
(((\cdots((\alpha_1 \models \alpha_2) \models \alpha_3)\cdots) \models \alpha_{n-1}) \models \\
\qquad \alpha_n) \models^* \alpha_1)
\end{array}\right)$$

and adequately

$$\Phi^{\models}_{\circlearrowleft}(\alpha_1, \alpha_n, \alpha_{n-1}, \cdots, \alpha_1) \rightleftharpoons$$

$$\left(\begin{array}{l}
\Phi^{\models}_{\circlearrowleft_1}(\alpha_1, \alpha_n, \alpha_{n-1}, \cdots, \alpha_1) \rightleftharpoons \\
(\alpha_1 \models^* (\alpha_n \models (\alpha_{n-1} \models (\alpha_{n-2} \models (\cdots(\alpha_2 \\
\qquad \models \alpha_1)\cdots)))));\\
\\
\Phi^{\models}_{\circlearrowleft_2}(\alpha_1, \alpha_n, \alpha_{n-1}, \cdots, \alpha_1) \rightleftharpoons \\
((\alpha_1 \models \alpha_n) \models^* (\alpha_{n-1} \models (\alpha_{n-2} \models (\cdots(\alpha_2 \\
\qquad \models \alpha_1)\cdots)));\\
\vdots\\
\Phi^{\models}_{\circlearrowleft_n}(\alpha_1, \alpha_n, \alpha_{n-1}, \cdots, \alpha_1) \rightleftharpoons \\
(((\cdots(((\alpha_1 \models \alpha_n) \models \alpha_{n-1}) \models \alpha_{n-2})\cdots)) \\
\qquad \models \alpha_2) \models^* \alpha_1)
\end{array}\right)$$

is the ascending and descending (counterascending) circularly serial sequence of informing in respect to circular subscript 1. □

Proof. The cyclicity in respect to entity α is in the α_1's property to be in the position, together with other entities or alone, of the informer (left of operator $\models^*$) and the observer (right of operator $\models^*$), simultaneously. As we see, the cyclicity for other entities than α_1 can not be derived merely from the premise $\Phi^{\subset}_{\circlearrowleft}(\alpha_1, \alpha_2, \cdots, \alpha_n, \alpha_1)$.

The last consequence considers only the ascending and descending circular sequences of entities $\alpha_1, \alpha_2, \cdots, \alpha_n$ in respect to entity α_1. Entities $\Phi^{\models}_{\circlearrowleft}(\alpha_1, \alpha_2, \cdots, \alpha_n, \alpha_1)$ and $\Phi^{\models}_{\circlearrowleft}(\alpha_n, \alpha_{n-1}, \cdots, \alpha_1, \alpha_1)$ are evident consequences of entity $\Phi^{\subset}_{\circlearrowleft}(\alpha_1, \alpha_2, \cdots, \alpha_n, \alpha_1)$. But, there exists a stronger circular and non-circular (linear serial) consequence of Definition 5, as presented by the next consequence. Q.E.D.

According to Consequence 8 it is possible to deduce a similar consequence concerning the ordered cyclical serialism of informational includedness, where complex circular informational entities (operands), considering $\alpha_i \subset \alpha_j$,

$$\Phi^{\models}_{\circlearrowleft_{ij}}(\alpha_i, \alpha_{i+1}, \cdots, \alpha_j, \alpha_i);$$
$$\Phi^{\models}_{\circlearrowleft_{ij}}(\alpha_i, \alpha_j, \alpha_{j-1}, \cdots, \alpha_i)$$

come to existence. The second formula can be viewed as a *countercycle* of the first formula, that is,

$$\Phi^{\models}_{\circlearrowleft_{ij}}(\alpha_i, \alpha_j, \alpha_{j-1}, \cdots, \alpha_i) \rightleftharpoons$$
$$\Phi^{\models}_{\circlearrowleft_{ij}}(\alpha_i, \alpha_{i+1}, \cdots, \alpha_j, \alpha_i)$$

where the difference is in $\circlearrowleft$ and $\circlearrowleft$ subscript operator, respectively.

Consequence 10 [A General Consequence of an Ordered Circular Serialism of Informational Includedness] *The following implication represents the most general system of the ordered circular serial includedness:*

$$\Phi^{\subset}_{\circlearrowleft}(\alpha_1, \alpha_2, \cdots, \alpha_n, \alpha_1) \Longrightarrow$$

$$\begin{pmatrix} \Phi^{\models}_{\circlearrowleft_{ij}}(\alpha_i, \alpha_{i+1}, \cdots, \alpha_j, \alpha_i); \\ \Phi^{\models}_{\circlearrowleft_{ij}}(\alpha_i, \alpha_j, \alpha_{j-1}, \cdots, \alpha_i); \\ 1 \leq i;\ i < j;\ j \leq n; \\ i = 1, 2, \cdots, n-1; \\ j = 2, 3, \cdots, n \end{pmatrix}$$

where $(i, i+1, \cdots, j, i)$ *and* $(j, j-1, \cdots, i, j)$ *is an ascending and descending circular interval (of natural numbers), respectively, and considering* $\alpha_i \subset \alpha_j$. □

A Comment. Circularly serial informational includedness causes an infinite number of possible cycles and subcycles of involved entities. This fact offers the possibilities of choice in a particular case and enables the syntactic and semantic diversity of arising informational cases.

4.6 Externalism, Internalism, and Phenomenalism of Includedness

Let us interpret additionally the appearance of includable externalism, internalism, and phenomenalism with the sense of their introduction into informational discourse.

Informational externalism of includedness concerning an entity says that the entity is a subunit of as yet undetermined informational unit ($\alpha \subset$). The question of the subunit embracing unit is left open and the identification of an adequate unit will appear as the consequence of the happening circumstances (e.g., within an arising formula system). Usually, on the most general level, we have the informational externalism (marked by $\alpha \models$). We arrive to the includable externalism through particularization of operator $\models$, replacing it by operator $\subset$. But, as we have recognized (Definition 1), the includedness (characterized by operator $\subset$) is a recursively and complexly determined form of informationalism (characterized by operator $\models$).

Informational internalism of includedness concerning an entity is a 'reverse' problem to informational externalism and says that the entity is a unit of as yet undetermined informational subunit(s) ($\subset \alpha$). The question of the unit including subunit(s) is left open and the identification (decomposition) of an adequate subunit will appear as the consequence of the happening circumstances (e.g., within an arising formula system). Usually, on the most general level, we have the informational internalism (marked by $\models \alpha$). We arrive to the includable internalism through particularization of operator $\models$, replacing it by operator $\subset$.

Informational phenomenalism of includedness concerning an entity is an informational system of includable externalism and internalism and says that the entity is simultaneously a subunit of as yet undetermined informational unit(s) and a unit of as yet undetermined informational subunit(s) ($\alpha \subset; \subset \alpha$). The questions of the subunit em-

bracing unit(s) and the unit including subunit(s) are left open and the identification (composition and decomposition) of adequate unit(s) and subunit(s) will come to the surface as a consequence of the happening circumstances (e.g., within a complexly arising formula system). Usually, on the most general level, we have the informational complex of externalism and internalism (marked by $\models \alpha; \models \alpha$). We arrive to the includable phenomenalism through particularization of operator types $\models$, replacing them by operator types $\subset$.

4.7 Metaphysicalism of Includedness

Includable metaphysicalism concerns informational parallelism and serialism of several distinguished entities and is a consequence of the general metaphysical structure belonging to an informing entity. To understand the includable metaphysicalism, we have to show the circular parallel and serial schemes (metaphysical shells, structures) of very particular subentities behind (within) the informing entity.

The metaphysical informing of an entity—its metaphysicalism—is constituted by its subentities, which are pragmatically classified as informing, counterinforming, and informational embedding. It is understood that each of these entities has two components: an entity as an informational operand and its explicitly informing (acting) component. Thus, the entire entity α has its specific informing component $\mathcal{I}_\alpha$ in the sense of $\mathcal{I}_\alpha \subset \alpha$. Furthermore, informing of entity α informationally includes the so-called counterinforming of α, marked by $\mathcal{C}_\alpha$. This fact is expressed by the includable formula $\mathcal{C}_\alpha \subset \mathcal{I}_\alpha$. Counterinforming as an active component produces the counterinformational entity γ_α, which is through counterinforming arisen counterinformation. It has to be informationally connected (included) to the frame informational entity α through a distinguished component of informing, called informational embedding and marked by $\mathcal{E}_\alpha$. It is understood that this embedding component is a consequence of the counterinformational entity γ_α in the sense $\mathcal{E}_\alpha \subset \gamma_\alpha$. Informational embedding as an active component of α produces the γ_α-connective informational entity in respect to the frame entity α. This component is marked by ε_α and the corresponding formula of includedness is $\varepsilon_\alpha \subset \mathcal{E}_\alpha$. Last but not least, the embedding informational product ε_α includes α through $\alpha \subset \varepsilon$. By this, the includable cycle of α's metaphysical components comes into existence.

Metaphysicalism of includedness pertaining to an informing entity can unite the metaphysical parallelism and serialism within the entity. This metaphysical complexity of includedness ensures the most powerful and perplexed informational spontaneity (arising) and circularity. By a pragmatical way of filling the complex metaphysical shell, intelligent informational entities can come into appearance.

4.7.1 Metaphysical Parallelism of Includedness

That which we have intuitively described as a basic metaphysical system of an informing entity is, according to Consequence 6, a circular parallelism of informational includedness.

Definition 6 [Metaphysical Parallelism of Informational Includedness Pertaining to an Entity] *Let entities $\mathcal{I}_\alpha$, $\mathcal{C}_\alpha$, γ_α, $\mathcal{E}_\alpha$, and ε_α be metaphysical components of entity α, called α-s informing, counterinforming, counterinformational entity, informational embedding, and informational embedding entity, respectively. Then the following metaphysical, that is circular includable parallelism of the form*

$$\text{entity } \alpha \left\{ \begin{array}{l} \left.\begin{array}{l} \alpha \subset \varepsilon_\alpha; \\ \varepsilon_\alpha \subset \mathcal{E}_\alpha; \end{array}\right\} \alpha\text{'s embedding} \\ \left.\begin{array}{l} \mathcal{E}_\alpha \subset \gamma_\alpha; \\ \gamma_\alpha \subset \mathcal{C}_\alpha \end{array}\right\} \alpha\text{'s counterinforming} \\ \left.\begin{array}{l} \mathcal{C}_\alpha \subset \mathcal{I}_\alpha; \\ \mathcal{I}_\alpha \subset \alpha \end{array}\right\} \alpha\text{'s informing} \end{array} \right.$$

exists. This kind of includedness is called the complete includable parallelism. □

Consequence 11 [Metaphysical Parallelism of Informing Pertaining to an Entity] *A consequence of Definition 6 is the following:*

$$\begin{pmatrix} \alpha \subset \varepsilon_\alpha; \\ \varepsilon_\alpha \subset \mathcal{E}_\alpha; \\ \mathcal{E}_\alpha \subset \gamma_\alpha; \\ \gamma_\alpha \subset \mathcal{C}_\alpha; \\ \mathcal{C}_\alpha \subset \mathcal{I}_\alpha; \\ \mathcal{I}_\alpha \subset \alpha \end{pmatrix} \Longrightarrow \begin{pmatrix} \alpha \models \varepsilon_\alpha; & \alpha \models \mathcal{I}_\alpha; \\ \varepsilon_\alpha \models \mathcal{E}_\alpha; & \mathcal{I}_\alpha \models \mathcal{C}_\alpha; \\ \mathcal{E}_\alpha \models \gamma_\alpha; & \mathcal{C}_\alpha \models \gamma_\alpha; \\ \gamma_\alpha \models \mathcal{C}_\alpha; & \gamma_\alpha \models \mathcal{E}_\alpha; \\ \mathcal{C}_\alpha \models \mathcal{I}_\alpha; & \mathcal{E}_\alpha \models \varepsilon_\alpha; \\ \mathcal{I}_\alpha \models \alpha; & \varepsilon_\alpha \models \alpha \end{pmatrix}$$

The columns right of $\Longrightarrow$ *are parallel metaphysical cycles and, simultaneously, they constitute a so-called double metaphysical cycle with its first (left column) and second (right column) transition. These cycles are countercyclical to each other.* □

The validity of the last consequence is evident and can be derived from Definition 1 and Consequence 3.

Consequence 12 [A Weak Metaphysical Parallelism of Informing Pertaining to an Entity] *For the consequent of Consequence 11 even a weaker (more natural) condition suffices, that is,*

$$\begin{pmatrix} \varepsilon_\alpha \subset \mathcal{E}_\alpha; \\ \mathcal{E}_\alpha \subset \gamma_\alpha; \\ \gamma_\alpha \subset \mathcal{C}_\alpha; \\ \mathcal{C}_\alpha \subset \mathcal{I}_\alpha; \\ \mathcal{I}_\alpha \subset \alpha \end{pmatrix} \Longrightarrow \begin{pmatrix} \alpha \models \varepsilon_\alpha; & \alpha \models \mathcal{I}_\alpha; \\ \varepsilon_\alpha \models \mathcal{E}_\alpha; & \mathcal{I}_\alpha \models \mathcal{C}_\alpha; \\ \mathcal{E}_\alpha \models \gamma_\alpha; & \mathcal{C}_\alpha \models \gamma_\alpha; \\ \gamma_\alpha \models \mathcal{C}_\alpha; & \gamma_\alpha \models \mathcal{E}_\alpha; \\ \mathcal{C}_\alpha \models \mathcal{I}_\alpha; & \mathcal{E}_\alpha \models \varepsilon_\alpha; \\ \mathcal{I}_\alpha \models \alpha; & \varepsilon_\alpha \models \alpha \end{pmatrix}$$

This consequence does not require the explicit condition $\alpha \subset \varepsilon_\alpha$ *which closes the includable metaphysicalism in a circular manner.* □

Proof. Because of the transitivity of informational includedness (Consequence 3), there is

$$\begin{pmatrix} \varepsilon_\alpha \subset \mathcal{E}_\alpha; \\ \mathcal{E}_\alpha \subset \gamma_\alpha; \\ \gamma_\alpha \subset \mathcal{C}_\alpha; \\ \mathcal{C}_\alpha \subset \mathcal{I}_\alpha; \\ \mathcal{I}_\alpha \subset \alpha \end{pmatrix} \Longrightarrow (\varepsilon_\alpha \subset \alpha)$$

This consequence yields (Definition 1)

$$(\varepsilon_\alpha \subset \alpha) \rightleftharpoons \begin{pmatrix} \alpha \models \varepsilon_\alpha; \\ \varepsilon_\alpha \models \alpha; \\ \Xi(\varepsilon_\alpha \subset \alpha) \end{pmatrix}$$

Thus, the necessary transitions $\alpha \models \varepsilon_\alpha$ and $\varepsilon_\alpha \models \alpha$ exist. Q.E.D.

Consequence 13 [A Further Metaphysical Parallelism Pertaining to an Entity] *A further useful consequence of the parallel metaphysical includedness is*

$$\begin{pmatrix} \varepsilon_\alpha \subset \mathcal{E}_\alpha; \\ \mathcal{E}_\alpha \subset \gamma_\alpha; \\ \gamma_\alpha \subset \mathcal{C}_\alpha; \\ \mathcal{C}_\alpha \subset \mathcal{I}_\alpha; \\ \mathcal{I}_\alpha \subset \alpha \end{pmatrix} \Longrightarrow \begin{pmatrix} \varepsilon_\alpha, \mathcal{E}_\alpha, \gamma_\alpha, \mathcal{C}_\alpha \mathcal{I}_\alpha \subset \alpha; \\ \varepsilon_\alpha, \mathcal{E}_\alpha, \gamma_\alpha, \mathcal{C}_\alpha \subset \mathcal{I}_\alpha; \\ \varepsilon_\alpha, \mathcal{E}_\alpha, \gamma_\alpha \subset \mathcal{C}_\alpha; \\ \varepsilon_\alpha, \mathcal{E}_\alpha \subset \gamma_\alpha; \\ \varepsilon_\alpha \subset \mathcal{E}_\alpha \end{pmatrix}$$

which follows directly from Consequence 3. □

4.7.2 Metaphysical Serialism of Includedness

In parallel to metaphysical parallelism of informational includedness there exists the metaphysical serialism of informational includedness which can offer the cyclically most perplexed, interwoven, and involved possibilities, by which intelligent, understanding, or cognitive scenarios (processes, entities) can be constructed.

Definition 7 [Partial Metaphysical Serialism of Informational Includedness Pertaining to an Entity] *Let entities* $\mathcal{I}_\alpha$, $\mathcal{C}_\alpha$, γ_α, $\mathcal{E}_\alpha$, *and* ε_α *be metaphysical components of entity* α, *called* α*'s informing, counterinforming, counterinformational entity, informational embedding, and informational embedding entity, respectively. Then, for example, the following metaphysical and reverse metaphysical, that is circular includable serialisms of the form*

$$\overbrace{\Big(\Big(\underbrace{(\alpha \subset \mathcal{I}_\alpha)}_{\text{informing}} \subset \underbrace{\mathcal{C}_\alpha)}_{\text{counterinforming}} \subset \gamma_\alpha\Big) \subset \underbrace{\mathcal{E}_\alpha\Big)}_{\text{embedding}} \subset \varepsilon_\alpha\Big) \subset \alpha}^{\text{metaphysical informing of } \alpha \text{ as a whole}}$$

and

$$\overbrace{\Big(\Big(\underbrace{(\alpha \subset \varepsilon_\alpha) \subset \mathcal{E}_\alpha)}_{\text{r-embedding}} \underbrace{\subset \gamma_\alpha\Big) \subset \mathcal{C}_\alpha\Big)}_{\text{r-counterinforming}} \underbrace{\subset \mathcal{I}_\alpha\Big) \subset \alpha}_{\text{r-informing}}}^{\text{reverse metaphysical informing of } \alpha \text{ as a whole}}$$

can exist, respectively. This kind of includedness is called the partial includable serialism. *In the second formula, r-embedding, r-counterinforming, and r-informing mark reverse embedding, reverse counterinforming, and reverse informing, respectively.* □

Definition 8 [Multiform Metaphysical Serialism of Informational Includedness Pertaining to an Entity] *According to Definition 7 for two basic forms (ascending, marked by* $\Upsilon^{\subset}_{\supset}(\alpha)$, *and descending or reverse includable metaphysical cycle, marked by* $\Upsilon^{\subset}_{\supset}$, *of an entity* α*), the multiform metaphysical serialism is obtained by considering of all possible positions of the parenthesis pairs, that is,*

$$\Upsilon^{\subset}_{\mathcal{O}}(\alpha) \rightleftharpoons \begin{pmatrix} ((((\alpha \subset \mathcal{I}_\alpha) \subset \mathcal{C}_\alpha) \subset \gamma_\alpha) \subset \mathcal{E}_\alpha) \subset \varepsilon_\alpha) \subset^* \alpha; \\ ((((\alpha \subset \mathcal{I}_\alpha) \subset \mathcal{C}_\alpha) \subset \gamma_\alpha) \subset \mathcal{E}_\alpha) \subset^* (\varepsilon_\alpha \subset \alpha); \\ (((\alpha \subset \mathcal{I}_\alpha) \subset \mathcal{C}_\alpha) \subset \gamma_\alpha) \subset^* (\mathcal{E}_\alpha \subset (\varepsilon_\alpha \subset \alpha)); \\ ((\alpha \subset \mathcal{I}_\alpha) \subset \mathcal{C}_\alpha) \subset^* (\gamma_\alpha \subset (\mathcal{E}_\alpha \subset (\varepsilon_\alpha \subset \alpha))); \\ (\alpha \subset \mathcal{I}_\alpha) \subset^* (\mathcal{C}_\alpha \subset (\gamma_\alpha \subset (\mathcal{E}_\alpha \subset (\varepsilon_\alpha \subset \alpha)))); \\ \alpha \subset^* (\mathcal{I}_\alpha \subset (\mathcal{C}_\alpha \subset (\gamma_\alpha \subset (\mathcal{E}_\alpha \subset (\varepsilon_\alpha \subset \alpha))))) \end{pmatrix}$$

and

$$\Upsilon^{\subset}_{\mathcal{O}}(\alpha) \rightleftharpoons \begin{pmatrix} ((((\alpha \subset \varepsilon_\alpha) \subset \mathcal{E}_\alpha) \subset \gamma_\alpha) \subset \mathcal{C}_\alpha) \subset \mathcal{I}_\alpha) \subset^* \alpha; \\ ((((\alpha \subset \varepsilon_\alpha) \subset \mathcal{E}_\alpha) \subset \gamma_\alpha) \subset \mathcal{C}_\alpha) \subset^* (\mathcal{I}_\alpha \subset \alpha); \\ (((\alpha \subset \varepsilon_\alpha) \subset \mathcal{E}_\alpha) \subset \gamma_\alpha) \subset^* (\mathcal{C}_\alpha \subset (\mathcal{I}_\alpha \subset \alpha)); \\ ((\alpha \subset \varepsilon_\alpha) \subset \mathcal{E}_\alpha) \subset^* (\gamma_\alpha \subset (\mathcal{C}_\alpha \subset (\mathcal{I}_\alpha \subset \alpha))); \\ (\alpha \subset \varepsilon_\alpha) \subset^* (\mathcal{E}_\alpha \subset (\gamma_\alpha \subset (\mathcal{C}_\alpha \subset (\mathcal{I}_\alpha \subset \alpha)))); \\ \alpha \subset^* (\varepsilon_\alpha \subset (\mathcal{E}_\alpha \subset (\gamma_\alpha \subset (\mathcal{C}_\alpha \subset (\mathcal{I}_\alpha \subset \alpha))))) \end{pmatrix}$$

Such an includable system enables that all possible serial metaphysical cycles of both informing (informer) and observing (observer) come into existence. Thus, in the first formula, α is the main (operator $\subset^*$) metaphysical observer, while in the last formula it is the main metaphysical informer. □

Which are the consequences of includably embedded entities (operands) in a metaphysical case? The reader can construct the answers to this question taking into account the consequences pertaining to circular serialism of includedness (Consequence 9 and 10). There are infinitely many serial and circular-serial metaphysical consequences originating in entities $\Upsilon^{\subset}_{\mathcal{O}}(\alpha)$ and $\Upsilon^{\subset}_{\mathcal{O}}(\alpha)$ of the last definition. Let us see only some of the most interesting.

Consequence 14 [A Short-sized Metaphysical Serialism Pertaining to an Entity] *By introspection of Definition 8, one can prove the following implications concerning the short-sized forms of inclusiveness and informing in a metaphysical case:*

$$\Upsilon^{\subset}_{\mathcal{O}}(\alpha) \Longrightarrow \begin{pmatrix} \alpha \subset \mathcal{I}_\alpha; \\ \varepsilon_\alpha \subset \alpha \end{pmatrix}$$

and

$$\Upsilon^{\subset}_{\mathcal{O}}(\alpha) \Longrightarrow \begin{pmatrix} \alpha \subset \varepsilon_\alpha; \\ \mathcal{I}_\alpha \subset \alpha \end{pmatrix}$$

cause implications

$$\Upsilon^{\subset}_{\mathcal{O}}(\alpha) \Longrightarrow \begin{pmatrix} \mathcal{I}_\alpha \models \alpha; \\ \alpha \models \mathcal{I}_\alpha; \\ \alpha \models \varepsilon_\alpha; \\ \varepsilon_\alpha \models \alpha \end{pmatrix}$$

and

$$\Upsilon^{\subset}_{\mathcal{O}}(\alpha) \Longrightarrow \begin{pmatrix} \varepsilon_\alpha \models \alpha; \\ \alpha \models \varepsilon_\alpha; \\ \alpha \models \mathcal{I}_\alpha; \\ \mathcal{I}_\alpha \models \alpha \end{pmatrix}$$

The last two consequences of short-sized informing pertaining to $\Upsilon^{\subset}_{\mathcal{O}}(\alpha)$ and $\Upsilon^{\subset}_{\mathcal{O}}(\alpha)$ have equal consequents, evidently. □

Of course, the so-called includable extensions have been not considered.

Consequence 15 [A Medium-sized Metaphysical Serialism Pertaining to an Entity] *By introspection of Definition 8, we can prove the following implications concerning the medium-sized forms of inclusiveness and informing in a metaphysical case:*

$$\Upsilon^{\subset}_{\mathcal{O}}(\alpha) \Longrightarrow \begin{pmatrix} (\alpha \subset \mathcal{I}_\alpha) \subset \mathcal{C}_\alpha; \\ ((\alpha \subset \mathcal{I}_\alpha) \subset \mathcal{C}_\alpha) \subset \gamma_\alpha; \\ (((\alpha \subset \mathcal{I}_\alpha) \subset \mathcal{C}_\alpha) \subset \gamma_\alpha) \subset \mathcal{E}_\alpha; \\ ((((\alpha \subset \mathcal{I}_\alpha) \subset \mathcal{C}_\alpha) \subset \gamma_\alpha) \subset \mathcal{E}_\alpha) \subset \varepsilon_\alpha; \\ \mathcal{E}_\alpha \subset (\varepsilon_\alpha \subset \alpha); \\ \gamma_\alpha \subset (\mathcal{E}_\alpha \subset (\varepsilon_\alpha \subset \alpha)); \\ \mathcal{C}_\alpha \subset (\gamma_\alpha \subset (\mathcal{E}_\alpha \subset (\varepsilon_\alpha \subset \alpha))); \\ \mathcal{I}_\alpha \subset (\mathcal{C}_\alpha \subset (\gamma_\alpha \subset (\mathcal{E}_\alpha \subset (\varepsilon_\alpha \subset \alpha))))) \end{pmatrix}$$

and

$$\Upsilon^{\subset}_{\mathcal{O}}(\alpha) \Longrightarrow \begin{pmatrix} (\alpha \subset \varepsilon_\alpha) \subset \mathcal{E}_\alpha; \\ ((\alpha \subset \varepsilon_\alpha) \subset \mathcal{E}_\alpha) \subset \gamma_\alpha; \\ (((\alpha \subset \varepsilon_\alpha) \subset \mathcal{E}_\alpha) \subset \gamma_\alpha) \subset \mathcal{C}_\alpha; \\ ((((\alpha \subset \varepsilon_\alpha) \subset \mathcal{E}_\alpha) \subset \gamma_\alpha) \subset \mathcal{C}_\alpha) \subset \mathcal{I}_\alpha; \\ \mathcal{C}_\alpha \subset (\mathcal{I}_\alpha \subset \alpha); \\ \gamma_\alpha \subset (\mathcal{C}_\alpha \subset (\mathcal{I}_\alpha \subset \alpha)); \\ \mathcal{E}_\alpha \subset (\gamma_\alpha \subset (\mathcal{C}_\alpha \subset (\mathcal{I}_\alpha \subset \alpha))); \\ \varepsilon_\alpha \subset (\mathcal{E}_\alpha \subset (\gamma_\alpha \subset (\mathcal{C}_\alpha \subset (\mathcal{I}_\alpha \subset \alpha))))) \end{pmatrix}$$

cause implication

$$\Upsilon^{\subseteq}_{\mathcal{O}}(\alpha), \Upsilon^{\subseteq}_{\mathcal{O}}(\alpha) \Longrightarrow$$

$$\left(\begin{array}{l}
(\alpha \models \mathcal{I}_\alpha) \models \mathcal{C}_\alpha; \\
\mathcal{C}_\alpha \models (\mathcal{I}_\alpha \models \alpha); \\
((\alpha \models \mathcal{I}_\alpha) \models \mathcal{C}_\alpha) \models \gamma_\alpha; \\
\gamma_\alpha \models (\mathcal{C}_\alpha \models (\mathcal{I}_\alpha \models \alpha)); \\
(((\alpha \models \mathcal{I}_\alpha) \models \mathcal{C}_\alpha) \models \gamma_\alpha) \models \mathcal{E}_\alpha; \\
\mathcal{E}_\alpha \models (\gamma_\alpha \models (\mathcal{C}_\alpha \models (\mathcal{I}_\alpha \models \alpha))); \\
((((\alpha \models \mathcal{I}_\alpha) \models \mathcal{C}_\alpha) \models \gamma_\alpha) \models \mathcal{E}_\alpha) \models \varepsilon_\alpha; \\
\varepsilon_\alpha \models (\mathcal{E}_\alpha \models (\gamma_\alpha \models (\mathcal{C}_\alpha \models (\mathcal{I}_\alpha \models \alpha)))); \\
\mathcal{E}_\alpha \models (\varepsilon_\alpha \models \alpha); \\
(\alpha \models \varepsilon_\alpha) \models \mathcal{E}_\alpha; \\
\gamma_\alpha \models (\mathcal{E}_\alpha \models (\varepsilon_\alpha \models \alpha)); \\
((\alpha \models \varepsilon_\alpha) \models \mathcal{E}_\alpha) \models \gamma_\alpha; \\
\mathcal{C}_\alpha \models (\gamma_\alpha \models (\mathcal{E}_\alpha \models (\varepsilon_\alpha \models \alpha))); \\
(((\alpha \models \varepsilon_\alpha) \models \mathcal{E}_\alpha) \models \gamma_\alpha) \models \mathcal{C}_\alpha; \\
\mathcal{I}_\alpha \models (\mathcal{C}_\alpha \models (\gamma_\alpha \models (\mathcal{E}_\alpha \models (\varepsilon_\alpha \models \alpha)))); \\
((((\alpha \models \varepsilon_\alpha) \models \mathcal{E}_\alpha) \models \gamma_\alpha) \models \mathcal{C}_\alpha) \models \mathcal{I}_\alpha
\end{array}\right)$$

Consequents of $\Upsilon^{\subseteq}_{\mathcal{O}}(\alpha)$ *and* $\Upsilon^{\subseteq}_{\mathcal{O}}(\alpha)$ *coincide, evidently.* □

Consequence 16 [A Long-sized Metaphysical Serialism Pertaining to an Entity] *By introspection of Definition 8, we can prove the following implication concerning the long-sized form of inclusiveness and informing in a metaphysical case:*

$$\Upsilon^{\subseteq}_{\mathcal{O}}(\alpha), \Upsilon^{\subseteq}_{\mathcal{O}}(\alpha) \Longrightarrow$$

$$\left(\begin{array}{l}
((((\alpha \models \mathcal{I}_\alpha) \models \mathcal{C}_\alpha) \models \gamma_\alpha) \models \mathcal{E}_\alpha) \models \varepsilon_\alpha) \models^* \alpha; \\
\alpha \models^* (\varepsilon_\alpha \models (\mathcal{E}_\alpha \models (\gamma_\alpha \models (\mathcal{C}_\alpha \models (\mathcal{I}_\alpha \models \alpha))))); \\
((((\alpha \models \mathcal{I}_\alpha) \models \mathcal{C}_\alpha) \models \gamma_\alpha) \models \mathcal{E}_\alpha) \models^* (\varepsilon_\alpha \models \alpha); \\
(\alpha \models \varepsilon_\alpha) \models^* (\mathcal{E}_\alpha \models (\gamma_\alpha \models (\mathcal{C}_\alpha \models (\mathcal{I}_\alpha \models \alpha)))); \\
(((\alpha \models \mathcal{I}_\alpha) \models \mathcal{C}_\alpha) \models \gamma_\alpha) \models^* (\mathcal{E}_\alpha \models (\varepsilon_\alpha \models \alpha)); \\
((\alpha \models \varepsilon_\alpha) \models \mathcal{E}_\alpha) \models^* (\gamma_\alpha \models (\mathcal{C}_\alpha \models (\mathcal{I}_\alpha \models \alpha))); \\
((\alpha \models \mathcal{I}_\alpha) \models \mathcal{C}_\alpha) \models^* (\gamma_\alpha \models (\mathcal{E}_\alpha \models (\varepsilon_\alpha \models \alpha))); \\
(((\alpha \models \varepsilon_\alpha) \models \mathcal{E}_\alpha) \models \gamma_\alpha) \models^* (\mathcal{C}_\alpha \models (\mathcal{I}_\alpha \models \alpha)); \\
(\alpha \models \mathcal{I}_\alpha) \models^* (\mathcal{C}_\alpha \models (\gamma_\alpha \models (\mathcal{E}_\alpha \models (\varepsilon_\alpha \models \alpha)))); \\
((((\alpha \models \varepsilon_\alpha) \models \mathcal{E}_\alpha) \models \gamma_\alpha) \models \mathcal{C}_\alpha) \models^* (\mathcal{I}_\alpha \models \alpha); \\
\alpha \models^* (\mathcal{I}_\alpha \models (\mathcal{C}_\alpha \models (\gamma_\alpha \models (\mathcal{E}_\alpha \models (\varepsilon_\alpha \models \alpha))))); \\
((((\alpha \models \varepsilon_\alpha) \models \mathcal{E}_\alpha) \models \gamma_\alpha) \models \mathcal{C}_\alpha) \models \mathcal{I}_\alpha) \models^* \alpha
\end{array}\right)$$

For the listed long-sized metaphysical forms of informing only one of entities $\Upsilon^{\subseteq}_{\mathcal{O}}(\alpha)$ *and* $\Upsilon^{\subseteq}_{\mathcal{O}}(\alpha)$ *must be given.* □

4.7.3 A Mixed Parallel-serial Metaphysical Case

The most complex case of an entity metaphysicalism can be achieved by the mixture of both principles, the parallel and the serial one, at any point of the metaphysical informing, as a consequence of the parallel, serial, and metaphysical informational includedness. One can easily and in an arbitrary manner construct such cases.

4.7.4 A Pragmatic Filling of the Parallel and Serial Metaphysical Shells

The basic question is, how can a metaphysical shell be filled to achieve, for example, intelligent functions of an informational entity. Candidates for such a filling of the metaphysical shells are principles of reasoning and understanding, by which reason and meaning are informed, respectively. A mode of reasoning producing reason can be seen as a counterinforming component, while a mode of understanding producing meaning can be seen as an embedding component for that which has arisen by reasoning. Thus reason is embedded into the existing informational entity by meaning, which is the connecting information between the arisen reason and the informational content of the informational entity.

Definition 9 [Reasoning and Understanding Components as an Informational Entity Includedness] *Reasoning* $\mathcal{R}$ *and understanding* $\mathcal{U}$ *are attributes of an intelligently informing entity* ι*, which can demonstrate its reasonable informing through an adequate filling of its metaphysical shells by intelligently informing, reasoning, and understanding components as counterparts to informing, counterinforming, and embedding, respectively. Let us define the following parallel arrays:*

$$\iota(\xi) \rightleftharpoons \begin{pmatrix} \iota^1(\xi); \\ \iota^2(\xi); \\ \vdots \\ \iota^{i_\iota}(\xi) \end{pmatrix}; \ \mathcal{I}_\iota(\xi) \rightleftharpoons \begin{pmatrix} \mathcal{I}^1_\iota(\xi); \\ \mathcal{I}^2_\iota(\xi); \\ \vdots \\ \mathcal{I}^{i_\mathcal{I}}_\iota(\xi) \end{pmatrix}$$

are the parallel arrays of intelligent intelligent entity components $\iota^m(\xi)$ *and its informing components of* $\mathcal{I}^n_\iota$ *concerning an exterior or interior (also complex) entity, marked by* ξ*. A pair of reasoning components of the form*

$$\mathcal{R}_\iota(\xi) \rightleftharpoons \begin{pmatrix} \mathcal{R}^1_\iota(\xi); \\ \mathcal{R}^2_\iota(\xi); \\ \vdots \\ \mathcal{R}^{i_\mathcal{R}}_\iota(\xi) \end{pmatrix}; \ \rho_\iota(\xi) \rightleftharpoons \begin{pmatrix} \rho^1_\iota(\xi); \\ \rho^2_\iota(\xi); \\ \vdots \\ \rho^{i_\rho}_\iota(\xi) \end{pmatrix}$$

depicts the reasonably informing components $\mathcal{R}_\iota^i(\xi)$ *together with the reason components* $\rho_\iota^j(\xi)$. *At last,*

$$\mathcal{U}_\iota(\xi) \rightleftharpoons \begin{pmatrix} \mathcal{U}_\iota^1(\xi); \\ \mathcal{U}_\iota^2(\xi); \\ \vdots \\ \mathcal{U}_\iota^{i_\mathcal{U}}(\xi) \end{pmatrix}; \ \mu_\iota(\xi) \rightleftharpoons \begin{pmatrix} \mu_\iota^1(\xi); \\ \mu_\iota^2(\xi); \\ \vdots \\ \mu_\iota^{i_\mu}(\xi) \end{pmatrix}$$

are understanding components $\mathcal{U}_\iota^k(\xi)$ *and meaning components* $\mu_\iota^\ell(\xi)$ *concerning entity* ξ. □

Let us show the filling of metaphysical shells in Consequence 16, representing the long-sized, that is, serially the most complex loops of informing, however expressed in the informationally includable form and, thus, giving the implemented filling of shells a choice of an infinite number of all possible extensions (symbols of indexed Ξ).

The filling of the shells can be understood as a particularization (decomposition) process by which several substitutions of symbols take place. These substitutions are as follows:

(1) Shell entity α is replaced by intelligent parallel array $\iota(\xi)$;

(2) shell's informing $\mathcal{I}_\alpha$ is replaced by intelligent parallel array $\mathcal{I}_\iota(\xi)$;

(3) shell's counterinforming $\mathcal{C}_\alpha$ is replaced by reasoning parallel array $\mathcal{R}_\iota(\xi)$;

(4) shell's counterinformational entity γ_α is replaced by reason parallel array $\rho_\iota(\xi)$;

(5) shell's embedding $\mathcal{E}_\alpha$ is replaced by understanding parallel array $\mathcal{U}_\iota(\xi)$;

(6) shell's embedding informational entity ε_α is replaced by meaning parallel array $\mu_\iota(\xi)$; and

(7) shell's operators $\models$ are replaced by specifically complex (universal) operators $\subset$.

Consequence 17 [Filling the Shell of a Long-sized Metaphysical Includedness] *Let us have the following metaphysical shell filling when entity* ι *observes entity* ξ*:*

$$\xi \models$$

$$\begin{pmatrix} ((((\iota(\xi) \subset \mathcal{I}_\iota(\xi)) \subset \mathcal{R}_\iota(\xi)) \subset \rho_\iota(\xi)) \subset \\ \quad \mathcal{U}_\iota(\xi)) \subset \mu_\iota(\xi)) \subset^* \iota(\xi); \\ \iota(\xi) \subset^* (\mu_\iota(\xi) \subset (\mathcal{U}_\iota(\xi) \subset (\rho_\iota(\xi) \subset \\ \quad (\mathcal{R}_\iota(\xi) \subset (\mathcal{I}_\iota(\xi) \subset \iota(\xi))))); \\ ((((\iota(\xi) \subset \mathcal{I}_\iota(\xi)) \subset \mathcal{R}_\iota(\xi)) \subset \rho_\iota(\xi)) \subset \\ \quad \mathcal{U}_\iota(\xi)) \subset^* (\mu_\iota(\xi) \subset \iota(\xi)); \\ (\iota(\xi) \subset \mu_\iota(\xi)) \subset^* (\mathcal{U}_\iota(\xi) \subset (\rho_\iota(\xi) \subset \\ \quad (\mathcal{R}_\iota(\xi) \subset (\mathcal{I}_\iota(\xi) \subset \iota(\xi))))); \\ (((\iota(\xi) \subset \mathcal{I}_\iota(\xi)) \subset \mathcal{R}_\iota(\xi)) \subset \rho_\iota(\xi)) \subset^* \\ \quad (\mathcal{U}_\iota(\xi) \subset (\mu_\iota(\xi) \subset \iota(\xi))); \\ ((\iota(\xi) \subset \mu_\iota(\xi)) \subset \mathcal{U}_\iota(\xi)) \subset^* (\rho_\iota(\xi) \subset \\ \quad (\mathcal{R}_\iota(\xi) \subset (\mathcal{I}_\iota(\xi) \subset \iota(\xi)))); \\ ((\iota(\xi) \subset \mathcal{I}_\iota(\xi)) \subset \mathcal{R}_\iota(\xi)) \subset^* (\rho_\iota(\xi) \subset \\ \quad (\mathcal{U}_\iota(\xi) \subset (\mu_\iota(\xi) \subset \iota(\xi)))); \\ (((\iota(\xi) \subset \mu_\iota(\xi)) \subset \mathcal{U}_\iota(\xi)) \subset \rho_\iota(\xi)) \subset^* \\ \quad (\mathcal{R}_\iota(\xi) \subset (\mathcal{I}_\iota(\xi) \subset \iota(\xi))); \\ (\iota(\xi) \subset \mathcal{I}_\iota(\xi)) \subset^* (\mathcal{R}_\iota(\xi) \subset (\rho_\iota(\xi) \subset \\ \quad (\mathcal{U}_\iota(\xi) \subset (\mu_\iota(\xi) \subset \iota(\xi))))); \\ ((((\iota(\xi) \subset \mu_\iota(\xi)) \subset \mathcal{U}_\iota(\xi)) \subset \rho_\iota(\xi)) \subset \\ \quad \mathcal{R}_\iota(\xi)) \subset^* (\mathcal{I}_\iota(\xi) \subset \iota(\xi)); \\ \iota(\xi) \subset^* (\mathcal{I}_\iota(\xi) \subset (\mathcal{R}_\iota(\xi) \subset (\rho_\iota(\xi) \subset \\ \quad (\mathcal{U}_\iota(\xi) \subset (\mu_\iota(\xi) \subset \iota(\xi))))); \\ ((((\iota(\xi) \subset \mu_\iota(\xi)) \subset \mathcal{U}_\iota(\xi)) \subset \rho_\iota(\xi)) \subset \\ \quad \mathcal{R}_\iota(\xi)) \subset \mathcal{I}_\iota(\xi)) \subset^* \iota(\xi) \end{pmatrix}$$

The listed long-sized metaphysical forms of intelligent informing, reasoning, and understanding constitute a parallel system of serial parallel formulas for ι*'s observing of* ξ. □

5 Includedness as a Logical Contradiction

The traditional (mathematical, logical) understanding of includedness (inclusion, inclusiveness) may seriously contradict the understanding of informational includedness (Being-in, involvement, interweavement, interrelation, interconnection of informational components, etc.). In the first case, the accent is given to the word *in,* while in the second case, the word *inter* (as interiority) is emphasized. Informational includedness expresses the inner character or the inward nature of informational something within informational something. For example, the so-called problem of *internal representation* [3, 11] concerns the problem of includedness. On the other hand, the philosophical Being-in seems to cover the substantial

part of the broadened realm of informational includedness. *Subjectivity and interiority are the notions acquired by the human mind* (W. James, 1890 [13]).

Let us see the controversial notions which may substantially touch the first and the second understanding of includedness. The difference between the traditional and informational understanding of includedness comes to the surface in case

$$(\alpha \subset \beta) \rightleftharpoons \begin{pmatrix} \beta \models \alpha; \\ \alpha \models \beta; \\ \Xi(\alpha \subset \beta) \end{pmatrix}; \ \beta \subset \alpha \rightleftharpoons \begin{pmatrix} \alpha \models \beta; \\ \beta \models \alpha; \\ \Xi(\beta \subset \alpha) \end{pmatrix}$$

where for includable extensions $\Xi(\alpha \subset \beta)$ and $\Xi(\beta \subset \alpha)$, there is

$$\Xi(\alpha \subset \beta),\ \Xi(\beta \subset \alpha) \in \mathcal{P}\left(\left\{\begin{matrix} (\beta \models \alpha) \subset \beta, \\ (\alpha \models \beta) \subset \beta, \\ (\beta \models \alpha) \subset \alpha, \\ (\alpha \models \beta) \subset \alpha \end{matrix}\right\}\right)$$

and, thus,

$$\Xi^{\beta,\alpha}_{\beta,\alpha}(\alpha \subset \beta) = \Xi^{\alpha,\beta}_{\alpha,\beta}(\beta \subset \alpha)$$

This case does not deliver a difference between includednesses $\alpha \subset \beta$ and $\beta \subset \alpha$ and is to this extent contradictory. But, the difference becomes quite senseful in case of

$$\Xi^{\beta}_{\beta}(\alpha \subset \beta) \rightleftharpoons \begin{pmatrix} (\beta \models \alpha) \subset \beta; \\ (\alpha \models \beta) \subset \beta \end{pmatrix};$$

$$\Xi^{\alpha}_{\alpha}(\beta \subset \alpha) \rightleftharpoons \begin{pmatrix} (\alpha \models \beta) \subset \alpha; \\ (\beta \models \alpha) \subset \alpha \end{pmatrix}$$

where, in the first case, β is the dominant entity possessing the informing control over the transitions $\beta \models \alpha$ and $\alpha \models \beta$, while in the second case this role belongs to α.

The contradictory case to the traditional understanding concerns the extensional example $\Xi^{\beta,\alpha}_{\beta,\alpha}(\alpha \subset \beta)$ since simultaneously the control of the dominant, β, and the inclusively subordinated component, α, is requested. But, this contradiction may appear as a (clear) prejudice in the realm of informational.

Similar situation appears at the serially circular (e.g., metaphysical) includedness, which enables the serially circular informing in one and the other direction (e.g.,

$$((((\iota \subset \mathcal{I}_\iota) \subset \mathcal{R}_\iota) \subset \rho_\iota) \subset \mathcal{U}_\iota) \subset \mu_\iota) \subset^* \iota$$

as an intelligently, that is reasonedly and understandingly structured informational shell). It is clear that in this case a serially embedded includedness and, at the end, circularly closed includedness must take place. The traditional doubts of this sort show how a spontaneous and circular informational phenomenalism can not only surpass but can also make the notional obstacles of such kind informationally (intelligently) productive and senseful.

6 Includedness and Reasoning (Inference)

Includedness (informational Being-in) and reasoning are essentially related informational entities. Reasoning (or inference) is possible only within a context of relatedness between a reason (cause, informational motive) and a certain informational consequence pertaining to the reason as a legalized fact. Historically, three basic ways of inference can be distinguished: deduction, induction, and abduction. All of them are philosophically and scientifically well-determined in respect to their formalistic power and practical disadvantages (reductionism, simplification, scientific straightforwardness, etc.). Within the discussion, their connectedness with the concept of informational includedness, that is, as an includable inference processibility, cannot be denied.

The includable informational modi presented are more concretized formulas of reasoning in one or another way (deductively, inductively, and/or abductively). For example, modus ponens is usually meant as a strict deductive principle, while modus tollens inclines to be inductive and modus obliquus abductive. Includable informational modi are typical scenarios (formulas) of informational inferring.

6.1 Includable Deduction

How does the informational Being-in concern the so-called deduction and what does the includable deduction mean? Does there exist a substantial connection between the Being-in as an informational phenomenon on one side and the deduction as a logical (inferential, derivative, conclusive) principle on the other side? Includable deduction seems to be our everyday principle of 'common-

sense' inference of which we are not being always sufficiently aware.

Let us keep in mind the following facts concerning the processes of deduction: deduction means inference by reasoning from generals (universals) to particulars. E.g., particularizing informational operands and especially operators is a kind of 'hidden' (unconscious) deduction. Deducing (or deriving) theorems (conclusions, consequences, lemmas, etc.) from systems of axioms (definitions, hypotheses, etc.) is a characteristic deductive procedure in abstract theories (systems, mathematics, abstract sciences). Deduction opposes induction by reduction, if reduction is meant to be particularization (derivation from universals). The principle of conditionalization (known as 'deduction theorem') was already taken for granted by Aristotle.

At the beginning, let us list three 'deductive' informational operators:

- $\Longrightarrow$ is the most common deductive operator and its meaning is the following: $\alpha \Longrightarrow \beta$ means *if entity (operand) α is given (informationally existent), then it is permitted to transit to entity (operand) β.*

- $\longrightarrow$ (or $\supset$) is a narrower deductive operator and its meaning is: $\alpha \longrightarrow \beta$ (or $\alpha \supset \beta$) means *if entity (operand) α, then β.* We rarely use this type of deductive operator.

- $\overset{\cdot\cdot\cdot}{=}$ (or $\prec$) denotes a complex and to some extent informationally precise deductive operator, which meaning is: $\frac{\alpha}{\beta}$ (or $\alpha \prec \beta$) means *from α there follows β (or α precedes β, also α implies β.)* Entity (operand) α usually denotes a complex (parallel) informational system.

In which way do the listed deductive operators concern informational includedness (informational operator $\subset$)?

In an experiential situation, deduction does not already concern the truth, but proceeds from a *hypothetical* informational entity (situation) to the *prognostic* informational entity. Also, weaker logical deduction rules can exist, for instance, those of the form $\alpha \Longrightarrow (\alpha \vee \beta)$ or $(\alpha, \neg\alpha) \Longrightarrow \gamma$, where γ is an arbitrary entity (from the false, $\neg\alpha$, an arbitrary formula can be logically deduced).

By Definition 1, the Being-in operator $\subset$ is defined complexly in regard to the general (yet non-particularized) operator $\models$ and to the operands α and β. Thus, we have the following consequence when particularizing of operator $\models$ to operator $\Longrightarrow$ is taking place.

Consequence 18 [Deduction Concerning Informational Includedness] *According to Definition 1, when implicatively particularizing operator $\models$, that is, $\models_{\Rightarrow}$ is equivalent to $\Longrightarrow$, there is*

$$(\alpha \subset_{\Rightarrow} \beta) \rightleftharpoons_{\mathrm{Def}} \begin{pmatrix} \beta \Longrightarrow \alpha; \\ \alpha \Longrightarrow \beta; \\ \Xi(\alpha \subset_{\Rightarrow} \beta) \end{pmatrix}$$

where for the extensional part $\Xi(\alpha \subset_{\Rightarrow} \beta)$ of the includable deduction $\alpha \subset_{\Rightarrow} \beta$, there is,

$$\Xi(\alpha \subset_{\Rightarrow} \beta) \in \mathcal{P} \left(\begin{Bmatrix} (\beta \Longrightarrow \alpha) \subset_{\Rightarrow} \beta, \\ (\alpha \Longrightarrow \beta) \subset_{\Rightarrow} \beta, \\ (\beta \Longrightarrow \alpha) \subset_{\Rightarrow} \alpha, \\ (\alpha \Longrightarrow \beta) \subset_{\Rightarrow} \alpha \end{Bmatrix} \right)$$

The most complex element of this power set is denoted by

$$\Xi^{\beta,\alpha}_{\beta,\alpha}(\alpha \subset_{\Rightarrow} \beta) \rightleftharpoons \begin{pmatrix} (\beta \Longrightarrow \alpha) \subset_{\Rightarrow} \beta, \alpha; \\ (\alpha \Longrightarrow \beta) \subset_{\Rightarrow} \beta, \alpha \end{pmatrix}$$

Cases, where $\Xi(\alpha \subset_{\Rightarrow} \beta) \rightleftharpoons \emptyset$ and $\emptyset$ denotes an empty entity (informational nothing), are exceptional (reductionistic). □

To get even a more transparent impression what is going on with the last consequence, we can write definiens of definiendum in Consequence 18 by sample formulas

$$\begin{pmatrix} \beta \longrightarrow \alpha; \\ \alpha \longrightarrow \beta; \\ \Xi(\alpha \subset_{\rightarrow} \beta) \end{pmatrix} \text{ and } \begin{pmatrix} \frac{\beta}{\alpha}; \\ \\ \frac{\alpha}{\beta}; \\ \Xi(\alpha \subset_{\overset{\cdot\cdot\cdot}{=}} \beta) \end{pmatrix}$$

We can now discuss the deductive character of the operation of informational includedness (operator $\subset$ in a universal or particular form) and vice versa. Where lies the deductive point of informational includedness?

Deduction by itself is nothing else than a kind of informational involvement. Otherwise, the concept of deduction (coming from the Latin 'deducere', the German 'herabführen', and the English 'lead away' or 'trace the course of') would not be possible. To deduce something informationally means to extract it informationally (in German, abtrennen) out of something, certainly not in an informationally total (strictly including), but also in an initializing or initially arising (involving) informational way. Includable deduction is an arising informational phenomenon, emerging out of a deductively happening (intentional) situation. Within this illumination we have to explain the connection existing between the definition of informational includedness and the principles of deduction, expressed in the form of the so-called deductive rules. We should also make a clear distinction between deductive and inductive nature of the inference rules. The deductive always roots in previously strict causes and arises as a clear and constructively structured consequence. The inductive makes an intuitive jump from the particular to general and afterwards proceeds deductively. The question is if this jump is deductively legal.

Consequence 19 [A Primitive Circular Structure of Deduction Concerning Informational Includedness] *Considering the most complex extensional element in Consequence 18,* $\Xi^{\beta,\alpha}_{\beta,\alpha}(\alpha \subset_{\Rightarrow} \beta)$, *there is,*

$$\begin{pmatrix} (\beta \Longrightarrow \alpha) \subset_{\Rightarrow} \beta, \alpha; \\ (\alpha \Longrightarrow \beta) \subset_{\Rightarrow} \beta, \alpha \end{pmatrix} \rightleftharpoons$$

$$\begin{pmatrix} \beta \Longrightarrow (\beta \Longrightarrow \alpha);\ (\beta \Longrightarrow \alpha) \Longrightarrow \beta; \\ \Xi^{\beta,(\beta\Longrightarrow\alpha)}_{\beta,(\beta\Longrightarrow\alpha)}((\beta \Longrightarrow \alpha) \subset_{\Rightarrow} \beta); \\ \alpha \Longrightarrow (\beta \Longrightarrow \alpha);\ (\beta \Longrightarrow \alpha) \Longrightarrow \alpha; \\ \Xi^{\alpha,(\beta\Longrightarrow\alpha)}_{\alpha,(\beta\Longrightarrow\alpha)}((\beta \Longrightarrow \alpha) \subset_{\Rightarrow} \alpha); \\ \beta \Longrightarrow (\alpha \Longrightarrow \beta);\ (\alpha \Longrightarrow \beta) \Longrightarrow \beta; \\ \Xi^{\beta,(\alpha\Longrightarrow\beta)}_{\beta,(\alpha\Longrightarrow\beta)}((\alpha \Longrightarrow \beta) \subset_{\Rightarrow} \beta); \\ \alpha \Longrightarrow (\alpha \Longrightarrow \beta);\ (\alpha \Longrightarrow \beta) \Longrightarrow \alpha; \\ \Xi^{\alpha,(\alpha\Longrightarrow\beta)}_{\alpha,(\alpha\Longrightarrow\beta)}((\alpha \Longrightarrow \beta) \subset_{\Rightarrow} \alpha) \end{pmatrix}$$

The most interesting cases of circular implication are

$$(\beta \Longrightarrow \alpha) \Longrightarrow \beta;\ \alpha \Longrightarrow (\beta \Longrightarrow \alpha);$$
$$\beta \Longrightarrow (\alpha \Longrightarrow \beta);\ (\alpha \Longrightarrow \beta) \Longrightarrow \alpha$$

where, in the first case, β *involves* α *and, then, this involvement involves* β *again, etc. In this way the process of involvement (informational includedness) proceeds (e.g. deductively improves in an implicative manner) circularly.* □

6.2 Includable Induction

A strict separation between deduction and induction seems to be probably impossible. For instance, induction concerns derivation from something similarly as deduction. Induction does not mean bringing something into existence from nothing—at least not in the traditional sciences. Within informational theory, induction (as well as deduction) concerns informational arising, for example, counterinforming and informational embedding of the arisen informational entities.

Our question remains, how does the informational Being-in concern the so-called induction and what does the includable induction mean? We have to repeat the following questions: Does there exist a substantial connection between the Being-in as an informational phenomenon on one side and the induction as a logical (intuitive, inferential, derivative, conclusive) principle on the other side? Includable induction is a deeply implanted everyday intuitive principle of common sense and of the informational nature of things (discourses, speech acts, behaviors).

Let us keep in mind the following facts concerning the processes of induction: induction is the informational action of introducing and initiating in (arising, counterinforming). It is, for example, introduction and initiation of knowledge of something, that which leads to something (new). It is the initial step in logical (informational, also intuitive) understanding (undertaking). In this sense, induction is a process of inferring a general law (principles, axioms, hypotheses) from the observation of particular instances (e.g., $\grave{\varepsilon}\pi\alpha\gamma\omega\gamma\acute{\eta}$ in Greek, means a bringing on, an advancing). Induction is a wider (transitive) sense of inference. If a theorem is true in one case, it is true in another case which may be called the *next* case. The prove is made by trial [13].

Induction also means inference by reasoning from particulars to generals (universals). E.g., universalizing (generalizing) informational operands and especially operators is a kind of 'hidden' (unconscious) induction. Inducing (or in-

tuitively deriving, introducing) systems of axioms (definitions, hypotheses, etc.) with the intention to deduce theorems (conclusions, consequences, lemmas, etc.) is a characteristic inductive procedure in abstract theories (systems, mathematics, abstract sciences). Induction opposes deduction by generalization, if generalization is meant to be universalization (intuitive derivation from particulars).

Thus, three 'deductive' informational operators, $\Longrightarrow$, $\longrightarrow$, and $\dddot{=}$, can function also inductively because of the informational-arising nature of informational entities. Circular and particularly metaphysical scenarios of informing are inductive in the sense of introducing new entities into informational cycles and, in parallel, initiating the interpreting formulas for already existing entities. In this sense, induction is a substantial phenomenon of informational decomposition and composition [9].

Informational Being-in offers opportunities which can be taken into consideration. These opportunities have their roots in the recursive character of informational includedness and in the additional possibilities of pragmatic nature of decomposition and composition of informational entities and systems (formulas). The concept of informational inclusivism conditions the concept of inductivism, being an informationally inclusive phenomenon, proceeding, for instance, from granted particularities to certain universalities, being informationally involved by the first ones. A radical initial intuitive step is the introduction of the so-called informatio prima (the first of informational axioms) [10].

6.3 Includable Abduction

Abduction is a special (in traditional science, illegal) way of deduction which may include elementary induction too. In the similar way as deduction and induction, abduction as such concerns informational includedness. It means a leading away in the informational sense. For instance, it can represent a syllogism, of which the major premise (antecedent) is certain, and the minor only probable, so that the conclusion has only the probability of the minor [13]. But, this view of abduction does not embrace its entire informational realm, which can consider an initial (introductory) entity and then proceed away (e.g., counterinformationally) to another possible (probable) entity by a degree of similarity, sporadicalness, relatedness, etc. This is a characteristic phenomenon of abduction, a progress from one informational situation (attitude) to another, when the first being once informationally legalized (demonstrated, approved) and afterwards employed to the proving of other situations (attitudes).

Abduction may represent an indirect proof, like the apagoge, which means syllogistic reasoning, by which a thing is not directly proved, but shows, for example, the absurdity or impossibility of denying the thing in a certain, particularly informational way. Sometimes, it is called *reductio ad absurdum.* A good example of abductive reasoning is perhaps the so-called informational modus obliquus (see later).

6.4 Includable MODUS PONENS

Includable modus ponens is an informational inference rule constructed in the sense as it is known in symbolic logic. This rule uses a true conjunction of an affirmative (true) statement and a true implication of the affirmative and some other statement. In this situation the truth can be decided for the other statement. In our case, instead of truth, we have a certain value of including informing, conjunction is replaced by an informational operator of parallelism (e.g., semicolon ';', symbol $\|$, or a proper parallel informational operator $\|\models$). We also introduce the informational operator of implication $\Longrightarrow$ with the meaning 'implies/imply'.

Inference Rule 1 [Includable MODUS PONENS] *Informational modus ponens can be expressed in terms of informational externalism, internalism, metaphysicalism, and phenomenalism giving 16 basic inference rules concerning an entity α's includedness. We list only four characteristic cases.*

The rule for an externalistic inference *on including externalism $\beta \subset$ from externalisms $\alpha \subset$ and $\beta \models$ is*

$$\frac{\alpha \subset;\ ((\alpha \subset) \Longrightarrow (\beta \models)) \subset}{\beta \subset}$$

A similar rule for an internalistic inference *on including internalism $\subset \beta$ from internalisms $\subset \alpha$ and $\Longrightarrow \beta$ is*

$$\frac{\subset \alpha;\ \subset ((\subset \alpha) \Longrightarrow (\models \beta))}{\subset \beta}$$

Trivially seems to be the rule for a metaphysicalistic inference on including metaphysicalism $\beta \subset \beta$ *from metaphysicalisms* $\alpha \subset \alpha$ *and* $\beta \models \beta$, *where*

$$\frac{\begin{pmatrix} \alpha \subset \alpha; \\ \Big((\alpha \subset \alpha) \Longrightarrow (\beta \models \beta)\Big) \subset \\ \Big((\alpha \subset \alpha) \Longrightarrow (\beta \models \beta)\Big) \end{pmatrix}}{\beta \subset \beta}$$

At last we have a case of the rule for a phenomenalistic inference *on including phenomenalism* $(\beta \subset; \subset \beta)$ *from phenomenalisms* $(\alpha \subset; \subset \alpha)$ *and* $(\beta \models; \models \beta)$, *where*

$$\frac{\begin{pmatrix} (\alpha \subset; \subset \alpha); \\ \Big(\Big((\alpha \subset; \subset \alpha) \Longrightarrow (\beta \models; \models \beta)\Big) \subset; \\ \subset \Big((\alpha \subset; \subset \alpha) \Longrightarrow (\beta \models; \models \beta)\Big)\Big) \end{pmatrix}}{(\beta \subset; \subset \beta)}$$

The last rule means to infer phenomenalistically by modus tollens in the sense of includedness upon a phenomenalistic case of informing. □

The listed informational rules of modus ponens are only the most characteristic ones. We did not present any of the possible cross-modal rules, that is from externalistic-internalistic to phenomenalistic-metaphysicalistic ones (additionally, 12 possible cases).

The rules of modus ponens belong to the most obvious (normal, generally agreed) rules of inference primarily because of their categorical value. However, within the informational logical realm, the rule of modus ponens is certainly only one of possible rules of inference. Applying only this kind of rules would mean to infer in a particularly reductionistic and informationally unidirectional way.

6.5 Includable MODUS TOLLENS

The informational modus tollens pertaining to informational includedness is a good example of the difference arising from the positions of categorical reasoning on one side and the informationally phenomenological reasoning—for example, includably-as-in-the-informationally-involved-way—on the other side. Includedness as an informational in-volvement must not be comprehended categorically, since reasoning in this way would lead to the categorical nonsense, traditional-logic controversy, and 'common-sense' (say, occurrent, in German, vorhanden) absurdity. From another point of view, the informational Being-in represents the most general term concerning the 'In', which at a given situation or attitude speaks for a particular situation or attitude. In this sense, informational Being-in is always particularizes and if not, the empty place in its whole meaning only waits to be complemented.

Inference Rule 2 [Includable MODUS TOLLENS] *Some cases of informational modus tollens can again be expressed in terms of the pure informational externalism, internalism, metaphysicalism, and phenomenalism concerning an entity* α*'s includedness. The modus tollens rule for an* externalistic inference *on non-including externalism* $\alpha \not\subset$ *from externalisms* $((\alpha \models) \Longrightarrow (\beta \models)) \subset$ *and* $\beta \not\subset$ *is*

$$\frac{((\alpha \models) \Longrightarrow (\beta \models)) \subset;\ \beta \not\subset}{\alpha \not\subset}$$

A similar rule for an internalistic inference *on non-including internalism* $\not\subset \alpha$ *from internalisms* $\subset ((\models \alpha) \Longrightarrow (\models \beta))$ *and* $\not\subset \beta$ *is*

$$\frac{\subset ((\models \alpha) \Longrightarrow (\models \beta));\ \not\subset \beta}{\not\subset \alpha}$$

Trivially seems to be the rule for a metaphysicalistic inference on non-including metaphysicalism $(\alpha \models \alpha) \not\subset (\alpha \models \alpha)$ *from metaphysicalisms* $((\alpha \models \alpha) \Longrightarrow (\beta \models \beta)) \subset ((\alpha \models \alpha) \Longrightarrow (\beta \models \beta))$ *and* $(\beta \models \beta) \not\subset (\beta \models \beta)$, *thus,*

$$\frac{\begin{pmatrix} \begin{pmatrix} (\alpha \models \alpha) \Longrightarrow \\ (\beta \models \beta) \end{pmatrix} \subset \begin{pmatrix} (\alpha \models \alpha) \Longrightarrow \\ (\beta \models \beta) \end{pmatrix}; \\ (\beta \models \beta) \not\subset (\beta \models \beta) \end{pmatrix}}{(\alpha \models \alpha) \not\subset (\alpha \models \alpha)}$$

At last we have a case of the rule for a phenomenalistic inference *on non-including phenomenalism* $(\alpha \not\subset; \not\subset \alpha)$ *from phenomenalisms* $(((\alpha \models; \models \alpha) \Longrightarrow (\beta \models; \models \beta)) \subset; \subset ((\alpha \models; \models \alpha) \Longrightarrow (\beta \models ; \models \beta)))$ *and* $(\beta \not\subset; \not\subset \beta)$, *so,*

$$\frac{\begin{pmatrix}\begin{pmatrix}((\alpha \models; \models \alpha) \Longrightarrow (\beta \models; \models \beta)) \subset; \\ \subset ((\alpha \models; \models \alpha) \Longrightarrow (\beta \models; \models \beta))\end{pmatrix}; \\ (\beta \not\subset; \not\subset \beta)\end{pmatrix}}{(\alpha \not\subset; \not\subset \alpha)}$$

The last case means the inferring by modus tollens in the sense of includedness upon a phenomenalistic case of informing. □

The rule of informational modus tollens shows clearly that operators $\subset$ and $\not\subset$ must be comprehended differently from the adequate categorical relational symbols in logic. It is to emphasize that if operator $\subset$ is concretely particularized in the upper rules then operator $\not\subset$ must receive the same concrete particularization. It is to say in general that operators $\subset$ and $\not\subset$ are of the same kind (meaning) in the given context.

Further on, operators $\subset$ and $\not\subset$ express the activity of informational in-volvement and non-involvement (embedding and non-embedding), respectively. In this respect, an informational entity α may *particularly* be involved (informationally embedded) in itself ($\alpha \subset \alpha$) or not ($\alpha \not\subset \alpha$). From this non-categorical point of view, informational operator $\subset$, expressing the informational Being-in, behaves as a regular informational operator.

6.6 Includable MODUS RECTUS

The intentional of an informational entity is that which informs actively and participates in the informational arising and constitution of the entity. As soon as we say that informational acts are intentional, the question arises, how the extraction (bringing to the surface) of intentional information, hidden *in* the background of an informing entity, would be possible. Intention is nothing else than an informational phenomenon of informing, pertaining to the question "Why does an entity inform in just a particular way and does not inform in another one?" In this context, intention appears as a reason (motive, cause, hidden explanation) of an informing entity.

Informational modus rectus is a rule for the inference which concerns the intentional informing of an informational entity. Includable modus rectus reduces this inference to the includable informing of an entity, which means that informational involvement, embedding, and connectedness pertaining to the intention as a ruling (motivating) informational phenomenon is being searched.

Let us construct a case of includable modus rectus (which predicts the intention of an informing entity) as a conclusion of particular conclusions, that is, as a modal inference of modal inferences.

Inference Rule 3 [Includable MODUS RECTUS] *The basic scheme of the includable modus rectus concerning entity α and its intention ι_α could be the following basic phenomenalistic form:*

$$\frac{\alpha; (\alpha \Longrightarrow (\alpha \models_{\iota_\alpha}; \models_{\iota_\alpha} \alpha))}{\iota_\alpha \subset \alpha}$$

Taking into account the entity α's externalism, internalism, metaphysicalism, and phenomenalism, there is,

$$\frac{\begin{pmatrix}\dfrac{\alpha \models; ((\alpha \models) \Longrightarrow (\alpha \models_{\iota_\alpha}))}{(\alpha \models_{\iota_\alpha}) \subset (\alpha \models)}; \\ \dfrac{\models \alpha; ((\models \alpha) \Longrightarrow (\models_{\iota_\alpha} \alpha))}{(\models_{\iota_\alpha} \alpha) \subset (\models \alpha)}; \\ \dfrac{\alpha \models \alpha; ((\alpha \models \alpha) \Longrightarrow (\alpha \models_{\iota_\alpha} \alpha))}{(\alpha \models_{\iota_\alpha} \alpha) \subset (\alpha \models \alpha)}; \\ \dfrac{(\alpha \models; \models \alpha); ((\alpha \models; \models \alpha) \Longrightarrow (\alpha \models_{\iota_\alpha}; \models_{\iota_\alpha} \alpha))}{(\alpha \models_{\iota_\alpha}; \models_{\iota_\alpha} \alpha) \subset (\alpha \models; \models \alpha)}\end{pmatrix}}{\iota_\alpha \subset \alpha}$$

This inference rule of includable modus rectus expresses the common phenomenon of intentional informing (intention ι_α with intentional informing $\Im_{\iota_\alpha}$) of entity α. Thus, intention ι_α as a distinguished entity informs within α as

$$\begin{pmatrix}(\iota_\alpha \models \Im_{\iota_\alpha}) \models \iota_\alpha; \\ \iota_\alpha \models (\Im_{\iota_\alpha} \models \iota_\alpha)\end{pmatrix} \subset \alpha$$

in an intentional manner. □

6.7 Includable MODUS OBLIQUUS

The Latin *obliquus* concerns that which is slanting, sideways, oblique, indirect, covert, envious [8], or also out-of-the-way. For instance, the abduction [4] as a mode of logical inference could be characterized as oblique in comparison to the deduction. But, the oblique mode of inference could be that which becomes interwoven in the realm of

the absurd, unbelievable, unforeseeable, contradictory, obscure, etc. In this respect, oblique conclusions may appear as the most surprising (e.g. counterinformational) cases of inference.

Inference Rule 4 [Includable MODUS OBLIQUUS] *Let o mark an oblique informational operand (entity) with oblique informing $\mathcal{O}_o$ and let α be a regular informational entity which informs includably, $\alpha \subset$, is informed includably, $\subset \alpha$, is metaphysically includable, $\alpha \subset \alpha$, and phenomenalistically includable, $\alpha \subset; \subset \alpha$, in the domain of belief β. Then we can obtain one of several possible formulas for the includable modus obliquus, for example:*

$$\frac{\begin{pmatrix} \dfrac{(\alpha \models) \models_\beta;\ ((\alpha \not\models) \Longrightarrow (o \models)) \subset}{(o \models_{\mathcal{O}_o}) \subset_\beta (\alpha \models_{\mathcal{I}_\alpha})}; \\ \dfrac{\models_\beta (\models \alpha);\ \subset ((\not\models \alpha) \Longrightarrow (\models o))}{(\models_{\mathcal{O}_o} o) \subset_\beta (\models_{\mathcal{I}_\alpha} \alpha)}; \\ \dfrac{\begin{pmatrix} \alpha \models_\beta \alpha;\ (((\alpha \not\models \alpha) \Longrightarrow (o \models o)) \subset \\ (((\alpha \not\models \alpha) \Longrightarrow (o \models o))) \end{pmatrix}}{(o \models_{\mathcal{O}_o} o) \subset_\beta (\alpha \models_{\mathcal{I}_\alpha} \alpha)}; \\ \dfrac{\begin{pmatrix} (\alpha \models_\beta; \models_\beta \alpha);\ (((\alpha \not\models; \not\models \alpha) \Longrightarrow o) \subset; \\ \subset ((\alpha \not\models; \not\models \alpha) \Longrightarrow o)) \end{pmatrix}}{(o \models_{\mathcal{O}_o}; \models_{\mathcal{O}_o} o) \subset_\beta (\alpha \models_{\mathcal{I}_\alpha}; \models_{\mathcal{I}_\alpha} \alpha} \end{pmatrix}}{\begin{pmatrix} (o \models) \subset (\alpha \models);\ (\models o) \subset (\models \alpha); \\ (o \models o) \subset (\alpha \models \alpha); \\ (o \models; \models o) \subset (\alpha \models; \models \alpha) \end{pmatrix}}$$

where $\models_\beta$ and $\subset_\beta$ are believable operators. □

Certainly, we must not forget that the last inferential scheme of modus obliquus is obtained on an intuitive basis and that many other senseful, obliquely structured inference schemes may exist.

7 Includedness as a Consequence of Informational Internalism

Informational internalism pertaining to an entity α was expressed formally by $\models \alpha$. Operator $\models$ was said to be the most general informational operator (an operational joker), which can be particularized according to the occurring informational circumstances. Informational includedness in the internalistic sense is a particularization, marked by $\subset \alpha$. The empty left side of operator $\subset$ points to the openness, to the question: *What can informationally be included in α?* or *What is the includable internalism of entity α?*

Includedness is a consequence of the particularization of informational internalism. This particularization was determined by Definition 1 as a recursive (recursively infinite) scheme with further 15 possibilities (if we exclude the empty case, as the 16^{th} possibility). Thus, the particularization from general internalism of the form $\models \alpha$ to the includable internalism of the form $\subset \alpha$ brought a complex recursive scheme, in which, the initial form $\models \alpha$ there appears as a particular case. This does not mean an informational contradiction but circularity.

Let us show how even the extreme extensional case $\Xi^{\beta,\alpha}_{\beta,\alpha}(\alpha \subset \beta)$ obtains its full significance in the so-called everyday speech [6]. Let the role of α be assigned to words and that of β to their contexts in speech. In [6], the following dictum seems to be highly senseful (pp. 80–81):

- *Words and ideas are inseparable. ... Words and ideas hold together. ... Every word gets its meaning from some kind of context and we recognize it in that, or similar contexts. The context suggests the word, the word suggests the context. The context may be* physical. *... The context may be* psychological. *... The context may be* verbal. *Every word that you understand when you read or listen has meaning in that, and similar verbal contexts. The word belongs in the context. The word lives in the context. The two are inseparable.*

We can understand that in the cited case not only $\alpha \models \beta; \beta \models \alpha$ holds, but also $\alpha \subset \beta; \beta \subset \alpha$ is informationally senseful. The last citation helps us to understand that in the case of $\alpha \subset \beta; \beta \subset \alpha$ there is no a problem of contradiction in comprehension of the informational Being-in. As it was said, the word belongs to the context and vice versa, the context is includably impacted by the word. This statement holds especially for the process of informational arising of context (e.g., speech), where words intentionally influence the arising of context and context influences the choice of the words constituting the arising context.

Informational includedness of something is a consequence of the perceiving abilities of the something observing entity. An informational depiction of something in the observing entity concerns the problem of informing between entities [11], where the depiction of something is called the internal representation (or real presence [7]) of something. Such representation is always informationally included in the observing entity, while the vice versa case does not hold at all (there is, for example, no informational influence of the observing entity on the observed entity).

8 Being-in and Being-in-the-world

Being-in-the-world is a philosophical term, being coined by Heidegger [2]. "It is a general basic state of an entity and of informational entity in particular (in this case, $\beta_{\text{Being-in-the-world}} \subset \alpha$, where α marks an entity in question). It is sketched in terms of an orientation towards Being-in as such. Being-in-the-world stands for a unitary phenomenon and cannot be broken up into contents which may be pieced together. But, it has several constitutive items in its structure." As we will understand, Being-in-the-world informationally dwells in Being-in which always pertains to an informational entity (as a property, involvement, characteristics).

Being-in-the-world is informationally particularized Being-in, where the world is still comprehended in a universally open way. The world is also a specific category of thinking which must not be equalized with the physical (space-temporal) world in which phenomena appear and disappear. An informational entity informs in the world if the surrounding world (environment, its exterior) impacts the structure of the entity's informing and the entity perceives also the responses to its own informing to the world.

Being-in-the-world is a condition sine qua non for the arising of the so-called intelligent informing. Intelligent can mean to inform inventively, ingeniously, creatively also in the sense of the chaotic, unforeseeable, with the intention to adapt, reach a goal, survive, solve a problem, etc. Informational Being-in-the-world is more concretized informing of something than the informational Being-in, which is a general framework for further informational particularization or certain universalization. If we introduce markers $\beta_{\text{Being-in}}$ and $\beta_{\text{Being-in-the-world}}$ then one can express this relation by the operator of informational includedness $\subset$. At least, there must hold $\beta_{\text{Being-in-the-world}} \subset \beta_{\text{Being-in}}$. Because of an informational interaction, there can also exist $\beta_{\text{Being-in}} \subset \beta_{\text{Being-in-the-world}}$. Entity α informs as being in the world, that is, $\alpha \subset \beta_{\text{Being-in-the-world}}$ and arises informationally within this circumstances.

9 Conclusion

Informational Being-in comes not only very close to the philosophical Being-in [2, 1], but can surmount it by theoretical-formal expressions (arising formulas) of informational language, showing the decompositional power of the initially set includable problems. Through the discussion of informational includedness in this paper we have learned its complexity in parallel, serial, and parallel-serial structures. We chose (Definition 1) an informationally logical and flexible case of the includedness definition. Certainly, other cases of even a more complex definition of informational includedness are possible.

The most pretentious case of an includable structure seems the metaphysical case, where further and more detailed decomposition (interpretation) together with introduction of parallel formulas is possible and senseful. The case of a complex parallel structure of serial-parallel and parallel-serial formulas can be viewed as the most appropriate candidate in conceptualizing an informational system performing as an intelligent entity.

Metaphysical includedness was composed of several reasonable chosen informational entities, that is components, which have been staying for the so-called informing, counterinforming, and informational embedding of the entity in question. The informing component seems to be a necessity for the explication of the essential and detailed characteristics of the informing entity, its own intention (informational perseverance as a consequence of the existing informational structure) in spontaneity and circularity.

The counterinforming component (as a counterpart of informing, e.g. a form of its informa-

tional 'subconsciousness') was involved in production of new, also contradictory, and yet not informationally embedded (connected) information, which is nothing else than a kind of 'originally' arisen phenomenon. This component (counterinforming with counterinformational entity) seems to be the most problematic concept in concern to its informational implementation, for example, by a computer program and, lastly, by an informational machine as a substantially differently structured and conceptualized nowadays computer system. In general, we have presupposed that informational operators perform in an informational, counterinformational, and informationally embedding manner.

At last, the arisen and to the informational entity arrived (from its exterior perceived) information has to be informationally connected to that what already exists, that is, to the informing (body) of the entity itself. This phase of metaphysical phenomenalism we have called informational embedding.

An entity's metaphysical shell as described, is the most rationally imaginable (minimalist) structure of informing. Its particularization means the filling and extending (interpreting) of the three main components. The metaphysical shell conceptualizes unlimited possibilities in the informational arising of the entity α. It is the way to its concrete and artificial implementation, for instance, within an informational machine $\mathcal{M}$. Thus, $\alpha \subset \mathcal{M}$, where $\mathcal{M}$ systemically supports the informing of α. Counterinforming $\mathcal{C}_\alpha$ with counterinformational entity γ_α supports the arising of originality o within an entity's counterinforming, that is, $((o \subset \mathcal{C}_\alpha) \subset \mathcal{I}_\alpha) \subset \alpha$, etc. Originality as counterinforming can inform in different ways: every time new value to something is given, it can be seen as an original informing; familiar entities can be looked in different light; bringing together known entities and link them in new ways is an original approach. Originality can grow out of that which is already known (e.g., producing knowledge by knowledge from knowledge [12]).

Informational Being-in as developed in this paper is the beginning step in making informational theory axiomatic and constructive and, through this, tracing the way to an informational system implementation. The problem of knowledge as informational entity will show how the formalistic (axiomatic and inferring) power of informational theory can lead to new concepts, techniques, methods and theoretical approaches, which can absorb the today scientific methodologies and connect them into an informationally arising system. The biggest challenge on this way is the so-called informational machine, which will perform as a real informational accelerator, offering the widest possible framework for informational experiments and applications. In this sense, the future informational machine must be capable to mimic the most complex parallel-serial systems of informational arising, supporting systemically the informing, counterinforming, and embedding of any informational entity. Within this perspective, informational Being-in with its externalism, internalism, metaphysicalism, and phenomenalism seems to be the keystone of the arising informational theory and methodology.

Let us close our theoretical and formalistic discourse on informational Being-in with a dictum of George Steiner ([7], pp. 174–175), which approves the potentiality of informational approach (the theory in this essay) in its wholeness:

- *Though acts of reception and of understanding are in some measure fictions of ordered intuition, myths of reason, this truth does not justify the denial of intentional context. It is an absurd to discard as mendacious, as anarchically opaque, the bearing of contextual probability and suggestion, as it is to invest in such probability any blind trust. ... We advance step by step towards a delineation of the given space; our perceptions are more and more justly incident to the circumference of possible intent and meaning. The congruence is never complete. It is never uniform with its object. If it was, the act of reception would be wholly equivalent to that of original enunciation.*

References

[1] H.L. Dreyfus, *Being-in-the-World,* The MIT Press, Cambridge, MA, 1991.

[2] M. Heidegger, *Being and Time,* Harper & Row, New York, 1962.

[3] F. Heylighen, *On Internal Representation,* Informatica 17 (1993) 294.

[4] A.C. Kakas, R.A. Kowalski, and F. Toni, *Abductive Logic Programming,* Journal of Logic and Computation 2 (1992) 719–770.

[5] J. Šlechta, *On a Quantum-Statistical Theory of Pair Interaction between Memory Traces in the Brain,* Informatica 17 (1993) 109–115.

[6] B. Sondel, *Everyday Speech,* Barnes & Noble, New York, 1965.

[7] G. Steiner, *Real Presences,* The University of Chicago Press, Chicago, 1989.

[8] A.P. Železnikar, *Informational Logic IV,* Informatica 13 (1989) No. 2, 6–23.

[9] A.P. Železnikar, *Metaphysicalism of Informing,* Informatica 17 (1993) 65–80.

[10] A.P. Železnikar, *Logos of the Informational,* Informatica 17 (1993) 245–266.

[11] A.P. Železnikar, *On Informing between Entities,* Informatica 17 (1993) 294–296.

[12] A.P. Železnikar, *Towards an Informational Understanding of Knowledge.* Cybernetics and Systems '94, Vol. II (Ed. R. Trappl), pp. 1587–1594, World Scientific, Singapore, 1994.

[13] *The Oxford English Dictionary,* Second Edition (on compact disc), Oxford University Press, Oxford, 1992.

GRAPHS AND THE THIRD NORMAL FORM FOR RELATIONAL DATABASE

Jože Nemec and Janez Grad*

University of Maribor, College of Agriculture, 62000 Maribor, Vrbanska 30, Slovenia
Phone: +386 62 226 611, Fax: +386 62 23 363
E-mail: joze.nemec@uni-mb.si

*University of Ljubljana, Faculty of Economics, 61109 Ljubljana, Slovenia
Phone: +386 61 1683 333, Fax +386 61 301 110

Keywords: relational database, relations, normal forms, normalization process, DB graphs, matrices

Edited by: Rudi Murn
Received: July 12, 1993 **Revised:** May 30, 1994 **Accepted:** June 9, 1994

Determination of higher order normal forms for a relational database (RDB) is frequently a time-consuming process. We can solve this problem by applying graph theory. In the paper the necessary characteristics of graphs that represent a given RDB are analyzed. The connectivity matrices for these graphs and the properties they must satisfy are also introduced and discussed. The established RDB graphs and the corresponding relationship matrices form an important basis of the algorithm for designing RDB with no redundant relations.

1 Introduction

The RDB application to data handling demands a careful design of RDB structure. The designing process consists of several stages. First the conceptual model of a DB has to be defined in which logical data structures of the information system (IS) are determined. An adequate conceptual model is an important prerequisite for a successful DB design. The RDB consists of a number of attributes that change during the time. Within the process of the RDB design the relations must be normalized. The characteristics of different normal forms are described in (Stout, Woodworth, 1983), (Elmasri, Navathe 1989) etc.

The normalization process becomes quite demanding in case of a large number of interrelated attributes. This asks for an experienced model designer. It is desirable to predetermine the process as much as possible in order to exploit the computer for performing the most tedious part of the task. Such procedures have already been developed in the past. In (Vetter, Maddison, 1981) a DB design procedure was described where the resulting model was obtained by ascertaining the redundant relationships. Another algorithm was presented in (Salzberg, 1986), and its modifications were published in (Atkins, 1988) and (Diederich, 1991),respectively. In the following paper we develop an RDB by means of graphs and state the basic characteristics of the graphs corresponding to the normalized RDB.

2 Graphs and relations

Let us present the relations in the form of graphs. Each attribute becomes a node within the graph. We denote the nodes as $x_1, x_2, \ldots, x_n$. A possible relation between two attributes is presented by means of a directed arc. For example, if x_1 stands for an employee's social security number (SSN) and x_2 stands for the employee's salary, we may express this relationship as illustrated in Figure 1.

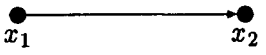


Figure 1: Directed relationship between nodes (attributes) x_1 and x_2

The directed arc begins in x_1 and ends in x_2. This is understandable because salary is defined by employee's social security number (SSN). The starting node represents the key attribute, while the ending node represents the non-key attribute of the relation R(x_1,x_2).

Let us briefly discuss the type of nodes that appear in graphs. We ascertain two groups of nodes that represent

(i) the simple attributes, and
(ii) the composite attributes, respectively.

A simple attribute represents some property of the entity, for example the colour of an article, birth date, etc. Two or more simple attributes may compose a composite attribute which serves as a key of some relation. By introducing composite attributes, we assure that to each value of the attribute, in which the arc starts, there belongs only one value of the attribute in which the arc ends. The composite attribute x_s defined by the simple attributes x_1, x_2, ..., x_k is shown in Figure 2. The arcs start in nodes that compose a composite key, and end in the composite key, which in turn is the key of the relation R(x_s,x_n). An arc starting in a simple attribute and ending in a composite attribute is a trivial arc. We shall mark trivial arcs by a broken line.

Suppose that we have defined the necessary data of a company in consideration. Since this data collection comprises the needs of different users (departments, services) it consists of a number of attributes and relationships between them. On the basis of the collected data we can draw a graph which consists of a set of nodes (attributes) and directed arcs (relationships) between them.

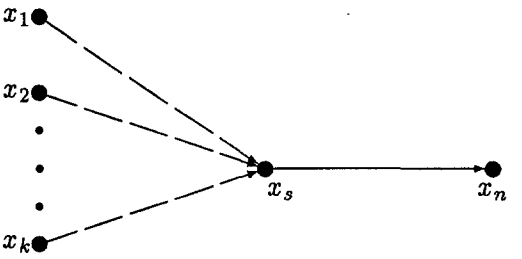


Figure 2: Composite attribute x_s defined by simple attributes x_1, x_2, ..., x_k

For each node x_i we can define the degree d(x_i). This is the number of arcs which either start or end in the node. We denote by $d^+(x_i)$ the outdegree of a node, i.e. the number of arcs which start in x_i, and by $d^-(x_i)$ the indegree of a node, i.e. the number of arcs that end in x_i. A simple attribute has $d^-(x_i)$ equal to one or zero, while a composite attribute has $d^-(x_i)$ greater than one.

If there exists a sequence of arcs between two nodes and the terminal node of the previous arc coincides with the starting node of the following arc, then the sequence forms a path between the two nodes. The path is called a closed path when the initial node and the terminal node of the path coincide, otherwise, the path is an open path. The path is called a cycle when all its arcs are different and any two nodes of it are different except for the initial one and the terminal one that coincide. The length of a path is the number of arcs that compose the path.

3 Basic characteristics of DB graphs

The process of a DB design results in a number of attributes and relations between them that determine the corresponding DB graph. In general, this graph represents a DB being in the first normal form (1NF). We want to transform 1NF into higher order normal forms, the graphs of which have some special properties. In the following paragraphs these properties will be discussed more in detail.

Theorem 1: *Suppose that the set of simple attributes X=(x_1, x_2, ...x_i, x_{i+1}, ...,x_n) defines the composite node x_s so that x_s and the set of simple attributes Y=(y_1, y_2, ...,y_{m-1}, y_m) form the relation R(X,Y), where X is the key of R. Suppose also that X'=(x_1, x_2, ...x_i) is a subset of X, and Y' is a subset of Y. Then no such Y' exists that R(X',Y'), where X' is the key of R.*

Suppose we have the graph as shown in Figure 3. x_1,x_2,...,x_n are simple attributes which compose the key x_s, and y_1,y_2,...,y_m are simple attributes dependent on x_s.

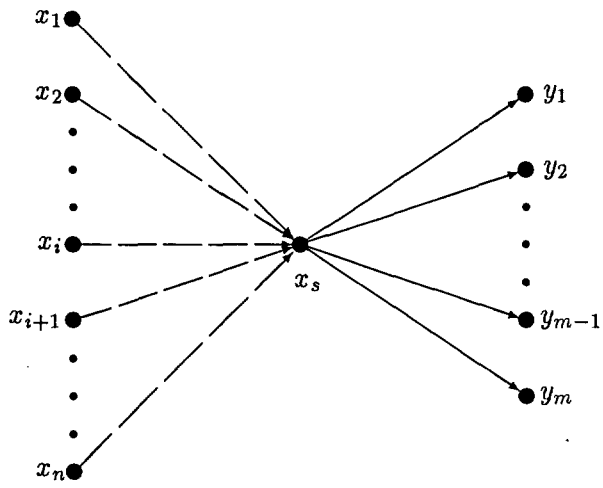


Figure 3: R(x_s,Y) graph of the composite attribute x_s

Let there be the attribute y_m which functionally depends upon the key composed of x_{i+1},...,x_n. The case is shown in Figure 4. x_s is composed of two subsets of attributes composing x_{s1} and x_{s2}. As y_m functionally depends upon the attributes of x_{s2}, we can replace the arc between x_s and y_m by the arc between x_{s2} and y_m. In this case y_m does not functionally depend upon attributes that compose x_s in Figure 3 and the corresponding relation is not in the second normal form (2NF).

While designing a DB we must analyze each attribute dependent on a composite key to find out whether it depends upon some subset of the key's attributes. If so we can decompose the key into a subset of keys where some attributes functionally depend upon one key only. In this case the DB is not in 2NF.

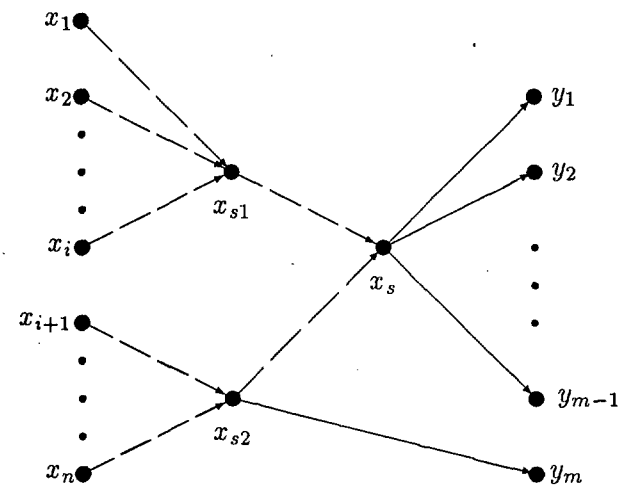


Figure 4: The key composed of two subsets of attributes

For the composite nodes some additional requirements must be met. For cycles the following theorem holds:

Theorem 2: *A cycle within a graph includes no node which is the composite key of some relation.*

Suppose such a cycle exists. Further, it is assumed that the initial (and therefore also the terminal) node is one of the attributes which form the composite key of some relation. Now we assume that we have the following sequence of arcs

$$x_1 \rightarrow x_2 \rightarrow x_3 \rightarrow x_4 \rightarrow \ldots. \rightarrow x_n \rightarrow x_1$$

Let x_2 be the composite key of R(x_2,x_3), and x_1 one of the attributes that form the composite key. If we substitute all the arcs between nodes x_3 and x_1 with the transitive arc, we obtain the sequence

$$x_1 \longrightarrow x_2 \longrightarrow x_3 \longrightarrow x_1$$

x_2 is the composite key of a relation according to the supposition. If we now replace the arcs

$$x_1 \longrightarrow x_2 \text{ and } x_2 \longrightarrow x_3$$

by the transitive arc

$$x_1 \longrightarrow x_3$$

we see that x_3 depends upon x_1 only. The obtained case is the same as the case shown in Figures 3 and 4. The relation R(x_2,x_3) is not in 2NF what completes the proof of the theorem.

Now we can prove that for any DB graph the following theorem holds:

Theorem 3: *When a graph has the arc starting in the node x_1 and ending in the node x_2 then it has no other path starting in x_1 and ending in x_2.*

Suppose that Theorem 3 is not true and that there are an arc and some other path both starting in x_i and ending in x_j. Then this arc is a transitive arc for the path, and x_j is transitively dependent on x_i. The corresponding DB is not in the third normal form (3NF) and the following corollary can be stated:

Corollary 3.1: *There is only one path between two nodes.*

In the opposite case, i.e. when there are two paths, we can replace one of them by an adequate transitive arc.

Consider now also the notion of the graph connection.

The directed graph is said to be

(i) a strongly connected graph, when for any two different nodes x_i and x_j there exist the directed paths from x_i to x_j and x_j to x_i.

(ii) a one-way connected graph, when for any two different nodes there exists the directed path starting in one and ending in the second node. The directed path in the opposite direction may also exist.

(iii) a weakly connected graph when any two different nodes are connected by a set of optionally directed arcs.

(iv) a disconnected graph when it is not a weakly connected graph.

Let us consider how the graph connection reflects the DB graphs. For the graph of a DB in 3NF the following theorem holds:

Theorem 4: *The graph of a DB in 3NF is a weakly connected graph.*

The statement is based on the fact that all the nodes within the strongly connected graph are connected in both directions. Therefore, they are, reciprocally dependent, which is not allowed within 3NF. Further, in the case of a disconnected graph we can divide the nodes into at least two subsets where the nodes of one subset are disconnected from the nodes of the other. Therefore, there is no relationship between any two nodes of different subsets, which means that we deal with two different databases.

Suppose that DB is a one-way connected graph. Let x_0 be the starting node, and x_1 and x_2 the terminal nodes of two different arcs. Accordingly, as the graph is a one-way connected graph, there exists at least one path between x_1 and x_2 starting in one node, say x_1, and ending in the other one. Then there exists the transitive dependency between x_0 and x_2, which must not be the case in 3NF. A DB graph may, therefore, not be a one-way connected graph.

According to the statement above, a DB in 3NF can only be a weakly connected graph and, therefore, $d(x_i) \neq 0$ for any node x_i. This means that each node in a graph is either the starting node or the terminal node of at least one arc.

Let us now focus our attention to the properties of the DB subgraphs. Suppose we have a DB subgraph with all its nodes being strongly connected. We call such a subgraph a strong graph component of the given graph. Each graph may have several strong graph components. Since such a subgraph may represent a subdatabase, which is not in 2NF, the following theorem holds:

Theorem 5: *A DB graph may not involve strong graph components.*

When dealing with a large organization, we trace the relationships in the corresponding DB by setting up an initial model, which as a rule involves some redundant relationships. The database administrator task is to minimize the number of relationships. The problem can be solved by determining the minimal cover of the DB. The searching process is not uniform, as there may exist more minimal covers for the same DB. In order to obtain one, we must determine the or-

der in which to remove the redundant arcs. We can set up different removal criterions, like the longest transitive arcs, the arcs with the minimum access value, etc. By setting up an appropriate criterion, the process can become automatic.

4 Relations and matrices

As already stated, a RDB can be presented and described as a graph with nodes representing attributes and directed arcs representing relations between the attributes. We want to present and describe such a graph by means of adequate matrices. We show that the matrices which describe a DB also have some particular properties.

Let us first define the node relationship matrix and state its properties.

Definition 1: *For each DB graph we can build a square node relationship matrix* $P = \| p_{ij} \|$ *so that* p_{ij}, *where* $p_{ij} \geq 0$, *denotes the number of arcs starting in i-th and terminating in j-th node.*

Now we compute matrices P^2, P^3,... We prove the following theorems:

Theorem 6: *The element* r_{ij} *of R, where* $R = P^n$, *equals the number of the directed paths from the i-th node to the j-th node with the length of n.*

We prove the theorem by means of the total induction. It holds for n = 1 by the definition of P. Suppose that it holds also for P^n. Now we compute $P^{n+1} = P^n$ - P the (i,j) element of which, say c_{ij}, is obtained from

$$c_{ij} = \sum_{k=1}^{m} r_{ik} p_{kj}$$

According to the supposition, r_{ik} equals the number of the directed paths from the i-th node to the k-th node with the length of n. The element p_{kj} = p $\neq$ 0 if there exist p arcs starting in the k-th and ending in the j-th node. Therefore the product r_{ik}-p_{kj} equals the number of the paths with the length of n+1, the last arc starting in the k-th node. By computing and adding together $r_{ik}p_{kj}$ for all k = 1,..., m, where m is the number of the nodes, we obtain c_{ij}. The value of c_{ij} equals the number of directed paths heading from the i-th node to the j-th node with the length of n+1. This is just what we had to prove.

The following corollary also can be proved:

Corollary 6.1: *The elements* r_{ij} *of R, where* $R = P^n$ *and P is the relationship matrix, can only take the values 0 or 1.*

Suppose this is not true and there exist two directed paths between nodes i and j. We can replace one of the two by a transitive arc between nodes i and j. This arc, however, represents also a transitive arc for the second path, and therefore this is not the graph of a DB in 3NF. This contradicts the demand for a DB being in 3NF.

Theorem 7: *If an element* r_{ij} *of* P^n *is equal to 1, then all the corresponding elements* a_{ij} *of matrices P,* P^2, P^3, ... P^{n-1} *are equal to zero.*

Suppose $r_{ij} \neq 0$ and one of the corresponding $a_{ij} \neq 0$. Then two directed paths exist between nodes i and j, and the DB is not in 3NF what we proved earlier.

Let us now form a sequence of matrices D_1, D_2, ..., D_i, where the elements d_{ij}^k of D_k are defined as follows:

$d_{ij}^k = 0$, if $a_{ij} = 0$ for P, P^2, ..., P^k, and

$d_{ij}^k = 1$, if at least one corresponding element $a_{ij} \neq 0$.

Matrix D_k tells us that $d_{ij}^k = 1$, if there exists at least one path starting in the i-th node and ending in the j-th node with the length not exceeding k.

Theorem 8: *There always exists such integer m, that* $D = D_m = D_{m+k}$, *for k = 1, 2, 3,...*

Proof. The sequence of matrices D_k determines the node relationships within the paths composed of no more than k arcs. Suppose that the longest path in the graph which includes no cycles comprises m arcs. The value of m is less than or equal to the number of arcs in the graph. An optional element d_{ij}^m of D_m is not equal to zero if a path exists starting in the i-th node and ending in the j-th node, otherwise $d_{ij}^m = 0$.

Let there exist some additional matrices D_{m+k} which satisfy the conditions stated above. Let element d_{ij}^s of D_s be equal to zero for some s, where s ¿ m, and the corresponding element of

D_{s+1} be equal to one. Then a path of length s+1 exists between nodes i and j which includes no cycle. This contradicts the statement that the longest path in the graph which includes no cycles comprises m arcs. □□

We prove now the following corollary related to Theorem 8.

Corollary 8.1: *Matrix D defines the transitive closure of a DB.*

Transitive closure incorporates all elementary relations and also the relations obtained by all possible transitive arcs. We can introduce a transitive arc between two nodes only when there exists a path between the initial node and the terminal node. The path exists only then when the corresponding element d_{ij} of matrix D equals one.

Theorem 9: *If D represents a normalized DB then $d_{ij} = 0$ if $d_{ji} = 1$.*

Proof. Suppose the statement is not true and that we can write D in a form

$$\mathrm{D} = D_s + D_u$$

where D_s is a symmetric matrix and D_u is an asymmetric matrix such that

$$d_{ijs} = 1 \text{ and } d_{iju} = 0 \text{ if } d_{ij} = d_{ji} = 1$$

and

$$d_{ijs} = 0 \text{ and } d_{iju} = d_{ij} \text{ if } d_{ij} \neq d_{ji}.$$

This means that (i) $d_{ijs} = 1$ only when there is a path from node i to node j and a path from node j to node i, and (ii) $d_{iju} = 1$ only when there is a path from node i to node j and no path in the opposite way.

Consider first the matrix D_s. Suppose there exist at least two elements, say d_{ijs} and d_{jis}, not equal to zero. There may in addition exist some more elements in the i-th row and the j-th column which are not equal to zero. Observe the relations between the set of nodes defined by the non-zero elements in the i-th row. There exist paths from the i-th node to each of them and also in the opposite directions. We can, therefore, reciprocally connect any two nodes of this set. A DB composed of these nodes is a strongly connected graph. This is contrary to our statement that RDB can only be a weakly connected graph. Therefore, the initial statement must hold, stating that $d_{ij} = 0$ if $d_{ji} = 1$.

The theorem leads to the following properties of the elements of P.

Corollary 9.1: *If $p_{ij} = 1$, then $p_{ji} = 0$, for all $i \neq j$. All diagonal elements $p_{ii} = 0$.*

The statement holds because otherwise we would have the case where $d_{ij} = d_{ji} = 1$. Since all diagonal elements $p_{ii} = 0$, the graph must not include loops.

5 Algorithm for finding DB in the third normal form

Suppose that we have found all possible relations among the attributes. These relations are not in 3NF, so we carry out the normalization process. The process consists of several steps as shown in Table 1.

(1)	Determining the relationship matrix P
(2)	Determining the strong components of the graph
(3)	Removing the arcs from the cycles
(4)	Finding and removing the transitive arcs
(5)	Determining the relations in the data base

Table 1: The necessary steps for obtaining data base in 3NF

Let us explain the steps more in detail.

(1) The relationship matrix P is established which includes all possible arcs (nontrivial as well as trivial). Put D^1 = P.

(2) Matrices P^k, for k=2,3,.. are calculated, and the elements d_{ij}^k of D^k are determined by comparing the corresponding elements p_{ij} of P^k and d_{ij}^{k-1} of D^{k-1} . If one of these elements is equal to or greater than one, then $d_{ij}^k = 1$, otherwise $d_{ij}^k = 0$. The process ends when $D^{k-1} = D^k$. Put D = D^k and D = $D_s + D_a$, where D_s is symmetric and D_a is asymmetric. The strong components of the graph are determined by the non-zero elements of D_s: all the non-zero elements (attributes) within a row or column define a strong component.

(3) Suppose there exists at least one strong component forming a subgraph with all the nodes (attributes) being connected via a path in both

directions. This subgraph has cycles and at least one cycle is broken by removing an arc out of it. The process can be made fully automatic by applying some weight to the arcs (for instance the probabilities of their use) and removing the arc related with the lowest value of the weight. Otherwise it is up to a DB administrator to determine which arc should be removed.

The subgraph obtained by removing an arc is analyzed in the same way as the whole graph. The process ends after removing the arcs from all cycles. Matrix D becomes asymmetric.

Here we must point out that no composite attribute is a part of a strong component. If some strong component has a node which represents a composite element, this composite element is not defined properly.

(4) All transitive arcs are found and removed. Suppose that the element a_{ij} of P^k is equal to n (n¿1). In this case n paths can exist between the i-th node and the j-th node. The ending arcs of these paths can be found by comparing the i-th row of P and the j-th column of D^{k-1}. If the m-th elements of the corresponding row and column are equal to one, then the arc connecting the m-th node and the j-th node is an ending arc of the transitive path.

Transitive arc may also be found when the element a_{ij} of P^k and the element d_{ij} of D^{k-1} are equal to one. The ending arc corresponding to a_{ij} was discussed earlier. The ending arc corresponding to d_{ij} can be obtained by comparing the elements of D^1, D^2,..,D^{k-2}. Assume that the first non-zero (i,j) element is in D^t. The (i,j) element in P^t is also equal to one and defines an ending arc of the transitive path.

A path involving a transitive arc does not contain a composite node. We can prove this fact by applying the matrix D^{k-1}. Each trivial arc ends in the composite attribute. Assume that there exist two paths between the i-th node and the j-th node and that one of the two contains a composite node x_s. Then the i-th node is one of the attributes which define the composite attribute x_s. The path between x_i and x_j is longer than the path between x_s and x_j. Therefore the element d_{sj} of D^{k-1} equals one. In this case the composite node is not determined properly.

In order to remove the transitive arcs, the same criterion can be applied as for removing the arcs from cycles. After removing them, a data base in 3NF can be determined. First, all trivial arcs are removed and the remaining arcs are used to construct matrix P. Some columns of P contain only the zero elements. Suppose that the i-th column is one of them. The i-th row contains at least one non-zero element a_{ij}. This element defines the first arc of the path with x_i being the starting node and x_j being the second node. Now put a_{ij}=0 and observe the j-th row of P. The possible non-zero element a_{jk} defines the second arc of the path. The third arc is found by putting a_{jk}=0 and observing the k-th row. The process repeats until a row with all zero elements is found. When all elements of P are equal to zero all the paths in a graph have been found.

Suppose that the nodes in these paths are as follows:

Path 1: $x_{1,1}, x_{1,2}, x_{1,3}, \ldots x_{1,m1}$
Path 2: $x_{2,1}, x_{2,2}, x_{2,3}, \ldots x_{2,m2}$
Path 3: $x_{3,1}, x_{3,2}, x_{3,3}, \ldots x_{3,m3}$
Path 4: $x_{4,1}, x_{4,2}, x_{4,3}, \ldots x_{4,m4}$

.
.
.

Path k: $x_{k,1}, x_{k,2}, x_{k,3}, \ldots x_{k,mk}$

Each two neighboring elements in a row define a relation. The first element in each row is a key of at least one relation.

(5) We can now determine the set, say M, of the connected nodes. The necessary statements and tasks are as follows:

(i) All the nodes in the first path are in M.

(ii) If any node in M is one of the nodes which define the composite node x_s, then x_s is in M.

(iii) Find out whether there is a path with at least one node in M.

(iv) Add to M all the nodes in this path which are not in M.

(v) Repeat steps (iii) and (iv) until all the connected nodes are found.

Suppose that after this process all the nodes are in M. Then all the nodes are part of a DB. If not,

then there are at least two unconnected components and the graph of the DB is disconnected. This but contradicts the theorem that DB is a weakly connected graph. In a case like this we must determine the relations between the nodes of different components.

The relations in a DB can be found by following and executing the following procedure:

(i) Consider all different first elements x_1 (key attributes) in the rows.

(ii) The second elements in the rows represent the key depending attributes.

(iii) Remove the first element of each row, and eliminate the rows with only one element left.

(iv) Repeat steps (i) to (iii) until all the rows are eliminated.

The programming process is relatively simple and so is the data preparation. Only matrix multiplications are needed in the algorithm. All the redundant arcs can be determined by comparing the corresponding elements in matrices. We must define the connected attributes and for each connection the number (probability) of its appearance. In the beginning the probabilities can only be estimated, and determined more accurately within later stages of the process. In this way an optimum organization can be obtained by iterative reorganizations of the DB.

The process of finding the 3NF of a DB based on the graph theory can be carried out entirely automatically. In a large organization, the attributes and relations form a large connectivity matrix P. The number of operations within the solution process is high, too. This number can be reduced if some columns and rows of P need no processing. If all elements in the i-th row of P are equal to zero and only one element in the i-th column is equal to one, then this node represents the depending attribute in one relation only. This node has no effect on the elimination process and therefore the corresponding row and column can be removed. Similarly, if all elements in the i-th column of P are equal to zero and only one element in the i-th row is equal to one, then this node represents a key attribute in one relation only, and the corresponding row and column can be removed. By eliminating all such rows and columns, we get a reduced matrix P'. Only P' is needed in the normalization process. The size of this matrix is usually much smaller than the size of P. So the number of operations is considerably reduced and the third normal form can be obtained quickly.

6 Conclusion

In the paper we point out some particular properties which graphs must satisfy in order to represent a normalized DB. Matrices which describe these graphs must also comply with some requirements. This all helps us in exerting control over possible redundancies in the DB. By applying matrices P, D and higher powers of P, we can ascertain the redundant relationships. Matrix properties help in describing a DB and can also be exploited in designing an algorithm for the DB construction process.

By introducing the criterions for establishing and deleting the redundant arcs within different stages of the solution process, we can make the process of building up a DB in 3NF fully automatic.

7 References

Atkins John, A Note on Minimal Covers, SIGMOD Record, Vol. 17, No. 4, December 1988, pp. 16-21.

Diederich Jim, Minimal Cover Revisited: Correct and Efficient Algorithms, SIGMOD Record, Vol. 20, No. 1, March 1991, pp. 12-13.

Elmasri Ramez, Navathe B. Shamkant, Fundamentals of Database Systems, The Benjamin Cummings Publishing Company, Inc., Redwood City, California.... Reading, Massachusetts. New York ..., Bonn... 1989.

Salzberg Betty, Third Normal Form Made Easy, SIGMOD Record, Vol. 15, No. 4, December 1986, pp. 2-18.

Stout F. Quentin, Woodworth A. Patricia, Relational Databases, The American Mathematical Monthly, Vol. 90, 1983, pp. 101-118.

Vetter M., Maddison R.N., Database Design Methodology, Prentice Hall, Englewood Cliffs, 1981.

ON BAYESIAN NEURAL NETWORKS

Igor Kononenko
University of Ljubljana, Faculty of electrical engineering & computer science,
Tržaška 25, SI-61001 Ljubljana, Slovenia
Phone: +386 61 1231121, Fax: +386 61 264990
e-mail: igor.kononenko@ninurta.fer.uni-lj.si

Keywords: Bayesian neural network, Hopfield's neural network, naive Bayesian classifier, continuous neural network, probability, entropy, machine learning, artificial intelligence, overview

Edited by: Matjaž Gams
Received: September 15, 1993 **Revised:** January 11, 1994 **Accepted:** April 12, 1994

In the paper the contribution of the work on Bayesian neural networks is discussed with respect to previous, current, and potential future research in machine learning. The discrete and the continuous Bayesian neural network model is compared with Hopfield's models. It is shown that the Bayesian neural network's equations are analogous to equations used to describe Hopfield's model. Two different models of the Bayesian neural network are compared with Hopfield's model, one based on Shannon's entropy (probability) and the other based on Good's plausibility (odds). A generalization of the naive-Bayesian classifier is described that enable the basic algorithm to detect the dependencies between neurons.

1 Introduction

Machine learning is a subfield of artificial intelligence (AI) research field (Michalski et al., 1983;1986). AI researchers are concerned with algorithms that would enable computers to behave intelligently. As intelligence is strongly related to learning, especially in recent years machine learning is becoming of central importance for AI research. Although opinions are not unique, machine learning can be defined as subsuming, besides symbolic learning approaches, also parts of pattern recognition and neural networks.

An artificial neural network is constructed from a set of artificial neurons connected with synapses. Artificial neurons are simple elements able to calculate the weighted sum of the contributions of other neurons. The output from a neuron is usually binary, with possible values 0 or 1. The features of artificial neural networks are: biological similarity, high scale parallelism, robustness with respect to missing and incorrect data and with respect to the damage of the network, locally available information, modest software requirements, and the ability of adaptation through learning.

The main problems when designing an artificial neural network are the selection of the appropriate topology, the selection of the learning rule, the convergence of learning, and the convergence of the execution. A major weakness is the inability of the explanation of the execution and the results.

The work on Bayesian neural networks (Kononenko, 1990a) tries to unify probabilistic, symbolic, and neural networks approaches to learning. Various results of the work on Bayesian neural networks were published in several journals and conference proceedings (Kononenko, 1989-1993; Kononenko & Bratko, 1991). Due to the "Jožef Stefan Golden Badge" reward for a distinguished Ph.D. dissertation in technical sciences in Slovenia I was asked by the editor of this paper to answer the following questions about my dissertation:

- How did the work come about?
- What was the key intellectual contribution of the dissertation?

- What was critically left out that is still an open issue?
- What could people build more easily as a result of this work?
- What part would I have done differently given what is known today?
- What current papers reflect the future of that work?

The next section introduces the Bayesian neural networks based on probability and some other contributions of the thesis (Kononenko, 1990a). In section 3 each of the subsections answers one of the above questions. The appendix describes the Bayesian neural network models based on probability ratio.

2 Bayesian neural networks

This section introduces work on Bayesian neural networks. The next subsection gives an overview of the thesis. Subsections 2.2 and 2.3 define discrete and continuous models of Bayesian neural networks based on probability and show the analogy with discrete and continuous Hopfield's models. In subsection 2.4 the information score of classifier's answers is defined. Subsection 2.5 defines the semi-naive Bayesian classifier.

2.1 Overview of the thesis

In (Kononenko, 1990a) it is shown that the "naive" Bayesian classifier can be implemented with an artificial neural network. A Bayesian classifier is called "naive" if it disregards the interdependences between attributes. Bayesian neural networks use for learning the basic Hebbian learning rule. This rule states that the weight of a synapse is increased if both connected neurons are active. The convergence of the execution of the multidirectional feedback Bayesian neural network is proved. The interpretation of the execution of one neuron is defined as the summation of information gains from other neurons.

A multidirectional feedback Bayesian neural network based on probability ratio is also defined and the convergence of the execution is proved. The execution of one neuron of such network is interpreted as the summation of the weights of evidence. The relation with systems for inductive learning of decision trees is given.

It is shown that the naive Bayesian formula can be transformed into the weighted sum which shows the analogy with the Hopfield's feedback neural network model. Analogously to the Hopfield's continuous model, both types of Bayesian neural networks are generalized to continuous neuron states together with the convergence proof.

For the comparison of the performance of different classifiers on different classification problems a method for estimating the information score of a classifier's answer was developed. This measure excludes the influence of prior probabilities of classes and allows the estimation of incomplete answers.

It was shown experimentally that the Bayesian neural network significantly outperforms Hopfield's model with respect to the classification accuracy and the information score while the complexities of the learning and the execution remain equal. In experiments with four medical diagnostic problems the Bayesian neural network slightly outperformed the naive Bayesian classifier as the iterations make the network less sensitive to noise and missing data. The naive Bayesian classifier and the network outperformed the diagnostic accuracy of physicians specialists.

To overcome the naivety of the Bayesian neural network which stems from the independence assumption, an algorithm was developed for learning "semi-naive" feedforward Bayesian neural networks. The idea is to optimize the tradeoff between the reliability of approximations of probabilities and the naivety with respect to the independence assumption. It was shown experimentally that the algorithm performs better than the naive Bayesian classifier.

2.2 Discrete Bayesian neural networks based on probability

Let objects in a given domain be described with a set of attributes, each having a fixed number of values. Let each value of an attribute be rep-

resented with one Boolean variable. Therefore, an object is described with a set of variables $V_i, i = 1..n$. Let the value of the V_j be unknown. If the independence of influences of other variables to the j-th variable is assumed the probability that $V_j = 1$ given the values of other variables can be computed with the following 'naive' Bayesian formula derived from the Bayesian rule (Good, 1950; Kononenko, 1989c) (for brevity conditions $V_i = 1$ will be written simply as V_i):

$$P(V_j|V_1,...,V_n) = P(V_j) \prod_{V_i=1,i\neq j} Q_{ji} \quad (1)$$

where

$$Q_{ji} = \frac{P(V_j|V_i)}{P(V_j)} = \frac{P(V_j\&V_i)}{P(V_j) \times P(V_i)} \quad (2)$$

The 'naive' Bayesian classifier proved to be very efficient in classification problems when compared to other classification methods and to human experts (Kononenko, 1993; 1993a). It can be naturally implemented by a neural network. In the *Bayesian neural network* (Kononenko, 1989a) every neuron in a network is connected with every other neuron via bi-directional connections, called synapses. The *learning phase* changes the weights associated with synapses according to the basic Hebbian learning rule (Hebb, 1949) so that each neuron can use (1) as a *combination function* to compute its *activation level.* Each neuron represents a single value of one attribute, i.e. an attribute with N_i values is represented with N_i neurons. Classes are represented as one additional attribute with one value for each class. Note that the network makes no difference between the attribute that represents classes and other attributes.

The condition $i \neq j$ in (1) is optional, depending on whether each neuron is connected to itself with a *feedback* connection or not. Here, for brevity, feedback connections are omitted. Their influence is studied in more details in (Kononenko, 1989a).

In (Kononenko, 1991b) the following interpretation of one neural network's iteration is proposed. The minus logarithm of (1) gives

$$-log_2P(V_j|V_1,...,V_n) = -log_2P(V_j) - \sum_{V_i=1,i\neq j} log_2Q_{ji} \quad (3)$$

"$-log_2P(Event)$" is interpreted as the *amount of information (in bits) necessary to find out that Event has happened* (or the entropy of the Event (Shannon & Weaver, 1949)). Therefore, (3) is interpreted as follows: the amount of information necessary to find out that j-th neuron is active, given the values of other neurons, is equal to that amount before knowing the values of other neurons minus the sum of information gains from active neurons for the same conclusion.

If $-log_2Q_{ji}$ is replaced with T_{ji}, $-log_2P(V_j)$ with I_j and the left-hand side of (3) with A_j ("A" stands for activation level), the classical weighted sum is obtained from (3) which is used as a combination function in Hopfield's (1982) model:

$$A_j = \sum_{V_i=1,i\neq j} T_{ji} + I_j = \sum_{i\neq j} V_iT_{ji} + I_j \quad (4)$$

In Hopfield's model T_{ji} are elements of the memory matrix obtained as the sum of outer products of training patterns (corrected so that states 0 are changed to -1) and I_j is a constant input to the j-th neuron. Note that the model with no feedback connections ($i \neq j$) corresponds to the memory matrix with zero-diagonal ($T_{ii} = 0$ for all i). Here V_i can be 0 or 1, like in the original Hopfield's model although the learning rule for Hopfield's model is designed for values 1 and -1. In Hopfield's model one memory element corresponds to the difference between a probability that the two connected neurons are in the same state minus the probability that they are in different states. One memory element of the Bayesian neural network corresponds to the probability that the two connected neurons are both active.

The proof of the convergence of the Hopfield's model is based on the idea of the energy of the model. The state of the model determines its energy which with iterations decreases until it reaches a minimum in a fixed point. The energy function used by Hopfield (1982) is:

$$E(V_1,...,V_n) = -\frac{1}{2}\sum_j\sum_{i\neq j} V_jV_iT_{ji} \quad (5)$$

Kosko (1988) uses the similar approach to prove the convergence of the execution phase of bi-directional associative memories. The proof of the convergence of the Bayesian neural network

(Kononenko, 1989a) assumes a similar function representing a measure of *similarity* between the current state of a network and the current activation levels of neurons. During the execution the similarity increases until a maximum is reached in a fixed point. The similarity function used is:

$$Sim(V_1,...,V_n) = \prod_{V_j=1} \frac{P(V_j|V_1,...,V_n)}{P(V_j)} \quad (6)$$

The logarithm of (6) shows the analogy with Hopfield's energy function. The following holds:

$$E(V_1,...,V_n) = -\frac{1}{2} log_2 Sim(V_1,...,V_n) \quad (7)$$

as

$$\begin{aligned} log_2 Sim(V_1,...,V_n) &= \sum_{V_j=1} log_2 \frac{P(V_j|V_1,...,V_n)}{P(V_j)} \\ &= \sum_j V_j log_2 \frac{P(V_j|V_1,...,V_n)}{P(V_j)} \\ &= \sum_j \left(V_j \times \sum_{i \neq j} V_i T_{ji} \right) \\ &= \sum_j \sum_{i \neq j} V_i V_j T_{ji} \end{aligned}$$

The major differences among the two functions are the same as for the differences among the combination functions. According to relation (7) it would be more appropriate to name (5) the *entropy* or the *information content* of the system. Namely the minus logarithm of (6) is interpreted according to the interpretation of (3) as the sum of information gains from all active neurons to the conclusion that other currently active neurons are in fact active. As every information gain appears in the sum twice, because of the symmetry, constant $\frac{1}{2}$ is added to (5). The fixed point is now interpreted as the state with (locally) minimal information content.

2.3 Continuous Bayesian neural network based on probability

In this subsection the discrete model of the Bayesian neural network is generalized to continuous states analogously to Hopfield's continuous model (1984). Instead of discrete states 0 and 1, the state of one neuron will now be represented by any value on the interval [0..1]. The state of a neuron will be proportional to the probability that the neuron is currently active.

Eq. (3) defines the information gain of i-th neuron to the conclusion that j-th neuron is active if the i-th neuron is active with probability 1. A more general definition of information gain is the product:

$$V_i \times log_2 Q_{ji} \quad (8)$$

where V_i represents the current state of i-th neuron (which stands for the probability that i-th neuron is active). (8) can be interpreted as the expected amount of information from i-th neuron to the conclusion that j-th neuron is active. The generalized eq. (3) contains the sum over all neurons (not only the active ones):

$$\begin{aligned} &-log_2 P(V_j|V_1,...,V_n) = \\ &-log_2 P(V_j) - \sum_{i \neq j} (V_i \times log_2 Q_{ji}) \end{aligned} \quad (9)$$

Applying the exponential function to (9) results in the combination function for the continuous model which is the generalization of (1):

$$P(V_j|V_1,...,V_n) = P(V_j) \prod_{i \neq j} {Q_{ji}}^{V_i} \quad (10)$$

Note that the correct generalization of the conditional probability $P(V_j|V_i = 1)$, given the uncertain evidence of $V_i = 1$, is $P(V_j|V_i = 1) \times V_i$ (Ihara, 1987) which leads to different definition than (10). However, (10) is a meaningful generalization if the influences of different neurons (although representing the same attribute) are regarded independently.

Hopfield (1984) defined the dynamics of his continuous model with the following equations:

$$C_j \frac{du_j}{dt} = \sum_{i \neq j} T_{ji} V_i - \frac{u_j}{R_j} + I_j = A_j - \frac{u_j}{R_j} \quad (11)$$

$$V_j = g_j(u_j) \quad (12)$$

where V_i is the output (state) of the i-th neuron, u_j is the input (activation level) of the j-th neuron and I_j, R_j and C_j are constants. "g_j" is the output function defining the input-output relation which is smooth and sigmoid with asymptotes 0 and 1. Eq. (11) states that the speed of

change of u_j is proportional to the difference between current u_j and the new one calculated with (4). Actually, Hopfield omits the condition $i \neq j$ as for all i in his model $T_{ii} = 0$.

Since (9) is analogous to (4), if same replacements are made as in (3) in order to obtain (4), the same dynamics described with (11) and (12) for Hopfield's model (Hopfield, 1984) can be used to define and prove the stability of the *continuous Bayesian neural network model.* The generalized similarity function (6) is therefore:

$$Sim(V_1, ..., V_n) = \prod_j \left(\frac{P(V_j|V_1, ..., V_n)}{P(V_j)} \right)^{V_j} \quad (13)$$

2.4 Information score

Let the correct class of a testing instance T_i be C and the prior probabilty of class C be denoted by $P(C)$. Let the probability, returned by a classifier, that a given testing object T_i belongs to class C be $P'(C)$. We define the *information score* $Inf(T_i)$ of classifier's answer as follows:

1. if $P'(C) > P(C)$ then

$$Inf(T_i) = -\log_2 P(C) + \log_2 P'(C) \quad [bits]$$

i.e., the amount of obtained information is the entire amount of information necessary to correctly classify an instance into class C minus the remainder of information necessary to correctly classify that instance.

2. if $P'(C) = P(C)$ then

$$Inf(T_i) = 0 \quad [bits]$$

i.e., the system didn't change the prior probability of the correct class therefore we didn't obtain any information.

3. if $P'(C) < P(C)$ then

$$Inf'(T_i) =$$

$$-\log_2(1 - P(C)) + \log_2(1 - P'(C)) \quad [bits]$$

i.e., the amount of information returned by the system is the entire amount of information necessary to decide that an instance *does not* belong to class C minus the remainder of the information necessary to make that decision. As this information is in fact wrong we define the information score of the system's answer in this case as negative:

$$Inf(T_i) = -Inf'(T_i) \quad [bits]$$

The *average information score* of an answer is calculated over all testing instances:

$$Inf = \frac{\sum_i^{\#testing\_instances} Inf(T_i)}{\#testing\_instances}$$

Note that this assumes that prior probabilities of classes are known or can be reliably approximated with relative frequencies from training instances. We define also *the relative information score* Inf_r as a normalization of the average information score of an answer with the expected necessary information to classify one instance (i.e. entropy):

$$Inf_r = \frac{Inf}{-\sum_C (P(C) \log_2 P(C))} \times 100\%$$

2.5 Semi-naive Bayesian classifier

Here we will limit our discussion on a feedforward Bayesian neural network that calculates the probability $P(C_j|V_1, ..., V_n)$ of class C_j of an object, described with values $V_1, ..., V_n$ of attributes. Note that for Bayesian neural networks, described in sections 2.2 and 2.3, the class attribute is just one of attributes that describe the object while here it is the target attribute.

When calculating the probability of class C_j in (1) the influence of attributes A_i and A_l is defined with:

$$\frac{P(C_j|V_i)}{P(C_j)} \times \frac{P(C_j|V_l)}{P(C_j)} \quad (14)$$

If, instead of assuming the independence of values V_i and V_l, the values are joint, the corrected influence is given with:

$$\frac{P(C_j|V_iV_l)}{P(C_j)} \quad (15)$$

For joining the two values two conditions should be satisfied: the values of (14) and (15) should be sufficiently different while the approximation of

$P(C_j|V_iV_l)$ with relative frequency should be sufficiently reliable. For the estimation of the reliability of the probability approximation the theorem of Chebyshev can be used. The theorem gives the lower bound on the probability, that relative frequency f of an event after n trials differs from the factual prior probability p for less than ε:

$$P(|f-p| \leq \varepsilon) > 1 - \frac{p(1-p)}{\varepsilon^2 n} \quad (16)$$

The lower bound is proportional to n and to ε^2. In our case we are interested in the reliability of the following approximation:

$$\hat{P}(C_j|V_iV_l)) = \frac{N_{C_jV_iV_l}}{N_{V_iV_l}} \quad (17)$$

Therefore the number of trials n in (16) is equal to $N_{V_iV_l}$, i.e. the number of training instances having values V_i and V_l of attributes A_i and A_l, respectively. As prior probability p is unknown, for approximation of p at the right-hand side of (16) the worst case can be assumed, i.e. $p = 0.5$.

It remains to determine the value of ε. As we are interested also if the values of (14) times $P(C_j)$ and (17) are significantly different we use ε that is proportional to the difference between the two values. The joint values will influence all classes $C_j, j = 1 \ldots m$. Therefore, ε will be the average difference over all classes:

$$\varepsilon = \sum_{j=1}^{m} P(C_j) \times \left| P(C_j|V_iV_l) - \frac{P(C_j|V_i)P(C_j|V_l)}{P(C_j)} \right| \quad (18)$$

It is necessary to determine the threshold for the probability (16) above which decides when it is useful to join two values of two attributes. Empirically (Kononenko, 1991c), a typical value of the threshold that gives satisfactory results is 0.5. Therefore, the rule for joining two values states: join two values if the probability is greater than 0.5 that the theoretically correct (unknown) influence of values V_i and V_l differs, in average over all classes, from the used (approximated) influence, for less than the difference between used influence and the influence of the two values without joining them:

$$1 - \frac{1}{4\varepsilon^2 N_{V_iV_l}} \geq 0.5 \quad (19)$$

The values can be iteratively joint so that more than two values can be joint together enabling the semi-naive Bayesian classifier to discover higher order dependencies.

3 Discussion

3.1 The origin of Bayesian neural networks

The naive Bayesian classifier (that assumes the conditional independence of attributes) is fast, incremental, has no problems with overfitting the training data, and can naturally deal with missing data. Despite its naivety, it achieves the impressive classification accuracy on many real world problems.

Our early experience with the naive Bayesian classifier was presented at the "International School for the Synthesis of Expert's Knowledge" Workshop (Kononenko et al., 1984). We compared the performance of the naive Bayesian classifier with Assistant, an inductive learning algorithm for generating decision trees. Although both systems achieved similar results we claimed that Assistant has an obvious advantage as the generated knowledge in the form of decision trees is transparent to human experts. However, at the workshop Professor Donald Michie pointed out that "if the same amount of effort had been devoted to the development of the naive Bayesian classifier as was used for the development of Assistant, the Bayesian approach would certainly outperform Assistant". Today it seems that his prediction was correct.

We performed a series of experiments in various medical diagnostic problems in order to develop medical diagnostic expert systems. Physicians were never realy satisfied with Assistant's decision trees although Assistant achieved excellent classification performance. They complained that decision trees contain too few attributes and therefore poorly describe the patients (Pirnat et al., 1989). On the other hand the naive Bayesian classifier uses all available attributes. It turned out, that a simple interpretation of its decisions as the sum of information gains for/against certain diagnoses is transparent to physicians and acceptable for everyday use of such diagnostic system (Kononenko, 1989b; 1990b; 1991b; 1993a).

In 1986 the PDP group, with the book (Rumelhart & McClelland, 1986), caused the beginning of the exponential growth of the research effort devoted to neural networks (Anderson & Rosenfeld, 1988). One of the most notable contributions are famous Hopfield's papers (Hopfield, 1982; 1984), that described the single layered feedback neural network architecture with interesting properties. It turned out that the naive Bayesian classifier can be implemented on the same neural network architecture as was used by Hopfield but with appropriately modified learning rule and combination function (Kononenko, 1989a). There is a strong analogy between Bayesian neural network and Hopfield's model (see sections 2.2 and 2.3). Discrete and continuous Bayesian neural networks were defined and empirically they significantly outperformed Hopfield's model with respect to classification accuracy (Kononenko, 1990a).

Although the naive Bayesian classifier is fast, incremental, has excellent performance on real life problems, and can explain its decisions as the sum of information gains, its naivety may result in poor performance in domains with strong dependencies among attributes. To avoid independence assumption we defined the "semi-naive" Bayesian classifier. The idea is to explicitly search for the dependencies between the values of different attributes and if such dependency is discovered the two values are joint (Kononenko, 1991c). The algorithm must solve the trade-off between the non-naivety and the reliability of probability approximations (see section 2.5).

A subproblem in comparison of different classifiers was that classification accuracy may be misleading especially for the classification problems with high variations in prior probabilities of different classes. This problem was particularly illuminated with experiments in two medical diagnostic problems. In the "breast cancer" problem classifiers typically achieved the classification accuracy of about 80% while in the "primary tumor" problem the classification accuracy was about 45%. The classification accuracy for breast cancer seems high while for primary tumor very poor. However, in breast cancer there are only 2 classes, and one has the prior probability equal to 80%! Therefore, a simple classifier that each object classifies into the majority class would also achieve a "high" classification accuracy. On the other hand, in primary tumor, there are 22 possible classes, and the majority class contains only 25% of cases. 45% of the classification accuracy is in fact a fairly good result in this problem!

Professor Michie helped us by suggesting entropy as the basis for the appropriate measure of classifier's performance. We developed the evaluation function called "information score" (Kononenko & Bratko, 1991), that can evaluate answers of classifiers in the form of a probabilistic distribution, appropriately considers differences in prior probabilities of classes, and has natural interpretation that stems from the information theory (see section 2.4).

3.2 The contribution of Bayesian neural networks

The key intellectual contribution is that the use of probability and information theory can naturally (and simply) solve several open issues in machine learning. Instead of using ad-hoc approaches or "fuzzy arithmetic" approaches (Kaufmann & Gupta, 1985) which are widely used in machine learning and neural networks, we used the *probability* (Good, 1950; 1964) as the basic tool for modeling, and the strongly related *information theory* (Shannon & Weaver,1949) as the basic tool for the interpretation.

We showed that the probability can be used to model neural networks by introducing the naive Bayesian formula as a combination function while preserving the basic Hebbian learning rule, which is one of the basic learning rules in neural networks that has also strong biological plausibility (Hebb, 1949). We also used the probability to detect the dependencies among attributes by considering the basic definition of the dependency and the reliability of probability approximations (see section 2.5).

The logarithm of the probability of an event can be interpreted as the information necessary to find out that the event has happened. The interpretation of the naive Bayesian formula directly

follows from this. It is simple, natural, and transparent to human users. Besides, it shows direct relationship of Bayesian neural networks with the Hopfiel's model and it shows also the analogy between the Hopfield's *energy* of the network's state and the *entropy* (or the *information content*) of the state (see section 2.2).

The definition of the "information score" (see section 2.4) of a classifiers answer naturally follows from the definition of the information. The information score has several advantages as was discussed in the previous subsection.

3.3 Open issues

Although the semi-naive Bayesian classifier partially solves the problem of naivety, there is still an open problem which seems to be the key issue of machine learning in general. Namely, for particular problems (e.q. parity problems of higher degrees) the discovering of dependencies between attributes may be either

1. unfeasible due to combinatorial explosion or
2. the discovered dependencies cannot be reliably estimated due to small number of training examples.

In such cases efficient heuristic algorithms are needed to discover the dependencies (first problem) or discover new attributes by deriving them from existing ones (second problem). For such new attributes it should be possible to reliably estimate probabilities from the given training set.

It is well known that the result of the learning strongly depends on the knowledge representation. If the attributes, used to describe objects, are primitive and low level, learning systems will not be able to extract regularities from data. On the other hand, if attributes are high level and informative, most of classification systems will achieve similar classification accuracy. In the former situation one of the two problems mentioned above should be solved. The development of efficient heuristic algorithms for solving these two problems seems to be currently the main research issue in machine learning.

3.4 Possible applications

The developed algorithms may be used either as a tool for analysing data or as a basis for an expert system shell. In fact, the majority of applications that were done in Ljubljana Artificial Intelligence Laboratories (Urbančič et al., 1991), that involved machine learning algorithms, used machine learning as an efficient tool for data analysis. On the other hand, general expert system shell based on Bayesian neural networks may be developed (Kononenko, 1991b) and several prototypes were already implemented (Ritoša, 1992; Grahor, 1992).

One promising wide area of potential applications is medical diagnosis (Kononenko, 1993a). Here the major requirement for any system for supporting medical diagnostic decisions is that decisions of the system must be transparent to physicians. The semi-naive Bayesian classifier seems to be the most appropriate for that purpose. Its decisions can be interpreted as the sum of information gains from different attributes (symptoms, laboratory tests, etc) for/against the diagnosis which is similar to the way physicians actually explain their decisions. We are currently developing one such application in the problem of the prediction of hip-bone break recovery (Kukar, 1993).

Other potential uses of the developed methods include the use of the information score as a simple, transparent, and unbiased evaluation criterion for estimating classifier's performance. Unfortunately, not many researchers have begun to use it in their experiments so far. A feedback Bayesian neural network can be used also as an auto-associative memory that is hopefully more efficient than the Hopfield's auto-associative memory. This, however, must yet be analysed.

3.5 Refinements using today's knowledge

After the dissertation was completed m-estimate of probabilities was developed and used independently by Cestnik (1990) and Smyth & Goodman (1990) to estimate the conditional probabilities in the naive Bayesian formula. Cestnik (1990) showed that m-estimate drastically improves the

classification performance of the naive Bayesian classifier in several real world problems.

The experiments described in the dissertation would be more attractive if m-estimate was used instead of the relative frequency. Besides, m-estimate should even improve the explanation ability of the naive and the semi-naive Bayesian classifier, as it eliminates the high fluctuations of probabilities when relatively small samples of instances are used for training.

3.6 Continuation of the work

The most important open issue described in the dissertation that was later completed is the problem of continuous data. The naive Bayesian classifier is designed to deal with discrete attributes while continuous attributes need to be discretized (its values grouped into intervals) in advance. The problem is how many intervals should one use. Too many intervals may be a too detailed split resulting in a small number of training instances corresponding to each interval. This causes the unreliable estimation of probabilities. On the other hand, too few intervals may result in the loss of the information content of a continuous attribute. Another problem with discretization is the loss of the order of values of the continuous attribute.

We developed a *fuzzy discretization* of continuous attributes that "softens" the bounds between neighbour intervals (Kononenko, 1991a; 1992a). Such discretization may use a lot of intervals without the loss of the reliability of probability approximations. It also implicitly keeps the information about the order of values.

Currently, the *multistrategy learning* is becoming the central research area. The idea is to combine several different learning strategies and/or apply several different learning algorithms on the same problem and then try to combine their results. One promising approach to combining the answers of different classifiers, that was used by Smyth et al. (1990), is (again) the naive Bayesian formula. We showed experimentally that the naive Bayesian combination of answers of different decision rules is acceptable and superior to several other combination methods (Kononenko & Kovačič, 1992; Kononenko, 1992b).

Another open issue, tackled with current research, was described in details in section 3.3. We developed the *successive naive Bayesian classifier* (Kononenko, 1993) and Langley (1993) developed the *recursive Bayes*. Both approaches try to overcome the independence assumption by several successive applications of the naive Bayesian classifier on intermediate results. Both approaches, however, are not satisfactory as the classification accuracy is not better than that of the naive Bayesian classifier and, besides, both systems loose the explanation ability of the simple naive Bayesian classifier. Therefore, currently the "semi-naive" Bayesian classifier seems the most promising.

Acknowledgements

I am indebted to many colleagues for their contribution to the present work. I would like to thank my mentor Prof. Ivan Bratko for many years of consultations, introduction, and guidance into the scientific work. Colleagues from the Artificial Intelligence Laboratory at the Faculty of Electrical Engineering and Computer Science and at the Jožef Stefan Institute have contributed to the dissertation with many suggestions and discussions. I am indebted to Bojan Cestnik for his cooperation in the development and implementation of ASSISTANT and for many informal discussions about machine learning. Informal discussions with colleagues Matevž Kovačič, Alen Varšek, Aram Karalič, Matjaž Gams and Nada Lavrač have often contributed to clearer notions and ideas. Collecting and assembling the experimental data would not be possible without the invaluable help of physicians specialists Dr. Matjaž Zwitter, Dr. Sergej Hojker and Dr. Vlado Pirnat from the University Medical Center in Ljubljana. Prof. Donald Michie has contributed with initial suggestions to the appearance of the article (Kononenko & Bratko, 1991) that defines and analyses the information score. This research was supported by Slovenian Ministry of Science and Technology. The reported work was done in the Artificial Intelligence Laboratory at the Faculty of Electrical Engineering and Computer Science in Ljubljana.

References

[1] Anderson J.A. & Rosenfeld E. (eds.) (1988) *Neurocomputing: Foundations of Research.* Cambridge-London: MIT Press.

[2] Cestnik, B. (1990) Estimating probabilities: A crucial task in machine learning, *Proc. European Conference on Artificial Intelligence*, Stockholm, August 1990, pp.147-149.

[3] Good I.J. (1950) *Probability and the weighing of evidence.* London: Charles Griffin.

[4] Good I.J. (1964) *The Estimation of Probabilities - An Essay on Modern Bayesian Methods*, Cambridge: The MIT Press.

[5] Grahor J. (1992) Simulation of continuous Bayesian neural networks and an expert system shell (in slovene), B.Sc. Thesis. University of Ljubljana, Faculty of electrical eng. & computer sc., Ljubljana, Slovenia.

[6] Hebb, D.O. (1949) *The Organization of Behavior.* New York: Wiley.

[7] Hopfield J.J. (1982) Neural networks and physical systems with emergent collective computational abilities. *Proc. of the National Academy of Sciences* 79:2554-2558.

[8] Hopfield J.J. (1984) Neurons with graded response have collective computational properties like those of two-state neurons. *Proc. of the National Academy of Sciences* 81:4586-4590.

[9] Ihara J. (1987) Extension of Conditional Probability and Measures of Belief and Disbelief in a Hypothesis Based on Uncertain Evidence. *IEEE trans. on pattern analysis and machine intelligence*, 9:561-568.

[10] Kaufmann A. & Gupta M. (1985) *Introduction to fuzzy arithmetic*, New York: Van Nostrand Reinhold.

[11] Kononenko, I. (1989a) Bayesian neural networks. *Biological Cybernetics*, 61:361-370.

[12] Kononenko, I. (1989b) Interpretation of neural networks decisions. *Proc. of IASTED Internat. Conf. on Expert Systems*, Zurich, Switzerland, June 26-28, pp. 224-227.

[13] Kononenko, I. (1989c) ID3, sequential Bayes, naive Bayes and Bayesian neural networks. *Proc. 4th European Working Session on Learning*, Montpellier, France, December 1989, pp.91-98.

[14] Kononenko I. (1989d) Neural networks and artificial intelligence, *Proc. 11th Internat. Conf. Computer at the university*, Cavtat, June 5-9 1989,pp 8.3.1-8.3.8. (also: ISSEK Workshop, Udine, Sept. 1989).

[15] Kononenko I. (1989e) Neural networks (in slovene), *Informatica*, 13:56-71.

[16] Kononenko I. (1990a) Bayesian neural networks (in slovene), Ph.D. Thesis, University of Ljubljana, Faculty of electrical engineering & computer science, Ljubljana, Slovenia.

[17] Kononenko I. (1990b) Comparison of inductive and naive Bayesian learning approaches to automatic knowledge acquisition. In: B. Wielinga et al. (eds.) *Current Trends in Knowledge Acquisition*, Amsterdam: IOS Press.

[18] Kononenko, I. (1991a) Feedforward Bayesian neural networks and continuous attributes, *Proc. Int. Joint Conf. on Neural Networks IJCNN-91*, Singapore, Nov. 1991, pp. 146-151.

[19] Kononenko I. (1991b) Bayesian neural network based expert system shell, *International Journal on Neural Networks*, 2:43-47.

[20] Kononenko I. (1991c) Semi-naive Bayesian classifier, *Proc. European working session on learning*, Porto, March 1991, pp.206-219.

[21] Kononenko I. (1992a) Naive Bayesian classifier and continuous attributes, *Informatica*, 16:1-8.

[22] Kononenko I. (1992b) Combining decisions of multiple rules, In: B.du Boulay & V.Sgurev (eds.) *Artificial Intelligence 5: methodology, systems, applications*, North Holland.

[23] Kononenko I. (1993) Successive naive Bayesian classifier, *Informatica*, 17:167-174.

[24] Kononenko I. (1993a) Inductive and Bayesian learning in medical diagnosis, *Applied Artificial Intelligence*, 7:317-337.

[25] Kononenko, I. & Bratko, I. (1991) Information Based Evaluation Criterion for Classifier's Performance. *Machine Learning Journal*, 6:67-80. (also: *Proc. ISSEK Workshop*, Udine, Sept. 1989).

[26] Kononenko, I., Bratko, I., Roškar, E. (1984) Experiments in automatic learning of medical diagnostic rules, Technical report, Faculty of electrical engineering & computer science, Ljubljana, Slovenia (presented at ISSEK Workshop, Bled, august, 1984).

[27] Kononenko I. & Kovačič M. (1992) Learning as optimization: Stochastic generation of multiple knowledge, In D.Sleeman & P.Edwards (eds.) *Machine learning: Proc. 9th Intern. Conf.*, Morgan Kaufmann, San Mateo, CA.

[28] Kosko, B. (1988) Bidirectional Associative Memories. *IEEE Trans. on Systems, Man, and Cybernetics*, 18:49-60.

[29] Kukar M. (1993) An application of machine learning in hip-bone break diagnosis (in slovene), B.Sc. Thesis. University of Ljubljana, Faculty of electrical eng. & computer sc., Ljubljana, Slovenia.

[30] Langley P. (1993) Induction of recursive Bayesian classifier, In. P.Brazdil (ed.) *Machine learning: Proc. European Conf.*, Springer-Verlag, pp. 153-164.

[31] Michalski, R.S., Carbonell, J.G. & Mitchell, T.M. (eds.) (1983/1986) *Machine Learning: An Artificial Intelligence Approach. Volume I* by Tioga Publ. Comp., 1983 and *Volume II* by Morgan Kaufmann Publ. Inc., 1986.

[32] Michie, D., A. Al Attar (1991) Use of Sequential Bayes with Class Probability Trees. In: J.E. Hayes-Michie, D.Michie & E. Tyugu (eds.) *Machine Intelligence 12*, Oxford: Oxford University Press.

[33] Pirnat V., Kononenko I., Janc T., Bratko I. (1989) Medical Estimation of Automatically Induced Decision Rules, *Proc. of 2nd Europ. Conf. on Artificial Intelligence in Medicine*, City University, London, August 29-31 1989, pp.24-36.

[34] Ritoša V. (1992) An expert system shell based on Bayesian neural networks (in Slovene), B.Sc. Thesis. University of Ljubljana, Faculty of electrical eng. & computer sc., Ljubljana, Slovenia.

[35] Rumelhart, D.E. & McClelland, J.L. (eds.) (1986a) *Parallel Distributed Processing, Vol. 1: Foundations.* Cambridge: MIT Press.

[36] Shannon & Weaver (1949) *The mathematical theory of communications.* Urbana: The University of Illinois Press.

[37] Smyth P. & Goodman R.M. (1990) Rule induction using information theory. In: G.Piarersky & W.Frawley (eds.) *Knowledge Discovery in Databases*, MIT Press.

[38] Smyth P., Goodman R.M., Higgins C. (1990) A hybrid Rule-based Bayesian Classifier, *Proc.European Conf. on Artificial Intelligence*, Stockholm, August, 1990, pp. 610-615.

[39] Urbančič T., Kononenko I., Križman V. (eds.) (1991) Review of applications by Ljubljana Artificial Intelligence Laboratories, Technical report IJS DP-6218, Jožef Stefan Institute, Ljubljana, Slovenia.

Appendix: Bayesian neural networks based on probability ratio

Discrete model

Here, we will briefly define the Bayesian neural network which is more appropriate for comparison with Hopfield's model as it assumes the symmetric interpretation of values of neurons. It is based on Good's (1950) *plausibility* as opposed to Shannon's *entropy* (Shannon & Weaver, 1949), or, from another point of view, it is based on *odds* defined as $\frac{P(X)}{P(\overline{X})}$ as opposed to *probability*.

The Bayesian neural network will implement the 'naive' Bayesian classifier based on odds. Each neuron will now represent one attribute, and one neuron will stand for a class. Each neuron has only two possible values (0 and 1) and therefore all attributes will have only two possible values and there will be only two possible classes. Of course, it will be possible to solve also problems with multivalued attributes and with more than two classes by appropriate coding (binarization of attributes and classes, i.e. one multivalued attribute will be represented with more binary attributes). Note that now the two values (0 and 1) are not interpreted anymore as inactive and active as each represents one

value of an attribute. Again, network makes no difference among attributes and classes.

As all attributes are binary, the value of i-th neuron (i.e. i-th attribute) will be represented with $V_i^{X_i}, X_i =$ 0 or 1. If the independence of influences of other neurons to the activation level of the current neuron is assumed, the analogous formula to (1) can be used:

$$\frac{P(V_j^1|V_1^{X_1},...,V_n^{X_n})}{P(V_j^0|V_1^{X_1},...,V_n^{X_n})} =$$

$$\frac{P(V_j^1)}{P(V_j^0)} \prod_{i \neq j} \frac{Z_{ji}(1,X_i)}{Z_{ji}(0,X_i)} \quad (20)$$

where

$$Z_{ji}(X,Y) = \frac{P(V_j^X, V_i^Y)}{P(V_j^X)P(V_i^Y)} \quad (21)$$

is the influence of i-th neuron on j-th neuron. Note that $P(V_j^0)$ can be calculated from $P(V_j^1)$ and that all $P(V_j^{X_j}, V_i^{X_i})$ can be calculated from $P(V_j^1, V_i^1), P(V_i^1)$ and $P(V_j^1)$. Therefore, if j-th neuron is to be able to calculate value of (21) then it needs the same information as in the Bayesian neural network defined in section 2.2 and *the same learning rule can be applied.*

In the execution phase the network again iterates. Each neuron calculates the quotient among the probability of state 1 and the probability of state 0 as defined in (20). Note that all neurons (not only the active ones as before) will influence the quotient. The threshold value for changing the neuron's state will now be the quotient among prior probabilities of state 1 and state 0 (the first factor at the right hand side of (20)).

The network iterates until there are no more changes in neuron's state (the network reaches a fixed point). In (Kononenko, 1989c) it is shown that this always happens in a finite number of iterations if the network works asynchronously (only one neuron changes its state at a time). The proof is analogous to that in (Kononenko, 1989a). The *measure of similarity* between current activation levels of neurons and their current states $X_i, i = 1..n$, in this case is:

$$Sim(V_1^{X_1},...,V_n^{X_n}) = \prod_j \frac{P(V_j^{X_j}|V_1^{X_1},...,V_n^{X_n})}{P(V_j^{X_j})} \quad (22)$$

The similarity is greater if the calculated probability of the current state of a neuron is greater than the prior probability of that state and vice versa. It is shown in (Kononenko, 1989c) that when one neuron changes its state the similarity increases. As there is a finite number of possible states of a network the similarity measure is bounded and therefore the network will always converge to a fixed point.

Continuous model

The *weight of evidence* from i-th neuron, in favor of the conclusion that j-th neuron is active if the i-th neuron is active with probability 1, is given with (Good, 1950; Michie & Al Attar, 1991):

$$log_2 \frac{Z_{ji}(1,1)}{Z_{ji}(0,1)}$$

The expected weight of evidence given that i-th neuron is active with probability V_i (which represents the current state of i-th neuron), is given with:

$$V_i \times log_2 \frac{Z_{ji}(1,1)}{Z_{ji}(0,1)} + (1 - V_i) \times log_2 \frac{Z_{ji}(1,0)}{Z_{ji}(0,0)} \quad (23)$$

Therefore, the generalized (20) is:

$$\frac{P(V_j^1|V_1,...,V_n)}{P(V_j^0|V_1,...,V_n)} =$$

$$\frac{P(V_j^1)}{P(V_j^0)} \prod_{i \neq j} \left[\left(\frac{Z_{ji}(1,1)}{Z_{ji}(0,1)} \right)^{V_i} \times \left(\frac{Z_{ji}(1,0)}{Z_{ji}(0,0)} \right)^{(1-V_i)} \right] \quad (24)$$

and the corresponding similarity measure:

$$Sim(V_1,...,V_n) =$$

$$\prod_j \left[\left(\frac{P(V_j^1|V_1,...,V_n)}{P(V_j^1)} \right)^{V_j} \times \left(\frac{P(V_j^0)}{P(V_j^0|V_1,...,V_n)} \right)^{(1-V_j)} \right] \quad (25)$$

Let $T_{ji}(X,Y)$ stand for $-log_2 Z_{ji}(X,Y)$. The minus logarithm of (24) does not correspond directly to (4) as it is of the form:

$$A_j = \sum_{i \neq j} \Big(V_i \times (T_{ji}(1,1) - T_{ji}(0,1)) +$$

$$(1 - V_i) \times (T_{ji}(1,0) - Tji(0,0)) \Big) + I_j \quad (26)$$

The equation that describes the dynamics of the continuous Bayesian neural network based on odds is obtained from (11) by replacing A_j with (26). As (26) differs from (4) the convergence proof is not so obvious as for the continuous Bayesian neural network based on probability.

From Hopfield's (1984) proof it can be shown that the sufficient condition for such a model to converge is given with the relation

$$\frac{dE_1}{dt} = \sum_j \left(\frac{dV_j}{dt} \times A_j \right) \quad (27)$$

where E_1 is a function of neural network's state and A_j the activation level of j-th neuron that appears in (11). Hopfield uses the following E_1 as the part of the energy function for the continuous model (again omitting the condition $i \neq j$):

$$E_1(V_1, ..., V_n) = -\frac{1}{2}\sum_j \sum_{i \neq j} V_j V_i T_{ji} + \sum_j I_j V_j \quad (28)$$

If, instead of A_j in (11), activation level defined with (26) is used and if, instead of (28), E_1 is defined using the minus logarithm of (25):

$$E_1(V_1, ..., V_n) = \sum_j I_j V_j +$$

$$\frac{1}{2}\sum_j \sum_{i \neq j} \Big(V_j V_i T_{ji}(1,1) + V_j(1 - V_i) T_{ji}(1,0) +$$

$$(1 - V_j) V_i T_{ji}(0,1) + (1 - V_j)(1 - V_i) T_{ji}(0,0) \Big) \quad (29)$$

then the relation (27) holds, namely:

$$\begin{aligned} \frac{dE_1}{dt} &= \sum_j \left(\frac{dE_1}{dV_j} \times \frac{dV_j}{dt} \right) \\ &= \frac{1}{2}\sum_j (A_j \times \frac{dV_j}{dt}) + \frac{1}{2}\sum_i (A_i \times \frac{dV_i}{dt}) \\ &= \sum_j (\frac{dV_j}{dt} \times A_j) \end{aligned} \quad (30)$$

as $T_{ji}(X, Y) = T_{ij}(Y, X)$.

Lecture Notes in Machine Learning

Xindong Wu
Department of Computer Science, James Cook University,
Townsville, QLD 4811, Australia.
Email: xindong@cs.jcu.edu.au

Keywords: Artificial intelligence, machine learning, symbolic approaches, educational paper

Edited by: Rudi Murn
Received: October 4, 1993 **Revised:** January 17, 1994 **Accepted:** March 25, 1994

Machine learning is a major area in artificial intelligence (AI), and has seen sustained research and a growing presence in lecture syllabuses over recent years. It has been commonly recognized as a feasible solution to the so called knowledge bottleneck problem in transforming knowledge from human experts to knowledge-based systems. Also, as learning is the essence of human intelligence, only when we have computer systems that can learn can we have real AI. Researchers have devised quite a few sound and efficient learning algorithms (such as ID3 and HCV); a number of universities have opened machine learning courses in their AI-related undergraduate and/or Master of Science programs.
This document contains a compressed set of lecture notes designed for the Machine Learning module in our Advanced Artificial Intelligence *course at James Cook University. They are biased towards a basic exposition of the practical symbolic approaches only.*

Introduction

Artificial intelligence (AI) is a subject concerned with the problem of how to make machines perform such tasks, as vision, planning and diagnosis, that usually need human intelligence and are generally difficult to be carried out with conventional computer science technology. Machine learning research in AI is concerned with the problem of how machines can acquire the knowledge that might enable them to perform those tasks. Along with the recognition of the so called knowledge bottleneck problem in transforming knowledge from human experts to knowledge-based systems, machine learning research has been expanding rapidly over recent years. Also, as learning is the essence of human intelligence, only when we have computer systems that can learn can we have real AI. Machine learning has been predicted to be one of the main research streams of AI and computer science for the 1990's.

In some ways, machine learning is a rather broad area, with different research seemingly addressed to very different problems. A reasonable curriculum should cover the following three parts to give a general introduction to the state of the art in machine learning. Firstly, we would need to describe various mechanical (or interview-based), interactive (or semi-automatic), and automatic knowledge acquisition techniques for knowledge-based systems development. These topics are very popular to expert systems like courses. Secondly, it is necessary to address a comprehensive set of machine learning paradigms, such as attribute-based induction, incremental induction, learning by analogy/case based reasoning, knowledge rich learning (such as AM and EURISKO), inductive logic programming, explanation-based learning, concept formation, statistical techniques, genetic algorithms, and connectionist learning. These paradigms have been adopted in an independent machine learning course at a few universities. Finally, we should also survey some advanced research-oriented topics such as computational learning theory (especially the PAC model) and knowledge discovery in databases.

However, since the lecture notes contained in this document are designed for the Machine

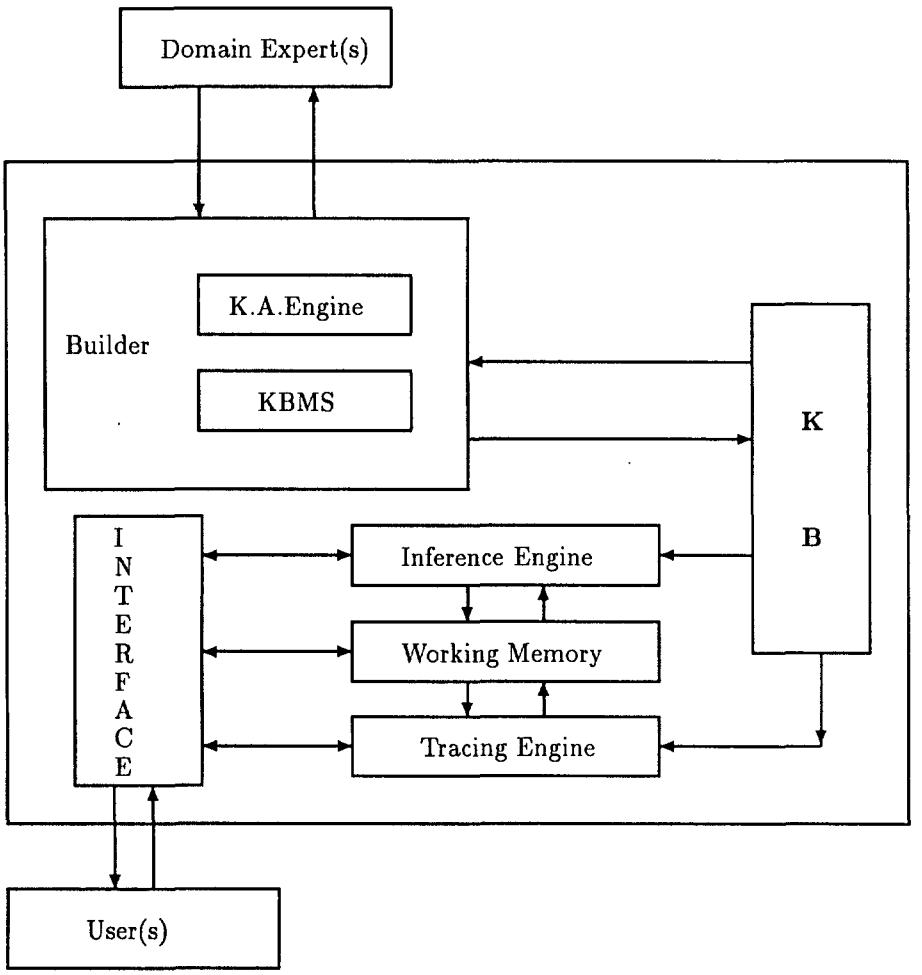


Figure 1: An Expert System

Learning module in an AI course, we will concentrate on the symbolic techniques (in particular rule induction techniques) used in AI. We do not cover genetic, connectionist and statistical paradigms because (1) the time and size are restricted, and (2) when and only when learning about what these independent topics are will it be easy for the students to know how to apply them to machine learning. We do not cover inductive logic programming and learning by analogy (or case-based reasoning) either, because research on inductive logic programming is still carried out in small-scaled laboratories only and learning by analogy is very domain dependent.

1 Background in Knowledge-Based Systems

1.1 Knowledge Acquisition in Knowledge-Based Systems

AI systems are knowledge processing systems. Knowledge representation, inference (including search and control) and knowledge acquisition are three fundamental techniques in AI.

Figure 1 shows the system structure of an expert system, where the arrows show the predominant direction of data flow through the system. The system structure consists of the following modules:

- an interface to facilitate human-computer communication,
- a working memory/data base, which stores

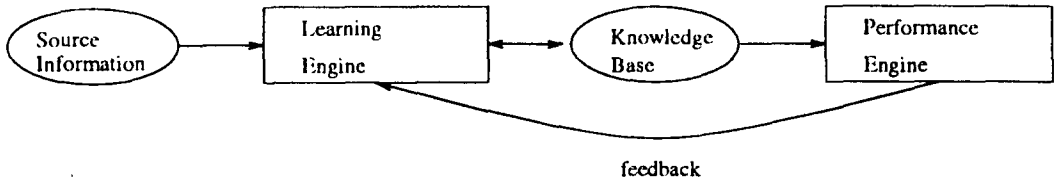


Figure 2: The Structure of a Learning System

the evidence and intermediate results of a specific problem to be solved,

- a knowledge base, KB, acquired from a domain expert or a group of experts,
- an inference engine to solve users' problems by applying the knowledge base,
- an explanation/tracing engine to tell the user how the solution has been achieved,
- a knowledge acquisition engine, K.A.Engine, to acquire and modify the knowledge in the knowledge base when necessary, and
- a knowledge base management subsystem, KBMS, to maintain the knowledge (e.g., detecting inconsistency such as repetitions, redundancy and contradictions) in the knowledge base.

From Figure 1, we can easily see one of the major differences between AI systems and conventional programming (such as software engineering and relational data base technology): Knowledge is clearly divided into three levels – data, knowledge and knowledge manipulation – in expert systems; While in conventional computer science, we deal with programs and data only.

The performance of an expert system relies on the knowledge embedded in the system, and therefore, knowledge acquisition is essential to the success of the system development. It has been identified as the knowledge bottleneck problem in expert systems or knowledge-based systems [1] development. Knowledge acquisition involves the initial acquisition of knowledge and knowledge refinement of the knowledge base during/after problem solving.

[1] Expert systems are also called knowledge-based systems since it has been recognized that those systems need general domain knowledge as well as expertise.

There are three different kinds of knowledge acquisition techniques:

- mechanical (or interview-based) ones,
- interactive (or semi-automatic) ones, and
- automatic ones.

Section 1.3 outlines the main differences between them.

1.2 Components in a Learning System

Figure 2 (adapted from [Cohen & Feigenbaum 82]) shows the structure of a simplified learning system.

In the diagram, the source information (e.g., relational tuples from a relational data base or advice from a human expert) is the input to the learning engine, which carries out the learning task and produces knowledge (e.g., rules) for the knowledge base. Sometimes, the learning engine uses existing knowledge in the knowledge base to produce new knowledge. The performance engine makes sure that the knowledge produced is useful. In the case of an expert system, the performance engine is its inference engine, which makes uses of the produced knowledge to solve users' problems. It is also quite common that the performance engine just tests the knowledge produced on more data (e.g., relational tuples which have or have not been presented in the source information). The performance engine produces feedback (which can sometimes be viewed as another kind of input) to the learning engine, according to which the learning engine may decide between the following actions: (1) continue the learning process, (2) ask for more information from the source information, and (3) stop.

1.3 Learning Strategies: Overview

The source information is the most important factor affecting the design of learning systems. It determines the task to be carried out in the learning engine. In general, different learning strategies have different levels of source information.

Rote learning

The input to the learning engine from the source information and the performance engine is exactly what is needed in the knowledge base. All the learning engine needs to do is remember and organize the input in the knowledge base. There are two different kinds of cases with rote learning. One is that the source information provides the exact knowledge for the performance engine. The other is that the learning engine remembers each specific user's problem (which is not mentioned in the diagram and is supposed to be solved by the performance engine) and the problem solving process or outcome of the performance engine in the knowledge base, so that the performance engine can take advantage of these memories to solve new problems when they are the same as the old ones.

Mechanical (or interview-based) knowledge acquisition mentioned in Section 1.1 falls into this category: knowledge engineers (the developers/programmers of expert systems) elicit knowledge from experts and then store it via an editor (the simplest learning engine) in the knowledge base.

Learning by being told or Learning by advice taking

The information provided by the source information is too abstract or too general to be adopted in the knowledge base directly, and thus, the learning engine needs to transform the input into a form readily usable by the performance engine, and fill in the details. This transformation is called *operationalization*. For example, when trying to form a description for a dog, the advice might be 'Don't count in cats'. The learning engine in this case needs to find some features of a dog in the description which can distinguish dogs from cats. To some extent, the operationalization is similar to the compiling task performed by high level programming compilers that convert unexecutable programs in high level languages into directly interpretable machine code. However, learning by advice taking is normally domain dependent and heuristic: the learning engine needs to have some background knowledge in the knowledge base, and the transformation process needs to hypothesize the missing details in the advice.

Interactive (or semi-automatic) knowledge acquisition mentioned in Section 1.1 normally makes use of both rote learning and learning by being told. Domain experts hold knowledge, most of which is readily usable by the performance engine, and input the knowledge out of the direct dialogue with an intelligent editor (the learning engine). What the editor needs to do is transform the experts' input into the knowledge base and check inconsistencies.

Learning from examples

The input to the learning engine is too specific and too detailed cases, e.g., medical cases. The learning engine is to generate general rules which are applicable to these cases and other unseen ones as well. For example, the rules can be the relationships between medical symptoms and certain diseases.

Learning from examples is the most attractive area in machine learning research and the main theme of our lectures. It has been seen as not only a feasible way but also the only way of avoiding the knowledge bottleneck problem in transforming knowledge from human experts to knowledge-based systems. While it is often difficult for an expert to articulate his expertise explicitly and clearly, it is usually relatively easy to document case studies of his skill at work.

According to the format of the examples and whether the knowledge base has already possessed relevant domain knowledge, learning from examples can be further divided into many learning paradigms: attribute based learning, incremental induction, concept formation (or unsupervised learning), inductive logic programming, explanation based learning, and so on.

Learning by analogy

Given as input a collection of past cases, which are relevant to the new problem to be solved by the performance engine, and the past experiences of solving these past cases, the learning engine is supposed to find out the hypotheses and/or strategies for the new problem.

Learning by analogy is one of the most efficient learning strategies used in human problem solving. It, with the name of *case-based reasoning*, has been viewed as one of the most popular research topics in expert systems. However, case-based reasoning needs effective support of domain knowledge to judge the similarity between old cases and the new problem and to adjust old experiences to it. We will not cover this topic in our lectures.

2 Attribute-Based Induction

2.1 The Paradigm

Existing work on machine learning has concentrated in the main on inducing rules from unordered sets of examples, especially attribute-based induction, an inductive formalism where examples are described in terms of a fixed collection of attributes. The learning systems in commercial use today are almost exclusively inductive ones.

Attribute based paradigm can be briefly described as follows. Given a discrete finite attribute space of a dimensions, $E = D_1 \times ... \times D_a$, where each D_j $(j = 1, ..., a)$ is a finite set of symbolic values or a numerical interval, an example, or a case, $e=(V_1, ..., V_a)$ is an element of E means $V_j \in D_j$. A positive example is such an example that belongs to a known class which, say, has a specific name in E. All the other examples which do not belong to the known class can be called negative examples (NE) at the moment we are considering the known class. The induction task is to generate a description, say production rules or a decision tree, that covers all of the positive examples (PE) against[2] NE or classifies them correctly.

2.2 Decision Trees and ID3

2.2.1 Developer and Background

ID3 [Quinlan 79] is the best known learning algorithm to date. It was developed by Quinlan out of the Concept Learning System (CLS) by Hunt [Hunt et al. 66].

CLS is a learning mechanism which accepts a set of training examples and constructs a representation in the form of a decision tree, which is equivalent to a disjunctive rule. The decision tree is structured so that each leaf node has a target output associated with it. An arbitrary input is processed by simply applying the tree to the input (i.e. propagating the input down through the tree). This produces a leaf node which in turn yields the target output.

Main steps in the CLS algorithm can be described as follows.

```
S1: T <- the whole training set.
    Create a T node.
S2: If all examples in T are positive,
    create a 'yes' node with T as its
    parent and stop.
S3: If all examples in T are negative,
    create a 'no' node with T as its
    parent and stop.
S4: Select an attribute X with values
    V1, ..., VN and partition T into
    subsets T1, ..., TN according to
    their  values on X.
    Create N Ti nodes (i=1,...,N) with
    T as their parent and X=Vi as the
    label of the branch from T to Ti.
S5: For each Ti do: T <- Ti and goto S2.
```

2.2.2 Algorithm Description

Quinlan modified the CLS algorithm in two ways.

First, he added a process known as *windowing*. This was designed to enable the algorithm to cope with very large training sets.

If the training set is very large then, rather than process the entire set in one, it may be more efficient to process a small sample first. If the sample is a representative of the complete set, the decision tree produced will be similar to the one which we would get by processing the entire training set. Once we have produced a tentative tree we can then gradually perfect it. To do this we simply search through the training set looking for any *(input, output)* pairs which are not properly represented and each time we find such an exception we modify the tree appropriately. However, windowing does not feature very strongly in recent work.

Second, and more importantly, Quinlan devised an information theoretic heuristic (the *entropy* measure) which decided how to split the inputs

[2] 'against' is used to mean that the description should cover none of the negative examples.

at each stage of the tree growing process thus enabling smaller and therefore more efficient decision trees to be constructed.

ID3 works as follows.

Suppose $T = PE \cup NE$ where PE is the set of positive examples and NE is the set of negative examples, $p = |PE|$ and $n = |NE|$. An example e will be determined to belong to PE with probability $p/(p+n)$ and NE with probability $n/(p+n)$. By employing the information theoretic heuristic, a decision tree is considered as a source of message, "PE" or "NE", with the expected information needed to generate this message, given by

$$I(p,n) = \begin{cases} -\frac{p}{p+n}log_2\frac{p}{p+n} - \frac{n}{p+n}log_2\frac{n}{p+n} & \text{if } p \neq 0 \ \& \ n \neq 0 \\ 0 & \text{otherwise.} \end{cases} \quad (1)$$

If attribute X with value domain $\{v_1, ..., v_N\}$ is used for the root of the decision tree, it will partition T into $\{T_1, ..., T_N\}$ where T_i contains those examples in T that have value v_i of X. Let T_i contain p_i examples of PE and n_i of NE. The expected information required for the subtree for T_i is $I(p_i, n_i)$. The expected information required for the tree with X as root, $EI(X)$, is then obtained as weighted average.

$$EI(X) = \sum_{i=1}^{N} \frac{p_i + n_i}{p+n} I(p_i, n_i) \quad (2)$$

where the weight for the i-th branch is the proportion of the examples in T that belong to T_i. The information gained by branching on X, $G(X)$, is therefore

$$G(X) = I(p,n) - EI(X). \quad (3)$$

ID3 examines all candidate attributes, chooses X to maximize G(X), constructs the tree, and then uses the same process recursively to construct decision trees for residual subsets $T_1, ..., T_N$. For each T_i $(i = 1, ..., N)$: if all the examples in T_i are positive, create a 'yes' node and halt; if all the examples in T_i are negative, create a 'no' node and halt; otherwise select another attribute in the same way as above.

2.2.3 An Example

Example 1. The decision tree generated by ID3 for the example set in Table 1 is shown in Figure 3.

As an exercise, the reader is advised to work out the tree by following the algorithm described in Section 2.2.2.

2.2.4 Advantages and Disadvantages

One of the great advantages of the ID3 method is the fact that it does not require users to specify background knowledge in the form of, say, generalization hierarchies. This means that ID3 can be applied to any syntactically well-formed training set. This, together with the high performance of the algorithm, has enabled ID3 to form the central component in several commercial packages.

However, ID3 also has some limitations:

First, its decision tree representation is less convenient for manipulations than the variable-valued logic (see Section 2.4) and production rules when a single decision tree is not sufficient to represent all the expertise of a domain. In this case, a number of decision trees of the domain need to be converted into decision rules before they can be used by a rule-based system. Although the conversion of decision trees to rules is not very difficult if we do not try to simplify the rules produced (see Section 2.3.5), the rules transformed from decision trees are still too simple to express things like memberships. Of course, for those domains where a decision tree is sufficient to express the expertise, we can use the decision tree directly by designing a non-rule-based problem solver.

Second, once an attribute is selected, all arcs labeled by values that attribute takes must be expanded. This can make resulting paths longer than those actually needed because, by the time specific concepts (leaves on the decision tree) are developed, irrelevant variables may have been introduced.

Third, the number of branches (paths) might still be large since at each arc, only one value can be labeled.

Finally, although the information theoretic heuristic can usually generate efficient decision trees, ID3 is still heuristic, which means it is not guaranteed to find the simplest decision tree that characterizes the given training instances because the information theoretic heuristic is by no means complete, and suffers from excessive complexity [Utgoff 89] and is therefore usually incomprehensible to experts since it needs to examine all candidate attributes to choose one at each non-leaf

Table 1: Cases of *Play* and *Don't Play* (adapted from [Quinlan 88b])

ORDER	*OUTLOOK*	*TEMPERATURE*	*HUMIDITY*	*WINDY*	DECISION
1	rain	hot	high	true	*Don't Play*
2	rain	cool	normal	true	*Don't Play*
3	overcast	mild	high	true	*Play*
4	overcast	mild	normal	false	*Play*
5	rain	hot	high	false	*Play*
6	overcast	cool	normal	true	*Play*
7	sunny	hot	normal	true	*Don't Play*
8	sunny	mild	high	true	*Don't Play*
9	sunny	mild	normal	false	*Play*
10	rain	cool	normal	false	*Play*
11	rain	hot	high	false	*Play*
12	sunny	hot	high	false	*Don't Play*
13	sunny	cool	normal	false	*Don't Play*
14	rain	mild	normal	true	*Don't Play*

node.

2.3 Recent Development of ID3

The original ID3 algorithm has been extended in several ways to improve its various capacities such as noise handling and incremental induction in its successors such as ID5R and C4. This section gives an account of these developments.

2.3.1 Noise Handling

ID3 can be easily adapted to handle noise by virtue of its top-down approach to decision tree construction. During induction, all possible attribute tests are considered when growing a leaf in a decision tree and the entropy measure is used to choose one to place at each node. In noisy environments, we can halt tree growth when no more significant information gain can be found. ID3's capacity to handle noisy data has been studied in [Quinlan 86] and [Quinlan *et al.* 87]. Noise handling in decision tree method based induction algorithms has been studied independently as a statistical technique [Breiman *et al.* 84] and shows a convergence between machine learning research in AI and statistics [Gams *et al.* 91].

However, the results produced by noise-tolerant algorithms are usually not completely consistent with the given training examples, which means those algorithms cannot guarantee to generate exact rules or decision trees in noise-free problems. This is not acceptable in cases where we need consistent rules to correctly classify all known examples.

2.3.2 Incremental Learning

There are several common problems in all kinds of inductive learning algorithms:

1. First, when an example set is very large, how can they speed up their learning processes?
2. Second, when an example set is not a static repository of data, e.g. examples may be added, deleted, or changed, the induction on the example set cannot be a one-time process, so how can induction algorithms deal with the changing examples?
3. Finally, when some inconsistency (e.g. noise) is found in an example set or a knowledge base just produced, how can they remove it?

One possible way to solve those problems is incremental learning, which means dividing a large example set into a number of subsets and treating each subset each time. Although no existing algorithm has found a complete solution to those problems, a lot of work has been done in this direction. For instance, ID4 [Schlimmer *et al.* 86], ID5R [Utgoff 89] and the windowing technique in ID3 can be viewed as good examples of research on incremental learning.

Generally speaking, incremental induction usually takes more time because it needs to restructure decision trees or rules when some new examples do not fit the decision trees or rules developed

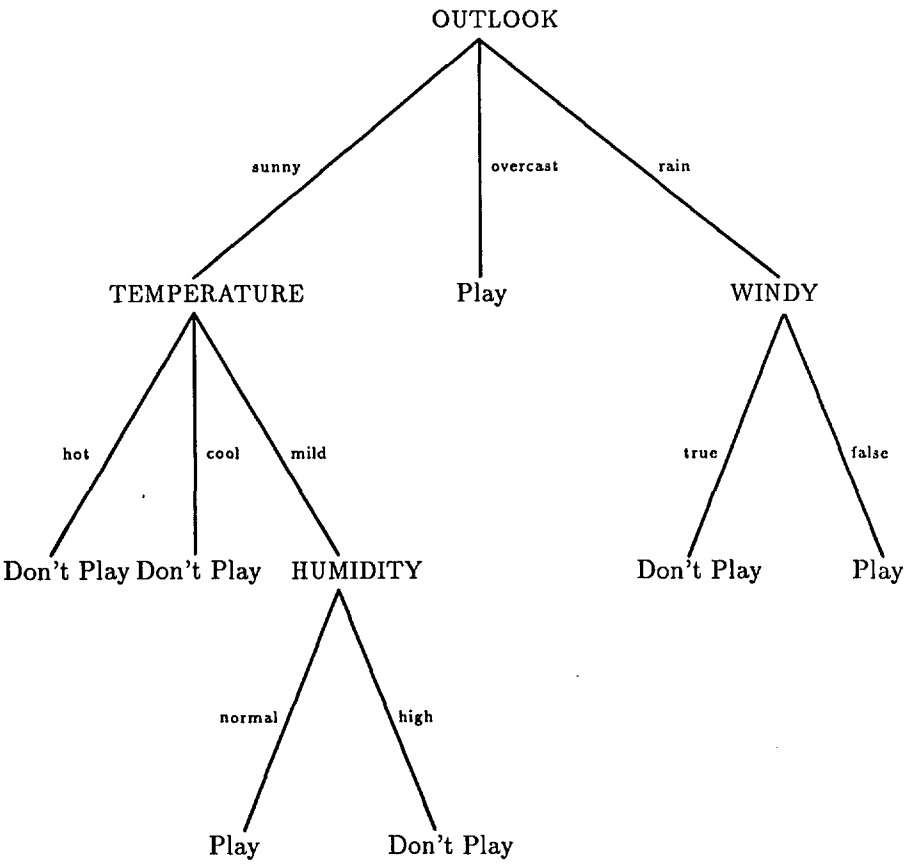


Figure 3: A Decision Tree (by ID3) for Table 1

so far. This is a common trade-off between time and space in Computer Science.

2.3.3 Constructive Learning

None of ID3-like algorithms need explicit, built-in background knowledge. That is why they are sometimes called empirical learning methods, which are different in nature from the knowledge-rich learning methods, such as AM [Lenat 79] and EURISKO [Lenat 83] developed by Lenat, learning by analogy (or case-based reasoning), explanation-based learning and inductive logic programming (see Section 4).

However, there is always implicit background knowledge embedded in the formulation of solution spaces and in the representation of examples. When a solution space turns out to be inadequate, which we often call the imperfect-knowledge problem, representation modification is needed and the modification process typically involves searching for useful new descriptive features (constructive induction) in terms of existing features or attributes.

Constructive learning has become a strong theme in inductive learning research. One of the difficulties in constructive learning is that the complexity in some cases (such as iterative feature construction) is extreme but there are situations in which it is a necessary part of learning.

2.3.4 Decision Tree Pruning

The basic ID3 algorithm tends to construct exact decision trees. However, in many real-world problems such as medical diagnosis and image recognition, the classification cannot be exact due to noise and/or uncertainty in data. As a result, a constructed tree by ID3 may not be able to capture the proper relations in data. Decision tree pruning mechanisms have been designed in many systems such as ASSISTANT [Cestnik *et al.* 87] and C4 [Quinlan *et al.* 87] to prevent this phenomenon. Once a non-leaf subtree meets a specific criterion (e.g. with an equal or smaller number of misclassifications), it is replaced by a leaf.

Pruning can usually simplify decision trees. The simplified trees can sometimes classify more accurately unseen cases in noisy environments. However, pruning decision trees is something very similar to noise handling mentioned in Section 2.3.1. It can simplify decision trees in noisy environments, but not improve the induction algorithms used to construct the trees. Also, we do not expect it to work properly in noise-free environments.

2.3.5 Decompiling Decision Trees into Production Rules

Decompiling decision trees has been implemented in C4 [Quinlan *et al.* 87] and C4.5 [Quinlan 92]. It contains three basic steps:

1. Traverse a decision tree to obtain a number of conjunctive rules. Each path from the root to a leaf in the tree corresponds to a conjunctive rule with the leaf as its conclusion.
2. Check each condition in each conjunctive rule to see if it can be dropped without more misclassification than expected on the original training examples or new test examples.
3. If some conjunctive rules are the same after Step 2, then keep only one of them.

Transformation of decision trees to production rules provides a way of combining different trees into the same knowledge base for more complicated domain. The final production rules produced are sometimes both simpler than the original decision trees and more accurate when classifying new examples in noisy environments.

However, dropping conditions from the decision-tree-traversal rules in Step 2 is something like a new induction algorithm which can work on the original example sets but in a way totally different from the ID3-like algorithms. Therefore, the time complexity for the transformation is expensive. We can also easily find some example sets where no conditions can be dropped from the decision-tree-traversal rules. In those cases, Step 2 is redundant.

2.3.6 Binarization of Decision Trees

Binarization in CART [Breiman *et al.* 84], ASSISTANT [Cestnik *et al.* 87] and NewID [Boswell 90] groups the attribute values into two subsets. It can usually produce smaller decision trees.

However, as indicated in [Quinlan 88b], there are two major problems in those systems. Firstly, binarization could lead to decision trees that are even more unintelligible to human experts than the ordinary case due to unrelated attribute values being grouped together and multiple tests on the same attributes in the binary decision trees. Secondly, binarization requires a large increase in computation to properly split the attribute values.

2.3.7 A New Selection Criterion for Decision Tree Construction

As ID3 has been found to operate unsatisfactorily when there are attributes with varying numbers of discrete possible values, [Quinlan 88b] proposes a new heuristic, called *the gain ratio criterion*, instead of the entropy measure, G(X) (see Section 2.2), adopted in ID3 for selecting tests in decision tree generation. In the gain ratio criterion, G(X)/IV(X) is used to replace G(X) where

$$IV(X) = \sum_{i=1}^{N} \frac{p_i + n_i}{p+n} log_2(\frac{p_i + n_i}{p+n}) \qquad (4)$$

and N, p, n are as mentioned in Section 2.2.2. When $IV(X)$ is not zero, [Quinlan 88b] suggests that "from among those attributes with an average-or-better gain, select the attribute that maximizes the above ration".

Here again, in some cases as shown in [Quinlan 88b], the new gain ratio criterion can outperform the entropy measure. However, in many other cases, even when there are attributes with varying numbers of discrete possible values, the new criterion cannot improve the decision trees produced by ID3 at all.

2.3.8 Structured Induction

ID3's simplicity is largely attributable to the following two restrictions placed on its application domains [Quinlan 88a]:

- The induction is a kind of classification, i.e., the knowledge we are trying to capture is that of assigning a case to one of a set of mutually exclusive classes.

– Each case is described in terms of a fixed collection of properties or attributes. An attribute may have a small set of discrete possible values or might be a real-valued numerical variable.

These limitations of ID3 exist also in the extension matrix approach based family of algorithms in Section 2.4 for complex applications.

Although induction offers a considerable short cut in comparison to those methods of rule generation such as explanation-based learning and inductive logic programming which couple both deduction and induction, decision tree method based algorithms provide large decision trees that are opaque to the user in large problem domains. Shapiro [1987] has developed the technique of structured induction. The basic idea is to split the whole complex problem which might be very large in size into a number of subproblems by using domain knowledge and apply the ID3 algorithm to each of the subproblems.

The structured induction technique can also be coupled with the extension matrix approach based family of algorithms in Section 2.4.

2.3.9 Conclusions

In addition to the development mentioned above, other features such as handling real-valued attributes have also been studied. However, as we have analyzed above, although each of them is useful in some specific cases, none of the extensions have generally improved ID3 in noiseless environments. The core of the decision tree method based family of algorithms is still the entropy measure to select an attribute by examining all candidate attributes during the splitting of examples.

2.4 Variable-Valued Logic and HCV

2.4.1 The Extension Matrix Approach

The new family of inductive algorithms based on the extension matrix approach was first developed in the University of Illinois by Hong *et al.* [Hong 85, Hong *et al.* 87] and then redesigned by the author [Wu 93]. In contrast to the decision tree method in ID3-like algorithms, the algorithms of the extension matrix approach based family take a new kind of matrix, called *an extension matrix*, as their mathematical basis.

Terminology and notation

Let a be the number of attributes $\{X_1, ..., X_a\}$ in an example space, n be $|NE|= | \{e_1^-, ..., e_n^-\} |$ where e_i^- $(i = 1, ..., n)$ is the i-th negative example and p be $|PE|=| \{e_1^+, ..., e_p^+\} |$ where e_i^+ $(i = 1, ..., p)$ is the i-th positive example. Let NE be expressed by

$$NEM = \{e_1^-, ..., e_n^-\}^T = (r_{ij})_{n*a} \qquad (5)$$

with the i-th negative example $e_i^-(i = 1, ..., n)$ being expressed on the i-th row of matrix NEM and $NEM(i, j) = r_{ij}$ indicating the value of e_i^- on attribute X_j is r_{ij}.

Definition 2.1. Let the k-th $(k = 1, ..., p)$ positive example be expressed as $e_k^+ = (v_{1_k}^+, ..., v_{a_k}^+)$, the matrix below is the *extension matrix* of e_k^+ against NE

$$EM_k = (r_{ij_k})_{n*a} \qquad (6)$$

where

$$r_{ij_k} = \begin{cases} * & when\ v_{j_k}^+ = NEM_{ij} \\ NEM_{ij} & when\ v_{j_k}^+ \neq NEM_{ij} \end{cases}$$

and '$*$' denotes a dead element which cannot be used to distinguish the positive example from negative examples.

Definition 2.2. In an EM_k, a set of n non-dead elements r_{ij_i} $(i = 1, ..., n,\ j_i \in \{1, ..., a\})$ that come from the n different i rows is called a *path* in the extension matrix.

A path $\{r_{1j_1}, ..., r_{nj_n}\}$ in an EM_k corresponds to a conjunctive formula

$$L = \bigwedge_{i=1}^{n} [X_{j_i} \neq r_{ij_i}] \qquad (7)$$

which covers e_k^+ against NE and *vice versa.*

Each $[X_{j_i} \neq r_{ij_i}]$ here is a *selector* in variable-valued logic [3]. If r_{ij_i} appears on m $(m \in$

[3]The variable-valued logic developed by Michalski [Michalski 75] is a calculus for representing decision problems where decision variables can take on some range of values. Its principal syntactic entity is a *selector* with the general form

$$[X \# R] \qquad (8)$$

where X is a variable or attribute, $\#$ is a relational operator (such as $=, \neq, <, >, \leq$, and $\geq$), and R, called a *refer-*

$\{0, ..., n\}$) rows in the same column j_i of an EM_k, we say it or $[X_{j_i} \neq r_{ij_i}]$ covers m rows of the EM_k.

Definition 2.3. Matrix $EMD = (r_{ij})_{n*a}$ with

$$r_{ij} = \begin{cases} * & \text{if } \exists k_1 \in \{i_1, ..., i_k\} : EM_{k_1}(i,j) = * \\ \vee_{k_2=1}^{k} EM_{i_{k_2}}(i,j) = NEM(i,j) & \text{otherwise} \end{cases} \quad (9)$$

is called the *disjunction matrix* of the positive example set $\{e_{i_1}^+, ..., e_{i_k}^+\}$ against NE or the disjunction matrix of $EM_{i_1}, ..., EM_{i_k}$.

Definition 2.4. In the EMD of a positive example set $\{e_{i_1}^+, ..., e_{i_k}^+\}$ against NE, a set of n nondead elements r_{ij_i} $(i = 1, ..., n, j_i \in \{1, ..., a\})$ that come from the n different i rows is also called a *path.*

A path $\{r_{1j_1}, ..., r_{nj_n}\}$ in the EMD of $\{e_{i_1}^+, ..., e_{i_k}^+\}$ against NE corresponds to a conjunctive *formula* or *cover*

$$L = \bigwedge_{i=1}^{n} [X_{j_i} \neq r_{ij_i}] \quad (10)$$

which covers all of $\{e_{i_1}^+, ..., e_{i_k}^+\}$ against NE and *vice versa.*

If there is no path which covers all the n rows in EMD, there is no common path and therefore no conjunctive formula cover in all the extension matrixes $EM_{i_1}, ..., EM_{i_k}$.

Definition 2.5. If there exists at least one path in the EMD of a positive example set $\{e_{i_1}^+, ..., e_{i_k}^+\}$ against NE, all the positive examples in the set intersect and the positive example set is called an *intersecting group.*

For a given set of examples, if PE and NE are persistent, which means they contain no common examples, there always exists at least one conjunctive formula cover for each intersecting example group against NE.

Optimization problems

There are two striking optimization problems in the extension matrix approach:

- The minimum formula (MFL) problem: Generating a conjunctive formula that covers a positive example or an intersecting group of positive examples against NE and has the minimum number of different conjunctive selectors.
- The minimum cover (MCV) problem: Seeking a cover which covers all positive examples in PE against NE and has the minimum number of conjunctive formulae with each conjunctive formula being as short as possible.

Since the extension matrix EM_k of each positive example e_k^+ against NE contains all such paths that each correspond to a conjunctive formula of e_k^+ against NE and an optimal cover of PE against NE is such a minimum set of formulae that is a logical combination of all the formulae from every EM_k $(k = 1, ..., p)$, both MFL and MCV problems have been proved to be NP-hard [Hong 85].

Heuristic strategies in AE1 [Hong 85]

As the nature of the MFL and MCV problems is NP-hard, when an example set or an attribute space is large the induction process based on the complete algorithms will become computationally intractable. Two strategies are adopted in AE1 to find approximate rather than optimal solutions for both MFL and MCV problems [Hong 85]:

1. Starting search from the columns with the most nondead elements, and
2. Simplifying redundance by deductive inference rules in mathematical logic.

There are two problems in AE1. First, its first strategy can easily lose optimal solutions in some cases. Taking the simple extension matrix below as an example, the first heuristic strategy in AE1 cannot produce the optimal formula ($[X_1 \neq 1] \wedge [X_3 \neq 1]$) since it will choose the selector $[X_2 \neq 0]$ at first. Second, simplifying redundance for MFL and MCV problems is NP-hard. No heuristic strategy for this process has been reported.

$$\begin{pmatrix} 1 & * & * \\ * & 0 & 1 \\ 1 & 0 & * \\ * & 0 & 1 \\ 1 & 0 & * \\ * & * & 1 \end{pmatrix}$$

ence, is a list of one or more values that X could take on. A well-formed rule in the logic is similar to a production rule but with selectors as the basic components of both its left-hand and right-hand sides.

Advantages and disadvantages

Time complexity and description compactness [4] are two important criteria for all induction algorithms. In the extension matrix approach, there are two extreme strategies, each of which places special emphasis on only one of the two criteria. The first is finding all possible formulae from each positive example's extension matrix first and then taking an exhaustive search among all the formulae to find the shortest combination which covers all the positive examples. This strategy can give the shortest description in the form of variable-valued logic but works in exponential time. The second is simply separating one positive example from NE by "memorizing" the positive example or all positive examples in PE from NE by "memorizing" each of the positive examples into a Boolean OR formula. This trivial heuristic can work quickly but generates an extremely large description. An OR formula of this kind cannot be used directly to classify new examples which have not been presented in the training example set while simplifying it into the shortest form also needs NP-hard time. Therefore, a good learning algorithm should be able to either avoid the NP-hard time or produce a briefer description which is at least able to correctly classify the PE and NE in a given training example set. We will show below that the HCV algorithm has made progress on both the time and the description compactness.

2.4.2 The HFL Algorithm

The HFL algorithm is designed to find a heuristic conjunctive formula which corresponds to a path in an extension matrix or a disjunction matrix when there is at least one path in the disjunction matrix. As a disjunction matrix can be processed in the same way as an extension matrix to find its conjunctive formulae, we will only refer to the extension matrixes below.

Four strategies in HFL

Four strategies are adopted in the HFL algorithm:

The *fast* strategy

In an extension matrix $EM_k = (r_{ij})_{n*a}$, if there is no dead element in a (say j) column, then $[X_j \neq r_j]$ where $r_j = \vee_{i=1}^{a} r_{ij}$ is chosen as the one selector cover for EM_k.

For example, selector $[X_5 \neq \{normal, dry - peep\}]$ below can cover all the five rows in the extension matrix.

$$\begin{pmatrix} absent & slight & strip & * & normal \\ * & * & hole & fast & dry-peep \\ low & slight & strip & * & normal \\ absent & slight & spot & fast & dry-peep \\ low & medium & * & fast & normal \end{pmatrix}$$

The *precedence* strategy

When a r_{ij} in column j is the only nondead element of a row i in an extension matrix $EM_k = (r_{ij})_{n*a}$, the selector $[X_j \neq r_j]$ where $r_j = \vee_{i=1}^{a} r_{ij}$ is called an inevitable selector and thus is chosen with top precedence.

For example, $[X_1 \neq 1]$ and $[X_3 \neq 1]$ are two inevitable selectors in the extension matrix below which we have mentioned in Section 2.4.1.

$$\begin{pmatrix} 1 & * & * \\ * & 0 & 1 \\ 1 & 0 & * \\ * & 0 & 1 \\ 1 & 0 & * \\ * & * & 1 \end{pmatrix}$$

The *elimination* strategy

When each appearance of some nondead element in the j_1-th column of some row is always coupled with another nondead element in the j_2-th column of the same row in an extension matrix $EM_k = (r_{ij})_{n*a}$, $[X_{j_1} \neq r_{j_1}]$ where $r_{j_1} = \vee_{i=1}^{a} r_{ij_1}$ is called an eliminable selector and thus eliminated by selector $[X_{j_2} \neq r_{j_2}]$ where $r_{j_2} = \vee_{i=1}^{a} r_{ij_2}$.

For example, attribute X_2 can be eliminated by attribute X_3 below.

$$\begin{pmatrix} 1 & * & * \\ * & 0 & 1 \\ 1 & 0 & 1 \\ * & 0 & 1 \\ 1 & 0 & 1 \\ * & * & 1 \end{pmatrix}$$

The *least-frequency* strategy

[4] The measures for description compactness adopted in this document are (1) the number of conjunctive formulae or rules, and (2) the number of all conjuncts or selectors in all the conjunctive rules.

When all inevitable selectors have been chosen and all eliminable selectors have been excluded but all the selectors chosen have not yet covered all the rows in an extension matrix, exclude a least-frequency selector which has least nondead elements in its corresponding column in the extension matrix.

For example, attribute X_1 in the following extension matrix can be eliminated and there still exists a path.

$$\begin{pmatrix} 1 & * & 1 \\ * & 0 & 1 \\ 1 & 0 & * \\ * & 0 & 1 \\ 1 & 0 & * \\ * & 0 & 1 \end{pmatrix}$$

All of the *fast, precedence* and *elimination* strategies are complete, which means if there exists one or more shortest conjunctive formulae in an extension matrix, they will not lose it.

Although the column with least nondead elements is not necessarily removed from all the optimal paths, the removal looks reasonable as choosing a column with fewer nondead elements means more columns thus more selectors may be involved in connecting a path. So the fourth strategy is a sensible heuristic. However, it is still heuristic. Firstly, this strategy is sensitive to the order of attributes in given examples. When we have two attributes with the same least nondead elements at some stage, different implementations of this strategy could produce different results. However, the order of attributes also matters in the ID3 algorithm. Secondly, removing the least-frequency selector could also lose optimal paths. These two problems also apply in a similar way to the first strategy of AE1 described in Section 2.4.1. However, as compared to AE1, we have adopted three complete strategies in HFL.

In the case of the extension matrix EM_k of a positive example $e_k^+ = (v_{1_k}^+, ..., v_{a_k}^+)$ against NE, a selector $[X_j \neq r_j]$ is equivalent to $[X_j = v_{j_k}^+]$ with existing examples in a given example set. Meanwhile, in the case of the disjunction matrix EMD of an intersecting group of positive examples $e_{i_1}^+, ..., e_{i_k}^+$ against NE, a selector $[X_j \neq r_j]$ is equivalent to $[X_j \in \vee_{k_2=1}^{k} v_{j_{i_{k_2}}}^+]$ in the context of existing examples. When X_j is a numerical attribute, $\vee_{k_2=1}^{k} v_{j_{i_{k_2}}}^+$ can be further grouped into a number of intervals none of which will contain any NEM_{ij} $(i = 1, ..., n)$.

2.4.3 The HCV Algorithm

Algorithm description

The basic idea for the HCV algorithm is to partition PE of a specific class into p' $(p' \leq p)$ intersecting groups first; call the heuristic Algorithm HFL to find a Hfl for each intersecting group; then give the covering formula by logically ORing all the Hfl's finally.

The GEM algorithm is designed to generate the disjunction matrix EMD of $e_{i_1}^+, ..., e_{i_k}^+$ against NE from $EM_{i_1}, ..., EM_{i_k}$ according to Definition 2.3. When there exists a dead element on the (i, j)-position of any of $EM_{i_1}, ..., EM_{i_k}$, $EMD(i, j) = *$. Otherwise, $EMD(i, j) = NEM(i, j) = EM_{i_{k_2}}$ $(k_2 \in \{1, ..., k\})$.

```
Procedure GEM({EM_i1, ..., EM_ik}; EMD)
  integer n, a, k
  matrix EM_i1(n,a), ..., EM_ik(n,a),
       EMD(n,a)
  for j1=1 to n do
    for j2=1 to a do
     if ∃k2∈{1, ..., k}:  EM_ik2(j1, j2)=*
     then EMD(j1, j2)←*
     else EMD(j1, j2)←EM_i1(j1, j2)
    next j2
  next j1
Return(EMD)
```

Algorithm IDEN below is designed to test whether there is a path in a disjunction matrix EMD with the result being returned by logical variable *Flag*. It tests each row of EMD to ascertain whether there is at least one nondead element on the row. If each row has at least one nondead element, then there exists at least one path in EMD and thus *Flag* is assigned to 'T'.

```
Procedure IDEN(EMD; Flag)
  integer n, a
  matrix EMD(n, a)
  logical Flag
  i←1, Flag2←'F'
  while i≤n & Flag2='F' do
    { j←1, Flag3←'F'
     while j≤a & Flag3='F' do
      if EMD(i, j)≠ * then Flag3←'T'
       else j←j+1
```

```
    if Flag3←'F' then Flag2←'T'
     else i←i+1
   }
  if Flag2='F' then Flag←'T'
    else Flag←'F'
Return(Flag)
```

Based on the GEM and IDEN algorithms above and the HFL algorithm outline in Section 2.4.2, the HCV algorithm is designed as follows where GEM and IDEN are used to partition PE into intersecting groups and HFL is used to find a conjunctive formula for each intersecting group.

```
Procedure HCV(EM1, ..., EMp; Hcv)
  integer n, a, p
  matrix EM1(n,a), ..., EMp(n,a), D(p)
  set Hcv
  S1:  D←0
  /* D(j)=1 (j=1, ..., p) indicates that EMj
  has been put into an intersecting group. */
     Hcv←φ /* initialisation */
  S2:  for i=1 to p do
     if D(i)=0 then
      { EM←EMi
       for j=i+1 to p do
        if D(j)=0 then
         { call GEM({EM, EMj}; EM2)
          call IDEN(EM2; Flag)
          if Flag='T' then
           { EM←EM2, D(j)←1 }
         }
        next j
        call HFL(EM; Hfl)
        Hcv ←HcvVHfl
      }
      next i
Return(Hcv)
```

The time complexity for Algorithm HCV is $O(pna^3 + p^2na)$ [Wu 93].

Algorithm HCV is a bidirectional algorithm. It first groups the positive example set in a top–down way and then calls algorithm HFL, which works in a bottom–up way. Its time is low-order polynomial as opposed to exponential in the first strategy mentioned in the "advantages and disadvantages" subsection of Section 2.4.1. Both Algorithm HFL and Algorithm HCV can always produce shorter formulae than the trivial strategy (the second strategy) also mentioned in the "advantages and disadvantages" subsection of Section 2.4.1, so long as the shorter formulae exist. If there exists at least one conjunctive cover in a given training example set, the formula produced by HCV must be a conjunctive one.

However, the intersecting groups partitioned by HCV and therefore the results returned by the HFL algorithm on each intersecting group or partition are sensitive to the order of examples in a given example set. For a given partition, the order of positive examples does not affect their disjunction matrix and therefore does not change the result of HFL.

An example run of HCV

Example 2. Considering PE (of *Play*) and NE (of *Don't Play*) in Table 1 (in Section 2.2), let us observe the results generated by the HCV algorithm.

For the given example set, $NE = \{e_1^-, e_2^-, e_7^-, e_8^-, e_{12}^-, e_{13}^-, e_{14}^-\}$, $PE = \{e_3^+, e_4^+, e_5^+, e_6^+, e_9^+, e_{10}^+, e_{11}^+\}$ and

$$NEM = \begin{pmatrix} rain & hot & high & true \\ rain & cool & normal & true \\ sunny & hot & normal & true \\ sunny & mild & high & true \\ sunny & hot & high & false \\ sunny & cool & normal & false \\ rain & mild & normal & true \end{pmatrix}.$$

The first intersecting group found in Step S2 by starting with the first positive example (e_3^+) and calling the GEM and IDEN algorithms is $\{e_3^+, e_4^+, e_6^+\}$ and the disjunction matrix EMD_1 against NE is

$$EMD_1 = \begin{pmatrix} rain & hot & * & * \\ rain & * & * & * \\ sunny & hot & * & * \\ sunny & * & * & * \\ sunny & hot & * & * \\ sunny & * & * & * \\ rain & * & * & * \end{pmatrix}.$$

Calling HFL, $[OUTLOOK \neq \{rain, sunny\}]$ which is equivalent to $[OUTLOOK = overcast]$ is chosen by the *fast* strategy and the first Hfl is thus

$$[OUTLOOK = overcast].$$

The second intersecting group found in Step S2 by starting with the third positive example

(e_5^+) and calling the GEM and IDEN algorithms is $\{e_5^+, e_{10}^+, e_{11}^+\}$ and the disjunction matrix EMD_2 is

$$EMD_2 = \begin{pmatrix} * & * & * & true \\ * & * & * & true \\ sunny & * & * & true \\ sunny & mild & * & true \\ sunny & * & * & * \\ sunny & * & * & * \\ * & mild & * & true \end{pmatrix}.$$

Running HFL, $[WINDY \neq true]$ and $[OUTLOOK \neq sunny]$ which are equivalent to $[WINDY = false]$ and $[OUTLOOK = rain]$ respectively are both chosen as inevitable selectors and they cover all of the five rows in EMD_2. Therefore, the second Hfl is

$$[WINDY = false] \wedge [OUTLOOK = rain].$$

The third intersecting group is $\{e_9^+\}$ and the disjunction matrix EMD_3 is

$$EMD_3 = \begin{pmatrix} rain & hot & high & true \\ rain & cool & * & true \\ * & hot & * & true \\ * & * & high & true \\ * & hot & high & * \\ * & cool & * & * \\ rain & * & * & true \end{pmatrix}.$$

Running HFL, $[TEMPERATURE \neq \{hot, cool\}]$ is first chosen as an inevitable selector and it covers rows 1, 2, 3, 5 and 6, attributes $OUTLOOK$ and $HUMIDITY$ are then excluded by attribute $WINDY$ and $[WINDY \neq true]$ is finally chosen as an inevitable selector on the fourth row after $OUTLOOK$ and $HUMIDITY$ have been crossed out. The equivalent Hfl for this intersecting group is

$$[TEMPERATURE = mild] \wedge [WINDY = false].$$

Therefore,

$$Hcv = [OUTLOOK = overcast] \vee$$
$$[WINDY = false] \wedge [OUTLOOK = rain] \vee$$
$$[TEMPERATURE = mild] \wedge [WINDY = false]$$

whose equivalent rule in variable-valued logic is:

[OUTLOOK=overcast]
∨
[WINDY=false]
[OUTLOOK=rain]
∨
[TEMPERATURE=mild]
[WINDY=false]
→
[DECISION=Play].

Meanwhile, the decision tree generated by ID3 and its equivalent rules are given in Section 2.2.

2.5 A Comparison with ID3 and HCV

One difference between HCV and ID3 is that the HCV algorithm only produces rules for positive examples while ID3 generates decision trees to classify both positive and negative examples. However, this is not an advantage of ID3 over HCV. For instance, if all the examples in an example set are people from different countries in the world, when we are told that some of them are British and the task is to find characteristics of British, we will only be interested in the description produced for British because all other examples which cannot be satisfied by the description will automatically belong to other countries. Although the ID3 algorithm will automatically produce a description for negative examples at the same time as it produces the description for positive examples, we do not think that is useful in many cases. A description for negative examples belonging to all other countries except Britain will not help anything because (1) it can be inferred from the description for positive examples, and (2) if we want to know which specific country a negative example belongs to, we need to run ID3 once again. The HCV (Version 1.0) program [Wu 92] implemented by the author has already been able to produce rules to classify more than two classes of examples. The entropy measure in ID3 can also be easily extended to chunk examples into more than two classes.

The following is a comparison between ID3 and HCV.

The reason for using decision trees rather than rules, such as the variable-valued logic rules adopted HCV, is said by [Jackson 90] to be that the ID3-like algorithms are comparatively simpler than other learning algorithms. From the fourth

disadvantage of ID3 (see Section 2.2) and the time complexity of HCV, we can say that the argument is now no longer convincing. Although the information theoretic heuristic is by no means complete, ID3 needs to examine all possible candidate attributes and their values to choose one attribute at each non-leaf node of its decision trees and thus its time complexity is still expensive [Utgoff 89]. In HCV, although all of the *fast, precedence* and *elimination* strategies are complete, which means if there exists one or more shortest conjunctive formulae in an extension matrix they will not lose it, the *fast* strategy can choose an optimal attribute as soon as it finds the attribute without any attention to other attributes and the *precedence* strategy can choose an inevitable attribute by examining only the values of one row in an extension matrix. High efficiency has been seen as an important requirement for machine learning algorithms and exponential or even medium-order polynomial complexity will not be of practical use in realistic applications. We have not provided the comparison of HCV and ID3 on time performance because there are still different results for ID3's time complexity. Therefore, we can not say in general that HCV outperforms ID3 in time. However, HCV's time complexity has been shown to be low-order polynomial and therefore computationally acceptable. The first significant feature of the HCV algorithm is that it supports a reasonable solution to the NP-hard problem in the extension matrix approach for inductive learning.

Contrasting to the second and third disadvantages of ID3, different values of the same attribute which take on only positive examples can be easily grouped into a selector in the variable-valued logic. In ID3, once an attribute is selected, all arcs labeled by values that attribute takes must be expanded. This can still make the number of branches (paths) large since at each arc only one value can be labeled, and resulting paths might be longer than those actually needed because, by the time specific concepts (leaves on the decision tree) are developed, irrelevant variables may have been introduced. All of the four strategies adopted in Algorithm HFL and the partitioning technique in HCV are designed to reduce the number of selectors. For those problems where the *fast, precedence* and *elimination* strategies are enough to produce their final formulae, we can guarantee that the formulae are optimal. If there exists at least one conjunctive cover in a given training example set for positive examples against negative examples, the formula produced by HCV must be a conjunctive one. However, the information theoretic heuristic in ID3 is not complete, which means it is not guaranteed to find the simplest decision tree that characterizes the given training instances. From various example sets the author has tested, the rules produced by HCV are all more compact [5] in terms of the numbers of conjunctive rules and conjunctions than the decision trees or their equivalent decision rules produced by ID3. So, the compactness of rules in HCV is its second feature. However, the *least-frequency* strategy and the partitioning of positive examples are still heuristic. We cannot guarantee the rules produced by HCV must be more compact than the decision trees generated by ID3 in all possible cases. There are still three kinds of possible results for a new example set: (1) HCV produces more compact rules as analyzed and shown above; (2) HCV and ID3 produce similar rules because ID3 can usually produce efficient decision trees, and (3) ID3 produces more compact rules than HCV when ID3 can produce the shortest decision tree while HCV cannot generate optimal rules. For instance, the order of cases in a given example set can effect the result of HCV but does not change the decision tree generated by ID3. Sometimes, we could possibly change the order of examples to make ID3 outperform HCV.

Also, all of the four strategies adopted in Algorithm HFL and the partitioning technique in HCV are more comprehensible than the information theoretic heuristic for most human experts who are not familiar with information theory.

However, there are also some disadvantages with HCV. Firstly, HCV as it stands has not yet provided efficient facilities to handle noise. We can adopt the TRUNC strategy developed in AQ15 [Michalski *et al.* 86] to help in this regard. Secondly, HCV takes longer time than the ID3-like algorithms to respond with some large example sets, although its theoretical time complexity is quite acceptable.

[5] This is still true when we only count the rules for positive examples.

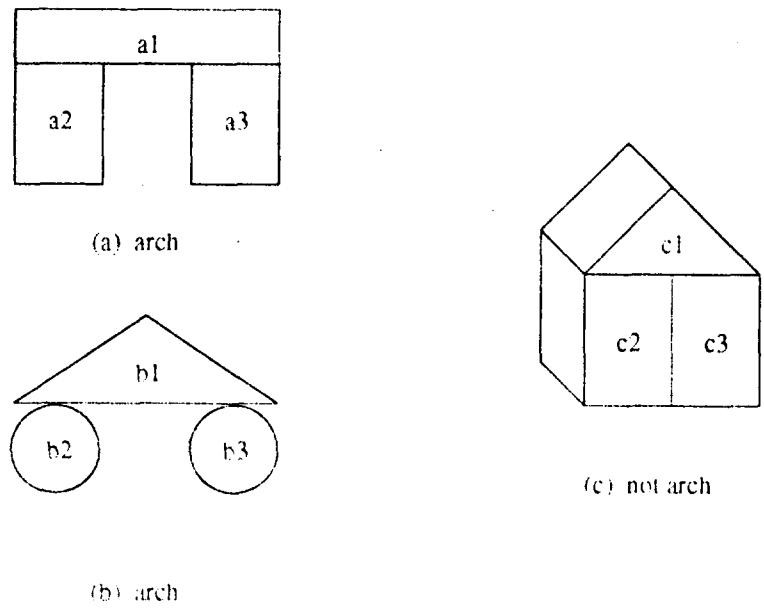


Figure 4: Training Examples of ARCH

3 Incremental Induction

3.1 Introduction

We have mentioned a little bit incremental induction in Section 2.3. However, that was something slightly different from what we are talking about in this section. Both ID3 and HCV process an example set as a whole and carry out induction as a one-time process. When example sets are too large, for example, each containing about a million examples, we have briefly outlined in Section 2.3 the need of incremental learning, which requires learning systems to use examples part by part. In this section, we are introducing two methods, ARCH and version spaces, which treat examples one by one by nature. Actually, the ID4 and ID5R algorithms mentioned in Section 2.3 are also of this type, but ARCH and the version space method in this section require background knowledge to form the generalization-specialization hierarchy. The hierarchy provides a basis, called *bias*, for choosing one generalization/specialization over another.

3.2 ARCH

Winston's work on concept learning [6] was performed on his ARCH program [Winston 70]. It has been suggested as a strong candidate to identify the precise point at which machine learning really came into being.

The task of the program is to learn concept descriptions by looking for relationships between semantic network representations of block world configurations. Two processes were particularly important in his formulation: (1) finding and exploiting commonalities among structural descriptions for the same type of configuration; and (2) finding significant differences between positive examples and negative examples. The two processes correspond to the present terminology *generalization* and *specialization*. His ARCH program effectively generalized the representation so as to cover all the positive examples and

[6] Concept learning finds the features of a concept by examining the examples of the concept. The results can be described in the form of "If <Features/Conditions> then <Concept>", and therefore concept learning is a kind of rule induction. When the examples are not classified with distinctive concepts, clustering them into groups is what we call *unsupervised learning*. Unsupervised learning will not be covered in this module due to time restrictions.

specialized it so as to exclude all the negative ones. Examples were presented incrementally, and a new positive example triggered generalization while a new negative example triggered specialization.

ARCH can be outlined as follows.

```
Step 1. Initialize the current concept
  description, C, to be the first
  positive example of the concept.
Step 2. For each of the remaining
  examples, E, do
  1. Match E with C to obtain the
     difference.
  2. If E is positive, then generalize
     C to cover the features of E in
     the difference. The generalization
     here can be carried out by either
     dropping a conjunction like X=a or
     replacing some value (e.g., wedge)
     by a more general term (e.g.,
     object).
  3. If E is negative, a condition from
     E in the difference is added to C.
```

From time to time, backtracking is needed in Step 2 to choose between the two types of strategies or even within each of them because there are probably more than one conjunction droppable and a specific value/term can possibly be replaced by several more general terms.

Figure 4 shows some possible training examples. Given enough positive and negative examples, the following is supposed to be inferred by ARCH:

```
        parts(X, [X1, X2, X3])
        on(X1, X2)
        on(X1, X3)
        not(touch(X2, X3)
   →    arch(X).
```

3.3 Version Spaces

In 1977, Mitchell described a method, called *"candidate elimination"*, which is similar to Winston's method in the sense that it is based on generalization and specialization but different in the way in which it explores the solution (or hypothesis) space. In Winston's ARCH program, hypotheses were generated and tested one by one, while in Mitchell's method, hypotheses are systematically deleted from a representation of the entire hypothesis space as they are found to be unsatisfactory. The novel feature of Mitchell's method was the way in which it allowed the hypothesis space to be efficiently represented as a version space.

Table 2: A Set of Poker Cards

Order	Card	Belongs to the Concept?
1	$A\spadesuit$	yes
2	$7\clubsuit$	yes
3	$8\heartsuit$	no
4	$9\clubsuit$	yes
5	$5\heartsuit$	no
6	$K\diamondsuit$	no
7	$6\diamondsuit$	no
8	$8\spadesuit$	no

The key feature of Mitchell's version space is the partial ordering of conjunctive concepts [7] according to their generality. For example, Figure 5 illustrates a general-to-specific ordering.

The aim of Mitchell's method is to ensure that, at all times, the version space contains the complete set of satisfactory representations. A simplified description of the candidate elimination algorithm is as follows. A generalized notion, called *the description identification*, of the version space method is described in [Mellish 91].

```
Procedure Candidate Elimination
 S1: Initialize the version space.
 S2: Set G to be the set of most general
     representations.
 S3: Set S to be the set of most specific
     representations.
 S4: For each new training example do
  S41: If it is positive then
     (1) remove from G all
     representations that do not cover
     this example,
     (2) update S so as to ensure that
     it still contains the set of
     maximally specific, common genera-
     lizations of the example and the
     previous representations in S.
  S42: If it is negative then
     (1) remove from S all
     representations that cover this
     negative example,
```

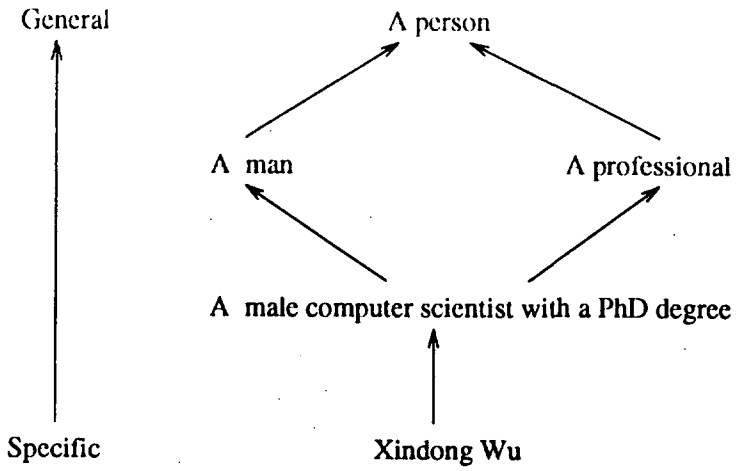


Figure 5: A General-to-Specific Ordering

Table 3: An Example Run of the Version Space Method on Table 2

Current card	G	S
	Any cards	{}
$A\spadesuit$	Any cards	$\{A\spadesuit\}$
$7\clubsuit$	Any cards	odd black cards
$8\heartsuit$	Black cards *or* odd cards	odd black cards
$9\clubsuit$	Black cards *or* odd cards	odd black cards
$5\heartsuit$	Black cards	odd black cards
$K\diamondsuit$	Black cards	odd black cards
$6\diamondsuit$	Black cards	odd black cards
$8\spadesuit$	odd black cards	odd black cards

```
    (2) update G so as to ensure that
    it contains the set of maximally
    general, common specializations of
    the example and the previous
    representations in G.
  S43: If G=S then exit.
Return.
```

Let us go through this algorithm below with a pack of cards in Table 2 (adapted from [Ginsberg 93]).

There are 8 cards, 3 positive and 5 negative to the target concept to be learned. Representations used here are things like "spade", "club", "heart", "diamond", "red" (heart or diamond), "black" (spade or club), "even card", "odd card", and so on. The example run is recorded in Table 3.

Clearly, black cards and odd cards are both more general than odd black cards, and an odd black card is more general than $A\spadesuit$ and $7\clubsuit$ in the domain. The final description for the concept is *odd black cards.*

4 Explanation-Based Learning

4.1 The Paradigm

The basic idea of explanation based learning is to use results from one specific problem solving effort to help you the next time around. The information provided for the learning engine is a target concept (in the form of a Prolog-like rule, *Head :- Body.*), a set of examples (positive only) and a domain theory expressed as a set of Prolog rules and facts. Some of the predicates used in the rules and facts in the domain theory are identified as *utilities* (or *operational* predicates). Explanation based learning processes an example each time with the domain theory, and then keeps the result of the processing in terms of the given utilities. The result can be taken as a lemma to the domain, which might be quite useful to subsequent problem solving.

Processing an example is a 2-stage operation. Firstly, we instantiate the head of the rule rep-

[7]Concepts, with or without specific names, described in terms of a set of conjunctions.

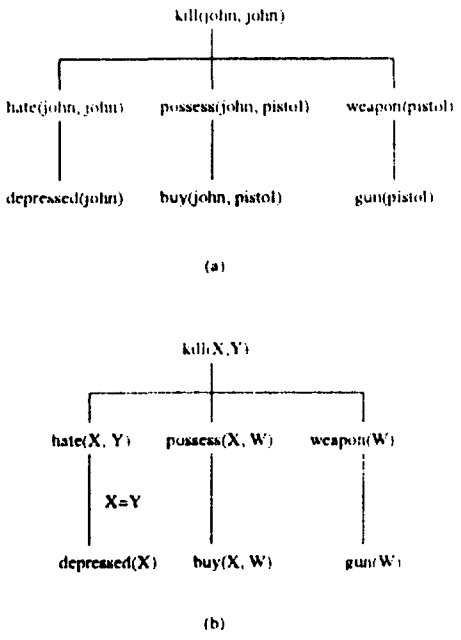


Figure 6: An Example of Explanation-Based Learning

resenting the target concept with the example by using the given utilities and produce a proof tree. Secondly, we regress the tree to form another rule of the target concept in terms of the given utilities. This rule forms another implementation of the target concept. Actually, the instantiation process is something more like logic deduction than induction, and the regression process is a kind of partial evaluation [Van Harmelen & Bundy 88].

4.2 An Example Run

Consider the following domain theory, expressed as Prolog clauses:

```
hate(X,X) :- depressed(X).
possess(X,Y) :- buy(X,Y).
weapon(X) :- gun(X).
drunk(john).
depressed(john).
buy(john, pistol).
gun(pistol).
```

with `drunk`, `depressed`, `buy` and `gun` as utilities, and the target concept as follows:

```
kill(X,Y) :- hate(X,Y), possess(X,W),
     weapon(W).
```

What follows `kill(john, john)`?

To answer this question, we first draw a complete proof tree (Figure 6 (a)) for `kill(john, john)`, and then generalize the constants with variables on the tree for the target concept `kill(X,Y)`, terminating at the given utilities (Figure 6 (b)).

The result is:

```
kill(X, X) :- depressed(X),
     buy(X,A), gun(A).
```

Note that the result is less general than the original concept `kill(X,Y)` and `drunk(john)` is not used.

4.3 Discussion

Explanation based learning is a knowledge-intensive learning paradigm which is sometimes capable of improving a system's performance impressively by using these newly-learned rules. However, this depends very much on the domain theory which is pre-programmed in. If the domain theory is not prefect, the learning process may not succeed in producing a useful result. The two classic papers on this topic are [DeJong *et al.* 86] and [Mitchell *et al.* 86], both published in volume 1 number 1 of the *Machine Learning* journal.

In contrast to the propositional rules produced by attribute-based induction, the rules produced by explanation based learning is in first-order logic, which allows variable sharing between conjunctions within a single rule. The use of first-order logic permits also the information source (see Figure 2 of Section (1) to supply background knowledge, such as relationship between entities. These are the major advantages of explanation based learning and inductive logic programming over the propositional attribute-based paradigm. Inductive logic programming also starts with a domain theory. The differences between this paradigm and explanation based learning are: Inductive logic programming (1) accepts both positive and negative examples, (2) does not assume a target concept rule, and (3) there are no predefined utilities. In general, inductive logic programming is a much more complicated paradigm than explanation based learning.

Conclusions

We have covered only four sections in this document: (1) Background in knowledge-based systems, (2) Attribute-based induction, (3) Incremental induction, and (4) Explanation-based learning. It is true that there are too many other methods (such as mentioned in the introduction) missing to call this document a complete introduction to machine learning. The reader is warned that machine learning is *not* just those issues covered in the four sections. Instead, as explained in the introduction, this document is designed to be part of an AI course, and we have aimed to cover practical symbolic approaches only. The four sections may also be extended and adapted to an 16-20 hours' short course in machine learning.

With respect to the issues covered in the four sections, the reader might easily question the representativeness of the author's HCV algorithm within the machine learning field. HCV is a newly developed algorithm, based on the little known extension matrix approach. It might be better to include systems like AQ and CN4 which are more often cited in the literature. However, HCV is the author's own work, so it is natural that it is included. More importantly, AQ-like algorithms and HCV share the same representation (*i.e.*, the variable-valued logic), but HCV has been shown [Wu 93] to be more compact in results and low-order polynomial in time complexity.

Acknowledgements

The author would like to express warm thanks to the two anonymous reviewers for their comments on an earlier draft of this document. Most of the comments have been addressed in the introduction and the conclusions.

References

[Boswell 90] R. Boswell, Manual for NewID Version 6.1, *TI/P2154/RAB/4/2.5*, The Turing Institute, Glasgow, Scotland, 1990.

[Breiman *et al.* 84] L. Breiman, J.H. Friedman, R.A. Olshen and C.J. Stone, *Classification and Regression Trees*, Wadsworth, Belmont, California, 1984.

[Cestnik *et al.* 87] B. Cestnik, I. Kononenko, and I. Bratko, ASSISTANT 86: A Knowledge-Elicitation Tool for Sophisticated Users, *Progress in Machine Learning*, I. Bratko and N. Lavrac (Eds.), Wilmslow: Sigma Press, England, 1987.

[Cohen & Feigenbaum 82] Paul R. Cohen and Edward A. Feigenbaum (Eds.), *The Handbook of Artificial Intelligence*, Volume III, William Kaufmann, Inc., 1982.

[DeJong *et al.* 86] G. DeJong and R. Mooney, Explanation–Based Learning: An Alternative View, *Machine Learning*, **1**(1986), 145–176.

[Gams *et al.* 91] M. Gams, M. Drobnič and M. Petkovšek, Learning from Examples — A Uniform View, *International Journal of Man-Machine Studies*, **34**(1991), 49–68.

[Ginsberg 93] M. Ginsberg, *Essentials of Artificial Intelligence*, Morgan Kaufmann Publishers, 1993.

[Hong 85] J. Hong, AE1: An Extension Matrix Approximate Method for the General Covering Problem, *International Journal of Computer and Information Sciences*, 14(1985), 6: 421–437.

[Hong *et al.* 87] J.R. Hong and C. Uhrik, The Extension Matrix Approach to Attribute-Based Learning, *Progress in Machine Learning*, I. Bratko and N. Lavrac (Eds.), Wilmslow: Sigma Press, England, 1987.

[Hunt et al. 66] E.B. Hunt, J. Marin and P.T. Stone, *Experiments in Induction*, Academic Press, New York, 1966.

[Jackson 90] P. Jackson, *Introduction to Expert Systems*, Second Edition, Addison-Wesley, 1990.

[Lenat 79] D.B. Lenat, On Automated Scientific Theory Formation: A Case Study Using the AM Program, *Machine Intelligence 9*, J. Hayes *et al.* (Eds.), New York: Halstead, 1979.

[Lenat 83] D.B. Lenat, EURISKO: A Program that Learns New Heuristics and Domain

Concepts – The Nature of Heuristics III: Program Design and Results, *Artificial Intelligence*, **21**(1983), 61–98.

[Mellish 91] C. Mellish, The Description Identification Problem, *Artificial Intelligence*, **52**(1991), 151–167.

[Michalski 75] R.S. Michalski, Variable-Valued Logic and Its Applications to Pattern Recognition and Machine Learning, *Computer Science and Multiple-Valued Logic Theory and Applications*, D.C. Rine (Ed.), Amsterdam: North-Holland, 1975, 506–534.

[Michalski *et al.* 86] R.S. Michalski, I. Mozetic, J. Hong and N. Lavrac, The Multi-Purpose Incremental Learning System AQ15 and Its Testing Application to Three Medical Domains, *Proceedings of AAAI 1986*, 1986, 1041–1045.

[Mitchell 77] T. Mitchell, Version Spaces: A Candidate Elimination Approach to Rule Learning, *Proceedings of the Fifth International Joint Conference on Artificial Intelligence*, Cambridge, Mass., 1977.

[Mitchell *et al.* 86] T. Mitchell, R. Keller and S. Kedar-Cabelli, Explanation – Based Generalization: A Unifying View, *Machine Learning*, **1**(1986), 47–80.

[Quinlan 79] J.R. Quinlan, Discovering Rules by Induction from Large Collections of Examples, *Introductory Readings in Expert Systems*, D. Michie (Ed.), Gordon and Breach, London, 1979, 33–46.

[Quinlan 86] J.R. Quinlan, Learning from Noisy Data, *Machine Learning*, Vol. 2, J. Carbonell and T. Mitchell (Eds.), Tioga, Palo Alto, USA, 1986.

[Quinlan 88a] J.R. Quinlan, Induction, Knowledge and Expert Systems, *Artificial Intelligence Developments and Applications*, J.S. Gero and R. Stanton (Eds.), North–Holland: Elsevier Science Publishers B. V., 1988, 253–271.

[Quinlan 88b] J.R. Quinlan, Decision Trees and Multi-Valued Attributes, *Machine Intelligence 11: Logic and the Acquisition of Knowledge*, J.E. Hayes, D. Michie and J. Richards (Eds.), Clarendon Press, Oxford, England, 1988, 305–318.

[Quinlan 92] J.R. Quinlan, *C4.5: Programs for Machine Learning*, Morgan Kaufmann Publishers, 1992.

[Quinlan *et al.* 87] J.R. Quinlan, P.J. Compton, K.A. Horn and L. Lazarus, Inductive Knowledge Acquisition: A Case Study, *Applications of Expert Systems*, Addison-Wesley, 1987.

[Schlimmer *et al.* 86] J.C. Schlimmer and D. Fisher, A Case Study of Incremental Concept Induction, *Proceedings of the Fifth National Conference on Artificial Intelligence*, Philadelphia, Morgan Kaufmann, USA, 1986, 496–501.

[Shapiro 87] A.D. Shapiro, *Structured Induction in Expert Systems*, Turing Institute Press in association with Addison-Wesley, Workingham, UK, 1987.

[Utgoff 89] P.E. Utgoff, Incremental Induction of Decision Trees, *Machine Learning*, 4(1989), 161–186.

[Van Harmelen & Bundy 88] F. Van Harmelen and A. Bundy, "Explanation-Based Generalisation = Partial Evaluation", *Artificial Intelligence*, **36**(1988): 401–412.

[Winston 70] P. Winston, Learning Structural Description from Examples, *Ph.D. Thesis*, Mass.: MIT AI Lab., USA, 1970.

[Wu 92] X. Wu, HCV User's Manual (Release 1.0 June 1992), *DAI Technical Paper No. 9*, Department of Artificial Intelligence, University of Edinburgh, Scotland, 1992.

[Wu 93] X. Wu, The HCV Induction Algorithm, *Proceedings of the 21st ACM Computer Science Conference*, S.C. Kwasny and J.F. Buck (Eds.), ACM Press, USA, 1993, 168–175.

Compiler Detection of Function Call Side Effects

David A. Spuler
Department of Computer Science
James Cook University
Townsville, QLD 4811
Australia
AND
A. Sayed Muhammed Sajeev
Department of Software Development
Monash University
Caulfield East, VIC 3145
Australia

Keywords: Side effect, function calls, static analysis

Edited by: Xindong Wu
Received: January 6, 1994 **Revised:** May 5, 1994 **Accepted:** May 11, 1994

The determination of whether an operation in a procedural language such as Pascal causes side effects has important applications in static error detection, program optimization, program verification and other software tool issues. For simple operations, such as assignment, there is obviously a side effect. However, to detect whether a function call causes a side effect, the body of the function must be analyzed. A function call can produce a side effect in a number of ways, such as: modifying a global variable, modifying an internal static variable, modifying one of its pass-by-reference arguments, producing output, consuming input etc. This paper explains the algorithms and data structures used to determine if a function call causes a side effect.

1 Introduction

Static analysis of computer programs is used in a variety of applications. Static analysis techniques provide means to study the properties of a program without executing it. For example, an optimizing compiler can employ data flow analysis to remove redundant code (dead-code elimination), to replace expensive operations by cheaper ones (reduction in strength) etc. [1]. Static analysis is extensively used in proving programs correct [2]. Error checkers like Lint [8] also use static analysis techniques to detect bugs and obscurities in programs.

Side effects are used as a method of communication among program units. However, the method is not considered very elegant because of its adverse effect on clarity of programs. Besides, side effects can prevent a compiler from generating optimized code. For example, in the loop:

```
while ( i < test(j) ) do
        i := i+1
```

an optimizing compiler would prefer to move the evaluation of the function call `test` to before the loop. (This optimization is called code motion.) This is possible only if it can be shown that `test` is free of side effects (and that `j` is not an alias for `i`).

This paper describes algorithms that can be used by an optimizing compiler to detect functions (and procedures) with side effects; it is also useful in error checkers to issue warnings on the problems side effects can create.

Algorithms have been proposed for side effect detection [3]. However, previous efforts were mostly on considering side effects through global variables and reference parameters. If optimizations such as in the example above are to succeed, other types of side effects must also be considered.

In the next section we define the term *side effect*

and identify the different causes of side effects. In Section 3, we describe in detail the algorithms and data structures used to detect functions with side effects. In Section 4 we point out some of the limitations of static analysis techniques in detecting side effects. Section 5 gives the conclusions.

2 What is a Side Effect?

The term *Side effect* refers to the modification of the nonlocal environment [7]. Generally this happens when a function (or a procedure) modifies a global variable or arguments passed by reference parameters. But there are other ways in which the nonlocal environment can be modified.

We consider the following causes of side effects through a function call:

1. Performing I/O
2. Modifying global variables
3. Modifying local permanent variables (like `static` variables in C)
4. Modifying an argument passed by reference
5. Modifying a local variable, either automatic or static, of a function higher up in the function call sequence (usually via a pointer).

All but item 3 obviously affect the nonlocal environment. We shall briefly describe these effects. Any function that performs I/O, whether it be console I/O or file I/O, is causing a side effect. Such functions do affect the environment outside the program. Failing to get the effect of I/O will obviously change the meaning of the program.

Another important way that a function can cause a side effect is by modifying a global variable that is used by some other function. Such a side effect can be caused by an explicit assignment to a global variable, or in less obvious ways such as calling a library routine to set a global clock (effectively a global variable).

A modification to an internal permanent variable (eg. a `static` variable in C) can also be a side effect if a later call to the same function uses the old value of the `static` variable. This is because a static variable, though declared within a function, retains its value after the the function terminates. Normally, the local environment of a function is destroyed at the termination of the function. Since static variables are not destroyed we can consider them as belonging to the nonlocal environment with the restriction that they are accessible only through the statements in the current function.

Modification of an argument passed by reference to a function is a form of side effect. In C, this is applicable to array variables only. However, languages like Pascal allows parameter passing by reference. Since pointers can set up *aliases*, modification of a variable through a pointer dereference can also cause a side effect. For example, the pointer could be pointing to a global variable.

3 Detection of Side Effects

To be certain that a function call produces no side effects, the following conditions are sufficient (but not necessary):

1. The function does not perform any I/O
2. No global variables are modified
3. No local permanent variables are modified
4. No pass-by-reference parameters are modified
5. No modification is made to nonlocal/static variables via pointers
6. Any function called also satisfies these conditions

The detection of side effects is done in two phases. In the first phase, each function in the program is considered separately (intraprocedural analysis) and all except condition 6 are checked. This will result in two lists: a list of functions which do not satisfy one or more of conditions 1 to 5, and therefore have the potential[1] to cause side effects, and another list of functions which satisfies conditions 1 to 5. The second list will be side effect free only if condition 6 is satisfied; this is checked in Phase 2 (Interprocedural analysis).

The algorithms are introduced here in an incremental fashion. First we consider the simple case (ignoring permanent variables and pointers) and later in Sections 3.4 and 3.5, we modify the

[1]The analysis is flow insensitive. That is, if a function can cause side effects in any of its execution path, then it will be flagged to have the potential to cause side effects.

```
Initialize the entries in the side effect table
            to (null set).
For each function f do
   For each statement s in f do
      If s modifies reference parameters
            r1,..,rn of f then
         seTable[f, ref] :=
            seTable[f, ref] ∪ [r1,..,rn];
      If s modifies nonlocal variables
            v1,..,vn of f then
         seTable[f, nonlocal] :=
            seTable[f, nonlocal] ∪ [v1,..,vn];
      If s is an input statement
         seTable[f, io] :=
            seTable[f, io] ∪ [R];
      If s is an output statement
         seTable[f, io] :=
            seTable[f, io] ∪ [W]
      end;
end;
For each function f in the seTable do
      if an entry in that row is non null then
         seTable[f, side effect] := true
end;
```

Figure 1: Algorithm for intraprocedural analysis to detect side effects

algorithm to consider the effect of pointers and permanent variables.

3.1 Intraprocedural analysis

In the intraprocedural analysis, statements in each function are analysed separately. Information about library functions is used to identify functions that perform i/o. A function that has statements which make calls to the i/o functions is identified as having the i/o side effect. A function can access nonlocal data objects through nonlocal variables (i.e., variables declared outside the function) and parameters passed by reference. The modification is detected by examining every assignment statement and input statement within the function body. If any of these variables appear in the left hand side of the assignment statement then the function is marked to produce a side effect. Note that in languages like C, short hand forms like i++ can be written to mean i = i+1; these forms are taken care of while scanning for assignments.

```
program demo;
var b : integer;

function f3 : integer;
var x : integer;
begin
   x := f1 (x);
   x := f2
end;

function f2 : integer;
var x : integer;
begin
   x  := 20;
   f2 := x
end;

function f1 (var a : integer)
         : integer;
begin
   read (a);
   a := f2;
   a := f3
end;

begin
   write (f1(b));
end.
```

Figure 2: An example program for side effect detection

3.1.1 The method

The statements in the function are scanned and the information collected is stored in a table called the side effect table (*se table*). The *se table* has a row for each function in the program. Columns are titled *Nonlocal Variables, Reference Parameters, I/O, Indirect Calls* and *Side Effect.* If a function is found by analysis to cause a side effect by modifying the nonlocal variables, reference parameters or through i/o, then the *Side Effect* entry for that function will have the value *true.* Moreover, if the side effect is caused by modifying one or more nonlocal variables, then the set of nonlocal variables which are modified in the function is given in the column titled *Nonlocal Variables.* The case for reference parameters is similar. For i/o, an input statement will be coded

Table 1: *Se Table* for Program demo

Fun.	Nonlocal	Ref	I/O	Indirect	S-effect
f1	ϕ	[a]	[R]	ϕ	true
f2	ϕ	ϕ	ϕ	ϕ	?
f3	ϕ	ϕ	ϕ	ϕ	?

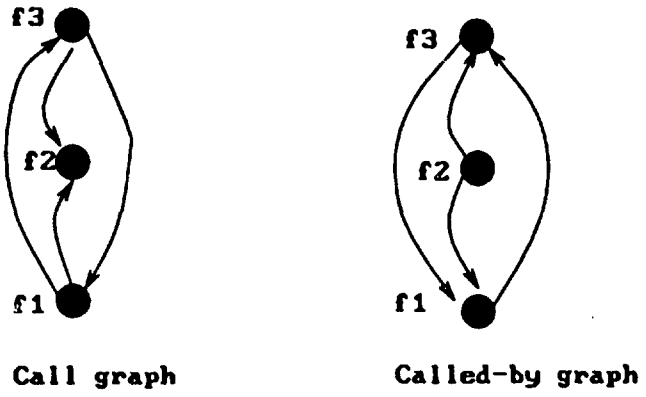


Figure 3: Call and called-by graphs for Program demo

as R and any output statement will be coded as W. The column *Indirect Calls* is left null in the first phase; it is later used to record the set of side effect causing functions which are called by the current function.

The algorithm to create the *se table* through intraprocedural analysis is given in Figure 1.

3.2 An example

Consider the Pascal program given in Figure 2. The *se table* for this program is shown in Table 1.

The table shows that *f1* has side effects through the reference parameter a as well as through i/o. No other function is determined to cause side effects at this stage.

3.3 Interprocedural analysis

The next phase is to do the interprocedural analysis to detect side effects created by calls made to other functions. For this, a *called-by graph* for the program is constructed. A called-by graph is similar to a *call graph* [1]. A call graph is a directed graph with a node for each function and an edge from the node for function A to the node for function B, if and only if, the body of function A contains a call to function B. In a called-by graph, the direction of the edges are reversed. Figure 3 shows the call graph and called-by graph of the program in Figure 2.

```
Unmark all rows in the se table;
While unmarked rows with true in the s-effect
        column exist in the table do
   Let f be the function name of an unmarked
        entry with true in the s-effect column;
   Mark the row corresponding to f;
   For every function g to which there is an
        edge from f in the called-by graph do
      Add f to the set of functions in
           the Indirect entry of g;
      Enter true in the s-effect entry of g;
   end;
end;
Replace question marks with false
        in the s-effect entry for
           all functions with question marks.
```

Figure 4: Algorithm for interprocedural analysis to detect side effects

Table 2: The *Se Table* after the interprocedural analysis

Fun.	Nonlocal	Ref	I/O	Indirect	S-effect
f1	ϕ	[a]	[R]	[f3]	true
f2	ϕ	ϕ	ϕ	ϕ	false
f3	ϕ	ϕ	ϕ	[f1]	true

The partially complete *se table* from the first phase and the called-by graph can be used to fill the remaining entries in the *se table*. This is done using the algorithm in Figure 4.

Intuitively, the algorithm starts with a function f, which is already found to cause side effect, and denotes all functions that call f as causing a side effect. This is repeated for all functions which cause side effects. If no function is found to cause a side effect at the end of phase 1, there is no need to do phase 2 because all functions are side effect free in the sense of Section 2.

Applying this algorithm to Table 1, gives Table 2.

All the functions found to be side effect free can be guaranteed to be side effect free in the sense of Section 2. However, as in the case of any static analysis technique, the converse need not hold. For example, our analysis will show that function f in Figure 5 is likely to cause side effects, but whether it will actually do so can be decided only at run time; it will depend on the input value of

```
program demo0;
var x : integer;

function f (a:integer) : integer;
begin
   if a = 0 then
      x := 10;
   f := 20;
end;

begin
   read(x);
   write (f(x))
end.
```

Figure 5: A program where it is not possible to statically detect side effect

Table 3: The *Se Table* for Program demo1

Fun.	Nonlocal	Ref	I/O	Indirect	S-effect
f1	ϕ	ϕ	ϕ	[f2]	true
f2	[y]	ϕ	ϕ	ϕ	true

the variable x.

In some cases, more complicated analysis can be performed to check whether some of the functions which are flagged to have side effects in the initial analysis indeed have a side effect. Consider the program in Figure 6.

Our static analysis will flag both f1 and f2 as having side effects. The *se table* created after both the intra and interprocedural analyses is:

Function f2 causes side effect because it modifies a non local variable (y). Function f1 causes side effect because it calls another function (f2) which has side effect. However, a look at f1 reveals that f1 does not have side effects because, f2 modifies a local variable of f1. The intrafunctional algorithm can be modified as in Figure 7 to incorporate this.

This algorithm uses two functions. Function *nl(g, nonlocal_set)* returns a set of variables which are elements of *nonlocal_set* but are also nonlocal to the function *g*. The function *ap(f, g, parameter_set)* returns a set of actual parameters of the function *g* corresponding to the formal parameters in the *parameter_set* for the function *f*.

Intuitively, if a side effect causing function *f* is called by *g*, then *g* will cause side effect only if

```
program demo1;
var x : integer;

function f1 : integer;
var y : integer;
   function f2 : integer;
   begin
      y := 10;
      f2 := 10;
   end;
begin
   y := 20;
   y := f2
end;

begin
   x := f1
end.
```

Figure 6: Another example program for side effect detection

Table 4: *Se Table* for Program demo1 after applying the modified algorithm

Fun.	Nonlocal	Ref	I/O	Indirect	S-effect
f1	ϕ	ϕ	ϕ	ϕ	false
f2	[y]	ϕ	ϕ	ϕ	true

nonlocal variables or reference parameters modified in *f* are also nonlocal variables or reference parameters of *g*. (Note that a nonlocal variable of *f* could be a reference parameter of *g* or a reference parameter of *f* could be a nonlocal variable of *g*.)

After applying the modified algorithm to the demo1 program in Figure 6, we get the *se table* in Table 4. The table shows that only function f2 has the potential to cause side effect.

3.4 Detecting side effects due to pointer dereferences

Nonlocal pointer variables can cause side effects. While other nonlocal variables cause side effects by direct modification, pointer variables can cause side effects by dereferencing. However, a number of side effects can occur in a non-obvious fashion by modification of a value via a pointer:

```
Unmark all rows in the se table;
While unmarked rows with the s-effect
        entry true exist in the se table do
  Let f be the function name of an unmarked
        entry with the s-effect entry true;
  Mark the row corresponding to f;
  For every function g to which there is an
      edge from f in the called-by graph do
    nlset := seTable[g, Nonlocal];
    rpset := seTable[g, Ref];
    seTable[g, Nonlocal] :=
        seTable[g, Nonlocal] ∪
          nl(g, seTable[f, Nonlocal]);
    actuals := ap(f, g, seTable[f, Ref]);
    seTable[g, Nonlocal] :=
      seTable[g, Nonlocal] ∪
        (actuals ∩ nonlocal vars. of g);
    seTable[g, Ref] :=
      seTable[g, Ref] ∪
        (actuals ∩ reference params. of g);
    If (nlset ∩ seTable[g, Nonlocal] ≠ φ) or
      (rpset ∩ seTable[g, Ref] ≠ φ ) then
      seTable[g, Indirect] :=
        seTable[g, Indirect] ∪ [f];
      seTable[g, s-effect] := true;
    end;
  end;
end;
Replace question marks with false in
      the s-effect entry for
        all functions with question marks.
```

Figure 7: Modified algorithm to find side effects caused by function calls

```
Iteration::
For each function f do
  nonLocalSet := set of all ptr type parameters
      and nonlocal variables appearing in f;
  modifiedSet := φ;
  For every element p of nonLocalSet do
    alias[p] := [p]; {Comment: alias is an
        associative array of sets }
  end;
  For every statement s in f do
    If s is an assignment of any element of
        alias[p] to any pointer q then
      Remove q from all alias sets;
      alias[p] := alias[p] ∪ [q];
      If q an element of modifiedSet then
        modifiedSet :=
          modifiedSet ∪ alias[p];
    If s modifies a data object by
        dereferencing through q then
      modifiedSet := modifiedSet ∪ the alias
        set in which q is a member;
  end;
  seTable[f, Pointers] := modifiedSet ∩
        nonLocalSet;
end;

Post iteration::
For every function f do
  if seTable[f, Pointers] ≠ φ then
    seTable[f, s-effect] := true;
end;
```

Figure 8: Modification to the intraprocedure algorithm to take care of pointers

- Pointer variables passed as *value* parameters can also cause side effects by dereferencing.
- Nonlocal pointers can be assigned to local pointer variables; the latter can then cause side effects by dereferencing.

In the earlier analysis we did not consider pointer dereferencing. To do this analysis, the *se table* is added one more column named *Pointers*. An entry in *Pointers* column shows the set of pointers that can cause a side effect through dereferencing. For every nonlocal pointer used in a function, a set of variables (local or nonlocal) which has the same pointer value is maintained. Any modification by dereferencing any of the pointers in this set is flagged to cause a side effect during the intraprocedural analysis (i.e., phase 1). Figure 8 shows the modification to the algorithm in Figure 1. (The algorithm must be modified in the iteration and the post-iteration parts.)

As an example of the application of the algorithm, consider the analysis of the procedure `f` in the program in Figure 9.

We have the *nonLocatSet* = *[a, c]*. Initially,

$$alias[a] = [a]; \qquad alias[c] = [c];$$
$$modified = \phi; \quad seTable[f, Pointers] = \phi$$

After analysing statement {1} in the procedure `f`, the set values become:

$$alias[a] = [a]; \qquad alias[c] = [c];$$
$$modified = [d]; \quad seTable[f, Pointers] = \phi$$

After analysing assignment {2}, they change to:

$$alias[a] = [a, d]; \qquad alias[c] = [c];$$

```
program demo2;
type
   R = record
      value : integer
   end;
   pointer = ^R;

var a, b : pointer;

procedure f (c : pointer);
var d : pointer; i : integer;
begin

   for i := 1 to 2 do
      if i>1 then d^.value := 25;  {1}
      else d := a; {2} {this stmt will
         be executed before stmt {1}}
end;

begin
   new(a); a^.value := 15;
   new(b); b^.value := 20;
   f(b);
end.
```

Figure 9: A pointer-based example for side effect detection

$$modified = [a, d]; \quad seTable[f, Pointers] = \phi$$

Therefore, at the termination of the analysis,

$$seTable[f, Pointers] = [a]$$
$$seTable[f, s\text{–}effect] = true$$

3.5 Detecting side effect due to permanent local variables

Some programming languages allow permanent local variables. `Static` variables in C and `own` variables in Algol are examples. Their values are retained from one function call to another, thus causing a side effect. Detecting this type of side effect is not a problem. The *se table* needs one more column for permanent local variables. Any modification to such variables will change the entries in that column during the intraprocedural analysis, just as in the case of other columns in the *se table.*

4 Limitations of the Method

There are situations where a function has the potential to cause a side effect, and yet a particular call to that function will not cause the side effect to occur. For example, the arguments to a function call may mean that the execution path containing the statements causing the side effect is not executed, as below:

```
function F (i : integer) : integer;
begin
   if i > 0 then
      writeln ('Hello');
   F := 0
end;

F(0);
```

The call to `F()` does not produce output, although `F()` must still be classed as an output-producing side effect function. The general problem of determining if a particular call to a function will invoke a particular execution path is non-computable (because being able to solve it would provide a solution to the halting problem), although many special cases are solvable, such as the simple one above. Analysis of the function body and possibly the entire program would be able to identify function calls that do or do not cause a side effect. However, the advanced methods necessary to detect which paths will be executed at run-time are beyond the scope of this paper. Thus, a limitation of our method is that, while it can identify function calls that definitely cause no side effect, any function call that is considered to cause a side effect may not actually do so at run-time. More extensive global analysis would be necessary to prove that a function call will cause a particular path inside the function to be executed. Fortunately, this limitation is not too damaging, since error detection of side effect related errors will, at worst, produce a spurious warning message, and program optimizations tend to rely on proving that a function call does *not* cause a side effect, rather than proving that it does.

5 Discussion

Our interest in function calls that cause side effects arose during the development of a static

checker for C. It was important to identify side effect operations so as to warn about expression statements that produced no side effects (null effect statements) and order of evaluation problems (where the order of occurrence of side effects was ambiguous). Lint also checks for such errors [8]. At the time the problem was not considered too important and it was decided to consider function calls as side effects for the first test, and not as side effects for the second test — both choices aim to reduce spurious warnings about correct code.

Detecting that a function call does not produce any side effects is important in compiler error detection to warn about "null effect" function call statements. Static detection of programming errors is not the only area where it is useful to know if a function call causes a side effect. It is also useful in program verification (side effects do make program verification difficult) and more importantly for compiler optimization (to remove redundant function calls, for example).

The algorithms developed analyse functions and detect those functions which do not cause side effects due to any of the reasons identified in Section 2. For those functions with potential side effects, the algorithms also identify the causes of those side effects. We are not aware of any side effect tools in actual use. Banning [3], Burke [4], and Cooper and Kennedy [5] have proposed algorithms to detect side effects but, their works were based only on side effects caused by global variables and parameter passing by reference.

Even when a function has a side effect, it may not adversely affect the calling function. For example, if the value of the global variable is not used at any time after the function returns, but instead the old value is overwritten, the change to the value of the global variable has had no effect. An example of this would be a function that uses a global variable, or even a global data structure, to perform some temporary calculations (e.g. a treesort routine uses a global binary tree as a temporary data structure to sort an array, and then destroys the binary tree). However, this does not mean that the called function does not cause a side effect— it simply means that the calling function is not affected by it.

Similarly a local static variable can only be accessed within the current function, and hence if it does not use the old value of the `static` variable, a modification to that variable does not affect the function. (Interestingly, any local `static` variable that is set before used inside a function need not be declared as "static", and hence any such instances probably deserve a compiler warning.) This situation can be detected by algorithms to determine if the variable is *set before it is used.* A number of algorithms have already been developed for the similar problem of variables *used before set* and these algorithms could easily be modified: Fosdick and Osterweil [6] discuss the algorithms in general and an example of their implementation is the Omega checker [9].

The analysis is more difficult for global variables because there is no scope restriction that only one function can use the variable (as there is for local `static` variables). It becomes necessary to perform global analysis to determine which functions actually modify or use the global variable. It is possible to generalize the algorithm for detection of set-before-used local `static` variables to global variables. A side effect due to the modification to a global variable within a function F is harmless if:

1. The global variable is set-before-used in the function F
2. Any other function setting or accessing this global variable must be called via F.

The second condition guarantees that any call to a function using the global variable must first pass through F, which sets the global variable and ignores any previous value. This condition can be detected by examination of the call graph. The lower level functions using the global variables form one or more sub-graphs of the call graph, where the only entry points to this sub-graph are directed edges originating at the node for F.

Procedural parameters (i.e., procedures passed as parameters to other procedures) are not included in our algorithms. In our experience, procedural parameters are not very common in programs, and the utility of the algorithms are not lost by leaving them out.

References

[1] A.V. Aho, R Sethi, and J.D. Ullman, *Compilers, Principles, Techniques and Tools*, Addison Wesley, Reading, MA (1986).

[2] K.R. Apt, A Static Analysis of CSP Programs, In *Logics of Programs: Proceedings 1983*, Lecture Notes in Computer Science, No. 164, 1-17, Springer Verlag, New York (1983).

[3] J.P. Banning, An Efficient Way to Find the Side Effects of Procedure Calls and the Aliases of Variables, *6th Annual ACM Symposium on Principles of Programming Languages*, Jan. (1979).

[4] M. Burke, An Interval-Based Approach to Exhaustive and Incremental Interprocedural Data-Flow Analysis, *ACM Transactions on Programming Languages and Systems*, Vol. 12, No. 3, 341-395, (1990).

[5] K.D. Cooper, K. Kennedy, Interprocedural Side-Effect Analysis in Linear Time, *ACM SIGPLAN Notices*, Vol. 23, No. 7, 57-66, (1988).

[6] L.D. Fosdick, L.J. Osterweil, Data Flow Analysis in Software Reliability, *ACM Computing Surveys*, Vol. 8, No. 3, 305-330 (1976).

[7] C. Ghezzi, M. Jazayeri, *Programming Language Concepts*, John Wiley & Sons, New York (1987).

[8] S.C. Johnson, *Lint: A C Program Checker*, Technical Report, AT&T Bell Labs, Murray Hill, New Jersey, (1978).

[9] C. Wilson, L.J. Osterweil, Omega—A Data Flow Analysis Tool for the C Programming Language, *IEEE Trans. on Software Eng.*, Vol. 11, No. 9, 832-838 (1985).

CONTROL ABSTRACTIONS IN MODULA-2: A CASE STUDY USING ADVANCED BACKTRACKING

Libero Nigro & Giuseppe Veneziano
Dipartimento di Elettronica, Informatica e Sistemistica
Universita' della Calabria, I-87036 Rende(CS) - Italy
E-mail: nigro@ccusc1.unical.it

Keywords: Control abstractions, Modula-2, reusable modules, backtracking, simulation

Edited by: Rudi Murn
Received: March 9, 1994 **Revised:** June 11, 1944 **Accepted:** June 27, 1994

This paper shows that Modula-2, extended with a general control abstraction called a thread, supports the construction of programmer-defined control modules. As an example, a realistic control regime providing advanced backtracking is presented.

1 Introduction

The control structures provided by Modula-2 include not only ordinary procedures but also coroutines. Unlike procedures, coroutines are a retentive control structure; coroutine activations can be retained arbitrarily long relative to other activations, and therefore be freely stored in data structures, passed as values or results, and so on. Other retentive control structures include backtracking [1,2] and simulation [3]. Most languages provide at most one retentive control structure and special-case its implementation; this is unfortunate since, as was argued by Lindstrom [4], new functionality may be achieved when control structures are combined (e.g., by combining backtracking with simulation, allowing the delivery of events in the simulation to be undone). A general framework for the addition of retentive control structures to a sequential programming language has been described in [5]. The proposal is centered upon a basic control abstraction called a *thread.* Threads are first-class control objects which support the construction of programmer-defined, reusable *control abstractions.* Threads can be hosted by a high-level language without changes to the run-time support. They have been embedded in Modula-2 and C++[6]. This paper surveys threads, shows their Modula-2 implementation and presents, as a case study, Lindstrom generalized backtracking [4], that is a realistic, advanced control regime which combines coroutining, backtracking and simulation facilities.

2 A Modula-2 General Control Abstraction

Threads were designed to provide a general abstraction for experimentation in sequential control. Threads are *control objects* in the same sense as SL5 *environments* [7] and Scheme *continuations* [8,9]. They contribute to the run-time representation of program control states. Primitive transformations on threads define program control events. Threads have first-class status, i.e. they may be assigned to data structures and passed to or returned from procedures. They are proposed in [5], where a formal operational model for sequential, unit-level control is presented. In this section we give only an informal introduction to threads and to their primitive control operations. Threads are mapped onto Modula-2 processes. Their main features are summarized in the following:

- Threads are dynamically created and destroyed.
- A thread is created for executing a *routine.* A routine coincides with a parameterless, globally-declared procedure whose general structure is shown in Figure 1.
- Passing value parameters to threads is simulated by means of an untyped, stack managed

```
PROCEDURE aRoutine(* a: Ta; b: Tb; ... *);
...
VAR
  a: Ta; b: Tb; (*formal parameters*)
...
BEGIN
   NEWROUTINE(TSIZE(Ta)+TSIZE(Tb)+...,WspSize);
   ...Arg(b); Arg(a); (*get argument values*)
   ...
END aRoutine;
```

Figure 1: Structure of a routine.

```
DEFINITION MODULE Routines;
   FROM SYSTEM IMPORT BYTE;
PROCEDURE NEWROUTINE
   ( ArgsSize, (*in*)
     WspSize: (*in*) CARDINAL );
PROCEDURE Arg
   ( VAR A: ARRAY OF BYTE (*out*) );
VAR VOID: RECORD END;(*the null object*)
END Routines.
```

Figure 2: The definition module Routines.

argument area.

- A thread can suspend itself and (re)activate another via a resume operation. In addition, the resume operation enables a value to be transmitted to the resumed thread.
- A thread can terminate itself and transfer the control to a continuation thread.
- The state of a thread can be restarted from its beginning or restored to a previous one saved by means of a copy operation.
- At the beginning of a program execution there exists the main thread only, automatically created and started by the run-time support for executing the main module. The main thread is subjected to some restrictions: it has no argument area, it cannot be copied, destroyed, restarted or restored.
- Thread storage is under user-control and may be managed via a garbage collection strategy.

The Modula-2 implementation of threads consists of two library modules, Routines and Threads, whose definitions are reported respectively in Figure 2 and 3. Threads module is essential in developing control abstractions. Routines module introduces a few entities which are useful in application programs using abstract control regimes implemented according to the library module Threads. Module implementations rely upon Modula-2 processes and depend neither on the run-time support nor on the operating system.

Declaring a routine. An invocation of the NEWROUTINE procedure:

NEWROUTINE(argssize, wspsize);

must be the first statement in a user-defined routine (see also Figure 1). Argssize and wspsize respectively specify the amounts of storage to be allocated for the argument area and for the execution of the routine as a Modula-2 process.

Taking the arguments of a routine. The next argument from the argument area of the running thread can be taken by invoking

Arg(A);

which pops the next argument and returns it into A.

Again referring to Figure 1, a routine body normally starts with a call of NEWROUTINE, followed by a number of Arg calls equal to the number of the expected arguments. Of course, the ordering of Arg invocations must follow the same ordering of argument bindings (see later in this paper). The rest of a routine body is structured, in general, according to the control operations of a selected control regime.

Establishing a new thread. A new thread T for the execution of a routine P, is generated, initialized and unmarked, by an invocation of the Create procedure:

T:=Create(P, Allocate);

Thread memory space is allocated through the user-provided Allocate procedure.

Binding an argument to a thread. An argument arg can be bound to a thread T by invoking

```
DEFINITION MODULE Threads;
   FROM SYSTEM IMPORT BYTE, ADDRESS;
TYPE
   THREAD;
   StorageProc =
      PROCEDURE( VAR ADDRESS, CARDINAL );
PROCEDURE Create
   ( P        : PROC (*in*);
     Allocate : StorageProc (*in*) ) : THREAD;
PROCEDURE Bind
   ( T   : THREAD (*in*);
     Arg : ARRAY OF BYTE (*in*) );
PROCEDURE Resume
   (     T      : THREAD (*in*);
 Value  : ARRAY OF BYTE (*in*);
     VAR Result : ARRAY OF BYTE (*out*) );
PROCEDURE Terminate
   ( T          : THREAD (*in*);
     Value      : ARRAY OF BYTE (*in*);
     Deallocate : StorageProc (*in*) );
PROCEDURE Copy
   (     Source   : THREAD (*in*);
     VAR Dest     : THREAD (*out*);
 Allocate : StorageProc (*in*) );
PROCEDURE Running() : THREAD;
PROCEDURE Routine
   ( T: THREAD (*in*) ): PROC;
PROCEDURE Restart
   ( T: THREAD (*in*) );
PROCEDURE Restore
   (     Source : THREAD (*in*);
     VAR Dest   : THREAD (*in/out*);
 Value  : ARRAY OF BYTE (*in*) );
PROCEDURE Main(): THREAD;
PROCEDURE Destroy
   ( VAR T          : THREAD (*in/out*);
 Deallocate : StorageProc (*in*) );
PROCEDURE Last(): THREAD;
PROCEDURE Mark
   ( T: THREAD (*in*) );
PROCEDURE GarbageCollect
   ( Old        : THREAD (*in*);
     Deallocate : StorageProc (*in*) );
END Threads.
```

Figure 3: The definition module Threads.

the Bind procedure:

Bind(T, arg);

which pushes arg, uninterpreted, into the argument area of T.

Transferring the control to a thread. An invocation of the Resume procedure:

Resume(T, value, result);

transfers control to thread T, which receives value as a result. The execution of the running thread is suspended within the Resume operation, which returns the expected result at the subsequent resumption only.

Terminating a thread. The running thread can terminate its execution, yielding the control to a continuation thread T, by using the Terminate procedure:

Terminate(T, value, Deallocate);

Thread T receives value as a result. The running thread is destroyed by invoking the Deallocate procedure.

Copying a thread. A copy of a given Source thread can be achieved by invoking the Copy procedure:

Copy(Source, Dest, Allocate);

which returns the copy into Dest. The copy is performed in such a way that if Source is destroyed and replaced by the copy, the computation is not modified. The copy includes the dynamic variables allocated by using the standard module Storage only if a separate heap is independently administrated by Storage in the workspace of each Modula-2 process. The Copy operation establishes between threads the equivalence relation "is a copy of".

Getting the running thread. The identity of the currently executing thread is returned by invoking the Running procedure:

C:=Running();

Getting the routine of a thread. The routine associated to a given thread T is returned, as a

PROC value, by invoking the Routine procedure:

R:=Routine(T);

Restarting a thread. A thread T can be restarted into the initial state established by the Create operation, by invoking the Restart procedure:

Restart(T);

Only the argument values of T are preserved.

Restoring a thread. A given thread Source can copied upon an existing thread Dest, by invoking the Restore procedure:

Restore(Source, Dest, value)

If Dest is the running thread, the operation transfers control to the copied thread which receives value as a result. Implementation may require that the two threads be in the relation "is a copy of".

Getting the main thread. The identity of the main thread is returned by invoking the Main procedure:

M:=Main();

Destroying a thread. A thread T can be destroyed, and its memory space released according to a Deallocate procedure, by invoking the Destroy procedure:

Destroy(T, Deallocate);

Getting the last thread. The identity of the most recently generated thread is returned by invoking the Last procedure:

T:=Last();

Marking a thread. A thread can be marked as not being garbage collectible, by invoking the Mark procedure:

Mark(T);

Collecting garbage threads. All the threads generated after a given Old thread, and whose state is unmarked, can be destroyed by invoking the GarbageCollect procedure:

GarbageCollect(Old, Deallocate);

All marked threads are then unmarked, with the exception of the main thread.

3 Programming Control Abstractions

Modules Routines and Threads enable the activities of programming control abstractions and writing application programs to be separately performed. These two activities require different competences. Therefore, in the context of control programmability, reusability is a major point: the application programmer needs a library of reusable control modules for selecting the control regimes suitable to his/her particular problem.

Programming a control abstraction is similar to programming a data abstraction. This analogy is made evident by the definitions which follow.

A control regime is composed of:

- a *control structure*, i.e. a data structure including threads among its components;
- a *set of control operations*, whose responsibility is to manipulate the control structure and transfer control between threads. Some operations provide for creation and termination of control regimes.

A *control module* is a library module which realizes a *control abstraction*, i.e. a control regime whose control structure is hidden and accessible solely via the control operations provided, or an *abstract control type*, i.e. a class of control abstractions. Obviously, to maximize reusability, a control module should always realize an abstract control type. So, in general, a Modula-2 control module exports both an opaque control type and a set of control procedures. Among these there must exist a procedure to instantiate a control regime and another which terminates it.

In a realization of a control module, thread storage management is a key point. When standard Storage module administers a separate heap for each process, there may exist problems concerning both thread lifetimes and thread dimensioning. In these cases, threads may be allocated in

```
DEFINITION MODULE TailRecursiveControl;
FROM SYSTEM IMPORT BYTE;
PROCEDURE Call
   (     Func    : PROC; (*in*)
 Args    : ARRAY OF BYTE; (*in*)
     VAR Result : ARRAY OF BYTE (*out*) );
PROCEDURE TailCall
   ( Func : PROC; (*in*)
     Args : ARRAY OF BYTE (*in*) );
PROCEDURE Return
   ( Result : ARRAY OF BYTE (*in*) );
END TailRecursiveControl.
```

Figure 4: The definition module TailRecursive-Control.

a global heap provided by a programmer-defined storage handler.

3.1 A Tail Recursive Control Module

The following shows a simple example of a control module which implements a *tail recursive* [10,11] control abstraction. The definition module, reported in Figure 4, besides the usual Call and Return control operations, exports also the operation TailCall.

Calling a function. A function Func with arguments Args can be called by invoking the Call operation:

Call(Func, Args, Result);

which only returns when the activation of Func returns, with Result then set to the value computed by Func.

Tail calling a function. The operation TailCall,

TailCall(Func, Args);

can be used instead of Call when the invocation of Func is the last action of the current function instance which returns the result evaluated by Func as its own result. The use of TailCall, when correct, enables memory space to be conserved, in that the frame of the current instance is deallocated in advance.

Returning from a function. An invocation of the operation Return:

Return(Result);

returns from the currently executing instance, with Result as the computed value. If the returning instance is that created by the first Call operation, control regime is then terminated.

An implementation of TailRecursiveControl module is shown in Figure 5. The control structure, i.e. a stack of threads, is obtained by linking nodes local to the different invocations of Call. The current thread is held out of the stack. The first Call pushes the thread of the original caller onto the stack. Such a thread is continued at control regime termination (see the Return operation).

An example of the use of the TailRecursiveControl module is shown in the Figure 6 where a routine which searches an item x in a binary search tree t and returns the corresponding (sub)tree or NIL, is reported. The two parameters of Search, i.e. t and x, are passed as fields of a record argument of type SearchPair. Routine Search may be invoked by a Call(Search, sp, r), after which r contains the result.

4 An Advanced Backtracking Regime

This section presents a backtracking control module which was programmed using threads. The control form is derived from that proposed by Lindstrom in [4] and consists of a combination of coroutining, backtracking and simulation facilities. Such a generalized backtracking regime allows an application to be structured as a tree of non-deterministic (ND) systems. The purpose of an ND system is to provide scope for the backtracking primitives (see below). Each ND system is governed by a *coroutine controlling instance* which actually represents the system. An ND system possesses, in addition to itself, all of its descendant ND systems. It possesses also a set of ordinary coroutines created at the time the ND system was current. Coroutining is possible both in an ND system and among ND systems.

The *control state* of a program during execution [12] consists of a *set of chains* established via the control (or dynamic) link of the various

```
IMPLEMENTATION MODULE TailRecursiveControl;
FROM SYSTEM   IMPORT BYTE, ADR;
FROM Storage  IMPORT ALLOCATE, DEALLOCATE;
FROM Terminal IMPORT WriteString, WriteLn;
FROM Routines IMPORT VOID;
FROM Threads  IMPORT THREAD, Create, Bind, Resume,
     Restart, Terminate, Running, Routine;
TYPE
   RecursionStack = POINTER TO RecursionNode;
   RecursionNode  = RECORD
       Caller : THREAD;
       Next   : RecursionStack;
    END;
VAR Top : RecursionStack; (*control structure*)
PROCEDURE PUSH( P : RecursionStack );
BEGIN
   WITH P^ DO Caller := Running();
      Next   := Top; END; Top:=P;
END PUSH;
PROCEDURE POP() : THREAD;
   VAR P : RecursionStack;
BEGIN P:=Top;
   WITH P^ DO Top := Next;
      RETURN Caller; END;
END POP;
PROCEDURE Call(     Func   : PROC;
    Args   : ARRAY OF BYTE;
VAR Result : ARRAY OF BYTE );
   VAR RN : RecursionNode; Calley : THREAD;
BEGIN PUSH( RecursionStack( ADR(RN)) );
   Calley := Create( Func, ALLOCATE );
   Bind( Calley, Args );
   Resume( Calley, VOID, Result );
END Call;
PROCEDURE Return( Result : ARRAY OF BYTE );
BEGIN Terminate( POP(), Result, DEALLOCATE );
END Return;
PROCEDURE TailCall(     Func : PROC;
Args : ARRAY OF BYTE );
    VAR Calley : THREAD;
BEGIN Calley:=Running();
   IF Routine( Calley ) = Func THEN
      Bind( Calley, Args );
      Restart( Calley );
   ELSE
      Calley := Create( Func, ALLOCATE );
      Bind( Calley, Args );
      Terminate( Calley, VOID, DEALLOCATE );
   END
END TailCall;
END TailRecursiveControl.
```

Figure 5: The implementation module TailRecursiveControl.

```
PROCEDURE Search
   (* sp:SearchPair *)(*: BinaryTree*);
VAR sp : SearchPair;
BEGIN
   NEWROUTINE( TSIZE(SearchPair), 600);
   Arg( sp );
   WITH sp DO
      IF (t=NIL) OR (t^.item=x) THEN
 Return( t );
      ELSIF t^.item>x THEN
 t:=t^.left; TailCall( Search, sp);
      ELSE
 t:=t^.right; TailCall( Search, sp );
      END;
   END;
END Search;
```

Figure 6: The routine Search.

(procedure or coroutine) instances. One chain, termed the *Operating Chain* (OC), is conventionally anchored by the abstract processor. All the others are *idle chains* and contain one suspended (or detached) coroutine, which acts as their anchor. Coroutine operations allow semisymmetric activations only [3]. The following are some useful definitions:

- A *coroutine* is said to be *active* if it belongs to the OC.
- The *currently executing coroutine* (self) is the youngest coroutine instance in the OC at any given moment.
- An *ND system* is said to be *active* if its controlling instance belongs to the OC.
- The *current ND system* is the active one whose controlling instance is furthest from the processor.

An ND system may possess a local *simulation system* made up of a *set of processes* (coroutines). One of these coroutines is designated the *simulation main*. It is itself a simulation process, and is responsible of setting up the simulation system. In addition to its processes, a simulation system is characterized by:

- a read-only variable *TIME*, initialized to zero, which provides the (virtual) time of the system;
- an *event list* which records the events generated by the processes.

```
DEFINITION MODULE NDTREE;
FROM SYSTEM IMPORT ADDRESS, BYTE;
TYPE
   COROUTINE;
PROCEDURE NewNDtree
   ( P : PROC (*in*);
     Arg : ARRAY OF BYTE (*in*) );
PROCEDURE NDCREATE
   ( P   : PROC; (*in*)
     Arg : ARRAY OF BYTE (*in*) ) : COROUTINE;
PROCEDURE CREATE
   ( P   : PROC; (*in*)
     Arg : ARRAY OF BYTE (*in*) ) : COROUTINE;
PROCEDURE CALL
   ( C : COROUTINE (*in*) );
PROCEDURE DETACH;
PROCEDURE TERMINATE;
PROCEDURE CHOICE
   ( N : CARDINAL (*in*) ) : CARDINAL;
PROCEDURE FAILURE;
PROCEDURE NEXTCHOICE() : CARDINAL;
PROCEDURE CUT;
PROCEDURE SELF() : COROUTINE;
PROCEDURE Failed
   ( C : COROUTINE (*in*) ) : BOOLEAN;
PROCEDURE SetGlobalVars
   ( Adr  : ADDRESS (*in*);
     Size : CARDINAL (*in*) );
PROCEDURE SIMCREATE
   ( P   : PROC (*in*);
     Arg : ARRAY OF BYTE (*in*) ) : COROUTINE;
PROCEDURE TIME() : CARDINAL;
PROCEDURE SCHEDULE
   ( C    : COROUTINE (*in*);
     Time : CARDINAL (*in*) );
PROCEDURE PASSIVATE;
PROCEDURE SimulationMain() : COROUTINE;
END NDTREE.
```

Figure 7: The definition module NDTREE.

A *simulation system* is *current* when the possessing ND system is current.

The NDTREE control module is a full implementation of Lindstrom control regime. A few new primitives were added in the interest of efficiency or to facilitate the expression of some specific programming tasks. The operations NEXTCHOICE and CUT was derived from [13]. The NDTREE definition module is shown in Figure 7.

The meaning of the NDTREE control operations is now clarified by giving an informal description of their semantics. Control operations are conveniently grouped according to coroutine, backtracking and simulation control forms upon which NDTREE is built.

4.1 Coroutine Control

The coroutine facilities offered by NDTREE are similar to those originally introduced by SIMULA-67. The opaque type COROUTINE abstracts the notion of coroutine instances which are expected to be dynamically allocated/deallocated. Coroutines are mapped on threads and ultimately on Modula-2 processes. Control exchanges among coroutines are basically realized through semisymmetric operations CALL/DETACH. In other words, COROUTINE instances are assumed to be orchestrated by a controlling supervisor which gets resumed at each coroutine DETACH, and is in charge of activating, via a CALL, another coroutine according to a control strategy. Of course, each called coroutine will be resumed from the control point where it last was left off.

Making a coroutine. A new coroutine can be established by invoking the CREATE procedure:

CREATE(P, arg);

which creates and returns a new ordinary coroutine for the execution of routine P. Arg is bound to this coroutine as an argument. The coroutine is detached and forms an idle chain. Its reactivation point is set at the first statement of P. The coroutine belongs to the current ND system. Calling CREATE when the main process of a simulation system is current, establishes the created coroutine as a process of that system.

Activating a coroutine. A given coroutine C can be activated by invoking the CALL procedure:

CALL(C);

which attaches the referenced coroutine C to its CALLer on the Operating Chain (OC). C must be the anchor of an idle chain. It becomes the current coroutine and its execution resumes at its reactivation point. C and its CALLer must belong to the same ND system, or C must be the controlling instance of an ND system which is a son of the current one.

Suspending a coroutine. The currently exe-

cuting coroutine can suspend its execution by using the DETACH procedure:

DETACH;

which causes the current coroutine to be detached from the OC, thus forming a new idle chain. Its reactivation point is set at the statement immediately following DETACH. Control is transferred to the CALLer, which resumes its execution at its reactivation point.

Terminating a coroutine. The currently executing coroutine can terminate itself by using the TERMINATE procedure:

TERMINATE;

which is similar to DETACH but the current coroutine is marked terminated and it is no longer CALLable. If the current coroutine is the controlling instance of the root ND system, the whole control regime is terminated.

Getting the current coroutine. The identity of the currently executing coroutine is returned by the SELF procedure:

C:=SELF().

4.2 Backtracking Control

NDTREE approach to backtracking relies on the notion of an ND system, which is governed by a coroutine controlling instance especially created to this purpose. ND control instance may be manipulated (i.e. CALLed and DETACHed) as an ordinary coroutine instance, but in addition it defines a scope of backtracking control for the special primitives CHOICE and FAILURE.

Making an ND system. A new ND system can be established by invoking the NDCREATE procedure:

nd:=NDCREATE(P, Arg);

which creates and returns the associated coroutine controlling instance whose routine is P. Arg is bound to this coroutine as an argument. The coroutine is detached and forms an idle chain. Its reactivation point is set at the first statement of P. The controlling instance is responsible of setting up the system. The new ND system is linked as a son to the current ND system.

Establishing a choice point. A new choice point within the current ND system, where one of N alternatives is to be chosen, can be established by invoking the CHOICE procedure:

c:=CHOICE(N);

Calling CHOICE causes the actual state of the current ND system and those of all possessed systems to be recorded and the value N returned. The values from N-1 down to 1 are returned through subsequent calls of FAILURE. If the value returned is 1, the choice point is cancelled. Calling CHOICE with N=1 causes N to be returned, and no choice point is established. Calling CHOICE with N=0 is equivalent to calling FAILURE.

Signalling a failure. An invocation of the FAILURE procedure:

FAILURE;

signals that the previous choices cannot possibly lead to a solution of the problem. FAILURE rolls the state of the current ND system and that of all the ND systems possessed back to the most recent choice point. The execution then continues with CHOICE, which delivers the next value. If there is no choice point, then FAILURE forces the termination of all the coroutines of the current ND system, including its controlling instance. Control is then transferred to the coroutine, if there is one, of the father ND system which first activated the current ND system. After such a termination, an invocation of Failed (C) with C being the controlling instance of the terminated ND system, returns TRUE. If the terminated ND system is the root one, then the whole control regime is terminated.

Immediate returning of the next choice value. If the current ND system has a choice point, then invoking the NEXTCHOICE procedure:

c:=NEXTCHOICE();

returns the next value of the most recent choice point but, unlike FAILURE, without a state roll back. If there is no choice point, or the last value returned by CHOICE or NEXTCHOICE is

1, then the invocation of NEXTCHOICE is equivalent to FAILURE.

Cancelling a choice point. If the last value returned by CHOICE or NEXTCHOICE is greater than 1, an invocation of the CUT procedure:

CUT;

cancels the most recent choice point of the current ND system. CUT always terminates by invoking FAILURE.

Inquiring an ND system against fatal failure. A call to the Failed procedure:

bool:=Failed(C);

returns TRUE iff the ND system of which coroutine C is the controlling instance, has been terminated by a FAILURE which has found no choice point.

Declaring global variables. A call to the SetGlobalVars procedure:

SetGlobalVars(Adr, Size);

"declares"
the global area identified by $< Adr, Size >$ as one which contains the global variables of the current ND system. Such a non interpreted area will be copied and restored automatically during CHOICE and FAILURE operations occurring within the ND system. NDTREE module implementation is unable to undo assignments to globals not selected by a SetGlobalVars operation, and allocation/deallocation of dynamic variables in a global heap.

4.3 Simulation Control

NDTREE offers simulation control features which can be used in combination with backtracking. The logical parallelism normally required by simulation is already provided by the coroutine control form. Simulation can be controlled in a simulation system, governed by a coroutine controlling instance especially created to this purpose. A simulation system is made up of a collection of simulation processes. The simulation control instance is itself a member of the simulation system and acts as its simulation main. It can manipulated, i.e. CALLed and DETACHed, as a normal coroutine. Simulation processes are a refiniment of ordinary coroutines. They can use the primitives SCHEDULE and PASSIVATE respectively for scheduling events in the future and for yielding control to the simulation controlling instance which is responsible of the basic actions of advancing the virtual time and delivering an event (i.e., activating a simulation process) to its relevant process.

Making a simulation system. A new simulation system is established by invoking the SIMCREATE procedure:

ss:=SIMCREATE(P, Arg);

which creates and returns its main process (coroutine), whose routine is P. Arg is bound to this coroutine as an argument. The coroutine is detached and forms an idle chain. Its reactivation point is set at the first statement of P. The coroutine belongs to the current ND system. The main process is responsible of creating the simulation processes. The simulation processes are private of the current simulation system and may use the operations SCHEDULE and PASSIVATE only. The new simulation system is possessed by the current ND system, which may activate it by CALLing its main process.

Getting the current simulation time. The current value of the simulation time is returned by an invocation of TIME procedure:

now:=TIME().

Scheduling an event. A new event $< C, Time >$ is created and added to the event list of the current simulation system, by a call to SCHEDULE procedure:

SCHEDULE(C, Time);

C must be a process of such a system. Time represents normally a future simulation time when reactivation of C is planned to occur.

Passivating a simulation process. A call to the PASSIVATE procedure:

PASSIVATE;

causes the following sequence of actions to be executed:

(a) the current simulation process is DETACHed;

(b) if the event list of the current simulation system is empty, all the processes of the system, including the main one, are terminated. Control is then transferred to the coroutine by which the main process was activated;

(c) otherwise, an event $< P, t >$ with minimum t is removed from the event list;

(d) if $t > TIME$ then TIME is set to t;

(e) process P is CALLed.

Getting the main simulation process. A call to the SimulationMain procedure:

sm:=SimulationMain();

returns the coroutine acting as the main process of the current simulation system.

4.4 Starting/Terminating the NDTREE Control Regime

An invocation of the NewNDtree procedure:

NewNDtree(P, Arg);

initializes the NDTREE control regime and makes the root ND system by creating its controlling instance, whose routine is P. Arg is bound to this coroutine as an argument. Control is then passed to the newly created coroutine, which is in charge of setting up the system by creating its component coroutines. The operation NewNDtree only terminates at control regime termination (see the operations DETACH, TERMINATE and FAILURE), at which time the invocator of NewNDTree continues.

4.5 The Notion of an ND System Execution State

For a fuller description of the CHOICE and FAILURE operations of NDTREE, the following reports Lindstrom definition of an *ND system execution state* [4, pag. 14].

"Let S be an ND system governed by control instance **v**. Then

(a) The *execution state* of S *is* its *choice-level state.*

(b) The *current choice-level state* of S consists of:

(i) the *current attempt-level state* of S, and

(ii) a stack of *CHOICE-execution records* corresponding to unexhausted CHOICE invocations local to S, in most recent topmost order. Each such record consists of

(1) the *attempt-level state* of S current when that CHOICE was encountered, and

(2) the *last value* delivered by that CHOICE invocation

(c) The *current attempt-level state* of S consists of

(i) the current attachment chain of coroutines anchored by **v** (along with their contained variable values);

(ii) all idle chains currently headed by non-ND coroutine instances possessed by **v** (along with their variable values);

(iii) the current state of the *simulation system* possessed by **v**, made up of the simulation processes (along with their contained variable values), the value of TIME and the event list (along with its contained event values);

(iv) the current choice-level state of each ND system governed by an ND control instance possessed by **v**."

Calling CHOICE(N) local to S causes the following actions to take place:

```
IF N=0 THEN a FAILURE is done
ELSIF N=1 THEN
   the execution continues with CHOICE
   which delivers N
ELSE
   (1) a copy of the current attempt-level
       state of S is paired with N and stacked
       on the CHOICE-execution stack within the
       execution state of S, and
   (2) execution continues with CHOICE, which
       delivers N.
```

The effect of calling FAILURE local to S is as follows:

```
IF the CHOICE-execution stack of execution
   state of S is empty THEN
   S is terminated as a whole
ELSE
   the record on top of that stack is popped,
   yielding an N value and an attempt-level
   state of S, which is made current by being
   installed in its execution state.
   A CHOICE(N-1) is then done.
```

5 A Programming Example

Lindstrom has furnished several examples which exemplify the possible programming paradigms which are supported by its control combination. In particular, the control form allows for:

- backtracking being applicable over attempt-level computations involving coroutines;
- coroutine management of multiple backtracking subsystems;
- a mixture of the two, e.g. backtracking control over a coroutine-managed package of backtracking subcomputations;
- optimization problems to be solved in connection with simulation.

The following shows a programming example derived from Lindstrom work [4, pag. 19], which makes use of most NDTREE operations.

> **Job Mix Problem:** A multiprogrammed computer system is given a set of one or more independent jobs to run. Each job consists of a well- formed sequence of two or more fixed duration phases, each beginning with the allocation or deallocation of a particular, unique and nonsharable device. We consider the computing load of each phase to be negligible, so there is no a priori limit to the number of jobs that can run concurrently, nor does that number have any impact on the real time of each phase in progress. Our question is: What is the earliest time at which that job mix may be finished?

This is an optimization problem over the space of all possible allocation sequences. Moreover, the possibility of deadlock situations makes only some of those sequences admissible. A suitable solution requires a control regime where basic simulation facilities can be used in combination with backtracking. An ND system may be planned with a simulation system nested in it. The backtracking process allows the ND system state to be rolled back as soon as an allocation sequence reveals to be suboptimal, in that takes a longer time than the current minimum. In particular, FAILURE allows device statuses, job states and the event list of the simulation system to be restored, and so it enables all the possible allocation sequences to be examined. This solution makes use of backtracking over a coroutine- based attempt-level programming. The situation considered here (see Figure 8) is slightly different, in that two backtracking systems co-operate in a (quasi)parallel fashion to find the earliest overall time of two job mix problems. The program establishes an ND tree where two sons of the root ND system are each dedicated to solve a job mix problem. There are two backtracking algorithms controlled as coroutines by a master represented by the controlling instance of the root node. This organization allows each son ND system to automatically prune from its context tree those paths which turn out to be suboptimal with respect to the current global minimum.

The following contains some comments about JobMix module. In Job routine, each phase begins with the allocation or deallocation of a device. In each case label, the use of CHOICE(2)=1 is worthy of note. It represents a binary fork which is useful to ensure full enumeration of all allocation sequences. In the Alloc alternative, if the requested device is free, then job occupies it immediately. Also, full evaluation of the IF condition implies the generation of a new choice point in the current ND system with CHOICE returning 2. Such a choice point ensures that after a FAILURE, the job is enqueued in the device queue, thus making the allocation of this device to another job possible. If the device is already busy, there is no choice but to enqueue job in device queue for later allocation. Note that the short circuit evaluation of the IF condition guarantees that CHOICE(2) is not invoked. In the Dealloc alternative, the device is first freed. Then, if jobs are in the device's waiting queue, a new choice point is established, which makes it possible not to allocate the device immediately to one of these waiting jobs. This precaution complies with the allocation strategy and it is a condition for the full

```
MODULE JobMix;
FROM NDTREE   IMPORT NewNDtree, COROUTINE, CREATE,
   NDCREATE, SIMCREATE, SetGlobalVars, CALL, DETACH,
   TERMINATE, SCHEDULE, PASSIVATE, SimulationMain,
   TIME, CHOICE, FAILURE, NEXTCHOICE, Failed;
FROM Routines IMPORT NEWROUTINE, Arg, VOID;
FROM InOut    IMPORT WriteString, WriteCard;
FROM SYSTEM   IMPORT SIZE, ADR;
CONST
   JobMax   = 3;(*max nr of jobs per simulation*)
   PhaseMax = 6;(*max nr of phases per job*)
   PhaseTimeMax= 10;(*max time per phase*)
   Tlarge  = 180; (*upper bound on mix finish time*)
   DevMax  = 5; (*max nr of devices*)
   SimMax  = 2; (*max nr of simulations*)
TYPE
   JobNr   = [1..JobMax]; (*job numbers*)
   PhaseNr = [1..PhaseMax]; (*phase numbers in a job*)
   SimNr   = [1..SimMax];   (*simulation numbers*)
   PhaseTime  = [0..PhaseTimeMax]; (*phase durations*)
   FinishTime = [0..Tlarge]; (*mix finish times*)
   DevNr      = [1..DevMax]; (*device numbers*)
   JobSet     = SET OF JobNr;(*sets of jobs*)
VAR
   PC : ARRAY SimNr,JobNr
   OF PhaseNr; (*Phase Counts*)
   Dv : ARRAY SimNr,JobNr,PhaseNr
   OF DevNr; (*Devices*)
   Req: ARRAY SimNr,JobNr,PhaseNr
   OF (Alloc, Dealloc); (*Request action*)
   PL : ARRAY SimNr,JobNr,PhaseNr
   OF PhaseTime; (*Phase Lenght*)
   NrJobs : ARRAY SimNr
   OF JobNr; (*nr of jobs per simulation*)
   j : JobNr;
   p : PhaseNr;
   s, sMin : SimNr;
   MinFinishTime : FinishTime;
   TlastFinish   : ARRAY SimNr OF FinishTime;
   JobRef : ARRAY SimNr, JobNr
       OF COROUTINE; (*job process names*)
   V : ARRAY SimNr OF (*glob vars of ND systems*)
 RECORD (*device*)
   DEVSTATUS : (*statuses*)
      ARRAY DevNr OF (Busy, Free);
   DEVQUEUE  : (*wait queues*)
      ARRAY DevNr OF JobSet;
 END;
PROCEDURE Boss; (*routine of root ND system*)
VAR NDS : ARRAY SimNr OF COROUTINE;
BEGIN NEWROUTINE(0, 600);
   FOR s:=1 TO SimMax DO (*create ND system sons*)
      NDS[s]:=NDCREATE(Minimize,VOID);
   END;
   (*initialize MinFinishTime to a generous value*)
   MinFinishTime:=Tlarge;
   s:=1;
   REPEAT (*coroutine control of ND system sons*)
      IF NOT(Failed(NDS[s])) THEN CALL(NDS[s]) END;
      IF s < SimMax THEN INC(s) ELSE s:=1 END;
   UNTIL Failed(NDS[1]) AND Failed(NDS[SimMax]);
   TERMINATE;
END Boss;
```

Figure 8: The module JobMix. (to be continued)

```
PROCEDURE Minimize; (*routine of an ND system son*)
VAR Sim :COROUTINE;
BEGIN NEWROUTINE (0, 600);
   SetGlobalVars(ADR(V[s]), SIZE(V[s]));
   (*"declare" globals*)
   Sim:=SIMCREATE(Supervisor, VOID);
   (*create nested sim system*)
   LOOP
      (*re-start simulation*)
      CALL(Sim);
      (*new minimum found: give back control to boss*)
      DETACH;
   END;
END Minimize;
PROCEDURE Supervisor;
   (*routine of main simulation system*)
   (*see Figure 9 for code*)
PROCEDURE Job;
    (*routine of a simulation process*)
    (*see Figure 10 for code*)
BEGIN (*JobMix*)
   (*assume initialization of simulation descriptions,
     i.e. of NrJobs, PC, Dv, Req and PL data structures*)
   ...
   NewNDtree(Boss, VOID);
   (*create and start root ND system*)
   WriteString('Simulation n.'); WriteCard(sMin,2);
   WriteString(' have minimum time equal to');
   WriteCard(MinFinishTime,3); WriteLn;
END JobMix.
```

Figure 8: The module JobMix.

```
PROCEDURE Supervisor;
VAR j : JobNr; d : DevNr;
BEGIN
   NEWROUTINE (0, 1000);
   FOR d:=1 TO DevMax DO (*initialize device d*)
      V[s].DEVSTATUS[d] := Free;
      V[s].DEVQUEUE[d]  := JobSet{};
   END;
   FOR j:=1 TO NrJobs[s] DO
     (*create and schedule job processes*)
      JobRef[s,j]:=CREATE(Job, j);
      SCHEDULE(JobRef[s,j], 0);
   END;
   SCHEDULE(SimulationMain(), Tlarge);
   PASSIVATE;
   (*simulation has run to completion*)
   FOR d:=1 TO DevMax DO
      IF V[s].DEVQUEUE[d] <> JobSet{} THEN
(*job mix did not finish, so back up*)
 FAILURE END; END;
   MinFinishTime:=TlastFinish[s];
   (*must be new best, so save*)
   sMin:=s;
   DETACH;
   FAILURE; (*keep on trying*)
END Supervisor;
```

Figure 9: The routine of main simulation system.

```
PROCEDURE Job;
VAR nr, j : JobNr; p : PhaseNr;
BEGIN
   NEWROUTINE(SIZE(nr), 1000); Arg(nr);
   FOR p:= 1 TO PC[s,nr] DO
      (*step through job phases*)
      CASE Req[s,nr,p] OF
 Alloc:   (*allocation phase*)
    IF (V[s].DEVSTATUS[Dv[s,nr,p]]=Busy) OR
       (CHOICE(2)=1) THEN
       (*enter job into device queue*)
       INCL(V[s].DEVQUEUE[Dv[s,nr,p]],nr);
       PASSIVATE;
       (*job may now be allocated*)
       EXCL(V[s].DEVQUEUE[Dv[s,nr,p]],nr);
    END;
    V[s].DEVSTATUS[Dv[s,nr,p]]:=Busy  |
 Dealloc: (*deallocation phase*)
    V[s].DEVSTATUS[Dv[s,nr,p]]:=Free;
    IF V[s].DEVQUEUE[Dv[s,nr,p]]<>JobSet{}
    THEN
       IF CHOICE(2)=1 THEN
 (*give device to a waiting job*)
  j:=CHOICE(NrJobs[s]);
  WHILE NOT(j IN
     V[s].DEVQUEUE[Dv[s,nr,p]]) DO
     j:=NEXTCHOICE();
  END;
  (*job j is chosen*)
  SCHEDULE(JobRef[s,j], TIME());
       END;
    END;
      END; (*CASE*)
      SCHEDULE(JobRef[s,nr], TIME()+PL[s,nr,p]);
      PASSIVATE;
      (*phase now completed*)
      IF TIME()>MinFinishTime THEN
(*already too long, so back up*)
FAILURE END;
   END;
   (*save job finish time; may be last of mix*)
   TlastFinish[s]:=TIME();
   PASSIVATE;
END Job;
```

Figure 10: The routine of a simulation process.

enumeration of all allocation sequences. After a subsequent FAILURE, CHOICE delivers 1, and one of the waiting jobs is chosen to be SCHEDULEd at the current time. If there is no waiting job, no further choice point is generated.

A notable difference between our solution and that provided by Lindstrom phrased in an extended Pascal, is that all our routines are globals, but, logically, Job is nested in Supervisor which is nested in Minimize which is nested in Boss. In one case only does the required nesting need to be explicitely simulated - when we want to ensure that the data structures of the devices (DEVSTATUS and DEVQUEUE) are accessed both by Supervisor and Job routines. A global area containing the named variables is "declared" by SetGlobalVars within Minimize routine.

6 Implementation Issues

The NDTREE control module was implemented as a state machine. A single control structure is managed and hidden within the implementation module. The realization relies upon threads and their mechanisms which allow a thread to be copied and subsequently restored to a saved state. Thread storage is *GarbageCollect*ed at control regime termination. The following gives a flavor of the implementation by describing some key points underlying the data structures mantained by the control operations.

A *backtracking flagging structure* was designed, which permits the incremental saving of the attempt-level state of an ND system. The structure has the property that it can allow flagging from multiple ND systems. A flagged entity (i.e. a coroutine, an ND system, a CHOICE-execution stack, a simulation system) is actually copied just before it is accessed and modified. For instance, a detached flagged coroutine will be copied at CALL time, just before it is attached to the OC. Since flags may cumulate on the same entity of a descendent ND system, as a result of multiple CHOICE invocations local either to different possessing ND systems or to the same possessing ND system but at different times, they are arranged into a linked list, referred to by the entity descriptor itself.

A *flag* stores as its value a reference to the ND system by which the save request on the flagged

entity was raised. When a flagged entity is subsequently accessed, a copy is formed and saved on the CHOICE-execution stack of the requesting system and the flag is removed. To save memory space, a single copy of a (possibly multiple) flagged entity is actually created and shared among the requesting ND systems. A reference count technique helps to detect when a given copy may be deallocated.

The *CHOICE-execution stack* of an ND system was implemented as a linked stack of *tags*, whose top is referred to by the ND system descriptor. Each tag is associated to a CHOICE invocation local to the ND system, and actually points to a linked choice-stack whose records hold saved portions of the system, which are needed to restore the attempt-level state that was current at that CHOICE invocation time. A tag active reference is one held by an ND system which may use the tag choice-stack for reconstructing a previous attempt-level state. A tag passive reference is one held by a flag which indicates the tag as the destination site for a copy of the flagged entity. A flag actually points directly to a tag. A tag, i.e. a choice-point, is retained as long as it is actively referenced.The FAILURE operation uses the information held by a tag choice-stack to reconstruct a system state, but does not cancel it. A copied entity is restored upon the original one, and is mantained for further re-use. As a consequence, in order to save the choice-level state of a descendent ND system (its tag stack) on the CHOICE-execution stack of a higher possessing system which has called CHOICE, it is sufficient to generate references to tags of the descendent tag stack and save them in the top choice-stack of the tag stack of the higher system. As a result of the outlined organization, memory space is saved as much as possible.

7 Conclusions

This paper shows how standard Modula-2, extended by a general control abstraction called a *thread*, enables the construction of user-defined, reusable control modules. Threads are first-class control objects which were achieved using Modula-2 processes. Their implementation is totally portable. Another realization of threads has been built in C++[6] with the aim of studying control extensions within an object-oriented framework. Although C++ is not equipped with a basic coroutine feature, nor offer facilities for accessing stack activation frames as, for instance, does Smalltalk [15], it can normally host threads via low-level support code. Several control abstractions have been programmed using threads (see also [5] and [6]). These experiments confirm that thread is a practical, yet powerful tool for control regime experimentation. As an example, this paper presents a Modula-2 realistic control module, called NDTREE, which is a full realization of Lindstrom backtracking regime [4]. NDTREE allows backtracking to be used in combination with coroutines and simulation. A programming example in which backtracking systems co-operate in a quasi-parallel fashion to solve an optimization problem in connection with simulation, it is also presented. NDTREE highlights some difficulties which exist when using threads for implementing backtracking. Since state saving/restoring operations rely upon copying threads, undoing of global assignments and of allocation/deallocation of dynamic variables in a global heap must in general be handled by "ad hoc" solutions. Minor problems caused by the use of thread in Modula-2 are a need to specify statically the workspace sizes of processes, a lack of routine parameters and that of generic operations. Such difficulties are partially overcome by the C++ realization of thread, where the workspace size is determined dynamically at a control transfer, and where it can be possible to type-check data communications among threads through the use of class member functions which host a control operation.

Acknowledments

The helpful discussions with Michele Di Santo, Wilma Russo and Francesco Tisato are acknowledged. The authors wish to thank the anonymous referees for their suggestions which improved the shape of the paper.

References

[1] Cohen J. (1979) Non-Deterministic Algorithms, *Computing Surveys*, 11, 2, p. 79-94

[2] Clocksin W.F. & Mellish C.S. (1981) *Programming in Prolog*, Springer-Verlag, Berlin

[3] Birtwitle G.M., Dahl O.-J., Myhrhaug & K. Nygaard (1974) *SIMULA Begin*, Wiley, New York

[4] Lindstrom G. (1979) Backtracking in a generalized control setting, *ACM Trans. Program. Lang. Syst.*, 1, 1, p. 8-26

[5] Di Santo M., Nigro L. & Russo W. (1990) Programmer- Defined Control Abstractions in Modula-2, *Computer Languages* 15, 3, p. 141-152

[6] Nigro L. (1994) Control extensions in C++, *Journal of Object-Oriented Programming*, 6, 9, p. 37-47

[7] D. Hanson & R. Griswold (1978) The SL5 procedure mechanism, *CACM*, 21, p. 392-400

[8] Haynes C.T. , Friedman D. P. & Wand M. (1986) Obtaining coroutines with continuations, *Comput. Lang.*, 11, 3/4, p. 143-153

[9] Haynes C.T. & Friedman D. P. (1987) Embedding continuations in procedural objects, *ACM Trans. Program. Lang. Syst.*, 9, 4, p. 582-598

[10] Steele G.L. (1977) Macaroni is better than spaghetti, *ACM SIGPLAN Notices*, 12, 8

[11] Read C. (1989) *Elements of functional programming*, Addison-Wesley, Reading, Massachussets

[12] Wang A. & Dahl O.-J. (1971) Coroutine Sequencing in a Block-Structured Environment, *BIT*, 11, p. 425-449

[13] Helsgaun K. (1984) Backtracking programming with SIMULA, *Comput. J.*, 27, p. 151-158

[14] Wirth N. (1985) *Programming in Modula-2*, 3nd Ed., Springer Verlag, Berlin

[15] Goldberg A. & Robson D. (1983) *Smalltalk-80: The language and its implementation*, Addison-Wesley, Reading, MA

World Organisation of Systems and Cybernetics

Foundation of the Organisation

The World Organisation of Systems and Cybernetics (WOSC) is a federation of national associations and institutions devoted to systems or cybernetics. It was founded in 1969 by Professor J. Rose, who is now Honorary Director of the organisation. Dr Norbert Wiener is the President in Memoriam. The President is Professor Stafford Beer.

Board of Directors

The Director-General of WOSC is Professor Robert Vallée (France) who is assisted by a Board of Directors in promoting systems theory and cybernetics in the 41 countries that are members. The members of the Directorate are:

Director-General

Professor R. Vallée (France)

Directors

Dr A. Andrew (U.K.) — *Affiliations*
Dr C. Bilciu (U.S.A.) — *Congresses*
Professor F.H. George (U.K.) — *Acad. Affairs*
Professor N.-C. Hu (P.R. of China)
Professor A. Lopes Pereira (Brazil)
Professor D. Dutta Majumder (India)
Dr C. Musés (U.S.A.) — *Research*
Professor M. Najim (Morocco)
Professor E. Nicolau (Romania) — *Education*
Dr R. Rodrig. Delgado (Spain) — *Ext. Aff.*
Professor B.H. Rudall (U.K.) — *Institute*

Communication between Member Countries

The promotion of systems and cybernetics in the 41 member countries is achieved mainly through the exchange of information at meetings, congresses and through official publications. *Kybernetes* has been chosen as the official journal of WOSC. *Details may be obtained from MCB University Press.)*

WOSC Institute of Systems and Cybernetics

Details of the proposals for the WOSC Institute and a progress report are included in *Kybernetes*, Vol. 20 No. 5, 1991, pp. 50–2 in the *News, Conferences and Technical Reports Section.* The designate Executive Director is Professor B.H. Rudall (U.K.).

WOSC Secretariat

More details about the work of WOSC can be obtained from: *WOSC Secretariat, Professor R. Vallée, Director-General, 2 rue de Vouillé, 75015 Paris, France.*

Honorary Fellowships

The WOSC has established a limited number of Honorary Fellowships confined to most eminent scientists in the fields of cybernetics, systems, computers and related disciplines. In effect, the group of honorary Fellows constitutes a "World Academy of Cybernetics and Systems Scientists" and will be of great value in advancing these interdisciplinary and transdisciplinary sciences.

The following have accepted the WOSC invitation:

D. Gabor, Nobel Laureate (U.K.) [In M.];
B.D. Josephson, Nobel Laureate (U.K.);
L. Pauling, Nobel Laureate (U.S.A.);
I. Prigogine, Nobel Laureate (Belgium);
H.A. Simon, Nobel Laureate (U.S.A.);
A. Balevski, Bulgarian Acad. Sci. (Bulgaria);
S. Beer, Manchester Business School (U.K.);
J. Bigelow, Princ. Inst. Adv. Stud. (U.S.A.);
C. Chagas, Vatican Acad. Sci. (Brazil);
A. Danzin, I.N.R.I.A. (France);
R. Ericson, Soc. Gen. Syst. Res. (U.S.A.);
F.H. George, Brunel Univ. (U.K.);
T.C. Helvey, Univ. Tenn. (U.S.A) [In M.];
C. Hammer, Sperry-Univac (U.S.A.);
Kožešnik, Cze.-Slo. Acad. Sci. [In M.]
L. Kumar, Goverm. of India (India);
M. Mănescu, Acad. Econ. Stud. (Romania);
M.D. Mesarović, West. Res. Univ. (U.S.A.);
M. Nałecz, Pol. Acad. Sci. (Poland);
J.D. Palmer, IEEE (U.S.A.);
S. Ramo, T.R.W. Inc. (U.S.A.);
J.G. Santesmases, A.E.I.A. (Spain) [In M.];
M. Valentinuzzi, Inst. Cyber. (Argentina);
R. Vallée, Univ. Paris-Nord (France);
Sir G. Vickers (U.K.) [In M.];
L.A. Zadeh, Univ. Cal. (U.S.A);
J. v.d. Zouwen, F. Univ. Am. (Holland)

A.P. Železnikar

Kybernetes

The International Journal of Systems & Cybernetics, an official journal of the WOSC

MCB University Press Limited
62 Toller Lane – Bradford – West Yorkshire
England BD8 9BY

Kybernetes is a resource for academics, researchers and consultants in cybernetics and systems, knowledge engineers and computer specialists. It is one of the most informative specialist publications in its area. As an aid to research, and as a stimulus to exploring the complex interaction between man, machine and environment, it is obviously of tremendous value to anyone needing up-to-date information for their work.

The journal features a section entitled "Contemporary Systems and Cybernetics" which contains selected reports and surveys of current research and development. The journal also contains a Book Review section (contributed by a team of distinguished reviewers, News, Conferences, Technical Reports, up-to-date Announcements and Special Announcements.

Selected and refereed articles will be published on a variety of important topics in systems and cybernetics. The 'Communication' section has been expanded to allow shorter articles and communications to be published. A section, entitled 'Forum', allows contributors to present their views on all matters concerning this multidisciplinary field, whether published in the form of short presentations or as correspondence.

The Norbert Wiener Centenary Award for the authors of the best articles is made in recognition of the excellence of contributions to the journal and to encourage new authors to publish their best work in Kybernetes. For entering the subscription to Kybernetes, contact:

Julie Kitson

Customer Service Executive

MCB University Press Limited

60/62 Toller Lane

Bradford, England BD8 9BY.

TEX and TUG News

Volume 3, Number 2, April 1994

In 'Typographers Inn', shady practice, ditto marks, books, interletter spacing, and 'Serif' magazine are reported.

In 'New Publication', books and articles are summarized. One of the most interesting is: M. Goossens, F. Mittelbach and A. Samarin: The LaTeX Companion, Addison-Wesley, 1994, 448pp. US$34.50, ISBN 0–201–54199–8 AW.

This book is based on LaTeX2$_\varepsilon$, the new version of LaTeX which is currently available in its beta test release. It explains tools and techniques that enhance the use of LaTeX and help format documents more quickly and more efficiently. Topics treated include customizing commands and environments, changing page layout, preparing indices and bibliographies, and the New Font Selection Scheme (NFSS), as well as using PostScript fonts and PostScript images.

In (LA)TeX News, the NTG's CD of 4AllTeX is reported. At the end of 1993 the NTG (the Dutch-speaking TeX users group) released the package "4AllTeX: A TeX Workbench for MS-DOS PCs" on 31 discs. This June a CD will be released, on whicj the most important TeX- and LaTeX-related packages are assembled, including public domain system-independent software. The CD-ROM should enable users to set up a 4AllTeX system with minimal effort. The price of the CD plus booklet is set at Dfl. 60,– or US$35. To order, contact the NTG Secretary, P.O. Box 394, 1740 AJ, Schagen, The Netherlands. E-mail: `ntg@nic.surfnet.nl`.

Within Reports and Meetings of the TUG'94 in Santa Barbara, California (31 July – 4 August) which is the main event for LaTeX users and hackers, there are participants from Slovenia. In the Preliminary program, the following reports from Slovenia can be found: under *Publishing and design,* Marko Grobelnik, Dunja Mladenić, Darko Zupanič and Borut Žnidar: *Integrated System for Encyclopedia Typesetting Based on TeX* and under *Posters, workshops and discussion sessions,* Marko Grobelnik: *Database Publishing.*

EuroTeX'94 (26–30 September) will take place at Sobieszewo on an idyllic island off the coast of Gdansk in Poland.

A.P. Železnikar

ERK'94

Electrotechnical and Computer Conference
Elektrotehniška in računalniška konferenca

26.–28. September 1994

Invitation

for the third **Electrotechnical and Computer Conference ERK'94**, to be held from 26–28 September 1994 in Portorož, Slovenia. Presentations will be given in English and Slovene.

The following areas will be represented at the conference:

- *electronics,*
- *telecommunications,*
- *measurement,*
- *automatic control and robotics,*
- *computer and information science,*
- *artificial intelligence and pattern recognition,*
- *biomedical engineering,*
- *power engineering.*

The conference is being organized by the **Slovenian Section of IEEE** and other Slovenian professional societies:

- Slovenian Society for Automatic Control,
- Slovenian Measurement Society (ISEMEC 94),
- SLOKO-CIGRE,
- Slovenian Society for Medical and Biological Engineering,
- Slovenian Society for Robotics,
- Slovenian Artificial Intelligence Society,
- Slovenian Pattern Recognition Society.

Time schedule: Papers due *July 20, 1994*
Notification of acceptance *August 10, 1994*

For all additional information, please contact the conference chairmen.

ECML-95

8th EUROPEAN CONFERENCE ON MACHINE LEARNING 25–27 April 1995,

Heraklion, Crete, Greece

First Announcement and Call for Papers

General Information:

Continuing the tradition of previous EWSL and ECML conferences, ECML-95 provides the major European forum for presenting the latest advances in the area of Machine Learning.

Program:

The scientific program will include invited talks, presentations of accepted papers, poster and demo sessions. ECML-95 will be followed by MLNet familiarization workshops for which a separate call for proposals will be published (contact `mlnet@computing-science.aberdeen.ac.uk`).

Research areas:

Submissions are invited in all areas of Machine Learning, including, but not limited to:
abduction
analogy
applications of machine learning
automated discovery
case-based learning
computational learning theory
explanation-based learning
inductive learning
inductive logic programming
genetic algorithms
learning and problem solving
multistrategy learning
reinforcement learning
representation change
revision and restructuring

Program Chairs:

Nada Lavrač (J. Stefan Institute, Ljubljana) and Stefan Wrobel (GMD, Sankt Augustin).

Program Committee:

F. Bergadano (Italy)
I. Bratko (Slovenia)
P. Brazdil (Portugal)
W. Buntine (USA)
L. De Raedt (Belgium)
W. Emde (Germany)
J.G. Ganascia (France)
K. de Jong (USA)
Y. Kodratoff (France)
I. Kononenko (Slovenia)
W. Maass (Austria)
R. Lopez de Mantaras (Spain)
S. Matwin (Canada)
K. Morik (Germany)
S. Muggleton (UK)
E. Plaza (Spain)
L. Saitta (Italy)
D. Sleeman (UK)
W. van de Velde (Belgium)
G. Widmer (Austria)
R. Wirth (Germany)

Local chair:

Vassilis Moustakis, Institute of Computer Science, Foundation of Research and Technology Hellas (FORTH), P.O. Box 1385, 71110 Heraklion, Crete, Greece (E-mail `ecml-95@ics.forth.gr`).

Submission of papers:

Paper submissions are limited to 5000 words. The title page must contain the title, names and addresses of authors, abstract of the paper, research area, a list of keywords and demo request (yes/no). Full address, including phone, fax and E-mail, must be given for the first author (or the contact person). Title page must also be sent by E-mail to `ecml-95@gmd.de`. If possible, use the sample LaTeX title page that will be available from `ftp.gmd.de`, directory `/ml-archive/general/ecml-95`. Six (6) hard copies of the whole paper should be sent by 2 November 1994 to:

Nada Lavrač & Stefan Wrobel (ECML-95)
GMD, FIT.KI, Schloß Birlinghoven,
53754 Sankt Augustin,
Germany

Papers will be evaluated with respect to technical soundness, significance, originality and clarity. Papers will either be accepted as full papers (presented at plenary sessions, published as full papers in the proceedings) or posters (presented at poster sessions, published as extended abstracts).

System and application exhibitions:

ECML-95 offers commercial and academic participants an opportunity to demonstrate their systems and/or applications. Please announce your intention to demo to the local chair by 24 March 1995, specifying precisely what type of hardware and software you need. We strongly encourage authors of papers that describe systems or applications to accompany their presentation with a demo (please indicate on the title page).

Registration and further information:

For information about paper submission and program, contact the program chairs (E-mail `ecml-95@gmd.de`). For information about local arrangements or to request a registration brochure, contact the local chair (E-mail `ecml-95@ics.forth.gr`).

Important Dates:

Submission deadline:	2 November 1994
Notification of acceptance:	13 January 1995
Camera ready copy:	9 February 1995
Exhibition requests:	24 March 1995
Conference:	25 – 27 April 1995

SCAI'95

Fifth Scandinavian Conference on AI

Trondheim, Norway, May 29 - 31, 1995

The biennial Scandinavian Conference on Artificial Intelligence is the open Scandinavian forum for scientific exchange and presentation of AI research. The aim of the conference is to cover all aspects of AI research, and to bring together basic and applied research. The technical program will include paper and poster presentations, invited talks and panels. An award will be given to the best student paper.

The major theme for SCAI'95 will be 'Theory meets Practice', with facilitation of feedback from real world applications to the researchers as a central goal. Industry is particularly encouraged to submit papers.

The fifth SCAI is hosted by the University of Trondheim and SINTEF DELAB, in cooperation with the Norwegian AI Society, NAIS.

Submission of papers

Authors are requested to submit 5 hard-copies of papers written in English. Submitted papers should be unpublished and present original work. Papers should be double-spaced and not exceed 6000 words. Each copy of the paper should include a separate title page containing the title, full names, postal addresses, phone numbers and e-mail addresses of all authors, an abstract of 100-200 words and an indicator whether a paper or poster presentation is preferred.

Papers should be sent to

SCAI'95, Agnar Aamodt, Dept. of Informatics, College of Arts and Science, The University of Trondheim, N-7055 Dragvoll, NORWAY
email: agnar@ifi.unit.no,
fax: +47-73591733, phone: +47-73591838 / -1840

Key Dates

January 10, 1995 – Papers due
February 25, 1995 – Notification of acceptance
March 25, 1995 – Camera ready paper due

Program committee

Agnar Aamodt,
Jan Komorowski,
Tore Amble,
Bernt Bremdal,
Roar Fjellheim,
Steffen Leo Hansen,
Johan Moller Holst,
Sture Hagglund,
Carl Gustaf Jansson,
Andrew Jones,
Mette Kloster,
Aarno Lehtola,
Morten Lind,
Mihhail Matskin,
Brian Mayoh,
Jorgen F. Nilsson,
Erik Sandewall,
Markku Syrjaenen,
Ingeborg Solvberg,
Henry Tirri,
Enn Tuygu,
Erling Woods,

Conference organizing committee

Inge Nordbo,
Arvid Holme,

Conference secretariat

SCAI'95, Inge Nordbo, SINTEF DELAB, N-7034 Trondheim, Norway
fax: +47 73 53 25 86
e-mail: scai95@delab.sintef.no

SEVENTH PORTUGUESE CONFERENCE ON ARTIFICIAL INTELLIGENCE

The Seventh Portuguese Conference on Artificial Intelligence (EPIA'95) will be held at Funchal, Madeira Island, Portugal, on October 3-6, 1995 under the auspices of the Portuguese Association for AI. As in previous issues ('89, '91, and '93), EPIA'95 will be run as an international conference, English being the official language. The scientific program encompasses tutorials, invited lectures, demonstrations, and paper presentations. Five well known researchers will present invited lectures. The conference is devoted to all areas of Artificial Intelligence and will cover both theoretical and foundational issues and applications as well. Parallel workshops on Expert Systems, Fuzzy Logic and Neural Networks, and Applications of A.I. to Robotics and Vision Systems will run simultaneously (see below).

Invited Lecturers

The following researchers have already confirmed their participation, as guest speakers:
Marvin Minsky, MIT (USA)
Manuela Veloso, CMU (USA)
Luis Borges de Almeida, IST (Portugal)
Rodney Brooks, MIT (USA)

Submission of Papers

Authors must submit five (5) complete printed copies of their papers to the "EPIA'95 submission address". Fax or electronic submissions will not be accepted. Submissions must be printed on A4 or 8 1/2"x11" paper using 12 point type. Each page must have a maximum of 38 lines and an average of 75 characters per line (corresponding to the LaTeX article-style, 12 point). Double-sided printing is strongly encouraged. The body of submitted papers must be at most 12 pages, including title, abstract, figures, tables, and diagrams, but excluding the title page and bibliography.

Electronic Abstract

In addition to submitting the paper copies, authors should send to epia95-abstracts@inesc.pt a short (200 words) electronic abstract of their paper to aid the reviewing process. The electronic abstract must be in plain ASCII text (no LaTeX)) in the following format:

TITLE: <title of the paper>
FIRST AUTHOR: <last name, first name>
EMAIL: <email of the first author>
FIRST ADDRESS: <first author address>
COAUTHORS: <their names, if any>
KEYWORDS: <keywords>
ABSTRACT: <text of the abstract>

Authors are requested to select 1-3 appropriate keywords from the list below. Authors are welcome to add additional keywords descriptors as needed. Applications, agent-oriented programming, automated reasoning, belief revision, case-based reasoning, common sense reasoning, constraint satisfaction, distributed AI, expert systems, genetic algorithms, knowledge representation, logic programming, machine learning, natural language understanding, nonmonotonic reasoning, planning, qualitative reasoning, real-time systems, robotics, spatial reasoning, theorem proving, theory of computation, tutoring systems.

Review of Papers

Submissions will be judged on significance, originality, quality and clarity. Reviewing will be blind to the identities of the authors. This requires that authors exercise some care not to identify themselves in their papers. Each copy of the paper must have a title page, separated from the body of the paper, including the title of the paper, the names and addresses of all authors, a list of content areas (see above) and any acknowledgments. The second page should include the same title, a short abstract of less than 200 words, and the exact same contents areas, but not the names nor affiliations of the authors. This page may include text of the paper. The references should include all published literature relevant to the paper, including previous works of the authors, but should not include unpublished works of the authors. When referring to one's own work, use the third person. For example, say "previously, Peter [17] has shown that ...". Try to avoid including any information in the body of the paper or references that would identify the authors or their institutions. Such information can be added to the final camera-ready version for publication. Please

do not staple the title page to the body of the paper. Submitted papers must be unpublished.

Publication

The proceedings will be published by Springer-Verlag (lecture notes in A.I. series). Authors will be required to transfer copyright of their paper to Springer-Verlag.

Associated Workshops

In the framework of the conference three workshops will be organized: Applications of Expert Systems, Fuzzy Logic and Neural Networks in Engineering, and Applications of Artificial Intelligence to Robotics and Vision Systems. Real world applications, running systems, and demos are welcome.

Planning to Attend

People planning to submit a paper or/and to attend the conference or attend a workshop are asked to send a note (inquiries address) standing their intention, as early as possible, to the conference organizers, in order to estimate the facilities needed for the conference.

Conference & Program Co-Chairs

Carlos Pinto-Ferreira Instituto Superior Tecnico ISR, Av. Rovisco Pais 1000 Lisboa, Portugal
Voice: +351 (1) 8475105
Fax: +351 (1) 3523014
Email: cpf@kappa.ist.utl.pt

Nuno Mamede Instituto Superior Tecnico INESC, Apartado 13069 1000 Lisboa, Portugal
Voice: +351 (1) 310-0234
Fax: +351 (1) 525843
Email: njm@inesc.pt

Program Committee

Antonio Porto (Portugal)
Lauiri Carlson (Finland)
Benjamin Kuipers (USA)
Luc Steels (Belgium)
Bernhard Nebel (Germany)
Luigia Aiello (Italy)
David Makinson (Germany)
Luis Moniz Pereira (Portugal)
Erik Sandewall (Sweden)
Luis Monteiro (Portugal)
Ernesto Costa (Portugal)
Manuela Veloso (USA)
Helder Coelho (Portugal)
Maria Cravo (Portugal)
Joao Martins (Portugal)
Miguel Filgueiras (Portugal)
John Self (UK)
Yoav Shoham (USA)
Jose Carmo (Portugal)
Yves Kodratoff (France)

Deadlines

Papers due: March 20, 1995
Author notification: May 15, 1995
Papers returned: June 12, 1995

Submission & Inquiries Address

EPIA95, INESC, Apartado 13069, 1000 Lisboa, Portugal
Voice: +351 (1) 310-0325,
Fax: +351 (1) 525843,
Email: epia95@inesc.pt

THE MINISTRY OF SCIENCE AND TECHNOLOGY OF THE REPUBLIC OF SLOVENIA

The Ministry of Science and Technology also includes the Standards and Metrology Institute of the Republic of Slovenia, and the Industrial Property Protection Office of the Republic of Slovenia.

Scientific Research and Development Potential

The statistical data for 1991 showed that there were 230 research and development institutions, organizations or organizational units in Slovenia, of which 73 were independent, 32 were at the universities, and 23 at medical institutions. The remainder were for the most part departments in industry. Altogether, they employed 13,000 people, of whom 5500 were researchers and 4900 expert or technical staff.

In the past 10 years, the number of researchers has almost doubled: the number of Ph.D. graduates increased from 1100 to 1484, while the number of M.Sc.'s rose from 650 to 1121. The 'Young Researchers' (i.e. postgraduate students) programme 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 fields of research:

	Ph.D.	M.Sc.
Natural Sciences	315	217
Engineering-Technology	308	406
Medical Sciences	262	174
Agricultural Sciences	122	69
Social Sciences	278	187
Humanities	199	68
Total	1484	1121

Financing Research and Development

Statistical estimates indicate that US$ 260 million (1.7% of GNP) was spent on research and development in Slovenia in 1991. 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 crisis. This crisis 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 R&D spending: its share of GNP doubled from the mid-seventies to 0.86% in 1993.

Overall, public funds available for Research & Development are distributed in the following proportions: basic research (35%), applied research (20%), R&D infrastructure (facilities) (20%) and education (25%).

Research Planning

The Science and Technology Council of the Republic of Slovenia, considering initiatives and suggestions from researchers, research organizations, professional associations and government organizations, is preparing the draft of a national research program (NRP). This includes priority topics for the national research policy in basic and applied research, education of expert staff and equipping institutions with research facilities. The NRP also defines the mechanisms for accelerating scientific, technological and similar development in Slovenia. 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 parliament as a whole. 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 provides organizational support for the NRP, but it 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: International quality standards of groups and projects, relevance to social development, economic efficiency and rationality of the project. The Ministry finances research or co-finances development projects through public bidding and partly 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 cooperation 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.

In evaluating the programs and projects, and in deciding on financing, the Ministry works closely with expert organizations and Slovene and foreign experts. In doing this, it takes into consideration mainly the opinions of the research leaders and of expert councils consisting of national research coordinators and recognized experts.

The Ministry of Science and Technology of the Republic of Slovenia. Address: Slovenska c. 50, 61000 Ljubljana. Tel. +386 61 131 11 07, Fax +38 61 132 41 40.

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 1259 199, Fax.:+386 61 219 385
Tlx.:31 296 JOSTIN SI
E-mail: matjaz.gams@ijs.si
Contact person for the Park: Iztok Lesjak, M.Sc.
Public relations: Natalija Polenec

REVIEW REPORT

Basic Instructions

Informatica publishes scientific papers accepted by at least two referees outside the author's country. Each author should 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. 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. The names of the referees should not be revealed to the authors under any circumstances. The names of referees will appear in the Refereeing Board. Each paper bears the name of the editor who appointed the referees.

It is highly recommended that each referee writes **as many remarks as possible directly on the manuscript**, ranging from typing errors to global philosophical disagreements. The chosen editor will send the author copies with remarks, and if accepted also to the Contact Person with the accompanying completed Review Reports. The Executive Board will inform the author that the paper is accepted, meaning that it will be published in less than one year after receiving original figures on separate sheets and the text on an IBM PC DOS floppy disk or through e-mail – both in ASCII and the Informatica LaTeX format. Style and examples of papers can be obtained through e-mail from the Contact Person.

Opinions, news, calls for conferences, calls for papers, etc. should be sent directly to the Contact Person.

Date Sent:

Date to be Returned:

Name and Country of Referee:

Name of Editor:

Title:

Authors:

Additional Remarks:

All boxes should be filled with numbers 1-10 with 10 as the highest rated.

The final mark (recommendation) consists of two orthogonal assessments: scientific quality and readability. The readability mark is based on the estimated perception of average reader with faculty education in computer science and informatics. It consists of four subfields, representing if the article is interesting for large audience (interesting), if its scope and approach is enough general (generality), and presentation and language. Therefore, very specific articles with high scientific quality should have approximately similar recommendation as general articles about scientific and educational viewpoints related to computer science and informatics.

☐ SCIENTIFIC QUALITY

☐ Originality

☐ Significance

☐ Relevance

☐ Soundness

☐ Presentation

☐ READABILITY

☐ Interesting

☐ Generality

☐ Presentation

☐ Language

☐ FINAL RECOMMENDATION

☐ Highly recommended

☐ Accept without changes

☐ Accept with minor changes

☐ Accept with major changes

☐ Author should prepare a major revision

☐ Reject

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 the original figures on separate sheets and the text on an IBM PC DOS floppy disk or by e-mail – both in ASCII and the Informatica LaTeX format. Style (attached) 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).

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 (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: .....

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 18 Number 2 June 1994 ISSN 0350-5596

Informatica

An International Journal of Computing and Informatics

Contents: