Deutsches
Forschungszentrum
für Künstliche
Intelligenz GmbH

Document
D-94-12

Proceedings des Studentenprogramms der 18. Deutschen Jahrestagung für Künstliche Intelligenz KI-94

Arthur Sehn, Serge Autexier (Hrsg.)

September 1994

Deutsches Forschungszentrum für Künstliche Intelligenz GmbH

Postfach 20 80
67608 Kaiserslautern, FRG
Tel.: (+49 631) 205-3211/13
Fax: (+49 631) 205-3210

Stuhlsatzenhausweg 3
66123 Saarbrücken, FRG
Tel.: (+49 681) 302-5252
Fax: (+49 681) 302-5341

Deutsches Forschungszentrum für Künstliche Intelligenz

The German Research Center for Artificial Intelligence (Deutsches Forschungszentrum für Künstliche Intelligenz, DFKI) with sites in Kaiserslautern and Saarbrücken is a non-profit organization which was founded in 1988. The shareholder companies are Atlas Elektronik, Daimler-Benz, Fraunhofer Gesellschaft, GMD, IBM, Insiders, Mannesmann-Kienzle, SEMA Group, and Siemens. Research projects conducted at the DFKI are funded by the German Ministry for Research and Technology, by the shareholder companies, or by other industrial contracts.

The DFKI conducts application-oriented basic research in the field of artificial intelligence and other related subfields of computer science. The overall goal is to construct *systems with technical knowledge and common sense* which - by using AI methods - implement a problem solution for a selected application area. Currently, there are the following research areas at the DFKI:

- Intelligent Engineering Systems
- Intelligent User Interfaces
- Computer Linguistics
- Programming Systems
- Deduction and Multiagent Systems
- Document Analysis and Office Automation.

The DFKI strives at making its research results available to the scientific community. There exist many contacts to domestic and foreign research institutions, both in academy and industry. The DFKI hosts technology transfer workshops for shareholders and other interested groups in order to inform about the current state of research.

From its beginning, the DFKI has provided an attractive working environment for AI researchers from Germany and from all over the world. The goal is to have a staff of about 100 researchers at the end of the building-up phase.

Dr. Dr. D. Ruland
Director

Proceedings des Studentenprogramms der 18. Deutschen Jahrestagung für Künstliche Intelligenz KI-94

Arthur Sehn, Serge Autexier (Hrsg.)

DFKI-D-94-12

Diese Arbeit wurde finanziell unterstützt durch das Bundesministerium für Forschung und Technologie (FKZ ITW-9201).

Proceedings des Studentenprogramms der 18. Deutschen Jahrestagung für Künstliche Intelligenz

KI–94

herausgegeben von

Arthur Sehn
Fachbereich Informatik
Universität des Saarlandes
Postfach 151150
66041 Saarbrücken
acsehn@cs.uni-sb.de

Serge Autexier
DFKI
Stuhlsatzenhausweg 3
66123 Saarbrücken
serge@dfki.uni-sb.de

21. – 22. September, 1994

Inhaltsverzeichnis

Vorwort

Das Gebiet der Künstlichen Intelligenz (KI) hat in den letzten Jahrzehnten einen großen Aufschwung erfahren. Das ursprünglich kleine Gebiet der KI hat sich in viele Teilgebiete aufgefächert, wie zum Beispiel:

- Computerlinguistik,
- Robotik,
- Computersehen,
- Wissensrepräsentation,
- Automatisches Beweisen,
- Planen,
- Expertensysteme.

Wegen der Vielfalt der Teilgebiete und deren Größe, ist es auch für den guten wissenschaftlichen Nachwuchs schwierig, sich zu artikulieren und einen geeigneten Rahmen für die Präsentation eigener Arbeiten auf dem Gebiete der KI zu finden. Deshalb findet im Rahmen der KI-94 am Mittwoch, den 21. September 1994 und Donnerstag, den 22. September 1994 erstmalig ein Studentenprogramm statt. Dieses ist bunt gefächert, und zeigt, wie die KI-94 insgesamt, das breite Spektrum der KI. Es werden einige Beiträge zu den vorher genannten Teilgebieten der KI vorgestellt.

Im ersten Teil des Studentenprogramms, am Mittwoch, den 21. September 1994, präsentieren fünf Studenten ihre aktuellen Arbeiten auf ihrem jeweiligen Teilgebiet der KI. Zur Eröffnung stellt Smailbegovic Fethula in seinem Vortrag "Physical information - the way to intelligent machine?" einige Gedanken über den Zusammenhang zwischen Information und Intelligenz vor. Es folgt ein Vortrag mit dem Thema "Resolving conflicts in distributed legal reasoning" von Stefanie Brüninghaus über ein Expertensystem mit Anwendung in der Rechtswissenschaft. Aus dem Bereich der Computerlinguistik hält Sascha Brawer einen Vortrag über "Mechanismen einer kontextfreien Grammatik für das Deutsche". Danach schließt Katerina Marinagi einen Beitrag über "TimePlan: Towards a time-based domain-independent planning system", einem zeitbasierten Planungssystem, an. Abgerundet wird dieser kurze Ausflug in die unterschiedlichen Anwendungsbereiche der KI durch Guido Lindner mit dem Thema "logikbasiertes Lernen in relationalen Datenbanken", hierbei wird die Anwendung eines logikorientierten Lernverfahrens auf relationale Datenbanken aufgezeigt.

Arthur Sehn und Serge Autexier

Physical information
The way to intelligent machine?

Fethulah Smailbegovic
Michaelplatz 4
53177 Bonn

April 1994

The space around us is everything that exists, ever existed and will ever exist. Even an innocent thought of a nature or of a space, make us excited, similar to some kind of remembrance an falling from great altitude. We are becoming aware that we are approaching to the greatest secrets of all secrets.

Lost somewhere between eternity and infinity, there is a man, whose attempts fron view of all space look insignifiant, even worthless. In the past thousand years man has come to incredible conclusions and discoveries about space that surrounds him. because he understands, that his future depends of what he knows about space. What is a really human being? That question lays in a basis of Artificial Intelligence. And today, AI becomes a significant part of our life and in our society, because, filfilling its task could bring our civilisation big step forward. Today AI loses its secrecy and becomes our great hope. But, in which direction we should try to find "lost treasure of AI"? Considering ideas of digital physics by Edward Fredkin from M.I.T. and experiments conducted by Benveniste, purpose of this paper is to show, that there is an area still unexplored, which in my opinion could hide answers on our questions. The roots of this "area" are in theory of information by C.E. Shannon, form which I observed troubles in definition, representation and influence of information. The basic problem is, that we can't find answer on question, on objectivity and subjectivity of information; Does the system which receives "information" can say, that the system which gives information exist? At this point, I am questioning the reality of entrance information. To this point of view, are connected idea of digital physics by Edward Fredkin, in which information builds matery and everything we know in a space around us, and contraverse experiments conducted by J. Benveniste (French immunologist), in which one matery reacts with other (THE CASE OF GHOST MOLECULES), that are theoretically "does not exist"! At this point, I am defining terms like, spectrum and physical information, trying to describe and to explain Benveniste's experiments and find implementation of Fredkin's idea.

Spectrum shows us, how many entrance information con one system receive in order

to react. Why spectrum? Intention is, with this to term to try "isolate" as much as possible smaller part of entrance information; and after that to explain it and determinate connection of entrance information wand system's respond on that (intelligent behaviour).

Knowing that system con conclude about entrance information that substance consist of informations, and knowing changeable nature of entrance information, only conclusion which strike me is that there is subjectivity of physical information. Does information builds substance we know? Yes, by this conclusions.

Abilities of physical information we don't know yet, but considering the great influence it may have on us, I would call it alpha (basic) key (it could open new horizonts), αk. That means that αk builds concepts we see, like substance, energy, elementary particles, force ...

If we implement algorithm approach instead of mathematical one, then Edvard Fredkin as only conclusion, is information as an elementary part of our world. For most of the scintists today, information is only one kind of energy, but by Fredkin information is more basical than an energy and substace.

αk builds connections and make structures of αk-s, what represents physical informations which have some meaning for some systems. We don't know yet features and activities of αk, but they are primery target of future researches, and especially its mathematical model. If we prove existence of αk we could, on the basis of entrance and outgoing informations, to predict functional connection between these signals. The prove that something exists here are Feinaman's diagramms of interactions between two electrons. They were not in any kind of contact but they reacted!

Can we prove existence of αk? It is very hard to find answer on this question. With todays methods of physics, with accelerators, is very hard to do it. Why? Because it is impossible to isolate αk alone, while there are reactins with equipment which gives us result. But theoretical way, with its proves in experiments, can give us an answer of existence of αk. It's not appropriate moment, at this stage, to speak about precise mathematical formulas. I think, we should accept method of global view by Pascal. The basic for the further work would give a proper computer support, which would consist of graphiv simulation of some parts of modell and executing some algorithms, for example; algorithm which would, on the basis of today knowledge about elementary particles and laws of physics, try to predict what is the next elementary particle in array of all known elementary particles. Purpose of this algorithm is to go deep into structure of our substance, and to try to find and follow processes, like "conversion", structure forming, influence and dealing with outside changes. Todays computers give us good basics for this. Artificial Intelligence, as a science, could bring great changes to our civilization, and bring us at the edge of new and better era. In order to reach intelligent machine, we have to solve great number of problems on our way to it. Problem of intelligent machine, of the whole intelligence, is dynamical, and we have to try to look "wider" the problem, to understand suuroundings and

influences on primary target of our research. Whatever truth will be, wether this field hide our answer or not, only breaking of antrophocentrical view our world can bring new areas of opportunities to us.

I hope I managed to explain new possibilities, which can bring field of physical information. Even a thought what physical information and Artificial Intelligence can bring us worth our devotion to finding solution of problem, intelligence, whatever price it would be.

Is it realistic to expect success? Yes. Technical progress today bring us more and more opportunities, in order to make survival for our civilisation secured.

Resolving Conflicts in Distributed Legal Reasoning

A Preliminary Report

Stefanie Brüninghaus
Lehrstuhl für Praktische Informatik I
Universität Mannheim
D - 68131 Mannheim, Germany

Phone: + 49 (621) 292 - 1196
Fax: + 49 (621) 292 - 5297
steffi@pi1.informatik.uni-mannheim.de

Intelligent Systems Program
University of Pittsburgh
Pittsburgh, PA 15260, USA
steffi@isp.pitt.edu

Abstract

This paper introduces DANIEL, *a* **D**istributed **A**rchitecture for the k**N**owledge-based **I**ntegration *of* **E**xpert *Systems in* **L**aw. *In this architecture, the legal sources are mapped to separate problem solvers which follow the appropriate paradigm – case-based or rule-based and are coupled via a blackboard. The integration of the problem solvers cannot be achieved by simply interleaving the respective reasoning chains or even "merging" their solutions without further analysis, since their knowledge widely overlaps and in addition often allows contradictory interpretations. In the proposed approach, the problem solvers work autonomously and concurrently, each of them derives an independent local solution. These are then assessed by an explicit coordination component, which on th eone hand applies legal meta-knowledge to handle conflicting interpretations. Additionally, a novel redundancy concept is introduced, which allows to explicitly represent the interrelationships between the reasoners and to handle conflicts efficiently.*

1 Motivation

The prospects of AI for legal reasoning have been recognized for more than 20 years [13]. In spite of several distinguished approaches [2, 5, 18], which gave raise to good hopes, still most of the systems are in a rather experimental state and the main problems are not satisfactorily solved. The principal problems can be identified as

- the lack of a *deep model* of legal reasoning,
- the existence of *inherently ill-defined* predicates,
- the common use of *open-textured* concepts, and
- the *frequent changes* in legislation and case law.

In this paper, a hybrid approach to legal reasoning will be discussed. It is based on the observation, that the bottleneck in modeling legal reasoning is not knowledge acquisition like in most other domains - the legal sources are all written down in a rather formalized, detailed way, numerous commentaries and monographs are easily accessible - but rather finding an appropriate way of reasoning with this knowledge.

The remainder of this article is organized as follows: Section 2 gives an overview over the proposed architecture, the main elements and the underlying design decisions. In section 3, the concurrent application of the reasoners will be thoroughly discussed and evidence will be provided for the usefulness of this decision. Then other approaches to legal or hybrid reasoning are presented and contrasted to DANIEL. Section 4 is dedicated to the main problem of this architecture, the conflict handling. It is argued that conflicts are a natural phenomenon in legal reasoning. In order to cope with these conflicts, their possible reasons and the different types structured in a novel redundancy concept. Section 5 summarizes the most important characteristics of DANIEL. The article concludes with a brief overview about the academic background and future prospects of this work.

2 Proposed Architecture

2.1 Legal Problem Solvers

The most important legal sources are case law and the statutes; in some fields, like tax law, there are also administrative regulations. For a comprehensive model of legal reasoning, each of these sources has to be considered, since none of them can be substituted by the others.

From an AI perspective, the legal sources can be characterized as follows:

- The statutes consist of a body of abstract rules intended to be applicable to a indefinite number of concrete cases of a certain type. They are therefore phrased in an abstract way and need to be interpreted when being applied to a concrete situation.
- Case law are the decisions of courts in previous cases. Their binding force is restricted to the dispute for which they were decided. However, since a judge will presumably decide the same way in a similar situation (defining this similarity is a non-trivial problem [2]), in particular the decisions of higher courts have a strong prejudicial effect. This phenomenon has been discussed controversely in jurisprudence; nevertheless it undeniably strongly influences legal interpretation.
- In particular in tax law, the administrative (resp. revenue) regulations play an important role in everyday legal practice. They embody an official interpretation of the statutes, mainly applicable in *standard situations*. The revenue regulations are enacted by the government, and not by the legislator, thus they do only bind the revenue offices and not the taxpayers.

Apparently, the legal sources require different representation paradigms, have distinct binding forces, and make use of dissimilar inference mechanisms. Therefore, each legal source has to be represented separately, applying the most suitable paradigm, as comprehensively discussed in [18].

From the characteristics of the legal sources, the proper representation can be easily derived: Case law is best represented in a case base and processed by a case-based reasoner, while the statutes are naturally mapped on a rule base. The administrative regulations are usually phrased in the form of rather general rules and therefore should be represented by (possibly annotated) rules.

2.2 Integration of the Problem Solvers

In order to bring together the problem solvers, they are coupled via a blackboard, which offers several advantages:

- Data can be smoothly exchanged via a blackboard, no particular exchange mechanisms have to be developed.
- Often, the statutes have to be applied recursively, which can be efficiently mapped on a hierarchical blackboard structure.

- For the flexibility of the blackboard approach, the architecture can be easily modified and extended.

2.3 Coordination Knowledge

The reasoning is controlled by a superior *coordination component*. Its knowledge can be either domain-dependent (i.e., knowledge from the particular application domain, the law) or domain-independent (i.e., knowledge that is applicable in every domain and only related to the characteristics of the problem solvers):

Domain-Dependent Legal meta-knowledge is on a level above factual knowledge (from statutes and cases). E.g., jurisprudence developed schemata and heuristics for solving cases. Legal argumentation has been an issue for several years, and rather comprehensive theories exist [1]. Moreover, there is a well-defined binding force assigned to the legal sources and rules for the applicability of norms do exist, like the well-known "lex specialis" rule.

Domain-independent This type of knowledge will be described in the light of one issue: Each of the reasoners has a particular problem solving behavior. Therefore a reasoner's solution can be better estimated when the respective problem solving process is recorded and accessible. E.g., when a case-based reasoner has retrieved a very similar case, then the confidence in this solution will be very high, and it will be assigned a high priority. On the other hand, when a rule-based reasoner can choose

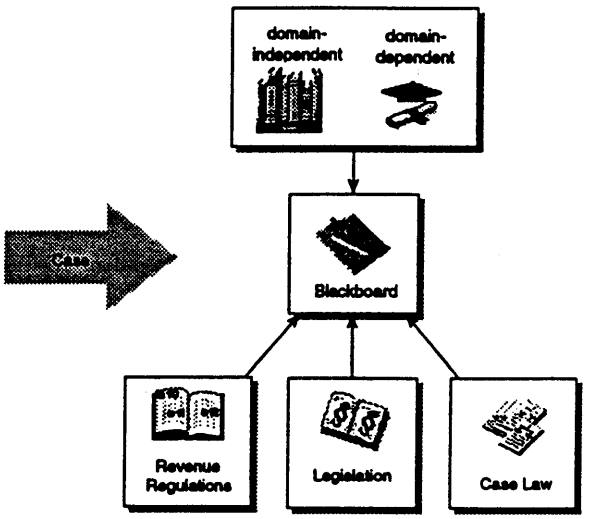


Figure 1: Overview of the DANIEL architecture

In order to illustrate the architecture, a brief, simplified example is given here: From the German Income Tax Law, it can be derived that built-in laundry facilities (i.e., not easily removable, since attached to the concrete) belong to a building, and that therefore the respective expenses must be manufacturing cost. The term manufacturing cost is not exactly defined in all details and therefore *open-textured*. In 1970, however, the German Supreme Tax Court decided that a washing machine screwed to the concrete is not included in the manufacturing cost of the building. Since the definition in the statutes is rather open-textured, while the cited case is an exact match, case law has to be applied.

3 Disacussion of the Control Strategy

There are numerous options for integrating case-based and rule-based co-reasoners [20]. This far the general idea has been that all reasoners contribute to a *common, overall* solution – a single thread of control is passed between the reasoners. In DANIEL, however, a different direction has been taken: the reasoners are supervised by a central module, they work concurrently and autonomously; each one of them derives a solution of its own. Then, their solutions are integrated by the central coordination component.

Intuitively, exploiting the flexibility of the blackboard paradigm by an opportunistic control strategy seems to be most advantageous. Moreover, the proposed control

does undoubtedly not prevent conflicts - it rather provokes them. For the following reasons, however, the proposed architecture is mandatory:

- The problem solvers can be assumed to have enough knowledge to be able to find at least a draft solution.
- The reasoners use different internal representations, and sometimes a term used by one reasoner does not have exactly the same meaning for the other (e.g., in the context of income taxation the terms business implies *profit* as a goal, while in the context of sales taxation, only *turnover* is required).
- The legal sources have different binding forces (i.e., case law has only a prejudicial force, legislation is generally valid and the revenue regulations only bind the administration).
- The partial results of the reasoners are not directly interchangeable - since non-monotonic reasoning is typical in law, and moreover, CBR has a non-monotonic character. Generally speaking, the reasoning process of CBR and RBR is basically different.
- A single state of affairs can be interpreted in different ways applying the legal sources, since the courts and the legislative instances are constitutionally independent forces, which are not mutually obliged.
- Both modes of reasoning have certain limitations and do not always work absolutely accurate. When the reasoners are interleaved, mistakes are forwarded from one reasoner to the other. By their concurrent application, the reasoners can provide mutual control.
- The knowledge of the reasoners is not independent from each other: Case law is usually concerned with the application of the currently valid legislation, and modifications of legislation are often influenced by case law.
- Since case law and legislation (notwithstanding the revenue regulations!) often refer to each other, identical knowledge may be applied multiple times when the reasoners are interleaved.

For these reasons, which do, nota bene, not exclusively apply in the legal domain, the reasoners in DANIEL are not interleaved, as mentioned above. Their inference chains are strictly separated, and no information is exchanged, in order to keep the respective reasoning "pure and unpolluted".

4 Related Work

Normative conflicts in legal reasoning are investigated by a number of researchers, among others [19, 17]. However, these approaches do only focus on the handling of conflicts in statutory interpretation. In legal CBR, the problem of selecting the most appropriate case form a set of retrieved, possibly conflicting cases, is usually not conflict resolution but rather a matter of similarity. Conflicts among case law and statutes have not been investigated this far.

In the AI and Law field, only few approaches to cooperative distributed problem solving (CDPS) exist. Doubtlessly the most sophisticated among these is Skalak's and Rissland's CABARET [18], which is a domain-independent hybrid reasoning shell developed with GBB. In CABARET, a case-based and a rule-based reasoner are coupled via a blackboard. The reasoners are interleaved by central component, which embodies domain-independent control heuristics. It is assumed, that the reasoners complement each other, and consequently, there is no explicit conflict resolution.

Most other systems use cases when rules run out or vice versa, like [22, 15]. In the Dutch system PROLEXS [23], rules and cases are applied in a fixed order, which is determined by situation-specific heuristics.

Another, however "illegal", system concerned with the integration of CBR and RBR is Golding's ANAPRON, which has been developed for the problem of pronouncing names. In ANAPRON, cases are used to discard respectively confirm the result of the rule-based inference process. The knowledge sources have to independent from each other in order apply this model - which is definitely not true in the legal domain.

In distributed artificial intelligence (DAI), negotiation is broadly applied to resolve conflicts among agents, see, e.g., [21, 16]. All these approaches, however are not directly applicable in the legal domain. In law, a certain predicate is either true or false, a concrete state of affairs is either subsumed by a legal rule or not, which is not negotiable. Therefore, in this approach, a strategy for finding the most probable solution among the possible interpretation is developed.

5 Redundancy Concept

A wicked side effect of the concurrent application of the reasoners is that conflicts are much more likely to occur - more than one (partial) solution is derived. Coordination

is done afterwards, and nothing is done to prevent conflicts. However, in this approach conflicts are explicitly modeled and used for increasing the accuracy of the system. For this purpose, a redundancy concept has been developed, which considers the overlapping expertise of the legal sources.

5.1 Relationship of the Knowledge Sources

When multiple knowledge sources contain knowledge applicable to a certain fact situation, this is an instance of *redundancy*. Here, redundancy is defined by the applicability of more than one problem solver in a certain situation, which means that they have overlapping areas of expertise. The goal of this research will be to find out how to deal with redundant legal sources and to describe a mechanism for exploiting redundancy in the coordination.

This far, four possible categories of interdependencies between the knowledge in a case base and the knowledge in a rule base have been identified, as illustrated by Fig. 2:

Complementary In this constellation the case base contains knowledge the rule base does not have and vice versa. The knowledge bases complement each other.

Inconsistent The knowledge bases contain contradicting pieces of knowledge. This conflict must be resolved on a higher knowledge-level. In DANIEL, the coordination component is responsible for this.

Identical The application of the knowledge bases to a specific fact situation leads to the same result. In this case, one knowledge base seconds the results of the other.

Mutual Influence The relationship between case law and legislation has not yet been represented properly. In the legal domain, a case is decided in the context of the current legislation, and legislation is often changed as a reaction to case law.

Since these categories depend on the situation in which the legal sources are applied, their pre-evaluation seems to be an unsolvable problem. In the following section, a preliminary taxonomy of the local solutions is presented, which is based on the different types of interrelationships between the knowledge sources.

5.2 Taxonomy of Local Solutions

In the following, the local solutions of a case-based and a rule-based reasoner are set against each other and arranged in a taxonomy, which is given in Fig. 3.

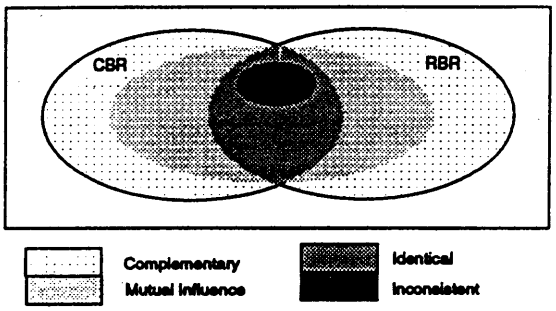


Figure 2: Overlapping of Knowledge Sources

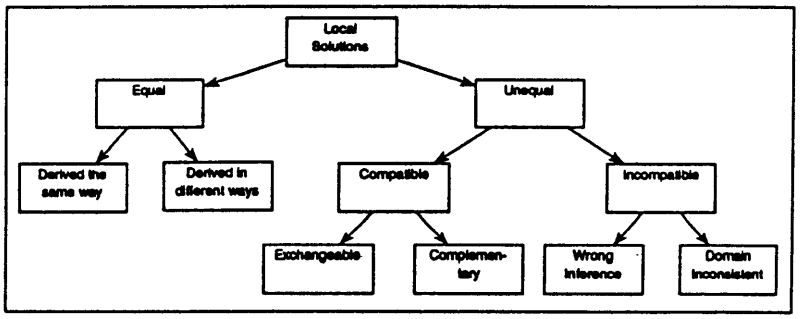


Figure 3: Taxonomy of local solutions.

Generally, the local solutions can be *equal* or *unequal*, which does *not necessarily* mean that they are contradictory. If they are equal, they can be derived either in the same way (following the same argumentation) or in different way. In the first case, the application of both knowledge sources was merely superfluous, while in the latter case the confidence in the solution is increased, because the correctness of the problem solvers is *proven*. Anyway, no further coordination is needed.

More effort has to be spent, however, if the local solutions are unequal. If they are *incompatible*, the conflict can not be resolved by the system itself. In order to achieve an assessment of the system status, two possible situations have to be distinguished. On the one hand, there can be inconsistencies caused by *inconsistent domain knowledge*, e.g., if a court explicitly contradicts a statute for constitutional reasons. Problems like this exceed the competence of a system and should cause a corresponding message to the user. In the following, it will be assumed that cases like this were eliminated in the knowledge acquisition phase.

On the other hand, the reasoners the reasoners do not always work accurately and

an improvement of the solution quality, but a *supervising function.*

The most interesting constellation occurs when both problem solvers produce non-identical local solutions, which are nevertheless consistent. In the first branch of this taxonomy *complementary local solutions* are depicted. This means that one of the problem solvers could not derive an accurate result for the lack of applicable knowledge. In the domain of taxation, sometimes open-textured terms do prevent the rule-based problem solver from working out an accurate solution, while the case-based reasoner can figure out an exact solution. Vice versa, a state of affairs that is certainly regulated by legislation will not have a proper case-based solution. In Fig. 2 this is referred to as *complementary knowledge.*

There are also situations where more than one reasoner can provide a valid solution. This is most likely to occur when rules set forth by the revenue regulations for simplifying the taxation procedure are involved. Often, lump sums are defined in order to remove the need to calculate exact numbers. In these situation both partial solutions are applicable and the most most favorable for the client can be chosen.

6 Conclusion

In this paper, an approach to the resolution of conflicts in an area where negotiation is not applicable has been introduced. Is based on the application of domain-dependent meta-knowledge as well as on the invention of a redundancy concept for explicitly modeling the overlapping expertise between the problem solvers.

For building expert systems, redundancy is desirable, as Buchanan points out [14]:

> ... One advantage of a modular representation of the domain knowledge is that it allows the system to explore multiple lines of reasoning ... Reasoning with uncertain or missing data, or with knowledge or data that is uncertain or incomplete, requires building redundancy into the reasoning to allow correct conclusions to be drawn in spite of these deficiencies ...

In DANIEL, redundancy is used to overcome the typical problems in legal reasoning, which are

- the lack of deep models for legal reasoning,
- the existence of inherently ill-defined predicates,
- and the frequent use of open-textured concepts [18].

Moreover, the inherent limitations of both reasoning modes, case-based and rule-based, can be minimized by their concurrent and autonomous application.

Since the problem of overlapping expertise and non-independent knowledge is not limited to the legal domain, this model can also be adapted and applied to other domains.

7 Background

This work is part my master's thesis, which I am currently preparing at the Chair for Applied Computer Science of Professor Dr. Franz Stetter, whose research focus is on Software Engineering. My supervisor is Dr. Michael Weiss, who graduated with a thesis on distributed AI for design, in which he developed HBBS, a hierarchical blackboard architecture.

After the completion of my Wirtschaftsinformatik degree at the University of Mannheim, I will start as a Ph.D. candidate in the Intelligent Systems Program at the University of Pittsburgh. This program was founded by Bruce G. Buchanan as a center for advanced education and research in artificial intelligence. There, I will continue my work in the field of AI and Law with Prof. Kevin D. Ashley [2, 4, 3].

I already presented my work in a several workshops and conferences [6, 11, 8, 9]. Recently, my student paper for the American National Conference on Artificial Intelligence (AAAI94) was accepted [10]. Furthermore, I could participate in workshops at conferences on object-orientation with my work in object-oriented modeling [12, 7].

References

[1] R. Alexy. *Theorie der juristischem Argumentation.* Suhrkamp, Frankfurt/Main, 1991. Theory of Legal Argument.

[2] K.D. Ashley. *Modeling Legal Argument.* MIT-Press, Cambridge, MA, 1990.

[3] K.D. Ashley and V. Aleven. Generating dialectical examples automatically. In *Proceedings of the Tenth National Conference on Artficial Intelligence*, pages 654 – 660, San Jose, CA, 1992.

[4] K.D. Ashley and V. Aleven. Using logic to reason with cases. In *Proceedings of the First European Workshop on Case-Based Reasoning*, pages 373–378, Kaiserslautern, 1992.

[5] K.L. Branting. Reasoning with portions of precedents. In *Proceedings of the Third International Conference on Artficial Intelligence and Law, Oxford*, pages 145 –, Oxford, 1991. Compares portions of precedents in order to solve new case.

[6] St. Brüninghaus. Modellierung der Rechtsanwendung in Steuerrecht. In L. Phillips and T.F. Gordon, editors, *Workshop of the Special Interest Group on Law and Computer Science at the German Workshop on Artificial Intelligence (GWAI)*, Bonn, 1992.

[7] St. Brüninghaus. An Approach to Object-Oriented Modeling of International Taxation. In F.J. Hauck and P. Steyert, editors, *Third European PhD Workshop at European Conference on Object-Oriented Programming (ECOOP)*, Kaiserslautern, 1993.

[8] St. Brüninghaus. An Architecture for the Coordination of Distributed Problem Solvers in Fiscal Law. In *Proceedings of the Conference in Celebration of the 25th Anniversary of the Istituto per la documentazione giuridica of the CNR*, pages 125 – 127, Florence, 1993.

[9] St. Brüninghaus. DANIEL - Eine verteilte Architektur für die Integration von Expertensystemen im Recht. In U. Hahn, editor, *Workshop der Fachgruppe Verteilte Künstliche Intelligenz*, 1994. forthcoming.

[10] St. Brüninghaus. DANIEL: Combining Case-Based and Rule-Based Reasoning in Law. In *Proceedings of the Twelfth National Conference on Artficial Intelligence*, Seattle, WA, 1994. AAAI, MIT Press. forthcoming Student-Paper.

[11] St. Brüninghaus and M. Weiss. A Coordination Architecture for Distributed Problem Solvers in Fiscal Law. In D. Jantezko and T.J. Schult, editors; *Proceedings of the Workshop on Case-Based Reasoning in Hybrid Systems*, Freiburg i.Br., 1993. Institut für Informatik und Gesellschaft.

[12] St. Brüninghaus and M.R. Weiss. Context. In P. Coad and B. Anderson, editors, *OOPSLA93 Abbendum to the Proceedings: Workshop on PATTERNS: Building Blocks for Object-Oriented Architectures*, Washington, DC, January 1994. ACM Press.

[13] B.G. Buchanan and T.E. Headrick. Some speculation about Artificial Intelligence and legal reasoning. *Stanford Law Review*, 23:40–62, 1970.

[14] B.G. Buchanan and R.G. Smith. Fundamentals of expert systems. In A. Barr, P.R. Cohen, and E.A. Feigenbaum, editors, *The Handbook of Artificial Intelligence, Vol. IV*, pages 149–192. Addison Wesley, Reading, MA, 1989.

[15] A.v.d.L. Gardner. *An Artificial Intelligence Appraoch to Legal Reasoning.* MIT Press, Cambridge, MA, 1987.

[16] M. Klein, editor. *Proceedings of the IJCAI-93 Workshop on Conflict Management*, Chambery, 1993.

[17] H. Prakken. A tool in modelling disagreement in law: preferring the most specific argument. In *Proceedings of the Third International Conference on Artficial Intelligence and Law, Oxford*, pages 165–174, 1991.

[18] E.L. Rissland and D.B. Skalak. CABARET: Rule Interpretation in a Hybrid Architecture. *International Journal on Man-Machine Studies*, 34(1):839–887, 1991. Also appeared as University of Massachusetts Technical Report COINS-TR 90-97.

[19] G. Sartor. *Artificial Intelligence and Law.* Complex. Tano, Oslo, January 1993.

[20] D. B. Skalak. Options for controlling mixed paradigm systems. In *Proceedings of the 1989 DARPA Case-Based Reasoning Workshop*, pages 318–323, Pensacola Beach, FL, 1989. Morgan Kaufman.

[21] E.P. Sycara. *Resolving Adversarial Conflicts: An Approach Integrating Case-Based and Analytical Methods.* PhD thesis, School of Information and Computer Science, Georgia Institute of Technology, Atlanta, GA, 1987. published as Technical Report GIT-ICS-87/26.

[22] G. Vossos, J. Zeleznikow, T. Dillon, and V. Vossos. An example of integrating legal case-based reasoning with object-oriented rule-based systems: IKBALS II. In *Proceedings of the Second International Conference on Artficial Intelligence and Law, Vancouver*, pages 31 – 41, Vancouver, 1989.

[23] R.F. Walker et al. PROLEXS: creating law and order in a heterogenous domain. *International Journal on Man-Machine Studies*, pages 35–67, 1991.

Mechanismen einer kontextfreien Grammatik für das Deutsche

Studentischer Beitrag zur 4. Deutschen Jahrestagung für Künstliche Intelligenz (KI-94)
Saarbrücken, 18. – 23. September 1994

Sascha Brawer · Stengelstraße 18 · D – 66117 Saarbrücken · e-Mail: brawer@coli.uni-sb.de

Zusammenfassung

Dieser Artikel stellt eine Grammatik für ein deutschsprachiges Text-to-Speech-System vor. Bei der Sprachsynthese ist eine kurze Verarbeitungszeit zwar von großer Bedeutung, die Praxis zeigt aber, daß man mit einer allzu oberflächlichen linguistischen Analyse kein hoch-

von geschriebener zu gesprochener Sprache mindestens eine syntaktische Verarbeitung vorzunehmen. Die Computerlinguistik arbeitet heute überwiegend mit Unifikationsgrammatiken, um natürliche Sprache syntaktisch zu verarbeiten. Um Rechenzeit einzusparen, beschränkte man sich beim hier besprochenen System allerdings auf die Unifikation von atomaren Termen; es wird (informell) gezeigt, daß dieser Verzicht unmittelbar zur Folge hat,

Ursprünglich lagen diese Grammatiken als Transitionsnetzwerke vor, die um Unifikation erweitert waren. Auf diese sogenannten »UTNs« (vgl. [Russi 1990]) wandte man ein an [Pereira/Warren 1980] angelehntes Verfahren an, um vollautomatisch Definit-Klausel-Grammatiken zu erhalten, wie sie auch zum Beispiel die Programmiersprache Prolog kennt. Obwohl DCGs sehr oft direkt in Prolog abgearbeitet werden, wird in SVOX aus Gründen der Effizienz ein in Modula-2 geschriebener Chart-Parser eingesetzt.
Meine Aufgabe war es nun einerseits, die Grammatiken zu verbessern, andererseits sollte die Syntaxanalyse beschleunigt werden. Die Änderungen am Parser sind jedoch nicht Gegenstand dieses Artikels: Im wesentlichen habe ich den Earley-Algorithmus implementiert und durch Techniken wie Überprüfung der First-/Follow-Relationen erweitert, die aus dem Compilerbau bekannt sind.
Es erwies sich als nötig, große Teile der Grammatik neu zu schreiben, wobei ich großes Gewicht auf die Verarbeitungszeit legte. Das Aufwendigste bei einer Unifikationsgrammatik ist das Unifizieren selber — es lohnt sich also, hier anzusetzen, bei dieser Operation, die unter Umständen pro Satz mehrere tausend Mal durchgeführt werden muß. Nun ist es vor allem das Unifizieren von *komplexen* Termen, das viel Zeit in Anspruch nimmt; um zwei Atome miteinander zu unifizieren, braucht es dagegen nicht viel mehr als einen simplen Vergleich.
Es liegt aus diesem Grund nahe, eine Grammatik zu schreiben, die sich mit atomarer Unifikation begnügt. Wie im nächsten Abschnitt gezeigt wird, kann eine solche Grammatik jedoch nur die Klasse der *kontextfreien Sprachen* verarbeiten. Kontextfreie Grammatiken sind nun allerdings in großen Kreisen der Computerlinguistik eher verpönt (vgl. zum Beispiel die Argumente gegen kontextfreie Grammatiken in [Haugeneder/Trost 1993]): Es heißt unter anderem, mit ihnen ließen sich komplexere linguistische Phänomene kaum beschreiben. Daß man mit kontextfreien Grammatiken durchaus einige Tiefe bei der linguistischen Analyse erreichen kann, soll der Rest dieses Artikels zeigen: Nach dem Einführen von zwei nützlichen Erweiterungen des Formalismus, die auf die Sprachklasse aber keinerlei Einfluß haben, wird erläutert, wie die SVOX-Satzgrammatik mit Koordination, Subkategorisierung und freier Wortstellung umgeht.
Es sei aber bereits an dieser Stelle angemerkt, daß im beschriebenen System keine semantische Verarbeitung stattfindet. In diesem Fall wäre eine »richtige« Unifikationsgrammatik, HPSG zum Beispiel, sicherlich ein geeigneteres Werkzeug als ein reiner DCG-Formalismus.
Weiterhin sei festgestellt, daß die nachfolgend vorgestellte Grammatik keinerlei theoretische Ansprüche erheben will: SVOX ist ein Programmpaket, das in der Praxis angewandt werden soll; es kann daher manche Frage einfach ignorieren, die man einer Grammatik-*Theorie* stellen müßte.

Warum ist die Grammatik kontextfrei?

Wie bereits erwähnt wurde, benutzt das System SVOX sowohl für die Darstellung des syntaktischen wie auch des morphologischen Wissens Grammatikregeln, die abgesehen von einer etwas anderen Syntax den Definit-Klausel-Grammatiken (DCGs) von Prolog entsprechen.
Mit Definiten Klauseln können durchaus auch kontext-*sensitive* Sprachen repräsentiert werden. Es sei hier lediglich auf [König/Seiffert 1989] verwiesen, wo sich auf Seite 112 eine

DCG für die kontextsensitive Sprache $a^n\ b^n\ c^n$ findet.[1] Obwohl der Parser von SVOX auch komplexe Terme miteinander unifizieren kann, also dieselbe Mächtigkeit wie der ursprüng-

Aus diesem Grund möchte man bestimmte Konstruktionen als wahrscheinlicher als andere einstufen. Die eine Erweiterung des Formalismus ist daher eine Bewertung jeder Regelanwendung in Form von »Strafpunkten«, die über die gesamte Ableitung hinweg aufsummiert werden. Die Ableitung mit der niedrigsten Punktzahl gilt als wahrscheinlichste Alternative und wird als einzige den nachfolgenden Komponenten übergeben.
Die zweite Erweiterung sind als »unsichtbar« markierte Regeln. Sie werden ganz normal verarbeitet, ihre Anwendung erscheint aber nicht im Baum, der an die anderen Module weitergegeben wird.
Nachfolgend werden einige konkrete Anwendungen dieser beiden Erweiterungen gezeigt, die für die Behandlung von Koordination, Subkategorisierung und freier Wortstellung hilfreich sind.

Pseudo-Operatoren

Das Einführen von echten Operatoren bzw. von Funktionen, wie es beispielsweise in HPSG möglich ist, hätte einen gewissen Aufwand bedeutet; insbesondere wäre es nötig geworden, den Modula-Quelltext des Parsers zu ändern, um neue Funktionen zu implementieren — was einerseits sehr unschön wäre und andererseits eine gewisse Vermischung der Wissensrepräsentation mit der Verarbeitung bedeuten würde. Durch die Möglichkeit, eine Regelanwendung nicht im Syntaxbaum erscheinen zu lassen, kann man jedoch »Pseudo-Operatoren« deklarativ einführen, welche zu einem gewissen Grad dieselbe Funktionalität bieten, ohne eine echte Erweiterung des Formalismus vornehmen zu müssen. Ich möchte das Prinzip nachfolgend am Beispiel des logischen »Oder« zeigen.
Für das Erzeugen der Satzmelodie ist es von grundlegender Bedeutung, zu wissen, ob der Satz eine Frage oder eine Aussage darstellt. Taucht irgendwo im Satz eine Konstituente auf, die durch bestimmte Wörter oder durch die Wortstellung auf eine Frage hindeutet, soll der gesamte Satz als Frage betrachtet werden. Die oberste Konstituente trägt daher ein Attribut »Frage«, dessen Wert von unten nach oben »vererbt« wird.[2] Dazu wird ein »Operator« benötigt, der das boolesche »Oder« berechnet. Diese »Funktion« kann folgendermaßen notiert werden:[3]

```
or(f, f, f).                    0 Strafpunkte · unsichtbar
or(t, _, t).                    0 Strafpunkte · unsichtbar
or(_, t, t).                    0 Strafpunkte · unsichtbar
```

Ein Beispiel für die Benutzung:

```
s(F) --> np(F1),
         vp(F2),
         or(F1,F2,F).           1 Strafpunkt · sichtbar
```

Neu ist eine solche deklarative Definition von Funktionen zwar nicht, nur wird sie eher auf dem Gebiet der Logischen Programmierung als beim Schreiben von Grammatiken benutzt. In einer »traditionellen« Definit-Klausel-Grammatik müßte der »Funktionsaufruf« entweder

[2] Es ist eigentlich nicht korrekt, bei einer Unifikationsgrammatik von »Vererbung« zu sprechen — Information wird beim Unifizieren nicht weitergegeben, sondern via *Structure Sharing* geteilt. Dennoch verdeutlicht dieser Sprachgebrauch vielleicht die Funktionsweise der Grammatik.

[3] Die Grammatiken des SVOX-Systems verwenden eine etwas andere Notation; ich habe sie hier zur besseren Verständlichkeit an den DCG-Formalismus von Prolog angeglichen.

durch partielle Evaluation direkt in die Regel integriert werden, welche den Operator benutzt (im Beispiel würde das zu drei S-Regeln führen), oder man ruft innerhalb der DCG Prolog-Klauseln auf, was wiederum eine Vermischung von Wissensrepräsentation und -verarbeitung wäre. Zudem werden die Grammatiken ja der Effizienz zuliebe nicht von Prolog abgearbeitet; die Beschränkung auf reine Grammatikregeln ohne prozedurale Elemente drängt sich daher schon aus diesem Grund auf.
Der Parser des SVOX-Systems entfernt die als »unsichtbar« markierten Regelanwendungen nachträglich aus der Ableitung — eine echte Erweiterung ist das nicht, aber recht nützlicher *syntactic sugar*.
Eine interessante Anwendung dieser »unsichtbaren Regeln« ist die sogenannte schwache Unifikation, von der die SVOX-Grammatik ausgiebig Gebrauch macht.

Schwache Unifikation

Auf die eben beschriebene Weise läßt sich zum Beispiel ein »Operator« definieren, dessen Anwendung nicht bestraft wird, wenn seine Argumente unifizierbar sind; andernfalls »kostet« die Regelanwendung 50 Strafpunkte.

```
WeakUnif50(A, A).                    0 Strafpunkte · unsichtbar
WeakUnif50(_, _).                   50 Strafpunkte · unsichtbar
```

Zugegebenermaßen ist es notwendig, innerhalb des Parsers gewisse Vorkehrungen zu treffen, damit ein solcher »Mißbrauch des Formalismus« nicht die Effizienz mindert.[4]

Koordination

Zum eigentlichen Thema dieses Artikels: Wie behandelt man bestimmte linguistische Phänomene, wenn man sich auf atomare Unifikation und somit auf kontextfreie Grammatiken beschränkt? Es sei an dieser Stelle nochmals ausdrücklich betont, daß die Problemstellung keine linguistisch saubere Behandlung von sprachlichen Phänomenen erforderte: So ist die mit Strafpunkten arbeitende Behandlung der Koordination linguistisch eher abwegig, sie liefert aber mit relativ niedrigem Rechenaufwand Resultate, die für die Sprachsynthese durchaus zu gebrauchen sind.
In diesem Abschnitt soll gezeigt werden, wie sich das doch eher komplexe Phänomen der Koordination einigermaßen zufriedenstellend durch eine kontextfreie DCG implementieren läßt. Als erstes seien einige sprachliche Beispiele aufgeführt:

A		Ich sah Anna.
A, B *Konjunktion* C		Ich sah Anna, Bruno und Cäcilie.
A, B	??	Ich sah Anna, Bruno.
A *Konjunktion* B, C	??	Ich sah Anna und Bruno, Cäcilie.
A *Konjunktion* B *Konjunktion* C		Ich sah Anna sowie Bruno und Cäcilie.
Konjunktion$_1$ A *Konjunktion*$_2$ B		Ich sah weder Anna noch Bruno.

[4] Für die Analyse der Wörter verwendet SVOX eine Modifikation des Earley-Algorithmus, für die Satzanalyse einen Bottom-Up-Chart-Parser. Lassen sich die beiden Atome im Beispiel unifizieren, werden *sowohl die gut als auch die schlecht bewertete* Kante in die Chart eingefügt. Der Parser müßte vor dem Einfügen einer leeren, inaktiven Kante überprüfen, ob eine ähnliche Kante nicht schon vorhanden ist, und gegebenenfalls lediglich die Zahl der »Strafpunkte« minimieren. Allerdings wurde dies bisher noch nicht implementiert.

Es ist zwar nicht der Fall, daß Konstruktionen wie »Ich sah Anna, Bruno« völlig ungrammatisch wären, sie sind bei Adjektiven sogar sehr häufig: »Das große, schöne Haus«. Sie sollen daher nicht völlig ausgeschlossen, sondern lediglich als eher schlecht bewertet werden, wobei die Zahl der Strafpunkte je nach Wortart variiert.
Betrachten wir nun die Grammatik für eine allgemeine Konstituente X, die koordinierbar sei. Gezeigt werden nur diejenigen Features, die für die Koordination von Bedeutung sind; es ist klar, daß zum Beispiel Agreement ebenfalls behandelt werden muß, oder daß Vorkehrungen zu treffen sind, damit eine Konstruktion wie »weder Anna als auch Bruno« ausge-

Der Wert des ersten Attributs ist gleich `t`, wenn die Konstruktion eine Konjunktion enthält; der Wert des zweiten zeigt, ob es sich überhaupt um eine koordinierte Konstruktion han-

mischt, wird hier große Schwierigkeiten haben. Dazu zählen die im SVOX-System verwendeten kontextfreien Grammatiken ebenso wie zum Beispiel LFG. Dagegen ist in HPSG die Trennung der Regeln für *immediate dominance* und *linear precedence* formalisiert (vgl. [Pollard/Sag 1987], insbesondere Kapitel 7).
Es gibt nun mehrere Möglichkeiten, das Problem auch mit einer Grammatik anzugehen, die *immediate dominance* und *linear precedence* nicht trennt. In Frage kommen beispielsweise besondere Regeltypen, deren rechte Seiten beliebig permutierbar sind; für LFG wurde ein Permutationsoperator vorgeschlagen.
Im folgenden sei erläutert, wie die hier beschriebene Grammatik mit Subkategorisierung allgemein und dem Phänomen der freien Wortstellung im Besondern umgeht. Als Beispiele dienen Verben, da die SVOX-Grammatik zur Zeit weder subkategorisierende Nomina (die Tatsache, daß ...) noch Adjektive (begierig, ihn zu sehen) kennt. Es wäre allerdings kein sehr großer Aufwand, die Grammatik entsprechend zu erweitern.
Verben subkategorisieren unter anderem nach folgenden Kategorien:[5]

Ich gehe.	Subjekt
Mich friert. Ich nehme *den Schal.*	Akkusativobjekt
Ich kaufte *der Marktfrau* einen Apfel ab.	Dativobjekt
Ich scheine *immer alles zu vergessen.*	»Zu«-Satz
Ich will *einen Salat essen.*	Infinitiv-Konstruktion
Ich verspreche, *daß ich gehe.*	»Daß«-Satz

Die Information, nach welchen Konstituenten ein Verb subkategorisiert, ist im Lexikon festgehalten. Üblicherweise wird dazu für das Deutsche wegen der freien Wortstellung eine Subkategorisierungs-Menge benutzt, bzw. (falls der Formalismus keine Mengen kennt) eine Subkategorisierungs-Liste, die als Simulation einer Menge dient. Nun wurde bereits zu Beginn dieses Artikels erläutert, daß man beim SVOX-System aus Effizienzgründen auf komplexe Features verzichten wollte. Stattdessen tragen Verben für jedes potentielle Element der Subkategorisierungs-Liste ein Attribut, dessen Wert anzeigt, ob das Verb nach der entsprechenden Konstituenten subkategorisiert.
Mit diesem Mechanismus werden (endliche) Mengen simuliert: Jedes Objekt kann höchstens einmal Element der Menge sein, und auf die einzelnen Elemente kann direkt zugegriffen werden, ohne eine Liste durchgehen zu müssen.
Fakultative Komplemente lassen sich auf diese Weise sehr effizient durch Unterspezifikation darstellen. Anstelle von zwei Einträgen für »geben« — einmal mit und einmal ohne Dativobjekt (»ich gab alles« vs. »ich gab dir alles«) — enthält das Lexikon nur einen Eintrag, bei dem der Wert des entsprechenden Attributs nicht spezifiziert ist. Konkret heißt dies, daß man in solchen Fällen die anonyme Variable angibt, die sowohl mit `t` (Komplement muß stehen) als auch mit `f` (Komplement darf nicht stehen) unifiziert werden kann.
Würde SVOX keine ausgefeilte Morphologie betreiben, sähen Lexikoneinträge ungefähr folgendermaßen aus — die Features stehen für Subjekt, Akkusativ-, Dativ-, Genetivobjekt, »zu«-Satz, Infinitivkonstruktion und »daß«-Satz:

```
verb(t,f,_,f,f,f,f,...) --> [geben].     »Ich gab (dir) alles«
verb(t,f,f,f,f,t,f,...) --> [wollen].    »Ich will gehen«
verb(t,f,f,f,f,f,t,...) --> [wollen].    »Ich will, daß Du gehst«
verb(t,f,_,f,t,f,f,...) --> [scheinen].  »Anna scheint (mir) zu wachsen«
```

[5]Diese Liste ist nicht vollständig; die besprochene Grammatik kennt noch weitere Kategorien wie zum Beispiel »Prädikative« — »Bruno ist groß«.

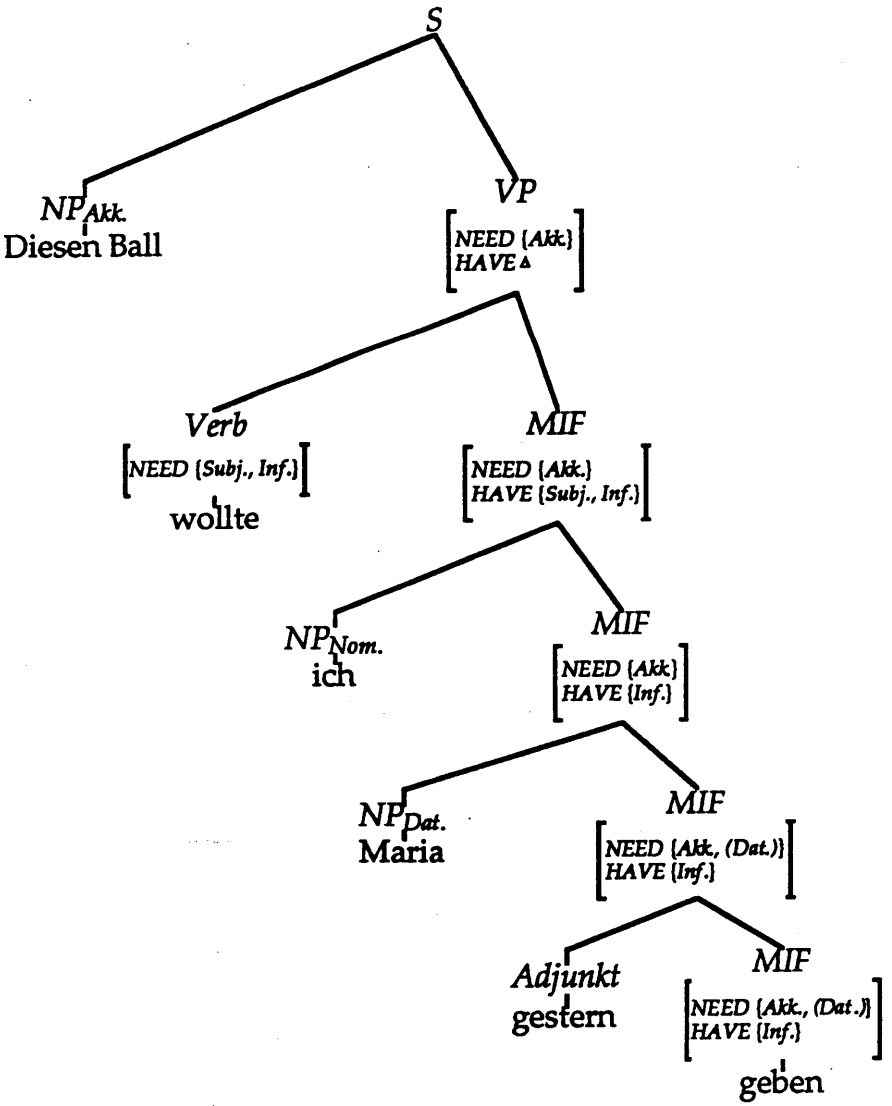
Die Wortgrammatik sorgt dafür, daß Verben im Infinitiv nicht nach einem Subjekt, sondern nur nach den anderen möglichen Komplementen subkategorisieren können. Dadurch ist es nicht notwendig, Konstruktionen mit Kontrollverben (Raising-/Equi-Verben) gesondert zu behandeln.

Nun ist bekannt, welche Kategorien vorhanden sein müssen, damit ein Satz vollständig ist; es bleibt die Frage, mit welchem Mechanismus die einzelnen Komplemente abgebunden werden. Betrachten wir dazu zunächst, an welchen Stellen im Satz Komplemente auftreten:

Gestern	hat Maria vor Zeugen	behauptet, ein Einhorn gesehen zu haben.
Vorfeld	**Mittelfeld**	**Nachfeld**
Genau eine Konstituente	*Null bis mehrere Konstituenten in beliebiger Reihenfolge*	

Ein Aussagesatz besteht aus dem Vorfeld (mit genau einer Konstituente), dem finiten Verb, dem Mittelfeld und eventuell aus einem Partizip; die Behandlung des Nachfelds geschieht mittels besonderer Mechanismen, die hier nicht erläutert werden.

Welche Komplemente vorhanden sein müssen, hängt von der Subkategorisierung des Verbs ab; die SVOX-Grammatik trennt daher erst das Vorfeld ab und unterteilt dann den Rest in die drei Teile finites Verb, Mittelfeld und Partizip. Die Hauptaufgabe der Satzgrammatik ist



Grafik: Subkategorisierung wird in der beschriebenen Grammatik ähnlich wie die Fernabhängigkeiten in HPSG behandelt, allerdings ohne leere Kategorien zu benötigen.

Die Nominativ-NP »ich« stellt ein Subjekt dar; die NEED-Menge von »Maria gestern geben« zeigt aber nicht an, daß ein Subjekt fehlen würde. Somit wird das Subjekt zur HAVE-Menge hinzugefügt.

Auf diese Weise wird das Mittelfeld schrittweise rekursiv aufgebaut. Die benötigten Grammatikregeln werden am Beispiel des Dativobjekts erläutert; die meisten anderen möglichen

Da Mengen durch Attribute mit booleschem Wertebereich simuliert werden, lassen sich diese beiden Regeln mit Hilfe einer Negation zu einer einzigen zusammenfassen: Das neue Mittelfeld hat genau dann einen Dativ überzählig, wenn das alte keinen benötigte. Somit enthält die Grammatik für jedes der möglichen Komplemente eine einzige Regel. Es wird hierbei angenommen, daß kein Dativ auf ein Mittelfeld trifft, das bereits einen nicht abgebundenen, »überzähligen« Dativ enthält. Dies ist eigentlich nicht korrekt, hat aber bisher zu keinen falschen Analysen geführt. Der Grund dafür ist, daß selten zwei gleiche Konstituenten nicht-lokal abgebunden werden.
Ist das Mittelfeld vollständig auf die beschriebene Art aufgebaut worden, wird es mit dem Verb zur »klassischen« Verbalphrase verknüpft. Dies bewerkstelligt ein besonderer Abbinde-Operator:

Verb	Mittelfeld		Verbalphrase	
NEED	NEED	HAVE	NEED	HAVE
t	f	t	f	f
t	f	f	t	f
f	t	f	t	f
f	f	t	f	t
f	f	f	f	f

Benötigt ein Verb beispielsweise einen Dativ und»bietet« das Mittelfeld einen Dativ an, hat die Kombination von beiden weder einen Dativ »zuviel« noch einen »zuwenig«. Die übrigen Zeilen können analog gelesen werden. Manche Kombinationen fehlen in der Tabelle; zum Beispiel ist es ausgeschlossen, daß ein Mittelfeld gleichzeitig einen Dativ überzählig hat und einen braucht, denn in diesem Fall wäre der Dativ schon beim Aufbau des Mittelfeldes abgebunden worden.
Als letztes muß die »Verbalphrase« mit dem Vorfeld verbunden werden: Ein Satz besteht aus einem Dativobjekt und einer Verbalphrase, die noch einen Dativ benötigt, aber nichts überzählig hat. Ähnliches gilt für die anderen möglichen Komplemente, zusätzlich kann im Vorfeld auch ein Adjunkt (»Gestern brannte es«) oder ein Expletivum (»Es brennt ein Feuer im Garten«) stehen; in diesen beiden Fällen müssen sowohl die NEED- als auch die HAVE-Menge leer sein.

Schlußbemerkungen

Das Anwendungsgebiet der Sprachsynthse läßt niedrige Rechenzeiten besonders wünschenswert erscheinen: Die Grammatiken des SVOX-Systems beschränken sich daher auf atomare Unifikation, was die syntaktische Analyse stark beschleunigt. Allerdings hat dies unmittelbar zur Folge, nur noch die Klasse der kontextfreien Sprachen verarbeiten zu können. Entgegen einer in der Computerlinguistik verbreiteten Meinung war es aber durchaus möglich, mit der beschriebenen Grammatik auch komplexere linguistische Phänomene zu behandeln.
Das vorgestellte Verfahren kann für Anwendungsgebiete, die lediglich eine syntaktische Analyse erfordern, einen Mittelweg zwischen einer linguistisch sehr sauberen, aber zeitintensiven Unifikationsgrammatik einerseits und einer nur oberflächlichen, daher qualitätsmindernden Analyse andererseits darstellen.

Dank

Abschließend möchte ich der Gruppe für Sprachverarbeitung der ETH Zürich danken; einmal für die interessanten Aufgabenstellungen, an denen ich als Student arbeiten durfte, ganz besonders aber Christof Traber und Schamai Safra, die so manche Diskussionen mit mir führten, von denen ich viel profitieren konnte.

Literatur

[Haugeneder/Trost 1993] Haugeneder, Hans/Trost, Harald: Beschreibungsformalismen für sprachliches Wissen. In: Görz, Günther (Hrsg.): Einführung in die künstliche Intelligenz. Bonn [u. a.]: Addison Wesley, 1993. ISBN 3-89319-507-6. Seite 372 – 424.

[König/Seiffert 1989] König, Esther/Seiffert, Roland: Grundkurs Prolog für Linguisten. Tübingen: Francke, 1989. ISBN 3-7720-1749-5.

[Pereira/Warren 1980] Pereira, F. C. N./Warren, D. H. D.: Definite Clause Grammars for Language Analysis — A Survey of the Formalism and a Comparison with Augmented Transition Networks. Artificial Intelligence, Vol. 13, 1980. Seite 231 – 278.

[Pollard/Sag 1987] Pollard, Carl J./Sag, Ivan A.: Information-based syntax and semantics. Vol. 1: Fundamentals.CSLI Lecture Note No. 13. Stanford: CSLI, 1987. ISBN 0-93707-24-5 (Paperback) und 0-937073-23-7 (gebunden).

[Russi 1990] Russi, Thomas: A Framework for Syntactic and Morphological Analysis and its Application in a Text-to-Speech System. Dissertation Eidgenössische Technische Hochschule Zürich Nr. 9328.

[Traber 1993] Traber, Christof: Syntactic Processing and Prosody Control in the SVOX TTS System for German. In: Proc. Eurospeech '93, vol. 3, S. 2099 – 2102, 1993.

TimePlan : Towards a time-based domain-independent planning system

C.C.Marinagi

Institute of Informatics and Telecommunications
N.C.S.R. Demokritos
15310 Aghia Paraskevi, Athens, Greece

e-mail : katerina@iit.nrcps.ariadne-t.gr
Tel : +301-6510310, Fax:+301-6532175

Abstract

In this paper we describe an experimental time-based planning system which is in development stage. Existing planning systems are compared with TimePlan as well as with its possible extension. TRL temporal reference language can be incorporated in TimePlan to provide techniques for temporal deduction. Additionally, TGS time graph management system can be used to handle temporal information fast and efficiently.

1. Introduction

A problem in the field of Artificial Intelligence is the design of systems, called planning systems, that can describe a sequence of actions to be executed by an agent, such as an intelligent robot.

Planning has been extensively studied during the last thirty years, and has been applied onto various areas such as process control, production control, project management, circuit design, spacecraft activities, etc...

However, realistic planning systems bear great temporal and conceptual complexity and major technical obstacles exist that still remain to be solved. To overcome such problems, issues such as time, space, uncertainty, and conceptual structures must be incorporated into planning systems. New representation schemes and planning algorithms could be designed and developed to support such issues.

During the long period of planning research, a lot of planning systems have been developed and a great progress has been made comparing to the

early years. A number of techniques have been discovered that aid in the development of applicable planning systems. For example, nowadays an hierarchical and partially-ordered architecture is generally accepted. In real world problems, where many, possibly parallel, processes are taking place over time, representing and reasoning about time has proved to be a strong necessity. Therefore, incorporating time in a planning system has become a requirement of contemporary planning systems. The question is which is the most efficient way to represent and reason about time in a planning system. Getting started with Vere's attempt [34] to take time into account by defining 'time windows', more recent approaches seem to prefer the integration of a more powerful temporal reasoning methodology. Some researchers satisfy this desire by adapting the time map management (TMM) [9]. However, one point of doubt of such an approach is that an integration between a time map management system and a nonlinear planner is not a trivial task [26]. Another point of doubt is that temporal projection (a kind of temporal reasoning used in TMM), may not be useful for planning [23],[5].

The fact that the usefulness of TMM in planning has recently come into question, has been the motivation of our approach. The initial idea was that we could build a planning system that would meet some standards and find an alternative methodology for time management and reasoning.

Time has already been represented in several planners, but most of the times in a rather naive manner. In order to incorporate temporal issues of advanced notational efficacy such as certain or uncertain temporal intervals and instances, new techniques and deeper modelling for the representation of time are required.

Such matters have also motivated us into following another approach to the representation and management of temporal aspects and the incorporation of these aspects into the planner. This approach will be realized by the utilization of the temporal reference language TRL [24], and the time management system TGS [16]. The basic idea is to use TRL as a temporal deduction component of the planner and TGS as a system for fast storing/retrieval of the temporal references.

This paper describes an effort to build a planning system, called TimePlan, which meets a set of requirements that many contemporary planners provide for. Some preliminary results of this system have been reported in [20]. TimePlan, at the current stage of development, does not

reach our aims, but consists a basis for further testing of ideas resulting from research. In order to provide the reader with the most essential background knowledge, in section 2, we briefly give the definitions of some basic planning terms as well as some of the important characteristics of existing planning systems. Old fashion techniques that have been abandoned are not referred at all. In section 3, it is explained which of these characteristics are considered desirable to be included to our system and in section 4 which of them have already been implemented. Finally, in section 5 we conclude and give future perspectives.

2. Background knowledge and previous work

2.1 Basic planning terms

There are several surveys in the literature of planning systems, [4],[15],[19] that can introduce the reader in planning issues. In the following we give the definitions of the basic planning terms :

A *planner* has to find a collection of temporally related actions that should be executed in order to achieve its goal.

An AI planning system generates an action sequence which can achieve the explicitly stated goals. The essential *inputs* of a planner are :

a) the current state of the world (the planning environment),

b) the operator schemata (or else the action specifications), which describe actions in terms of their preconditions and effects and

c) a set of goals ,which is a description of desired future conditions.

The *output* of a planner is a plan which satisfies the goals. Thus, a *plan* is a representation of a course of actions that is a solution to a given problem.

2.1 Classification of planning systems

When the plan representation language and plan generation algorithms are expected to work for a large variety of application domains the planner is characterised as *domain-independent.* On the other hand, when specific heuristics are used for the control of the operations of a planner, the planner is characterised *domain-dependent.* The planning research that focuses on

domain-independent planners has a history of over 30 years and is still an area of great interest.

Hierarchical planning resulted from the need to reduce the combinatorial problem in complex problem domains, that had appeared in non-hierarchical planners (e.g. [13],[31],[32]). A simplified version of the initial problem occurs in a higher level problem space or *abstraction space* and the most detailed version occurs in the lowest level or *ground space*. This approach has the advantage that it effectively reduces search by ignoring details. (e.g.[27],[33],[35]).

Partially-ordered planners are these systems that use a partial ordering of plan structures, while total-ordered planners use a sequence of operators to represent a plan in search space. All partially-ordered planners adopt the *least-commitment* strategy, which postpones the commitment to a particular ordering as long as possible, until there is some reason to make an ordering decision. (e.g.[27],[33],[35]).

There are also other architectures not discussed here. A more detailed presentation of classification issues can be found in [19].

2.2 Data structures

In order to represent the plan, a *data structure* is needed. NOAH-like planners [27], that search through the space of partial plans, use a data structure called *procedural network*. The *nodes* of this network represent actions and goals and the *arcs* represent the temporal precedence over them. This *action-ordering* representation contrasts *state-based* plan structures that were used before (e.g. STRIPS [13]) and required complete specification of intermediate states. Procedural networks, are appropriate for hierarchical planning, because any node at some level of abstraction, is attached to its detailed expansion in the next level. They are also appropriate for partially-ordered planning because they can represent parallel nodes as long as they remain unordered. Later on, when nodes can be merged, old arcs are cut and new ones are built to represent the precedence that has been decided.

Temporal planners extend classical procedural network to incorporate the notion of time. DEVISER [34] introduces time windows for activities and goals. Since each activity is not instantaneous as in classical planners, the node that represents it, has a respective 'package node' that carries the time

window and duration associated with the activity. FORBIN [10] uses a 'task network' for plan representation where there are not nodes and arcs, but time lines that are split and joined.

2.3 Domain Representation

As far as *domain representation* is concerned, planning systems have introduced different formalisms.

following the term action will be used to refer to both notions, because a common formalism is used for representing both of them.

An action is called *'primitive'* when it can be executed directly by the planner or it is called *'macro'* when it can be decomposed in more detailed

unimportant effects. Thus it checks if any effect at an action node is kept true up to the goal node , assuming that for any effect a 'scope' must be specified.

NONLIN makes the distinction between important and unimportant effects using the 'goal structure' (GOST) for the important ones. GOST gives a set of *'ranges'* for which an effect must be protected to have a particular value. NONLIN first check TOME to ensure that adding an effect will not cause a conflict with any parallel nodes, and then check GOST to ensure that protection ranges are respected.

2.4 Planning Algorithm

The *planning algorithms* that have been used by hierarchical partially-ordered systems, may differ in their details but there are basic steps that can be recognised to most of them, having in mind that the usual searching technique is backward chaining. These steps are illustrated in figure 1.a.

The details of this algorithm differ from system to system. For example NONLIN's algorithm on which our approach is based extends the above in that the goal expansion step is analysed in two substeps:

1) *Linking* : Look if the goal is already true somewhere in the plan and introduce a link there. The goal becomes a *phantom.*

2) If step 1) fails then do *Node Expansion* : choose an operator to expand the goal.

2.4.1 Conflict detection and correction

Conflict detection and correction made by step c) of the basic algorithm can be implemented using different ways. A conflict happens when two parallel nodes have contradictory TOME assertions . A check on TOME detects them and a check on GOST can see if for one or both of them there is a protected area to hold. NOAH uses 'critics', which are procedures that use heuristics to detect and correct for interactions , consulting TOME assertions. NONLIN does not need critics but consult GOST to detect for interactions and aid in suggesting appropriate orderings. This idea was followed by many consequent systems that have proposed similar techniques. (e.g. O-PLAN[8] , DEVISER [34], SIPE [35]).

Temporal planners impose time constraints on goals and actions to be satisfied or performed. Some solutions may need to be abandoned during the

correction of interactions. In planning systems that resources are handled, the mechanism that allocate and deallocate them check also for usage conflicts. If such conflicts occur they can be avoided.

a) An unsolved goal is chosen from the goal list to be satisfied.
b) An operator is chosen to expand the goal and the goal is expanded. *(Node Expansion)*
c) Check if the effects of the chosen operator generate inconsistencies and correct them by ordering. *(Conflict detection & correction)*
d) Remove the achieved goal from the list of unsolved goals and add the preconditions of the operator in the list . Preconditions are now the new goals.
e) Go to step a).

Figure 2.a The basic steps of a Planning Algorithm

2.4.2 Guiding search

During the execution of planning algorithms some choices must be made that can be kept as *backtrack choice points*. These are:

a) which action reduction schema to use to expand a node
b) how to order two conflicting nodes

NOAH performs deterministically, without backtracking at all. Most of the other hierarchical partially-ordered planners keep choice points for backtracking. They just use different techniques to search the space of plans. Backtracking can be classical chronological [13],[32],[34],[35], or dependency-directed [28],[6] when the system backtracks directly to the choice point that is responsible for failure.

On one side, avoiding backtracking can cause loss of solutions and on the other side uncontrolled backtracking can be very expensive. Therefore, planning systems employ various techniques in order to prune the search space. *'Meta-rules'* (e.g. MOLGEN [29],[30]) and *human interaction* (e.g. SIPE [35]) can manage control decisions. *Time* and *resource usage constraints* can also play an important role in taking decisions. Finally, *heuristic techniques* can be used to guide search which may be domain-dependent or independent. For example, *temporal coherence* (TC) [12] and

condition typing [33] use domain knowledge to reduce search. A system that uses TC as a pruning technique is O-PLAN [8]. Condition typing is the technique that distinguish different types of conditions (e.g. NONLIN [33], O-PLAN [8]). Another heuristic technique is to define action reduction schemata that contain information about their *priority* to be selected, [10].

2.5 Temporal Planning

In traditional STRIPS-like systems time is represented implicitly. Actions are instantaneous and only one action can occur at a time (state-based systems). Later, NOAH-like systems allow of parallel but still instantaneous actions. The only temporal relation that can be represented is action precedence. An action can be strictly before another or it can occur simultaneously with other actions. All these planners do not take into consideration numerical time. They allow parallel actions because there is no way to impose time constraints on actions or goals. As a result they can not describe more complicated temporal knowledge such as concurrency, duration, overlapping, delayed effects and natural death, that temporal logics intend to.

Temporal planning takes time into account, explicitly. There are different approaches on what concerns temporal representation and reasoning in planning systems. Vere was the first that used temporal annotation in DEVISER [34]. The system assigns 'time windows' to actions or goals which are actually temporal intervals with numerical endpoints. DEVISER can deal with time constraints and duration but it is inadequate to represent complex temporal situations.

Allen [1] , McDermott [21] and Kowalski with Sergot [17] introduce *temporal logics* which are used for the building of planners . A global notion of time is developed that is independent of agent's actions. Some examples of planning systems that are based on temporal logics are : Timelogic [3] that is

representing and reasoning about temporal information. Temporal planners (FORBIN [10], TRIPTIC [26]) are based on TMM. Another planning system that employs a time network management system (TNMS), which is similar to TMM, is O-PLAN2 [11]. Despite the fact that the metric temporal reasoning and incremental updates supported by TMM are useful in planning , the problem is that it is not certain that persistence, projection and overlap chaining are suitable forms of inference for current planning techniques. This problem is pointed out in [23] by Nebel and Backstrom and is also admitted by Boddy [5] who comments that " the correct level of integration between a planner and a temporal reasoning system such as the TMM is an open question".

3. Design Requirements of the TimePlan system

TimePlan system is an hierarchical partially-ordered planner based on the class of NOAH-NONLIN planners. In this planning system time is incorporated in a DEVISER-like manner.

After evaluating the characteristics of the existing planning systems we have decided to include a union of the most necessary characteristics. We use Tate's [33] terminology to describe most of our issues. In the following, we will describe the next TimePlan's basic design requirements :

- the operator schema, that is how actions are represented
- the plan network, that is the data structure that is used for plan representation
- the planning algorithm, that is the basic steps that the planner follows to produce a plan
- how time requirements are incorporated into the system.

3.1 Operator representation

The desirable operator schemata that we adopt for action description includes information that is illustrated in figure 3.a.

The *operator's name* is a unique name that characterises the operator. The *operators identification* is the parameterised action description. Several schemas that may have the same operator_id, are alternatives for expanding a node.

a) Operator_name

b) Operator_id

c) Type : primitive or macro

(if type : macro

then also define [action list] and action ordering)

d)1) Duration 2) (start time, end time)

e) Resources

(id_name, type)

f) Assumptions

(conditions that simply must be true or else the operator is not applicable)

g) Subgoals

(conditions that must become satisfied)

h) Primary effects

(effects that provide a reason to use this schema)

i) General effects

Figure 3.a Operator schema

The *type* may be primitive or macro. In case of a macro action the schema contains a list of the actions that belong to the lower level on which the macro action will be refined and also an action ordering description associated with the schema.

The *duration* is numerical. The *start* or the *end time* of the action can be defined, in case that the action has a fixed time of occurence (DEVISER's scheduled events). Otherwise, temporal bounds will be determined during plan construction.

The objects that are used as *resources* can be described by their identification name and their type (for example shared, dedicated or consumable).

Assumptions are conditions that are just checked to be true, like O-PLAN's *only-use-if* conditions. They can be useful during action selection to reject some alternatives. *Subgoals* correspond to normal preconditions of STRIPS that attempt to be satisfied through linking or expansion.

Effects are also separated into *primary* and *general effects* to differentiate between those that may be used to satisfy conditions and those that are general and considered as side-effects. The effects are not separated into added and deleted as STRIPS postconditions. They are expressed as propositions which can be negated or not.

3.2 Plan network

The *plan network* is a dynamically changing collection of nodes. In the network appear three types of nodes : *action nodes, goal nodes* and *phantom nodes.* There are also two special nodes the *start* and *stop* nodes that correspond to initial and final state respectively.

A goal node represents a condition which must be satisfied. In order to satisfy this condition an appropriate action is selected and an action node along with its subgoals replaces the initial goal.

Therefore each action node in the network is linked with its parent action node and children goal nodes (action subgoals). These goal nodes will be consequently replaced by other action nodes, etc. This operation is called *node expansion.*

Each action node has an associated TOME assertion, where its instantiated effects (both primary and general) are kept. Additionally, when an effect is protected to hold up to a particular node, then it is kept in GOST along with its protection range. A subgoal is protected to hold up to the action node that needs it as precondition.

Node expansion finishes when the condition corresponding to a goal node is satisfied by an initial condition. In this case this goal node is linked

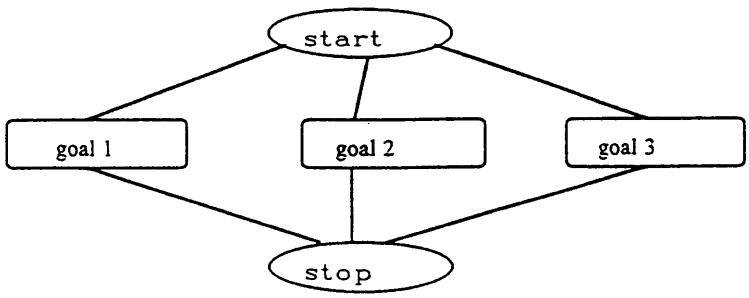


Figure 3.b Beginning plan generation

to a start node, and the goal becomes a *phantom* node. This operation is called *linking*. Another case of linking appears when the current goal can be found to be satisfied somewhere else in the network, i.e. if it is a TOME assertion of a parallel action node. Two nodes of the network are called parallel if they have a common ancestor node.

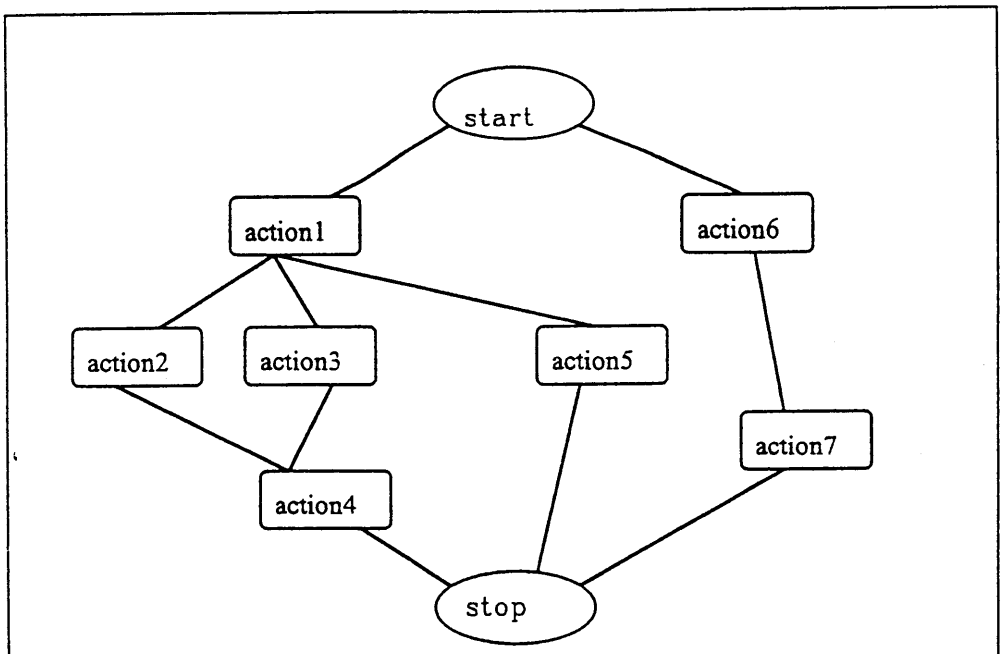


Figure 3.c Final plan

At the beginning of plan generation, the plan network consists of the start node, the stop node and the goal nodes to be satisfied (Figure 3.b). At the end of plan generation the plan network has no goal nodes, as they have been replaced by action nodes or phantom nodes. Phantom nodes are actually ignored. Therefore, the plan which is going to be generated contains only the action nodes of the plan network, with the computing ordering in the network (Figure 3.c).

During plan generation, ordering decisions forced by conflict resolution process may put new links while cutting old ones. The resulting network may still have parallel branches, in case that there are actions that can be executed in parallel by one or many agents.

3.3 Planning algorithm

Getting started, the domain writer introduces all operator schemata, as well as the facts that are true at initial state and the facts that will have to be

true at final state. Top goals are kept in a goal list. The system takes to satisfy one goal after the other. If a solution can not be found the system can replan by reordering goals and try the new sequence.

As a result, the planner can be characterized as *goal driven* that does a *depth-first search* through the space of possible plans.

As planning proceeds the system always works at a node, referred as 'current node' in the following. Each node has an associated number. This node number is unique for every goal node and every phantom node. An action node takes the same number node with the goal node that has previously been expanded by selecting this action node.

As it is already mentioned in paragraph 2.4, according to Tate's terminology, three fundamental operations are adopted here : *linking*, *node expansion* and *conflict resolution & ordering.* The details of the implemented algorithm are explained below :

The first top goal is selected from goal list to be satisfied. If it can not be linked to initial state , it has to be expanded. The planner selects an action with a primary effect that matches the goal and the subgoals of this action become the new goals which are put on top of goal list. The effects of the action become the TOME assertions of the action node. The primary effect that matched the subgoal becomes a GOST assertion.

When an action that has been selected, has more than one subgoals, then parallel branches are created for each one. Interactions, being helpful or harmful, may be appeared between parallel nodes. For this reason, each time a goal or subgoal is selected from goal list, all parallel nodes are checked for any harmful interactions (conflicts). In order to detect them, the planner compares the goal with TOME assertions of parallel actions nodes. Conflicts are resolved by ordering conflicting nodes while respecting protections. A primary effect is protected to hold up to the beginning of the actions of which it is a precondition and in this case is also a GOST assertion.

After conflict detection and resolution of a goal node, the planner looks if any helpful interactions exist, in other words, tries linking operation. The current goal node, may be true somewhere in the existing plan or else at initial state. Such a case appears when the goal matches with a TOME assertion of a parallel action node or else when the goal matches with one of the initial conditions.

Then if linking fails, the goal is expanded by selecting an action. Candidate actions are these that have a primary effect that matches with the goal. Time and resource constraints, assumptions, as well as heuristic rules, may reject some alternatives. If more than one candidate action remain, one is selected and the others are kept for backtracking.

When an action is selected, the planner compares its TOME assertions with those of parallel action nodes. If a conflict is found an ordering has to be decided, respecting GOST assertions.

Actually, each time a conflict is resolved there are two possible orderings that can be applied. Time and resource constraints must be evaluated and heuristics may be applied to help rejecting one of the two possible orderings. Then, if still two alternative orderings remain, this becomes a choice point for backtracking.

The plan construction continues by applying the three fundamental operations from the beginning, having selected the next unsolved goal from the goal list.

3.4 Design issues on incorporating time requirements

Each operator schema contains temporal information related to the duration of the action, the starting time point and the finishing time point of the action. Such information can be used in cases of real-world planning applications where time plays an important role.

The temporal parameters of the operator schema incorporate some additional complexity into the algorithm of the planner. The operator schema, the list of goals, and the planning algorithm must take temporal information into consideration. Some of the points that this should be done are the following :

- Operator schema contains information about the duration of an action. The start or the end time of the execution of an action may also be defined.
- Each goal in the goal list is augmented with a temporal template indicating the expected time that this goal must be achieved. The expected time of the achievement of top goals is defined from the beginning. As planning proceeds new subgoals will have to be true at the start time of the actions of which they are preconditions.

• It is possible to define uncertain expected time intervals for top goals. This uncertainty will influence the time intervals of all selected actions, except for those that have fixed time of occurence.

• When an action with uninstantiated temporal bounds is selected to expand a goal, then its bounds are evaluated taking into consideration the time that the goal has to be achieved and the duration of the action.

• The action which is going to be chosen to expand a goal must additionally satisfy the temporal constraints of this goal, e.g. an action with a long duration will not be selected by a goal whose temporal distance from the beginning of plan time is shorter.

• The effects of an action that are TOME assertions, are implied to start exactly at the end of the execution of the action. After action selection, it is not determined how long its effects hold. That is, they are not protected to hold up to a particular node. Later on, during conflict resolution, an ordering may be imposed and as a result the end times of some effects may be constrained.

• After linking a goal node to another action node, as well as after ordering two nodes to solve a conflict, the time bounds of nodes are constrained. Whatever changes take place are propagated throughout the plan network.

4. Implementation Issues of time plan system

4.1 What has already been implemented

The planning system TimePlan that has been implemented up to the present meets a great part of the design requirements. In this first attempt in order to have a system that would work, it was not possible to implement all features because of difficulties well-known in planning literature. The main difference is that the operator schema that has been used does not include information about resource usage.

A prolog-like description of the algorithm is presented below, much abstracted from the real code, that just gives an idea of how one could start implementing a planning system. TimePlan has been entirely written in Arity Prolog

```
topgoals ([G1,G2,... Gn]).
initialstate ([ F1,F2,....Fn]).

planner :- topgoals(G), take_to_satisfy(G).

take_to_satisfy([G|Goals]) :- linking (G),!,
                              take_to_satisfy(Goals).
take_to_satisfy([G|Goals]) :-
                    expand(G, Action, Subgoal, Effects),
                    tomeassert(Action),
                    gostassert(Action),
                    detect_resolve_action_conflict( Action),
                    detect_resolve_subgoal_conflict( Subgoals),
                    take_to_satisfy(Subgoals),
                    take_to_satisfy(Goals) .

linking(G) :- satisfied(G, Node),
              order(G,Node),
              propagate_changes_of_time_bounds(G),
              assert(phantom(G, at(Node))).

detect_resolve_action_conflict( Action, Node) :-
                    check_tome(Action, Node),
                    check_gost(Action, Node),
                    resolve_by_ordering(Node,Paral_Node),
                    propagate_changes_of_time_bounds(Action).

detect_resolve_subgoal_conflict( [S|Subgoals]) :-
                    check_tome(S, Node),
                    check_gost(S, Node),
                    resolve_by_ordering(Node, Paral_Node),
                    propagate_changes_of_time_bounds(S),
                    detect_resolve_subgoal_conflict(Subgoals).
```

The predicates that are not defined here, may be defined according to the features that the planning system adopts and the techniques that it expects to implement.

5 Conclusions and future perspectives

In this paper we have described TimePlan system which is still in development stage. Background knowledge useful for comparison with other relative systems and explanation of our intentions have been given. The desigh issues of the final system have also been presented. In the following we make the future perspectives more specific and discuss about them.

TimePlan provides some primitive operations concerning time management. However, it does not provide any facilities for temporal deduction. There are cases that we are interested in the value of a given condition during plan generation. If this condition happens to be a TOME assertion of an action node, then its value can be computed immediately. However, there are cases where the value of such a condition may only be deduced by some other instantiated conditions. To answer questions like that, some deduction mechanism needs to be incorporated in the planner. Especially when the planner represents time, this deduction mechanism should be temporal.

TRL [24] is a temporal reference language that can provide techniques for temporal deduction. We could therefore extend TimePlan with a temporal deduction mechanism based on the principles of TRL.

TGS [16] is a time graph management system that stores/retrieves temporal information in an application independent manner. By incorporating the principles of TGS into TimePlan we will be able to handle efficiently large amounts of temporal information making the planner fast and efficient.

Another direction of research concerns the incorporation of uncertain durations. In this way we will be able to represent goals and actions with dynamically changing durations, certain or uncertain, and produce plans with elastic temporal duration for their actions. An elastic temporal duration is a duration that can be compressed, shortened, or expanded.

In temporal planning systems, resource handling is a serious matter as well. The planning procedure have to ensure that resource conflicts and deadlocks do not occur. Resources can be defined as a separate component

of the operator schema (as in SIPE [35]). Alternatively, resources may be specified as conditions, that must be reserved as long as the particular action proceeds or for a definite period of time [7],[10],[34]. We intend to incorporate resources into TimePlan in one or the other way. TGS's [16] suggestion that a resource can belong to the set of resource types {shared, unary shared, dedicated, unary dedicated, consumable, partially consumable}, will be considered.

Acknowledgements

I want to thank Dr.C.D.Spyropoulos, Dr.T.Panayiotopoulos for their supervision and support of this work.

References

[1] J. F. Allen, "Towards a General Theory of Action and Time", Artificial Intelligence 23, pp.123-154, 1984.

[2] J. F. Allen, "Planning as Temporal Reasoning", In Proceedings of the Conference on Principles of Knowledge Representation and Reasoning, pp.3-14, 1991.

[3] J. F. Allen and J. A. Koomen, "Planning Using a Temporal World Model", in Proceedings of the 8th IJCAI, Karlsruhe, Germany, pp.741-747, 1983.

[4] A. Barr and E. A. Feigenbaum, "The Handbook of Artificial Intelligence", VolII, Morgan Kaufmann, Palo Alto, Calf., pp. 515-562, 1981.

[5] M. Boddy, "Temporal Reasoning for Planning and Scheduling", SIGART Bulletin, Vol 4, No 3, 1992.

[6] D. Chapman, "Planning for Conjunctive Goals" Artificial Intelligence 32, pp.333-377, 1987.[TWEAK]

[7] P. Cheesman, "A Representation of Time for Automatic Planning", IEEE, pp. 513-518.

[8] K. Currie and A. Tate, "O-PLAN: the open system architecture", Artificial Intelligence 52, pp.49-86, 1991. [O-PLAN]

[9] T. Dean and D. McDermott, "Temporal Data Base Management", Artificial Intelligence 32, pp.1-55, 1987.

[10] T. L. Dean, R. J. Firby and D. Miller, "Hierarchical Planning Involving Deadlines,Travel Time and Resources", Computational Intelligence, Vol.4, pp.381-398, 1988. [FORBIN]

[11] B. Drabble, R. Kirby, "Associating A.I. Planner Entities with an underlying Time Point Network", In J. Hertzberg, editor, European Workshop on Planning. EWSP '91, Sankt Augustin, FRG, March 1991, Proceedings, pp. 27-38. [O-PLAN2]

[12] M. Drummomd, K. W. Currie, "Exploiting Temporal Coherence in Non-Linear Plan Construction", Computational Intelligence, March 1989.

[13] R. Fikes and N. Nillson,"STRIPS: A new approach to the Application of Theorem Proving to Problem Solving", Artificial Intelligence 2 ,pp.189-208, 1971. [STRIPS]

[14] M. Georgeff, "Communications and Interactions in Multi-Agent Planning Systems", In Proceedings of the Third National Conference on Artificial Intelligence,Menlo Park, Calif, 1982.

[15] J. Hendler, A. Tate and M. Drummond, "AI Planning:Systems and Techniques", AI Magazine, pp. 61-77, Summer 1990.

[16] S. Kokkotos, C. D. Spyropoulos, "TGS : A kernel graph system for time management", DEMO 90/6, N.C.S.R. "Demokritos", June 1990.

[17] R. Kowalski, M. Sergot, "A Logic-based Calculus of Events", New Generation Computing, No4. pp 67-95, 1986.

[18] A.L. Lansky, "A representation of parallel activity based on events, structures, and causality", in Proceedings 1986Workshop on Reasoning about Actions and Plans, Timberline, OR, 1987. [GEMPLAN].

[19] C. C. Marinagi, C. D. Spypopoulos, "A Review on Domain-Independent Planning", 4th Hellenic Conference organised by the Greek Computer Society, 1993.

[20] C. C. Marinagi, T. Paraskiotopoulos, C. D. Spyropoulos, "TimePlan :

[22] D. P. Miller, "Planning by search through simulations" , PhD Thesis, Yale University, Dec. 1985.

[23] B. Nebel, C. Backstrom, "On the Complexity of Temporal Projection and Plan Validation", Proceedings AAAI-92, 10th National Conference on Artificial Intelligence, 1992, pp.748-753.

[24] T. Panayiotopoulos, C.D. Spyropoulos, E.V. Ioannidis, "TRL: A formal Language for Temporal References Part I : Specifications and Semantics", DEMO, 93/10, N.C.S.R. Demokritos, June 1993.

[25] E. Pednault, "Solving Multi-Agent Dynamic World Problems in the Classical Planning Framework" ,In Reasoning about Actions and Plans: Proceedings of the 1986 Workshop, eds. M.Georgeff, A.Lansky, San Mateo,Calif, Morgan Kaufmann, 1987.

[26] E. Rutten, "A temporal non-linear planner: TRIPTIC", Arbeitpapier, 582, GMD, 1991. [TRIPTIC]

[27] E. D. Sacerdoti, "A Structure for Plans and Behaviour", Amsterdam: Elsevier-North Holland, 1977. [NOAH]

[28] R. M. Stallman and G. J. Sussman, "Forward reasoning and Dependency-Directed Backtracking in a System for Computer-Aided Circuit Analysis", Artificial Intelligence 9, pp.135-196, 1977.

[29] M. J. Stefik, "Planning with Constraints", Artificial Intelligence 16 , pp.111-140, 1981. [MOLGEN]

[30] M. J. Stefik, "Planning and Metaplanning", Artificial Intelligence 16 , pp.141-169, 1981. [MOLGEN]

[31] G. A. Sussman, "A computational Model of Skill Acquisition", MIT AI Lab Memo, AI-TR-297, AI Lab., Massachusetts Institute of Technology, 1973.[HACKER]

[32] A. Tate, "Interacting Goals and their use", In Proceedings of the 4th IJCAI , Menlo Park, Calif, pp.215-218, 1975. [INTERPLAN]

[33] A. Tate, "Generating Project Networks", In Proceedings of 5th IJCAI, Menlo Park, Calif., 1977. [NONLIN]

[34] S. A. Vere, "Planning in Time: Windows and Durations for Activities and Goals", IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol.PAMI-5, No.3, pp.246-267, May 1983. [DEVISER]

[35] D. E. Wilkins, "Domain-independent Planning:Representation and Plan Generation ", Artificial Intelligence 22, pp.269-301, 1984. [SIPE]

Logikbasiertes Lernen in relationalen Datenbanken

Guido Lindner
Universität Dortmund
FB Informatik VIII
44221 Dortmund
e-mail: lindner@ls8.informatik.uni-dortmund.de

Mai 1994

Zusammenfassung

Die Verbindung von Verfahren des maschinellen Lernens mit relationalen Datenbanken ist für verschiedene Bereiche, z.B. Knowledge Discovery, Deduktive Datenbanken und maschinelles Lernen selbst, von steigendem Interesse. Dieser Artikel beschreibt die Anwendung des logikorientierten Lernverfahrens RDT auf relationle Datenbanken. In diesem Rahmen wird die Frage der Repräsentation der Datenbank für das logikbasierte Lernen diskutiert. In ersten experimentellen Test über dem im maschinellen Lernen bekannten KRK–Sachbereich und einer Anwendung aus dem Bereich der Roboternavigation werden einige Vorteile und Nachteile von RDT/DB deutlich.

1 Einleitung

Im Rahmen der Wissensentdeckung in relationalen Datenbanken versucht man immer mehr auch Verfahren des maschinellen Lernens einzusetzen. Hier bieten sich logikbasierte Verfahren aufgrund ihrer Leistungsfähigkeit an. Im logikbasiertes Lernen liegt das Lernergebnis in eingeschränkter Prädikatenlogik vor und ist von einem System interpretierbar. Andererseits besteht auch von der Seite des maschinellen Lernens ein großes Interesse, durch eine Anbindung an Datenbanken, sich größere Beispielmengen zugänglich und handhabbar zu machen.

Im nächsten Abschnitt beschreibe ich deshalb erst einmal die Berührungspunkte der verschiedenen Disziplinen, die ein Interesse am Einsatz von Lernverfahren in relationalen Datenbanken haben. Im weiteren werde ich die Anwendung des logikbasierten Lernverfahrens RDT [Kietz und Wrobel, 1992] auf relationale Datenbanken beschreiben. Im Abschnitt 3 stelle ich das Verfahren vor und beschreibe dann in den folgenden Abschnitten besondere Punkte bei der Anpassung an relationale Datenbanken. Der Abschnitt 6 zeigt dann die Unterschiede zwischen RDT, angewandt auf relationale Datenbanken (RDT/DB), und dem ursprünglichen RDT auf.

Im Abschnitt 7 beschreibe ich erste experimentelle Vergleichstests zwischen RDT und RDT/DB, an Hand von zwei Testszenarien. Zum einen wurde hierfür von mir der im maschinellen Lernen bekannte Sachbereich KRK [Quinlan, 1983] bearbeitet, und zum anderen eine reale Anwendung aus dem Bereich der Roboternavigation.

2 Induktives Lernen und relationale Datenbanken

2.1 Maschinelles Lernen im Knowledge Discovery

Im Rahmen des Knowledge Discovery (KDD) erhält das Maschinelle Lernen (ML) eine immer größere Bedeutung. Aus der Definition für Knowledge Discovery von Frawley, Piatetsky-Shapiro und Matheus wird die Ähnlichkeit der Zielsetzung deutlich[Frawley *et al.*, 1991].

Definition 1 (Knowledge Discovery/Data Mining)
Knowledge Discovery ist das Entdecken von potentiell nützlichen Informationen aus Daten.

Gegeben sei eine Menge von Fakten $\mathcal{F}$ (Daten), eine Sprache $\mathcal{L}$ und ein Kriterium C.
Ziel des Knowledge Discoverys ist es, eine Aussage S in der Sprache $\mathcal{L}$, die eine Beschreibung einer Teilmenge $\mathcal{F}_s$ von $\mathcal{F}$ ist, zu entdecken. Die Aussage S muß einfacher als die Aufzählung aller Fakten $\mathcal{F}_S$ sein und dem Kriterium C genügen.

Aus der Definition 1 sieht man die dem Lernen aus Beispielen ähnliche Zielsetzung. Auch hier geht man von einer Sprache $\mathcal{L}_H$ zur Beschreibung der Hypothese und einer Menge von Beispielen (Fakten) aus. Das Ergebnis eines maschinellen Lernvorgangs ist eine Hypothese, die den Kriterien des Benutzers genügt. Diese Hypothese könnte man auch als eine Menge von Schemata im Sinne des Knowledge Discovery ansehen. In diesem Sinne bezeichnen Matheus, Chan und Piatetsky-Shapiro Knowledge Discovery als eine Teilmenge des maschinellen Lernens [Matheus *et al.*, 1993].

Definition 2 (Lernen aus Beispielen)

Lernen aus Beispielen ist die Suche nach einer Beschreibung eines Begriffs C, zu dem man eine endliche Anzahl an Beispielen gegeben hat.

Gegeben sei eine Menge von Beispielen in einer Sprache $\mathcal{L}_B$, eine Sprache $\mathcal{L}_H$ zur Beschreibung des zu lernenden Begriffs (Hypothesensprache) und ein Kriterium zur Akzeptanz einer Hypothese. Eine Aussage $H \in \mathcal{L}_H$, die dem Akzeptanzkriterium genügt, ist eine Beschreibung des Begriffs C.

Lernen als Suche zu definieren, wurde erstmals von Mitchell 1982 vorgestellt [Mitchell, 1982]. In dieser Form des Lernens schließt man vom „Einzelfall "[1] auf eine allgemeine Beschreibung. Diese Art des logischen Schließen bezeichnet man als induktiven Schluß. Aus diesem Grund spricht man hier vom induktiven Lernen.

Aufgrund der Ähnlichkeit der zwei Forschungsbereiche entstand die Idee Verfahren des induktiven Lernens im Rahmen des KDD einzusetzen. Bisher wurden schon einige induktive Lernverfahren im Rahmen des KDD/Data Mining eingesetzt. Holsheimer und Siebes [Holsheimer und Siebes, 1993] beschreiben den Einsatz verschiedener induktiver Verfahren im KDD, z.B. von ID3 [Quinlan, 1986], AQ15 [Michalski *et al.*, 1986], CN2 [Clark und Niblett, 1989], ...

[1] In Anführungszeichen, da es meist doch eine Menge von Beispielen ist

2.2 Induktives Lernen in deduktiven Datenbanken

Eine weitere Anwendung und damit eine weitere Motivation, ein Verfahren des induktiven Lernens auf relationale Datenbanken anzuwenden, ist die Möglichkeit, diese als induktive Komponente in deduktiven Datenbanken einzusetzen. Deduktive Datenbanken bestehen aus explizieten Daten und implizieten Daten in Form von Regeln. Eine solche induktive Komponente könnte beim Aufbau und der Pflege einer deduktiven Datenbank durch das Entdecken von bisher nicht bekannten Abhängigkeiten hilfreich sein. So beschreiben Džeroski und Lavrač den Einsatz von LINUS [Lavrač und Džeroski, 1992] zum Entdecken von virtuellen Relationen in deduktiven Datenbanken [Džeroski und Lavrač, 1993].

Bei den bisher in den Abschnitten 2.1 und 2.2 genannten Verfahren handelt es sich ausschließlich um Attributwerte-Verfahren. Hier ist es nun von Interesse, ein logikorientiertes Verfahren wie RDT in Zusammenhang mit relationalen Datenbanken einzusetzen.

3 Das Lernverfahren RDT

RDT [Kietz und Wrobel, 1992] ist Bestandteil des Modellierungssystems MOBAL [Morik *et al.*, 1993] und wurde an der GMD entwickelt.

Bevor ich auf die Besonderheiten beim Lernen mit RDT über relationale Datenbanken eingehen, will ich das Verfahren RDT aus der Sicht des maschinellen Lernens charakterisieren.

RDT ist ein Verfahren aus dem *inductive logic programming* (ILP). Die Lernaufgabe von RDT kann wie folgt beschrieben werden:

Gegeben: Hintergrundwissen und eine Menge positiver und negativer Beispielen für einen zu lernenden Begriff C in funktionsfreier Klausellogik.

Ziel: Finde eine Hypothese H in funktionsfreier Klausellogik, die einem vom Benutzer definierten Akzeptanzkriterium genügt.

Das Akzeptanzkriterium setzt sich aus den folgenden Faktoren zusammmen:

Sei $P(X_1, \ldots, X_m)$ eine Menge von Prädikaten, die von den Attributen $X_1, \ldots, X_m$ abhängen. Weiterhin sei dann H eine Hypothese der Form $H : P(X_1, \ldots, X_m) \rightarrow q(X_1, \ldots, X_n)$. Dann sind mögliche Faktoren des Akzeptanzkriteriums:

- Anzahl der positiven Beispiele, die durch die Hypothese abgedeckt sind:
  $| pos(H) | := | \{(c_1, \ldots, c_n) \mid P(c_1, \ldots, c_m) \wedge q(c_1, \ldots, c_n)\} |$
- Anzahl der negativen Beispiele, die durch die Hypothese abgedeckt sind:
  $| neg(H) | := | \{(c_1, \ldots, c_n) \mid P(c_1, \ldots, c_m) \wedge not(q(c_1, \ldots, c_n))\} |$
- Anzahl der Instanzen, für die noch nicht bekannt ist, ob die Schlußfolgerung erfüllt ist:
  $| pred(H) | := | \{(c_1, \ldots, c_n) \mid P(c_1, \ldots, c_m) \wedge unknown(q(c_1, \ldots, c_n))\} |$
- Anzahl aller Beispiele der Hypothese:
  $| total(H) | := | \{(c_1, \ldots, c_n) \mid P(c_1, \ldots, c_m)\} | = | pos(H) \cup neg(H) \cup pred(H) |$
- Anzahl der Beispiele, für die die Schlußfolgerung gilt:
  $| concl(q) | := | \{c_1, \ldots, c_n) \mid q(c_1, \ldots, c_n)\} |$
- Anzahl der Beispiele, für die die Schlußfolgerung nicht von der Hypothese abgedeckt wird:
  $| uncover(H) | := | concl(H) \backslash pos(H) |$

Da bei RDT die Hypothesensprache eingeschränkte Prädikatenlogik ist, wird gemit ein sehr großer Hypothesenraum beschrieben. Bei RDT wird der Hy-

pothesenraum durch die Vorgabe von Schemata von Regeln 1. Stufe eingeschränkt. Diese definieren die syntaktische Form der möglichen Regeln. Ein Regelschema[2] ist also eine Regel, die anstatt der Sachbereichsprädikate Prädikatvariablen enthält.

Die Regelschemata werden nach einer erweiterten Form der θ–Subsumption [Plotkin, 1970], ihrer Allgemeinheit nach partiell geordnet. RDT beginnt mit dem generellsten Regelschema und geht dann zu den spezielleren (Top–Down–Strategie). Hierdurch ergibt sich eine weitere Einschränkung des Hypothesenraums. Spezialisierungen von akzeptierten oder von nach dem Ak-

solchen Verknüpfungsbeziehung mit den bereits belegten Prädikatvariablen stehen (siehe Beispiel Sortentaxonomie).

Weiterhin können von RDT zur Instanziierung der Prädikatvariablen die Prädikattopologie und die Sortentaxonomie ausgenutzt werden.

Prädikatentopologie

Prädikate können strukturiert, in Form eines Graphen, dessen Knoten Prädikate zugeordnet sind, repräsentiert werden. Eine Hypothese ist mit einer Prädikattopologie kompatibel, wenn die Prädikate der Prämisse in einem Vorgängerknoten oder im Knoten des Prädikats der Konklusion sind. Daraus folgt, daß die Suche nach Instanzen für die Prädikatvariablen auf wenige Topologieknoten beschränkt werden kann. Eine solche Topologie berechnet in MOBAL die Systemkomponente PST ([Klingspor, 1991],[Morik *et al.*, 1993, S.149-168]).

Sortentaxonomie

Es ist effektiv, nur sortenverträgliche Hypothesen zu testen. Ich will das an einem kleinen Beispiel deutlich machen.
Gegen sei folgendes Regelschema:

```
m_example(P1,P2,C):P1(Y) & P2(X,Y) --> C(X)
```

Nach der *relation chain* wird erst `P2` instanziiert, da nur die Variable `X` durch den Zielbegriff gebunden ist. Für `P2` können nur Prädikate ausgewählt werden, die die Sorte von `X` als als Sorte ihres ersten Arguments haben. Die Sortenbeziehungen werden in MOBAL durch die Systemkomponente STT berechnet ([Morik *et al.*, 1993, S.109-148])

Nachdem ich nun RDT beschrieben habe, will ich die Vorgehensweise noch einmal anschaulich skizzieren.

rdt(Q):-

- Konstruiere partielle Ordnung über die Metaprädikate (MP) für **Q**.
- Gehe TOP–DOWN durch die Ordnung der MP.
  - Instanziiere ein weiteres Prädikat P in MP, das folgendem genügt:
    1. Stelligkeitskompatibel
    2. Prädikatentopologiekompatibel
    3. Sortenkompatibel
  - Redundanztest θ–Subsumption mit bisher akzeptierten oder zu speziellen Hypothesen.
  - Berechnung der Faktoren für das Akzeptanzkriterium.
  - Auswertung des Akzeptanzkriteriums.

4 Repräsentation der Datenbank für RDT/DB

Um über eine relationale Datenbank mit dem Lernverfahren RDT lernen zu können, muß in einem ersten Schritt die tabellarische Darstellung in geeigneter Form als Prädikate dargestellt werden. Diese Transformation muß eine injektive Abbildung sein, damit man bei den Lernläufen wieder auf die DB zugreifen kann.

Problemstellung:
Gegeben sei eine Menge von Relationsschemata R und r∈R über die Attribute $a_1, ..., a_n$.
Finde eine geeignete Prädikatendarstellung für r.

4.1 Mögliche Repräsentationen

Bei vielen bisherigen Anwendungen von induktiven Lernverfahren wurden die einzelnen Tabellen (Relationen) der Datenbank durch logische *joins* zu einer *universal relation* [Ullmann, 1989] zusammengefaßt. So konnte man schnell und einfach existierende attribut–orientierte Lernverfahren einsetzen.

Eine solche Vorgehensweise ist für ein logikbasiertes Verfahren wie RDT sicherlich nicht geeignet, da man die einzelnen Relationen der Datenbank

Zur Umsetzung der Relationen in Prädikate gibt es zahlreiche Möglichkeiten, von denen ich die folgenden drei in Betracht gezogen habe.

Repräsentation I:

Man übernimmt den Tabellennamen als Prädikatnamen und die Attribute der Relation als Prädikatattribute.

Abbildungsvorschrift: R über $a_1, ..., a_n \rightarrow R(a_1, ..., a_n)$.

Repräsentation II:

Eine andere Möglichkeit ist die Zerlegung der Relation in $n+1$ Prädikate der Form:

Abbildungsvorschrift: R über $a_1, ..., a_n \rightarrow \forall(a_x(R, a_j, ..., a_l, a_x))$,
$R(a_j, ..., a_l)$,
$a_j, ..., a_l$ sind Primärschlüsselattribute der Relation R, und x läuft im Intervall $[1, ..., n]$ ohne $j \ldots l$.

Repräsentation III:

Die dritte Form ist ähnlich der Repräsentation II. Auch hier erhält man $n+1$ Prädikate, nur mit dem Unterschied, daß die Relationsbezeichnung (Tabellenname) mit in den Prädikatnamen gezogen wird.

Abbildungsvorschrift: R über $a_1, ..., a_n \rightarrow \forall R\_a_x(a_j, ..., a_l, a_x)$,
$R(a_j, ..., a_l)$,
$a_j, ..., a_l$ sind Primärschlüsselattribute der Relation R, und x läuft im Intervall $[1, ..., n]$ ohne $j \ldots l$.

4.2 Diskussion der Repräsentationen

Der Vorteil der Repräsentation I liegt in der sehr einfachen Überführungsvorschrift. Dem stehen allerdings zwei gravierende Nachteile gegenüber. Der erste Punkt ist, daß nicht mehr auf einzelne Attribute der Relation zugegriffen und somit nur die ganze Relation als Zielbegriff verwendet werden kann. Es dürfte aber gerade von Interesse sein, auch über einzelne Attribute zu lernen. Der zweite Nachteil ergibt sich aus der angestrebten Anwendung.

In dem logikorientierten Lernverfahren RDT wird die Beschreibung für einen Begriff gesucht. Als Beschreibungssprache dienen dabei die dem System bekannten Prädikate. Ist nun ein Attribut „frei" wählbar bei der Beschreibung des zu lernenden Begriffs, so erhöht sich das Kriterium *total* von RDT um den Faktor der Anzahl der Sortenelemente. Diese hätte zur Folge, daß das Kriterium *pred* als Akzeptanzkriterium von RDT nicht mehr handhabbar wäre. Man verliert dadurch ein Kriterium.

Regel auftreten können und Prädikate nicht doppelt dargestellt werden. Je-

genübergestellt.

Allerdings reicht die Repräsentation III alleine nicht aus um z.B auch Verkettungen über Attribute, die keine Schlüsselattribute sind, zu lernen. An einem Beispiel will ich die Problematik veranschaulichen.

Gegeben sind folgende zwei Metaprädikate[3]:

where $v(P(c_1, \ldots, c_m))$;

- Negative Beispiele stellen in relationalen Datenbanken ein besonderes Problem dar, da in relationalen Datenbanken keine negierten Relationen existieren. Durch die Einbindung von RDT/DB in MOBAL hat der Benutzer die Möglichkeit, negative Fakten des Zielbegriffs in die MOBAL–Wissensbasis einzugeben und diese beim Lernen zu berücksichtigen.

  $\mid neg(H) \mid \Rightarrow$
  select count(*) from $tabelle(P), tabelle(q)$
  where $v(P(c_1, \ldots, c_m) \wedge not(Q))$,
  $not(Q)$ ist die Menge der negativen Instanziierungen des Zielbegriffs q aus der MOBAL–Wissensbasis.

- $\mid pred(H) \mid = \mid total(H) \mid - \mid pos(H) \mid$

- $\mid total(H) \mid \Rightarrow$
  select count(*) from $tabelle(P)$
  where $(v(P(c_1, \ldots, c_m))$

- $\mid concl(H) \mid \Rightarrow$
  select count($ps(q)$) from $tabelle(q)$;

- $\mid uncover(H) \mid = \mid concl(H) \mid - \mid pos(H) \mid$

6 Das Verfahren RDT/DB

6.1 Einschränkungen des Hypothesenraums

RDT/DB gleicht in der Grundstruktur RDT. Wie bei RDT wird in RDT/DB der Hypothesenraum durch die vorgegebenen Regelmodelle definiert und nach der θ-Subsumption geordnet.

Unterschiede zu RDT gibt es bei der Instanziierung der Regelschemata. Wie in RDT wird in RDT/DB weiterhin die *relation chain* und die Prädikatentopologie für die Belegung der Prädikatvariablen benutzt.

In RDT/DB verwende ich allerdings keine Sortentaxonomie, da eine Berechnung der Sorten und der Vergleich der einzelnen Sorten über die Datenbank

aufgrund der zu erwartenden Größenordnung der einzelnen Sorten zu aufwendig wäre.

Stattdessen teste ich die Datentypverträglichkeit der gebundenen Attributvariablen. Eine Hypothese, in der eine Variable an zwei verschiedene Datentypen der Datenbank gebunden ist, kann kein positives Beispiel mehr abdecken.

Eine weitere Einschränkung des Hypothesenraums erreiche ich durch die Streichung redundanter Prädikate, ohne den θ-Subsumptionstest von RDT zu verwenden. Aufgrund der von Repräsentation der Datenbank als Prädikate und den Informationen über die Primärschlüssel der Datenbank kann man leicht verhindern, daß ein Prädikat mit gleicher Variablenbelegung in einer Regel zweimal instanziiert wird.

Neben den Unterschieden in der Hypothesenraumeinschränkung enthält RDT/DB einen zusätzlichen Parameter. Mittels dieses Parameters ist es dem Benutzer überlassen zu bestimmen, ob Hypothesen, die als zu speziell erkannt wurden, im Redundanztest berücksichtigt werden. Dann kann es passieren, daß Speziallisierungen der Hypothese noch getestet werden, obwohl die Hypothese als zu speziell bekannt ist. Das zusätzliche Testen der Hypothesen ist in vielen Anwendungen weniger rechenaufwendig als die Redundanztests (siehe Abschnitt 7).

6.2 Negative Beispiele in RDT/DB

In relationalen Datenbanken sind keine negativen Beispiele gegeben, diese können aber für das Lernen von großer Bedeutung sein, da ein negatives Beispiel oft stärker abgrenzt. Durch negative Beispiele vermeidet man es zu allgemeine Regeln zu lernen. An dieser Stelle habe ich ausgenutzt, daß RDT/DB als externes Tool in das System MOBAL mit eingebunden ist. Der Benutzer kann so über das System MOBAL negative Beispiele, die beim Lernen über die Datenbank berücksichtigt werden (siehe Abschnitt 5), eingeben.

6.3 Algorithmus

Bevor ich im nächsten Abschnitt die ersten experimentellen Tests mit RDT/DB vorstelle, möchte ich die Vorgehensweise von RDT/DB skizzieren.

rdt_db(Q):-

- Konstruiere partielle Ordnung über die Metaprädikate (MP) für **Q**.
- Gehe TOP–DOWN durch die Ordnung der MP.
  - Instanziiere ein weiteres Prädikat `P` in MP das folgendem genügt:
    1. Stelligkeitskompatibel
    2. Prädikatentopologiekompatibel
    3. Datentypkompatibel
    4. Test auf redundant instanziierte Prädikate
  - Redundanztest θ–Subsumption mit bisher akzeptierten oder zu speziellen Hypothesen.
  - Berechnung der Faktoren für das Akzeptanzkriterium.
  - Auswertung des Akzeptanzkriteriums.

7 Lernläufe mit RDT/DB

7.1 Tests im Lernszenario KRK

7.1.1 Sachbereich

Das KRK–Schachendspiel wurde als Beispiel für das Maschinelle Lernen zuerst von Quinlan beschrieben [Quinlan 1983]. KRK steht für king–rook versus king, d.h. es spielen der weiße König und der weiße Turm gegen den schwarzen König. Das Lernproblem besteht darin, die illegalen Stellungen der drei Figuren auf dem Schachbrett von den legalen abzugrenzen, d.h. der Begriff `illegal` soll gelernt werden. Es existieren insgesamt 64^3 (=262 144) mögliche Stellungen, von denen (unter der Annahme, daß Weiß am Zug ist) ca. 33% als `illegal` zu klassifizieren sind.

Man kann drei Arten illegaler Stellungen unterscheiden:

1. Ein Feld ist mit mehr als einer Figur belegt.
2. Die beiden Könige sind unmittelbar benachbart.
3. Der schwarze König steht im Schach, d.h. er steht entweder auf derselben Spalte oder derselben Zeile wie der weiße Turm, wobei der weiße König nicht zwischen den beiden steht.

white king x	white king y	white rook x	white rook y	black king x	black king y	ID
4	5	2	2	2	4	452224
...	...	...	...	...	...	...

Tabelle 2: illegal mit ID

x	y
1	1
1	2
...	...

Tabelle 3: adjacent

Dieser Sachbereich wurde von mir aufgrund seiner Überschaubarkeit und relativen Einfachheit ausgewählt. Weiterhin ist dieser Sachbereich ein oft verwendetes Testszenario im maschinellen Lernen, so daß man Vergleiche mit anderen Verfahren anstellen kann. So wurde der KRK–Sachbereich zu Vergleichtests von LINUS/FOIL [Lavrač und Džeroski, 1992], und RDT/FOIL [Lindner und Robers, 1994] verwendet. Aus der Sicht des Knowledge Discovery ist diese Anwendung allerdings nicht akzeptabel. Hier wird besonderer Wert auf die „reale"Anwendung gelegt. Um diesem Punkt gerecht zu werden habe ich ein Testszenario aus dem Bereich der Roboternavigation (siehe Abschnitt 7.2) für weitere experimentelle Tests mit RDT/DB ausgewählt.

7.1.2 Testaufbau für KRK

Der Sachbereich ist in der Datenbank durch die Relation illegal (siehe Tabelle 2) mit ca. 3500 Einträgen repräsentiert, wobei das Schachbrett in ein numerisches Koordinatensystem eingeteilt ist.

Weiterhin ist in der Datenbank als Hintergrundwissen die Relation adjacent

```
m3(P1,P2,P3,P4,P5,C):P1(E,X1) & P2(E,Y1) & P3(E,X2) & P4(E,Y2) &
                     P5(X1,X2)& P5(Y1,Y2)  --> C(E).
```

Die Ergebnisse habe ich in der Tabelle 4 zusammengefaßt.

Lernverfahren	Beispielanzahl	Regeln	Lernzeit	θ-Test bei zu spez. Hypothesen
RDT	354	< 600	< 20 Std.	ja
RDT/DB	3504	191	12 Std. 32 Min.	ja
RDT/DB	3504	191	8 Std. 5 Min.	nein

Tabelle 4: Lernergebnisse zu illegal

7.2 Tests im Lernszenario Roboternavigation

7.2.1 Sachbereich

Das Lernszenario ist aus dem aktuellen Forschungsprojekt B-Learn II[4] entnommen. In dem Lernszenario geht es um die Bewegung eines Roboters in einem einfachen Raum (siehe Bild 1), in dem sich dieser Roboter durch die Messungen seiner 24 Sensoren orientieren soll. Hierzu werden die Messungen auf verschiedenen Ebenen zu Definitionen von Begriffen abstrahiert. Eine genaue Beschreibumg der verschiedenen Abstraktionsebenen geben Morik und Rieger in [Morik und Rieger, 1993]. Die Messungen der einzelnen Sensoren werden in Intervalle unterteilt, die einen linearen Messungsverlauf beschreiben. Diese Intervalle werden *basic features* genannt. Für den Trace 24 und den Sensor 5 erhält man folgende Sequenz von *basic features* (siehe [Klingspor, 1994]).

- `no_measurement(t24,0r,s5,1,14,_)`
- `decreasing (t24,0r,s5,14,15,-22)`
- `incr_peak (t24,0r,s5,15,16,47)`
- `decreasing (t24,0r,s5,16,18,-30)`
- `no_measurement(t24,0r,s5,18,22,_)`
- `decreasing (t24,0r,s5,22,26,-30)`

[4]Das Projekt B-Learn II (P7274) wird von der EG und dem Forschungsministerium von NRW unterstützt

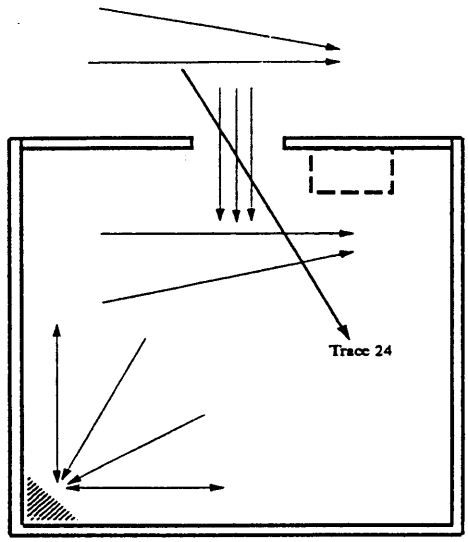


Abbildung 1: Roboterbewegung in einem Raum

Die *basic features* haben als Argumente die Tracenummer[5], die relative Orientierung des Sensors, die Sensorbezeichnung, Start- und Endzeit der Messung und den Gradienten. Jedes *basic feature* ist als Relation in der Datenbank repräsentiert, insgesamt gibt es 9 verschiedene *basic features*. Neben den *basic features* gibt es noch *sensor features*, die eine von einem Sensor gemessene Kantenkonstellation des Raums beschreibt.

Lernziel ist, Abfolgen von *basic features* zu lernen, die ein *sensor feature* beschreiben, d.h ein *sensor features* beschreiben das Verhalten eines Sensors vom Zeitpunkt t1 bis tn. Insgesamt gibt es 4 *sensor features*. Die Argumente der *sensor features* sind die Tracenummer, die Sensorbezeichnung, die Start- und Endzeit der Abfolge und die Bewegungsrichtung.

7.3 Testaufbau

Neben den *basic features* und den *sensor features* wurden fünf Metaprädikate für die Lernläufe verwendet, von denen ich exemplarisch drei angeben möchte:

```
m1(P1,P2,M,C)       :P1(T,S,X1,X2) & P2(T,S,X2,X3) & M(0)
                     --> C(T,S,X1,X3,0)
m2(P1,P2.P3,M,C)    :P1(T,S,X1,X2) & P2(T,S,X2,X3)
                     & P3(T,S,X3,X4) & M(0)
```

[5]Schlüsselattribute sind unterstrichen.

```
                        --> C(T,S,X1,X4,O)
m3(P1,P2.P3,P4,M,C):P1(T,S,X1,X2) & P2(T,S,X2,X3)
                    & P3(T,S,X3,X4) & P4(T,S,X4,X5) & M(O)
                    --> C(T,S,X1,X5,O)
```

Weiterhin mußte ich, da das Konstantenlernen in RDT/DB noch nicht implementiert ist, Prädikate für die Bewegungsrichtung vorgeben.
Die Lernergebnisse für das *sensor feature* s_jump in Tabelle 5 dargestellt.

Lernverfahren	Anzahl der Fakten	Regeln	Lernzeit	θ-Test bei zu spez. Hypothesen
RDT	2335	43	10 Std. 32 Min.	ja
RDT/DB	2265	39	¡ 24 Std.	ja
RDT/DB	2265	39	2 Std. 46 Min.	nein

Tabelle 5: Lernergebnisse zum *sensor feature* s_jump

7.4 Erste Schlußfolgerungen aus den Tests

Die Tests im KRK-Sachbereich zeigen, daß in diesem Szenario mit RDT/DB gelernt werden kann und das große Faktenbasen (hier ca. 17525 Fakten) bearbeitet werden können, die nur noch Beschränkungen von Seiten der Datenbank unterliegen. Die hohen Lernzeiten sind durch die für die Verfahren nicht besonders geeignete Repräsentation zu erklären. (Vergleiche auch [Lindner und Robers, 1994])

Im Lernszenario Roboternavigation war das Ziel eine „reale“ Anwendung durchzuführen. Eine solche Anwendung, ausgehend von einer nicht überschaubaren und interpretierbaren Menge an Meßdaten, scheint mir besonders für das Knowledge Discovery interessant. Das Lernergebnis ist, bis auf die vier nicht gelernten Regeln, identisch mit den im Projekt erzielten Ergebnissen.

8 Ausblick

Der nächste Schritt in der Anwendung von RDT auf relationale Datenbanken wird die Implementierung des Konstantenlernens sein. Außerdem werden

ich noch weitere Test mit RDT/DB, RDT und vielleicht einigen anderen Verfahren aus dem maschinellen Lernen durchführen. Dazu gehört dann auch eine Analyse der Vor- und Nachteile der Verfahren. Anfang August wird dann eine ausführliche Beschreibung über die Anwendung des Lernverfahrens RDT auf relationale Datenbanken als Diplomarbeit an der Universität Dortmund vorliegen.

Literatur

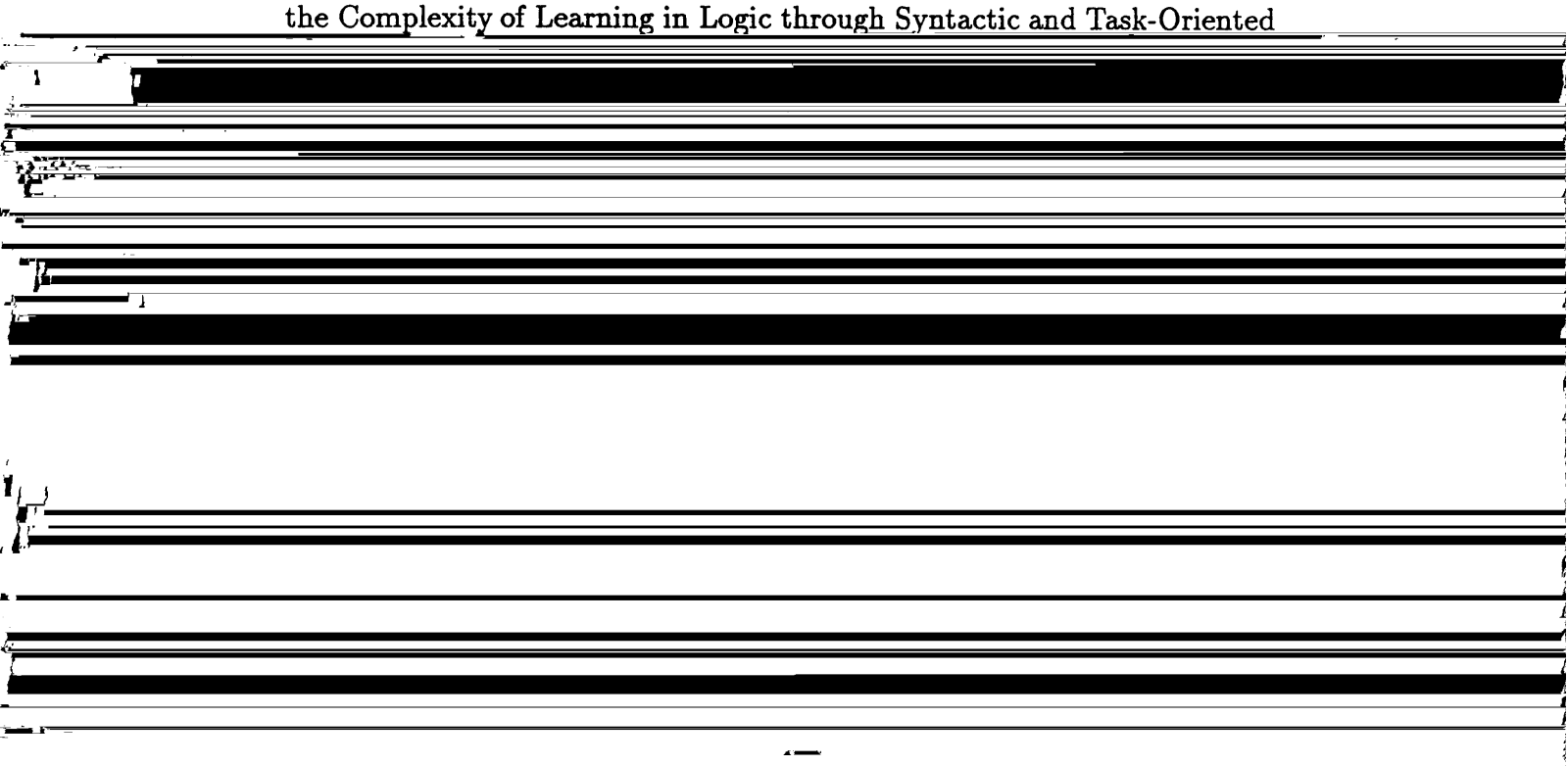
[Clark und Niblett, 1989] P. Clark und T. Niblett. The CN2 Induction Algorithm. *Machine Learning*, 3(4):261–284, 1989.

[Džeroski und Lavrač, 1993] S. Džeroski und N. Lavrač. Inductive Learning in Deductive Databases. *IEEE Transactions on Knowledge and Data Engineering*, 5(6):939–949, Dezember 1993.

[Frawley *et al.*, 1991] W. Frawley, G. Piatetsky-Shapiro und C. Matheus. Knowledge Discovery in Databases: An Overview. In G. Piatetsky-Shapiro und W. Frawley (Hrsg.), *Knowledge Discovery in Databases*, S.1–27, Cambridge, Mass., 1991. AAAI/MIT Press.

[Holsheimer und Siebes, 1993] M. Holsheimer und A. Siebes. Data Mininig: The Search for Knowledge in Databases. CS-R9406, CWI, P.O. Box 94079, 1090 GB Amsterdam, Netherland, 1993. ISSN 0169-118X.

[Kietz und Wrobel, 1992] Jörg-Uwe Kietz und Stefan Wrobel. Controlling the Complexity of Learning in Logic through Syntactic and Task-Oriented



Kapitel 16, S. 335 – 360. Academic Press, London, 1992. Also available as Arbeitspapiere der GMD No. 503, 1991

[Lindner und Robers, 1994] G. Lindner und U. Robers. Experimentelle Analyse zweier logik –basierter Lernverfahren. Forschungsbericht 6, Universität Dortmund, Fachbereich Informatik, Lehrstuhl VIII, 1994. ISSN 0943-4135.

[Matheus *et al.*, 1993] C.J. Matheus, P. Chan und G. Piatetsky-Shapiro. Systems for Knowledge Discovery in Databases. *IEEE Transactions on Knowledge and Data Engineering*, 5(6):903–913, Dezember 1993.

[Michalski *et al.*, 1986] Ryszard Michalski, Ivan Mozetic, Jan Hong und Nada Lavrač. The multi-purpose incremental learning system AQ15 and its testing application on three medical domains. In *Procs. of the National Conference on Artificial Intelligence*, S.1041 – 1045, San Mateo, CA, 1986. Morgan Kaufmann.

[Mitchell, 1982] Tom M. Mitchell. Generalization as Search. *Artificial Intelligence*, 18(2):203 – 226, 1982.

[Morik *et al.*, 1993] K. Morik, S. Wrobel, J.-U. Kietz und W. Emde. *Knowledge Acquisition and Machine Learning - Theory, Methods, and Applications.* Academic Press, London, 1993.

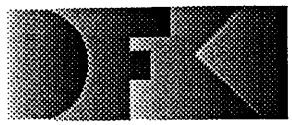
[Morik und Rieger, 1993] Katharina Morik und Anke Rieger. Learning Action-Oriented Perceptual Features for Robot Navigation. In Attilio Giordana (Hrsg.), *Learning Robots - Proceedings of ECML Workshop.* 1993.

[Plotkin, 1970] Gordon D. Plotkin. A note on inductive generalization. In B. Meltzer und D. Michie (Hrsg.), *Machine Intelligence*, Kapitel 8, S.153–163. American Elsevier, 1970.

[Quinlan, 1983] J. R. Quinlan. Learning Efficient Classification Procedures and Their Application to Chess End Games. In R.S. Michalski, J.G. Carbonell und T.M. Mitchell (Hrsg.), *Machine Learning - An Artificial Intelligence Approach*, S.463 – 482. Tioga, Palo Alto, CA, 1983.

[Quinlan, 1986] R.J. Quinlan. Induction of Decision Trees. *Machine Learning*, 1(1):81–106, 1986.

[Ullmann, 1989] J.D. Ullmann. *The New Technologies*, Band II der *Principles of Databases and Knowledge-Base Systems.* Computer Science Press, 1989.



Deutsches Forschungszentrum für Künstliche Intelligenz GmbH

DFKI
-Bibliothek-
PF 2080
67608 Kaiserslautern
FRG

DFKI Publikationen

Die folgenden DFKI Veröffentlichungen sowie die aktuelle Liste von allen bisher erschienenen Publikationen können von der oben angegebenen Adresse oder per anonymem ftp von ftp.dfki.uni-kl.de (131.246.241.100) unter pub/Publications bezogen werden.

Die Berichte werden, wenn nicht anders gekennzeichnet, kostenlos abgegeben.

DFKI Publications

The following DFKI publications or the list of all published papers so far are obtainable from the above address or via anonymous ftp from ftp.dfki.uni-kl.de (131.246.241.100) under pub/Publications.

The reports are distributed free of charge except if otherwise indicated.

DFKI Research Reports

RR-93-12
Pierre Sablayrolles: A Two-Level Semantics for French Expressions of Motion
51 pages

RR-93-13
Franz Baader, Karl Schlechta:
A Semantics for Open Normal Defaults via a Modified Preferential Approach
25 pages

RR-93-14
Joachim Niehren, Andreas Podelski, Ralf Treinen:
Equational and Membership Constraints for Infinite Trees
33 pages

RR-93-15
Frank Berger, Thomas Fehrle, Kristof Klöckner, Volker Schölles, Markus A. Thies, Wolfgang Wahlster: PLUS - Plan-based User Support Final Project Report
33 pages

RR-93-16
Gert Smolka, Martin Henz, Jörg Würtz: Object-Oriented Concurrent Constraint Programming in Oz
17 pages

RR-93-17
Rolf Backofen:
Regular Path Expressions in Feature Logic
37 pages

RR-93-18
Klaus Schild: Terminological Cycles and the Propositional μ-Calculus
32 pages

RR-93-20
Franz Baader, Bernhard Hollunder:
Embedding Defaults into Terminological Knowledge Representation Formalisms
34 pages

RR-93-22
Manfred Meyer, Jörg Müller:
Weak Looking-Ahead and its Application in Computer-Aided Process Planning
17 pages

RR-93-23
Andreas Dengel, Ottmar Lutzy:
Comparative Study of Connectionist Simulators
20 pages

RR-93-24
Rainer Hoch, Andreas Dengel:
Document Highlighting —
Message Classification in Printed Business Letters
17 pages

RR-93-25
Klaus Fischer, Norbert Kuhn: A DAI Approach to Modeling the Transportation Domain
93 pages

RR-93-26
Jörg P. Müller, Markus Pischel: The Agent Architecture InteRRaP: Concept and Application
99 pages

RR-93-27
Hans-Ulrich Krieger:
Derivation Without Lexical Rules
33 pages

RR-93-28
Hans-Ulrich Krieger, John Nerbonne, Hannes Pirker: Feature-Based Allomorphy
8 pages

RR-93-29
Armin Laux: Representing Belief in Multi-Agent Worlds viaTerminological Logics
35 pages

RR-93-30
Stephen P. Spackman, Elizabeth A. Hinkelman: Corporate Agents
14 pages

RR-93-31
Elizabeth A. Hinkelman, Stephen P. Spackman: Abductive Speech Act Recognition, Corporate Agents and the COSMA System
34 pages

RR-93-32
David R. Traum, Elizabeth A. Hinkelman: Conversation Acts in Task-Oriented Spoken Dialogue
28 pages

RR-93-33
Bernhard Nebel, Jana Koehler: Plan Reuse versus Plan Generation: A Theoretical and Empirical Analysis
33 pages

RR-93-34
Wolfgang Wahlster:
Verbmobil Translation of Face To Face Dialogs

RR-93-42
Hubert Comon, Ralf Treinen: The First-Order Theory of Lexicographic Path Orderings is Undecidable
9 pages

RR-93-43
M. Bauer, G. Paul: Logic-based Plan Recognition for Intelligent Help Systems
15 pages

RR-93-44
Martin Buchheit, Manfred A. Jeusfeld, Werner Nutt, Martin Staudt: Subsumption between Queries to Object-Oriented Databases
36 pages

RR-93-45
Rainer Hoch: On Virtual Partitioning of Large Dictionaries for Contextual Post-Processing to Improve Character Recognition
21 pages

RR-93-46
Philipp Hanschke: A Declarative Integration of Terminological, Constraint-based, Data-driven, and Goal-directed Reasoning
81 pages

RR-93-48
Franz Baader, Martin Buchheit, Bernhard Hollunder:

10 pages

RR-93-35
Harold Boley, François Bry, Ulrich Geske (Eds.): Neuere Entwicklungen der deklarativen KI-Programmierung — *Proceedings*
150 Seiten
Note: This document is available only for a nominal charge of 25 DM (or 15 US-$).

RR-93-36
Michael M. Richter, Bernd Bachmann, Ansgar Bernardi, Christoph Klauck, Ralf Legleitner, Gabriele Schmidt: Von IDA bis IMCOD: Expertensysteme im CIM-Umfeld
13 Seiten

RR-93-38
Stephan Baumann: Document Recognition of Printed Scores and Transformation into MIDI

Cardinality Restrictions on Concepts
20 pages

RR-94-01
Elisabeth André, Thomas Rist: Multimedia Presentations: The Support of Passive and Active Viewing
15 pages

RR-94-02
Elisabeth André, Thomas Rist: Von Textgeneratoren zu Intellimedia-Präsentationssystemen
22 Seiten

RR-94-03
Gert Smolka: A Calculus for Higher-Order Concurrent Constraint Programming with Deep Guards
34 pages

RR-94-07
Harold Boley: Finite Domains and Exclusions as First-Class Citizens
25 pages

RR-94-08
Otto Kühn, Björn Höfling: Conserving Corporate Knowledge for Crankshaft Design
17 pages

RR-94-10
Knut Hinkelmann, Helge Hintze:
Computing Cost Estimates for Proof Strategies
22 pages

RR-94-11
Knut Hinkelmann: A Consequence Finding Approach for Feature Recognition in CAPP
18 pages

RR-94-12
Hubert Comon, Ralf Treinen:
Ordering Constraints on Trees
34 pages

DFKI Technical Memos

TM-92-04
Jürgen Müller, Jörg Müller, Markus Pischel, Ralf Scheidhauer:
On the Representation of Temporal Knowledge
61 pages

TM-92-05
Franz Schmalhofer, Christoph Globig, Jörg Thoben:
The refitting of plans by a human expert
10 pages

TM-92-06
Otto Kühn, Franz Schmalhofer: Hierarchical skeletal plan refinement: Task- and inference structures
14 pages

TM-92-08
Anne Kilger: Realization of Tree Adjoining Grammars with Unification
27 pages

DFKI Documents

D-93-14
Manfred Meyer (Ed.): Constraint Processing – Proceedings of the International Workshop at CSAM'93, July 20-21, 1993
264 pages
Note: This document is available only for a nominal charge of 25 DM (or 15 US-$).

D-93-15
Robert Laux:
Untersuchung maschineller Lernverfahren und heuristischer Methoden im Hinblick auf deren Kombination zur Unterstützung eines Chart-Parsers

D-93-16
Bernd Bachmann, Ansgar Bernardi, Christoph Klauck, Gabriele Schmidt: Design & KI
74 Seiten

D-93-20
Bernhard Herbig:
Eine homogene Implementierungsebene für einen hybriden Wissensrepräsentationsformalismus
97 Seiten

D-93-21
Dennis Drollinger:
Intelligentes Backtracking in Inferenzsystemen am Beispiel Terminologischer Logiken
53 Seiten

D-93-22
Andreas Abecker:
Implementierung graphischer Benutzungsoberflächen mit Tcl/Tk und Common Lisp
44 Seiten

D-93-24
Brigitte Krenn, Martin Volk:
DiTo-Datenbank: Datendokumentation zu Funktionsverbgefügen und Relativsätzen
66 Seiten

D-93-25
Hans-Jürgen Bürckert, Werner Nutt (Eds.):
Modeling Epistemic Propositions
118 pages
Note: This document is available only for a nominal charge of 25 DM (or 15 US-$).

D-93-26
Frank Peters: Unterstützung des Experten bei der Formalisierung von Textwissen
INFOCOM:
Eine interaktive Formalisierungskomponente
58 Seiten

D-93-27
Rolf Backofen, Hans-Ulrich Krieger, Stephen P. Spackman, Hans Uszkoreit (Eds.):
Report of theEAGLES Workshop on Implemented Formalisms at DFKI, Saarbrücken
110 pages

D-94-01
Josua Boon (Ed.):
DFKI-Publications: The First Four Years
1990 - 1993
75 pages

D-94-02
Markus Steffens: Wissenserhebung und Analyse zum Entwicklungsprozeß eines Druckbehälters aus Faserverbundstoff
90 pages

D-94-03
Franz Schmalhofer: Maschinelles Lernen:
Eine kognitionswissenschaftliche Betrachtung
54 pages

D-94-04
Franz Schmalhofer, Ludger van Elst:
Entwicklung von Expertensystemen:
Prototypen, Tiefenmodellierung und kooperative Wissensevolution
22 pages

D-94-06
Ulrich Buhrmann:
Erstellung einer deklarativen Wissensbasis über recyclingrelevante Materialien
117 pages

D-94-07
Claudia Wenzel, Rainer Hoch:
Eine Übersicht über Information Retrieval (IR) und NLP-Verfahren zur Klassifikation von Texten
25 Seiten

D-94-08
Harald Feibel: IGLOO 1.0 - Eine grafikunterstützte Beweisentwicklungsumgebung
58 Seiten

D-94-09
DFKI Wissenschaftlich-Technischer Jahresbericht 1993
145 Seiten

D-94-10
F. Baader, M. Lenzerini, W. Nutt, P. F. Patel-Schneider (Eds.): Working Notes of the 1994 International Workshop on Description Logics
118 pages
Note: This document is available only for a nominal charge of 25 DM (or 15 US-$).

D-94-11
F. Baader, M. Buchheit, M. A. Jeusfeld, W. Nutt (Eds.):
Working Notes of the KI'94 Workshop:
KRDB'94 - Reasoning about Structured Objects: Knowledge Representation Meets Databases
65 Seiten

D-94-12
Arthur Sehn, Serge Autexier (Hrsg.): Proceedings des Studentenprogramms der 18. Deutschen Jahrestagung für Künstliche Intelligenz KI-94
69 Seiten

Proceedings des Studentenprogramms der 18. Deutschen Jahrestagung für Künstliche Intelligenz KI-94

Arthur Sehn, Serge Autexier

D-94-12
Document