No 3 − March 2003

www2.cs.ucy.ac.cy/projects/colognet

In this issue

Editorial

Antonis Kakas, Department of Computer Science, University of Cyprus, antonis@ucy.ac.cy

This is the third issue of the CoLogNet Newsletter, the official newsletter of the Network of Excellence in Computational Logic (CoLogNet). Like the previous issues of July 2002 and November 2002, the March Newsletter reports on the activities of the network and more generally on the international developments in the scientific field of computational logic

The third issue includes different and interesting contributions. You will come across reports of activities in recent international meetings in the areas of formal methods, logic−based agents, and automated reasoning. Of particular interest is the article on Combinatory Declarative Programming, as well as the report on the programming competition took place at the University of Applied Sciences in Moldova. Finally, the newsletter provides calendar information about upcoming events related to the network.

We would like to emphasize once more that the success of the newsletter depends on the continued interest of the CoLogNet members and other researchers outside the network to contribute with their own reports, articles, news items, reviews and any other appropriate information. In particular, letters or comments on the newsletter itself with suggestions and criticisms for the improvement and expansion of the newsletter will be most welcome and greatly appreciated. In the newsletter web site you have the opportunity to open or take part in a discussion on a topic of your choice.

We would like to thank all the contributors for helping us put together an interesting third issue of the newsletter. We hope that with the support of the network members and the community at large we will be able to continue improving the newsletter in the future.

Executive Council report

Jörg Siekmann

The first year of CoLogNET ended on 31 December 2002, which has been an important year for CoLogNET: In addition to consolidating and enlarging the activities of CoLogNET, the infrastructure was built up and many member nodes have been established. Here is a summary of recent activities and new developments.

Executive Council and Research Areas

The second Executive Council (EC) meeting will be held on 28 April 2003 in Lisbon. Similar to the past EC meeting in Madrid, a general business meeting will be co−located to the EC meeting as a half−day meeting on 27 April 2003. For various reasons the EC meeting in Lisbon will be of significance, as the network was assigned to a new project officer, Fabrizio Sestini, who has recently joined FET. The EC meeting in Lisbon will be the first informal review meeting with an external expert as proposed by the Commission. The CoLogNET periodic progress report for the first term will be sent with a small delay to the Commission in March.

Task Force Relations

Most of the industry−related activities in 2003 will organised by TF1 and TF2. Workshops with panellist from industry will be co−located with major conferences i.e. FMEÈ3, CPÈ3, ICLPÈ3. As agreed at the EC meeting in Madrid the main emphasis should be put on the involvement of industrial efforts within other work packages and the entire network. There should be more time for discussion and demo sessions of tools and software to bring in a wider audience not only from academia but also from industry. The proposals for industry−related initiatives that have been presented at the EC meeting in Madrid seemed rather promising and the future will show in how far these activities can be realized.

Infrastructure Building

The main functions of the network are to:

- coordinate research activities and infrastructure facilities
- coordinate and support research training and education initiatives
- promote cross−fertilisation and exploitation of results by the European industry
- promote worldwide cooperation
- establish IFCoLog as a legal entity
- contribute to standard debates and develop technological roadmaps for various sub−areas of CL.

To support these activities, a CoLogNET portal was developed, which will be officially launched at the EC meeting in Lisbon on 28 April 2003. It is already online now, prior to the official website launch.

Access to the site of IFCoLog and CoLogNET can be obtained via an attractive and general entry point www.compulog.info which provides a short introduction What is Computational Logic and links to further information in CoLogNET and the International Federation of Computational Logic (IFCoLog). The websites of IFCoLog and CoLogNET have been completely redesigned with a similar architectural structure. The CoLogNET portal offers a wide spectrum of services for internal management aspects as well as external representation of CoLogNET. Internal management aspects will be realised via the protected info site at www.colognet.org . There is an access hierarchy implicitly involved for the information in this slot. Access to protected info can only be provided for CoLogNET principal contractors and registered individual members

and member nodes. The CoLogNET portal also provides exhaustive information on the management structure of the Consortium and the Network areas as well as on education and training activities, jobs and grants as well as a Who is Who in Logic. Future members can register online and submit their application for membership in CoLogNET to gain visibility in the field of Computational Logic.

CoLogNET disseminates information at major conferences that are of interest for the networks members through its mobile booth, a set of slides, information flyers and posters and newsletter issues. Mobile booth, information flyer and slides can be ordered from the co−ordinating site, DFKI. CoLogNET also disseminates information through its area and work package leader who are also strongly committed to further promote their area−related activities. In addition the network provides many other infrastructure facilities within its specific areas and via maintenance of mailing lists.

With the rapid increase in the number of network nodes electronic mailing lists have become very important. The following shows various electronic facilities that are currently on offer:

1. With respect to managerial tasks an exhaustive set of mailing lists have been created at the very beginning of the project which take account of the partners involved in specific work packages, areas and task forces: WP1@colognet.org , Area1@colognet.org , TF1@colognet.org and respectively WP2, WP3, WP4, WP 5, WP6, WP7, WP8, WP9, WP10, WP11, WP12, WP13, Area2, Area3, Area4, Area5, Area6, Area7, Area8, Area9, TF2, TF3.
2. The main mailing list EC@colognet.org contains all members of the CoLogNET Executive Council and the most important notifications are forwarded via this list.
3. colognet−nodes@colognet.org contains both the officially approved and registered members of CoLogNET. This includes the nodes contact persons as well as the nodes mailing lists which take account of the node s individual members. It can be used to announce matters of general interest to all NOE nodes and also to other potentially interested parties. General matters like conference, workshops, electronic debates, industry−related events can be announced on this list.
4. e−eu−contacts@colognet.org pays particular attention to nodes and contacts that have been established in Eastern European Countries.

As already pointed out access to protected info can only be provided for CoLogNET principal contractors and registered individual members and member nodes. This will allow registered members remote access to CoLogNET related documents and shared knowledge around Europe. Further mailing lists will be established and maintained such as a list which contains only the contact person of the nodes of CoLogNET.

The lack of communication, insufficient new email addresses and electronic affiliation listing was the reason why only a few of the former nodes of Compulog were incorporated into CoLogNET. As the search of the new email addresses turned out to be rather time consuming it is expected that the former members of Compulog will now be integrated into CoLogNET within the next period.

Nodes and Membership

So far the Executive Council has approved 47 new nodes for membership in CoLogNET which sums up to about 150−200 new individual members of CoLogNET. CoLogNET nodes are required to provide the network with specific data on their group members, field of research, size of organisation, scope of activities and selected publications. These data are stored in a database and can be viewed from the CoLogNET portal. The entire package of membership documents will be forwarded to the Commission for approval. As soon as the Commission approves the request for membership, the CoLogNET nodes are official activists in the network, including the old nodes of COMPULOG. We expect a total number of about 120 nodes in Europe.

IFCOLOG

The network provided substantial support to the International Federation on Computational Logic, whose infrastructure and website have been completely renewed, see www.ifcolog.org .

Competition "Programmer '02"

Nicolae Pelin
Rector of the University of Applied Sciences of Moldova
e−mail: edu@usam.md

On 1 December 2002, the University of Applied Sciences of Moldova carried out a programming competition named "Best programmer 2002". The main purpose of this competition was to demonstrate that human oriented problems i.e. Artificial Intelligence direction are best solved using Logic Programming Languages. However every competition participant had a possibility to choose any of three programming language (PROLOG, PASCAL, C++), even after they saw the problems, to change the programming language on the competition runtime, or even to use two different programming languages for different problems.

The submission deadline was 23 November 2002. Information about this competition was distributed to different schools, Lyceums, Universities of Moldova. Twenty−one submissions received from eleven different institutions, by 23 November 2002 which were divided in three different levels: Specialists, University and college students, Lyceum pupils.

On 1 December 2002 only thirteen participants were able to participate in two levels: Lyceum pupils and Specialist. Computers were randomly distributed between all the participants half an hour before competition (for preparing for competition), and then disconnected from LAN.

The competition last for three hours, and provided three problems: Pawns, Rebus and The Farmer. The problems were selected by the competition committee from a multitude of other problems 15 minutes before competition starts (by voting of committee members). The competition committee consisted of three members, specialists from three different Universities:

1. Vitalie Cotelea, Ph.D., Academy of Economic Sciences
2. Victoria Boghanastiuc, Ph.D., Politechnical University of Moldova
3. Serghei Pelin, master of science, The University of Applied Sciences of Moldova

Immediately after the time was up, the competition committee thanked every participant and wrote down the results. The winners are:

- Lyceum level: Alexander Pelin (picture on the left)
  - ♦ lyceum pupil, XII−grade, real profile, specialization − "Informatics and mathematics", Lyceum nr. 2 − "Gaudeamus", Chisinau, Moldova)
- Specialists level: Ian Orlovski (picture on the right)
  - ♦ master of science, logic programming lecturer at the University of Applied Sciences of Moldova

The announcement and awarding of the winners took place on 5 December 2002 before the scientifico−metodical conference for the 10th anniversary of The University of Applied Sciences of Moldova.

Every participant was awarded by a certificate of participation, a set of three books on Logic Programming published in the University of Applied Sciences of Moldova and a CD with different information and programming languages.

The competition winners were awarded with Visual Prolog Professional (on behalf of Prolog Development Center, Copenhagen), financial contribution from COLOGNET − 100 EURO, and a financial contribution from University of Applied Sciences of Moldova − 500 Lei.

The results of this competition shows, as it was expected, that using of logic programming languages permits to obtain higher results, in shorter terms. This can be a good example for further popularization of Logic Programming in Moldova.

We hope to organize this competition in the future.

A Double Journal Issue and a Book: Logics of Formal Specification Languages

Dines Bjorner
Computer Science and Engineering, Informatics and Mathematical Modelling
Technical University of Denmark
E−−Mail: db@imm.dtu.dk
www.imm.dtu.dk/colognet

Abstract

Work Package 7 of the CoLogNET effort aims at publishing a pair of issues of the Slovak Academy of Science Journal CAI: Computers and Informatics, and, to also publish an edited, perhaps extended version of the CAI papers, as a book. The common, working title is: *Logics of Specification Languages.* This page will explain the current status of this endeavor. Throughout, when we refer to *`Specification'*, we mean *`Computing System Specification'* or a specification related to computing systems.

What is, and Why a Formal Specification Language ?

We explain and motivate the concept of `Formal Specification Languages'.

What is a Formal Specification Language?

A language is a set of sentences, or statements.

Formal Languages

A formal language has:

- a formal syntax,
- formal semantcs,
- a formal proof system,
- but an informal pragmatics.

That is, is a language:

- whose set of sentences can be mathematically delineated,
- in which every valid sentence can be ascribed a precise mathematical meaning,
- and for which a (mathematical logic) proof system can be established.

We shall not here be concerned with the syntax, nor with the general semantics of formal languages.

Formal Specification Languages

A formal specification language is a formal language that can be used to describe computing systems, and at a level of abstraction that may not lend itself to mechanization. That is, where specifications emphasize properties or abstractions, or both, such that proofs of properties of a specification, or of relations between two, given specifications can be carried out −− by man or machine, or both in collaboration.

Why Formal Specification?

Understanding computing systems is difficult, and developing them is hard. Abstract models, usually, is a key to approximate understanding, and is usually a good basis from which to develop less abstract models. Abstract models of computing systems, typically of software, are usually expressed in some formal specification language. It has shown to be easier, quicker and more trustworthy to develop first abstract models, and from them, in successive stages of development, ``the real thing", for example software −− such that the result (software) possess such properties as for example *correct with respect to abstract model.*

Some Formal Specification Languages

Depending on the application area, a number of formal specification languages are now reasonably readily available. Alphabetically we list a few:

- **ASM:** Abstract State Machines
- **B** (For: Bourbaki, a group of French Mathematicians)
- **CafeOBJ**
- **CASL:** Common Algebraic Specification Language
- **DC:** The Duration Calculi
- **Maude**
- **RSL:** RAISE Specification Language
- **STeP:** Temporal Logic of Reactive Systems (Stanford Temporal Prover)
- **TLA+:** Temporal Logic of Actions
- **VDM−SL:** Vienna Development Method Specification Language
- **Z** (for Zermelo's Set Theory)

What is `Logics of Specification Languages' ?

A specification language semantics has been formally defined in some meta−language, typically in some mathematical notation, say MN.

And a specification language has been given a proof system: A set of axiom schemes, inferences rules, and such lemmas and theorems which can be proven to hold on the basis of the axiom schemes and the rules of inference. Together they form an initial theory, the proof system, PS, of the formal specification language.

By the `Logics of Specification Languages' we mean the two logics underlying the mathematical notation MN, and the proof system PS.

The Logic of Specification Language Definition Languages

Let SL be any of the specification languages implied above.

Now the first aim of publishing the CAI double journal issue and book is to seek answers the following questions: ``*In which kind of notation: Mathematics, some mathematical logic, or some formal specification language, is the semantics of SL expressed ? Which are the special features of that notation ? Are there special logic properties or logic features of the semantics definition ? Any related logic issues ?"*

The Logic of Specification Language Proof Systems

Let SL be any of the specification languages implied above.

Now the second aim of publishing the CAI double journal issue and book is to seek answers the following questions: ``*What is the logic of the proof system: Two valued, three valued, over total, or over partial functions ? Any proof of consistency and relative completeness ? Any further logic issues ?*

Outline of a Double Journal Issue: Logics of SLs

The current list of authors is:

1. ASM: Wolfgang Reisig
2. B: Dominique Cansell and Dominique Mery
3. CafeOBJ: Kokichi Futatsugi and Radzvan Diaconescu
4. CASL: Till Mosakovski, Andrzej Tarlecki and Don Sannela
5. DC: Duration Calculi: Zhou Chaochen and Michael R. Hansen
6. Maude: Jose Meseguer
7. RAISE: Chris George, Anne E. Haxthausen and Søren Prehn
8. "STeP": Amir Pnueli and Zohar Manna −− Temporal Logic of Reactive Systems
9. TLA+: Stephan Mertz and Lesli Lamport
10. VDM: Peter Gorm Larsen and John Fitzgerald
11. Z: Jonathan Bowen, Steve Schneider, Martin C Henson and Steve Reeves

One or two of the above may drop out of the race to meet publication deadlines !

Current Status

As of November 28, 2002, most authors are busy working on draft versions of their promised papers. The editor of CAI, Prof. Rowan is eagerly awaiting papers for the first of the two journal issues. Two publishers have been contacted. Both have expressed interest.

CoLogNET INDUSTRY DAY − I

Dines Bjorner
Computer Science and Engineering, Informatics and Mathematical Modelling
Technical University of Denmark
E−−Mail: db@imm.dtu.dk www.imm.dtu.dk/colognet

Abstract

Work Package 7 of the CoLogNET effort is gathering a group of leaders from the European IT/Software industry. Specifically those industries which rely significantly on the use of `formal methods.' The reason for doing this will be explained on this page −− as will the events taken accordingly.

Formal Methods

By a method we mean

- A set of principles
- for *selecting* and *applying*
- techniques
- for *analyzing* problems
- and *synthesizing* (constructing) solutions (artifacts) to these problems
- using tools.

We emphasize the triplet of *principles, techniques* and *tools***.** By a *formal method* is meant a method some of whose techniques and tools are based on *mathematical theories***.** In the context of the CoLogNET endeavor `formal methods' apply to software (as well as hardware) problems and artifacts.

Among tools we typically such as formal specification languages, theorem provers, model checkers and other software packages for the support of the use of specification languages and development of software (as well as hardware).

CoLogNET + FME Industry Day

FLoC'02, the Federated Logic Conference, held July 20 −− August 1, 2002, in Copenhagen, featured a full day meeting, a *FLoC'02 Formal Methods Industrial Day*, with presentations and discussions, on industry issues wrt. formal methods.

The event was co−sponsored by *CoLogNET* and *FME: Formal Methods Europe*. FME is an association of industry and university people aiming to further the uptake of formal methods in industry.

Some 10−12 prominent, management−level industry people were invited. They represented leading−edge European industries that significantly rely on the research, development and use of formal methods.

Aims Objectives: Topics Covered

Aims

The aims of the a *FLoC'02 Formal Methods Industrial Day* were to present and discuss such issues as may be found specifically relevant to the European Formal Methods consuming (hardware/software) industry. Some

of the topics covered were:

- Increasing awareness of the possibilities of `formal methods' amongst potential, future customers;
- Expectations from university and college education & training wrt. formal methods.
- Expectations from European Community R&D programmes (such as the 6th framework IST, etc.).
- Co−ordination of relations to standardisation and other regional and international trade, profession, etc., bodies;
- Issues of
    - ♦ Accreditation
        - ◊ of formal methods−centered hardware/software house,
        - ◊ and of formal methods−based university curriculum.
    - ♦ Certification
        - ◊ of systematically/rigorously/formally developed software packages and systems,
        - ◊ and of software engineers who are professional in the areas of formal methods.
- (``Second sourcing") reassurance for existing customers.

Objectives

It was an objective of the a *FLoC'02 Formal Methods Industrial Day* to explore whether there is a need to form:

- *A Formal Methods Hardware/Software House Association*

  It was suggested that such an association could have as its objectives:

    - ♦ To influence, on national bases, university and college curricula wrt. `formal methods'.
    - ♦ To propagate, on national bases, that specific software and/or hardware packages or systems be formally developed, that is: certified and by accredited software and/or hardware houses.
    - ♦ To show potential clients that a formal methods industry is now an established fact, and that the association, so to speak, helps guarantee that clients can rely on ``second source" formal methods companies.
    - ♦ To represent industry interests in national, regional and international bodies such as ISO, ITU, CENELEC, etc.

Program

The program of presentations was:

Chairman: Dr John Fitzgerald

- Registration 8:30−9:00
- Opening: On CoLogNET and FME: Dines Bjørner 9:00−9:30
- Dr Martyn Thomas,Thomas Associates, UK 9:30−10:00
  Formal methods in industry: Disease and remedies
- Dr Dominique Bolignano, Trusted Logic, France 10:00−10:30
  Experience from applying formal methods to critical security issues
- Refreshment Break 10:30−11:00
- Peter Gorm Larsen, Systematic Software Engineering, Denmark11:00−11:30
  Practical use of VDM technology in industry
- Dr Jan Peleska, Verified Systems Intl. GmbH, Germany 11:30−12:00
  Formal methods−based testing for large scale industrial application

- Denis Sabatier, Clearsy, France 12:00−12:30
  The use of the B formal method to produce accurate and complete technical documents
- Lunch 12:15−14:00
- Dr Robin E. Bloomfield, Adelard, UK: 14:00−14:30
  Drivers in the application of formal method
- Summary Discussion −− and: What Next ? 14:30−15:30
- Refreshment Break 15:30−16:00

Conclusion

Amongst participants, active as well as passive, there was agreement that the a *FLoC'02 Formal Methods Industrial Day* event was successful: That a first start had been made on the ideas laid out in the Aims & Objectives.

FM'03: The Formal Methods Symposium to be held in Pisa in September 2003 will feature a second full day a *FLoC'02 Formal Methods Industrial Day*. It is planned, by that date, to formally announce the formation of a *Formal Methods Hardware/Software House Association.*

IJCAI−03 Workshop on Agents and Automated Reasoning

Volker Sorge
University of Birmingham, UK
Acapulco, Mexico, August 11, 2003
http://www.cs.bham.ac.uk/~vxs/ijcai03/index.html

Following recent interest in both the use of agents in automated reasoning, and in the application of automated reasoning to multi−agent systems, this workshop will bring together researchers from both communities to stimulate further cross fertilization of ideas.

The workshop is primarily aimed at two groups: Firstly, researchers developing distributed reasoning systems to exchange ideas on what agent techniques and architectures are particularly relevant for their field; Secondly, researchers in the multi−agent community who are interested in using automated reasoning techniques for their work or as a testing ground for their ideas.

Areas of Interest

Topics of interest include, but are not limited to:

- Distributed reasoning systems
- Agent−based reasoning systems
- Architectures for reasoning agents
- Communication protocols
- Distributed mathematical knowledge bases
- Reasoning techniques for multi−agent systems
- Logical foundations of multi−agent systems
- Specifications and ontologies for specialized reasoning agents
- Centralized mathematical knowledge bases accessed through a distributed architecture

Motivation

Over recent years in the automated reasoning community, there has been much interest in the distribution, combination and integration of techniques and systems using flexible methods. In this community, the multi−agent paradigm is becoming very influential as the mathematical systems are being modelled as agents in an agency. With the further distribution of reasoning systems over the internet, and the use of the semantic web to represent and distribute mathematical knowledge, this influence will continue to increase.

Conversely, multi−agent societies for mathematical reasoning, such as theorem proving and/or symbolic computation, are ideal and highly
non−trivial testbeds for the development of advanced multi−agent techniques themselves. In addition, formalisation of multi−agent architectures has led to increased usage of automated techniques for reasoning about agencies, and logics for doing so have emerged.

Thus, the link between agent technology and automated theorem proving is increasingly important. While distributed approaches to theorem
proving have succeeded to some extent, it has become clear that more flexible approaches will be required for more complex problems. This
has prompted a renewed interest in AI techniques, in particular multi−agent systems within distributed theorem proving. Moreover, this
interest has recently broadened to the integration of mathematical systems such as computer algebra systems,

theorem provers and constraint solvers, in order to solve problems which have resisted solution by single systems.

In addition, with the increased complexity of multi−agent architectures and increased sophistication of the interaction between agents, there is a need to formalise and reason about multi−agent systems. With the usage of complex, multi−agent systems set to explode because of the semantic web and grid computing, there has never been a more important time to devise logics and begin to reason about various aspects of agent technology.

Thus, we believe it is vital to the development of both automated reasoning and multi−agent systems that the two communities share discussions; this IJCAI workshop is intended to initiate such an interaction. Organising the workshop at such a large conference as IJCAI will enable us to attract researchers from these two, usually separate, communities, and this will stimulate the desired cross−fertilization.

Important Dates:

- Submissions received by: April 1st, 2003
- Author notification by: April 22nd, 2003
- Camera−ready version due: May 15th, 2003

Copies of accepted papers will be provided as a pre−proceedings at the workshop itself. Information about the workshop, together with abstracts of accepted papers, will be available via the WWW page: http://www.cs.bham.ac.uk/~vxs/ijcai03/index.html

Organizing Committee

- Volker Sorge/ Department of Computer Science, University of Birmingham, UK/ Email: V.Sorge@cs.bham.ac.uk
- Simon Colton/ Department of Computing, Imperial College, L/ Email: sgc@doc.ic.ac.uk
- Michael Fisher/ Department of Computer Science, University of Live/ Email: M.Fisher@csc.liv.ac.uk
- Jeremy Gow/ Interaction Centre, University College, L/ Email: j.gow@ucl.ac.uk

Logic−based Agent Implementation

A special issue of the Annals of Mathematics & Artificial Intelligence

Michael Fisher
Department of Computer Science
University of Liverpool
M.Fisher@csc.liv.ac.uk

Overview

The view of computational components as `agents' is widely used in contemporary software applications, such as Internet navigation, information management, autonomous process control, and e−commerce. The popularity of the agent paradigm stems not only from its intuitive and appealing nature, capturing the notions of flexibility and evolving behavior, but also from the range of theories, tools and techniques that have been developed over recent years for agent−based systems.

However, the languages in which agents are typically implemented are often standard (usually, object−oriented) languages, with few `agent' concepts included as a central part. Thus, the abstractions that agent−based systems developers must work with are not always appropriate for producing effective agent applications, especially where `intelligent' or `rational' behavior is required. In addition, with agent−based systems beginning to be used in both safety/mission critical (e.g. autonomous control) and business critical (e.g. e−commerce and security) software, it is clear that more precise, and logically well−founded, development techniques will be required for agent−based applications in the future.

Thus, the aim of this special issue is to bring together high−quality papers exhibiting leading edge research where a "logic−based" approach is taken to the implementation of agent−based systems (often comprising `rational' or `intelligent' agents). Our view of `logic' here is not restricted to classical logic and logic programming techniques and, indeed, much of the leading work in this area does not easily fit within such a classification. We here take a broad view of logic, and consider implementation techniques for agents that are inherently based in some logical formalism. This logical formalism may have been developed for a variety of reasons. For example, the use of logic may enhance the abstractions used in the language, the potential for verification, the type of tools that can be utilized or the breadth of behaviors that can be developed.

The Journal

The Annals of Mathematics and Artificial Intelligence (Editor−in−Chief: Martin Golumbic. Associate Editors: Endre Boros; Sarit Kraus; Anil Nerode) is intended to represent a wide range of topics of concern to scholars applying quantitative, combinatorial, logical, algebraic and algorithmic methods to Artificial Intelligence areas as diverse as decision support, automated deduction, reasoning, knowledge−based systems, machine learning, computer vision, robotics and planning. The journal is aimed at: applied logicians, algorithms and complexity researchers, Artificial Intelligence theorists and applications specialists using mathematical methods. For more details, see http://www.kluweronline.com/issn/1012−2443

Important Dates

- Submission deadline: 1st March 2003
- Author notification: 1st July 2003
- Revised papers due: 1st September 2003

Topics of Interest

The topics of interest of this special issue include (but are not limited to):

- implementation techniques (together with accompanying software engineering methods) for agent−based or multi−agent systems based on: logic programming (including, concurrent/parallel logic programming); constraints (including, constraint logic programming and concurrent constraint programming); intentional programming; situation calculus; executable modal/temporal logics; and multi−paradigm approaches
- advanced techniques within logic−based agents, such as: knowledge representation, nonmonotonic reasoning; planning, problem solving and deliberation; learning, evolution and adaption; cooperation, teamwork, negotiation and social ability; and decision theory for multi−agent systems
- applications of logic−based agents, for example in: robotics, autonomous process control; knowledge/information management, WWW search/ navigation; e−commerce and B2B applications; embedded components; telecommunications; and general purpose agent programming.

Submission of Manuscripts

We invite submission of full (typically, 20−40 pages) high−quality research papers for this special issue. These should neither have been previously published in their full form, nor be under review elsewhere. Submissions consisting of the paper (preferably PDF or Postscript via email but, alternatively, five hard−copies) including an abstract in which the role of logic in the particular approach used is clarified, should be sent to Michael Fisher (address given below)to arrive no later than 1st March 2003. Formatting instructions can be found at the journal's web site.

Queries concerning this special issue should be directed to any of the guest editors. Up−to−date information will also be available from http://www.csc.liv.ac.uk/~michael/LBAI03

Special Issue Editors

Juergen Dix
Department of Computer Science,
University of Manchester, Manchester, U.K.
email: dix@cs.man.ac.uk url: http://www.cs.man.ac.uk/~jdix

Michael Fisher
Department of Computer Science,
University of Liverpool, Liverpool L69 7ZF, U.K.
email: M.Fisher@csc.liv.ac.uk url: http://www.csc.liv.ac.uk/~michael

Hector Levesque
Department of Computer Science,
University of Toronto, Toronto, Canada
email: hector@cs.toronto.edu url: http://www.cs.toronto.edu/~hector

Leon Sterling
Department of Computer Science and Software Engineering,
University of Melbourne, Australia
email: mailto:leon@cs.mu.oz.au url: http://www.cs.mu.oz.au/~leon

Computational Logic and Agent−Based Systems −− Bridging the Gap

Michael Fisher
Department of Computer Science
University of Liverpool, Liverpool L69 7ZF, UK
M.Fisher@csc.liv.ac.uk http://www.csc.liv.ac.uk/~michael

This article provides a report on the first event within the "Logic and Agents" activity in CologNet. (Further details can be found at http://www.csc.liv.ac.uk/~michael/LBAI−Symp03/)

Background

The main aim of this activity, on "Logic and Agents", is to bring together expertise from the Computational Logic and Agent−Based Systems communities, via interaction between the CologNet and AgentLink networks of excellence, respectively. Formally, this activity is one of the main areas within CologNet and is also a special interest group within AgentLink.

The activity is funded by both networks, showing their commitment to this important area. This formalised activity aims to allow expertise/ideas to flow between the networks, thus providing more chance of productive technology transfer. Thus, in the future, this activity will contribute to the uptake of logic−based agent technology and logical methods for agent−based systems. This, in turn, may impact on forthcoming standards, for example where semantics/verification/compliance is an issue.

AgentLink

AgentLink [http://www.agentlink.org] is a large, and successful, EU Network of Excellence aimed at agent−based computing. AgentLink coordinates research and development activities in agent−based systems and supports a range of activities aimed at raising the profile, quality, and industrial relevance of agent systems research and development in Europe.

Among its many activities, AgentLink supports Special Interest Groups, "focusing on the development of communities around areas of strategic importance". The joint activity on "Logic and Agents" is a Special Interest Group within AgentLink.

Symposium

The work of this "Logic and Agents" activity is centred around two symposia, one on "logic−based agent implementation", the other on "logic−based agent verification", both of which provide very strong links between the networks. The first of these was held on 3rd February 2003 in Barcelona; the second is planned for 2004.

The aim of the symposium on "logic−based agent implementation" was to consolidate expertise, stimulate research/ collaboration in each area, showcase successful applications and enhance the uptake of logic−based agent technology. This symposium was co−located with a large event within the agent community, comprising the AgentLink Summer School, Agentcities/AgentLink Agent Technology Conference, and AgentLink Special Interest Group meetings. The aim was that each symposium should be co−organised by a leading EU researcher in the topic area. We are pleased that such a researcher agreed to be involved in this way, and thank Juergen Dix (Univ. Manchester, UK) who co−organised this symposium on "logic−based agent implementation".

Funding was provided by both AgentLink and CologNet. In addition, we would like to acknowledge support of Toshiba (Japan) and the AgentLink local organisers for this event. Thus, we supported a number of speakers, allowing them to participate in the event. We were also able to attract high−profile non−EU keynote speakers, one from Japan, the other from the USA. We were also pleased to involve industrial speakers, one from Europe, one from Japan.

Programme

The symposium was a one day event held on 3rd February 2003 at the Universitat Politecnica de Catalunya (UPC) in Barcelona. There were two non−EU keynote speakers: V.S. Subrahmanian (Univ. Maryland, USA) talking about "Scaling Heterogeneous Agent Systems" and Hisashi Hayashi (Toshiba, Japan) talking about "Plangent: A Dynamic Planning Mobile Agent System Based on Logic Programming".

In addition, we had eight well−known experts from across the breadth of "logic−based agent implementation":

- Wiebe van der Hoek (Univ. Liverpool, UK) − "Goals in Agent Programming Languages"
- Seif Haridi (SICS, SE) − "Implementation of Agents in Mozart"
- Giuseppe De Giacomo (Univ. Roma, IT) − "Deliberation and Execution in the Situation Calculus"
- Chiara Ghidini (Univ. Liverpool, UK) − "Using Executable Logics to Program Individual Rational Agents"
- Luis Moniz Pereira (Univ. Lisboa, PT) − "Implementing Rational Features for Agents in Logic Programming"
- Sverker Janson (SICS, SE) − "Trading Agents, Constraints, and Logic Programming"
- Monique Calisti (Whitestein Technologies, CH) − "Constraint Satisfaction Techniques for Intelligent Reasoning Agents"
- Maurizio Martelli (Univ. Genova, IT) − "The use of Logic Programming for Prototyping MAS Applications"

Summary

The presentations not only gave a broad view of the state of the art in logic−based agent implementation throughout both communities, but also allowed dissemination of key breakthroughs in this area. We believe that this event will enhance collaboration between groups, networks, academia/industry, etc. Finally, not only did this symposium allow participants to showcase successful applications concerning the implementation of agents using logic−based approaches, it also provided an opportunity to identify important future research issues and key requirements for wider use of these techniques.

Combinatory Declarative Programming

James Lipton
Visiting Ramón y Cajal Researcher
Technical University of Madrid
and
Dept. of Mathematics and Computer Science
Wesleyan University

Introduction

For over a decade several groups of researchers, working independently, have developed a variable−free, combinatory relational approach to declarative programming and applied it to different components of the programming cycle (development, refinement, transformation, static analysis, compilation) or used it as a declarative extension in its own right.

Each member group has had its own unique perspective on the use and development of these techniques, although there is at present a substantive core where all these approaches overlap, suggesting that a new area within declarative programming has come into being in this work.

The techniques draw on ideas from an unusually wide range of areas: relation calculus, database query evaluation, rewriting, combinatory logic, program synthesis and transformation, denotational semantics, program algebra, computational set theory, set constraints, and graph and category theory.

We briefly summarize some of the components of the work below, and the contributions of the research groups.

The Relation Calculus

In the 1940's Tarski introduced a first−order axiomatization of calculus of binary relations with operations $\cup, \cap, \circ, \smile, ^-$ (union, intersection, composition, converse, complementation) and constants $id, di, \mathbf{0}, \mathbf{1}$

(identity, diversity−its complement, and the empty and universal relation, consisting of all pairs) leading to the subsequent definition of relation algebras. Tarski and his collaborators found that the relation algebra equipped with the relational counterpart of constructors and destructors of pairs, and certain variants could provide a variable−free foundation for most of mathematics. Much of this work is set forth in [60], where the authors speculated that some of their results might be useful in computer science. It has certainly proved to be the case.

Variable−free approaches (of an entirely different character) to logic and computation have a long history: they were first discovered by Shönfinkel in the 1920's, and developed extensively by Curry and Feys in the 1940's and 50's, among others. Their work was later exploited by researchers in semantics and in the compilation of functional programming [61]. A combinatory approach to program development and analysis was developed by Backus [27] in his point−free Algebra of Programs. A combinatory−logic based approach to higher−order unification was described in [6].

Relations

The calculus of relations has been used extensively in computer science, of course, and logic programming itself can be described as a relations−based paradigm. But our focus here is on their combinatory development. Combinatory, or point−free relation calculi have been applied to the specification and synthesis

of functional programs, data types, and the derivation of general programming principles, by Roland Backhouse and his collaborators at Eindhoven[25,26], and further by Bird and De Moor [32], using Freyd's *allegories* [46], a categorical relational formalism.

Allegories have also been used in relational approaches to hardware design [33,48,37].

The database community has worked with query processing and optimization using relation algebra since Codd's pioneering work in the 1970s, probably the greatest single application of the formalism in computer science. Although this work is not directly related to the subject at hand, some of the researchers in combinatory logic programming came from the relational database community.

Other Variable−free Formalisms

No attempt is being made at an exhaustive description of the combinatory approach, elusive a concept as it is. Quine's predicate−functor logic, used by Hamfelt, Fischer Nilsson and others (see below) for *Combilog* is another example, as are the many codifications of logic in other theories, e.g. graphs, throughout the literature of mathematical logic.

Category Theory itself may be described, at least in part, as an effort to eliminate variables from mathematics. It has also been exploited to give a new foundation for logic programming by Martini, Corradini, Asperti, Montanari, Power, Kinoshita, Diaconescu, Finkelstein, Freyd, Amato, McGrail, Lipton and Pym, to name a few of the researchers in the field. This approach to logic programming is quite different in character, and deserves a treatment of its own. It will not be further discussed here. See e.g. [3,4,47,1,42,41,2].

Combinatory Logic Programming

It is, perhaps, surprising that these techniques were not applied (to this author's knowledge) until relatively recently (early 1990's) to logic programming. Since then however, three or four research groups have begun to systematically develop combinatory approaches to logic programming in a number of ways. Combinatory relation calculi similar to Tarski's calculus of relations and Quine's predicate functor logic are used as a variable−free target language by three groups of researchers, whose work is further described below:

- Formisano (Perugia), Omodeo (L'Aquila), Policriti (Udine) and Simeoni (Venice).
- Hamfelt (Uppsala), Fischer Nilsson (DTU−Denmark), Vitoria (Lisbon)
- Lipton (Wesleyan), Broome (Maryland), Chapman (New York)

The idea of taking a combinatory approach to logic programming originated (to the author's knowledge) with Jørgen Fischer Nilsson's 1990 paper [8] where the term ``Combinatory Logic Programming'' first appeared in print. A categorical variable−free formulation of a very different nature, due to Martini and Asperti, appeared the year before [23], but it is not a combinatory approach (see comments on categorical formulations, above). Broome and Lipton [35,43,44] began working in the area in the early 1990s.

An example

It may be most instructive to see a small example of a combinatory translation of a logic program. This example is deceptively simple, but gives some idea of the fundamental notions involved.

Consider the following Horn Clause program defining the transitive closure of a graph

```
conn(X,X).                          edge(a,b).
conn(X,Y):-edge(X,Z),conn(Z,Y).     edge(b,c).
                                    edge(a,l).
                                    edge(l,c).
```

and the queries

```
| ?- conn(a,c).
| ?- conn(X,c).
```

This rather carefully chosen example can be easily reformulated in first−order−variable−free relational terms denoting binary relations on the Herbrand universe $\mathcal{H}$, as follows. We introduce the binary relation variables $conn$ and $edge$ and translate the program into a pair of relation equations. We denote composition of relations by `` ; ''.

$$\begin{aligned} edge &= \{(a,b),(b,c),(a,l),(l,c)\} \\ conn &= id \cup edge; conn \end{aligned}$$

where id denotes the identity relation. Thus the second line states, in a point−free way, that

$$(x,y) \in conn \equiv y = x \vee \exists z\,(x,z) \in edge \wedge (z,x) \in conn.$$

The queries are then represented by the relation expressions:

$$\{(a,c)\} \cap conn \quad \text{and} \quad (\mathbf{1};\{(c,c)\}) \cap conn$$

where $\mathbf{1}$ is the universal relation $\mathcal{H} \times \mathcal{H}$, (hence $(\mathbf{1};\{(c,c)\})$ represents the set of all pairs whose second component is c).

Some questions that arise naturally at this point are: can such a translation be defined for an arbitrary Horn Clause program, or even for extensions of conventional programs, what kind of relation calculus is suitable for it, and can the resulting relational expressions be easily executed, controlled and optimized?

The research groups whose work is discussed below apply different techniques to obtain point−free versions of arbitrary logic programs and provide different notions of target language execution, sometimes with different aims. Fischer Nilsson and Hamfelt define a programming language, *Combilog*, based on a small set of predicate−forming operators and a set of inference rules, and a meta−interpreter to execute expressions in the language. They are able to achieve very compact representations of programs by means of recursion operators such as *foldr*. As an example, the program for *append*

$$\begin{aligned} &append([],X,X) \\ &append([X|T],Y,Z) \quad \leftarrow \quad append(T,Y,U) \wedge cons(X,U,Z) \end{aligned}$$

in the *fold*−extended COMBILOG becomes $append \leftarrow make[2,1,3](foldr(cons, id))$

where $make[2,1,3]$ swaps the two first arguments (since *foldr* differs from *append* in that the second argument, not the first one, is the recursion parameter).

Broome, Chapman and Lipton define a (binary) relational programming language, *Ralog*, a relational abstract machine based on rewriting, and a relational constraint processing engine based on constructive negation. Formisano, Omodeo, Policriti and Simeoni also use the Tarski−Givant formalism as a basis for their work, but exploit it in different ways: for automated deduction with different graph and set theories, and for formal comparison and analysis of these ontologies. All three groups make use of a variable−elimination algorithm to translate predicate logic into a point−free combinatory formalism.

Our summary here is not exhaustive, there have been other groups involved in similar projects: Bellia and Occhiuto develop a declarative algebra of programs (C−expressions) that captures unification, rewriting and narrowing in [29]. A compositional relational approach to logic programming is also described in [52]

The descriptions below have been contributed by members of the research groups involved, and edited for this report by the author.

Hamfelt and Fischer Nilsson: Compositional Predicate Operators

It is well−known that functional programs can be coined into variable free program expressions by appealing to a system of higher−order functions taking ordinary program functions as constants. Our concern is relational programming using logical clauses rather than functional programming.

We introduce a variable−free compositional form of definite clausal logic programs based on a system of predicate−forming operators proposed by Quine. The semantics is given by a simple fixpoint construction accomplished solely by iterated union, intersection, a generalised projection, and list construction and deconstruction. In addition a system of proof rules devises a proof theoretical semantics which is proved equivalent to the suggested fixpoint semantics. Moreover the predicate operators are proved sufficient with respect to expressivity for definite clauses over list terms and a logic programming metainterpreter is set up. With recursion operators being added we have results on applications to compositional programming and inductive logic programming reported in [12].

We define a novel view of logic programming in which the resolution mechanism for definite clauses is supplemented with goal and problem reduction oriented proof structure reflecting the high level structure of a program shaped by certain operators. The operators used for composing programs come with perspicuous inference rules which lay open the goal−directed computation behavior. The program operators constitute a tool for composing programs at a structural level above clauses.

The dispensing with variables facilitates manipulation of programs and transformations by means of algebraic rewriting rules, since the only remaining variables are variables ranging over subprograms, that is predicate terms. The compositionality of the predicate term semantics ensure that programs represented by predicate subterms function fully independently of the embracing context, further easing use of algebraic rewriting. This is in contrast to usual clausal formulations, where variables in terms links to the embracing constructions. This is analogous to the the λ−calculus *vis à vis* combinatory logic.

In particular, the variable−free form of operator programs facilitates manipulation of programs in a metalogic programming setting. This is exploited in an innovative method for induction of logic programs in [12].

Formisano, Omodeo, Policriti & Simeoni: A description of our research activity on the calculus of relations

The Tarski−Chin−Givant formalism $\mathcal{L}^{\times}$ is a modernized version of the Pierce−Schröder formalism of dyadic relations. It offers an assembly language for equational logic of a lower level than first−order logic. Its constructs operate on relations, very much like the constructs of Codd's relational algebra operate on relations of any degree; in $\mathcal{L}^{\times}$, however, relations can have an infinite cardinality, and complementation w.r.t. the full domain of discourse is available as an operation.

The possibilities of use of the calculus of relations in automated deduction are described in [20] and in [21], where the authors outline different research themes and pave the way for the subsequent work. In particular, in [21] we have abstractly specified important data types such as nested lists and sets ultimately based on individuals, on top of which one can easily define further types (e.g., a line editor).

In [16] we touch different themes related to relational reasoning, among others:

- The emulation of $\mathcal{L}^{\times}$ through first−order logic (e.g., within Otter, or within a more markedly equational theorem prover). Far from being a vicious circle, this has led to a finite axiomatization of ZF set theory.
- Initial steps in the development of a platform specifically oriented to relational reasoning.

Automated techniques to translate first−order formulas into the relational formalism are dealt with in [17]. Algorithmic techniques are presented (by means of several examples taken from different application scenarios) together with an analysis of their computational complexity. The exposition is made clear by exploiting a graphical representation for expression over relations.

The possibility of exploiting a general−purpose theorem prover in order to mechanize relational reasoning has been investigated in [18,19,22]. In these works we design and develop a *structured* proof−methodology aimed at using a first−order theorem prover (in the case, Otter from the Argonne National Lab.) as a *kernel engine* for reasoning with relations. In particular, in [18,19], we show how the adoption of a rigorous and structured approach in developing the experiments immediately allows one to build a hierarchy of axiomatizations. At each layer of this hierarchy one introduces new relation constructs and proves their properties. This approach permits a better guidance of the behavior of the underlying theorem−prover, since during the proof−search, the *focus* tends to be posed on the laws/constructs strictly related to the theorem being proved. Higher is the layer in the hierarchy of axiomatizations, higher is the semantical complexity of the laws to be proved and of the involved constructs.

As an interesting side−effect, this approach guarantees great independence from the specific theorem−prover used as kernel inference−engine.

Broome, Lipton & Chapman: Declarative Relational Programming

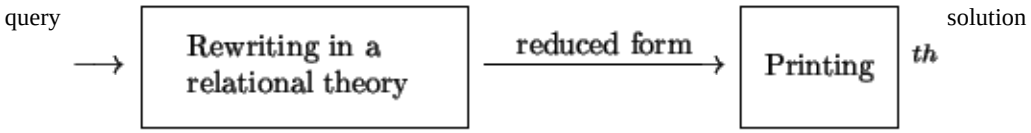
Our research explores how Logic Programming itself may be profitably understood, extended and compiled in terms of an underlying equational relational calculus, in which relation variables play a fundamental role, similar in some regards to the role of function variables in the lambda−calculus. The variables in the resulting terms correspond to predicate (second order) variables in the original program, whereas all first order variables are eliminated.

In this work we use an equational extension of the relation algebra formalism (with a fixed−point operator and relations corresponding to permutations, projections and constructors of terms) as an *executable* algebra of logic programs. We translate logic programs into combinatory relation expressions, which are then executed via rewriting and output−formatting algorithms.

Since at any moment during execution, the partially−evaluated program is a bona−fide ``declarative" mathematical object (a relation expression), the language makes available to the programmer direct manipulation of the fringe of the SLD tree. This provides a new approach to control, optimization, and manipulation of state. The relation calculus admits a more general notion of query and output (compound relation and set−expressions), as well as propagation of relational constraints during execution and extensions to higher−order logic.

Our Research in this area makes use of other relational formalisms (Freyd's allegories) which incorporate type and constraint information into the compilation language, and a more general notion of program, queries and outputs (first−order queries, extensions to equational logic and constructive negation as well as higher order logic).

Computation in this formulation means *relation calculus rewriting* of queries (relation expressions) to members of some distinguished class of relation expressions (usually recursion−free terms), followed by *representation* of such expressions in a human readable form in a way that depends on the data domain, a process here called *printing*.



In the *Ralog* programming language, data means terms in a Herbrand Universe, and printing is application of the quantifier elimination algorithm in constructive negation.

Future Directions

The research in Combinatory Declarative Programming raises many interesting questions. What special techniques will be needed for efficient compilation of these new combinatory expressions? What point−free calculi are most suitable for different declarative extensions? There is a great deal of formal similarity between the techniques outlined here and calculi used for program synthesis. It would seem that this opens up new possibilities for transfer of techniques and semantics between the two areas.

Fourth Panhellenic Logic Symposium

Announcement and call for papers
July 7−10, 2003, Thessaloniki, Greece

Lefteris Kirousis Department of Computer Engineering and Informatics
University of Patras
Email: kirousis@ceid.upatras.gr

The Panhellenic Logic Symposium is a biannual scientific event established in 1997. It is open to researchers from all countries who work on Logic broadly conceived. The language of the Symposium is English. The Fourth Panhellenic Logic Symposium will be hosted at the Conference Center of the International Fair of Thessaloniki, located downtown in close proximity to the University campus. The scientific program of the symposium will consist of hour−long invited talks, tutorials and presentations of accepted papers.

Original papers on all aspects of Logic are solicited. Authors are invited to submit an extended abstract of at most five pages. In addition, the contact author of each paper should send a cover page with his/her addresses (postal and email) and a statement classifying the paper in one of the following areas: Mathematical Logic, Set Theory, Logic in Computer Science, History of Logic, Philosophy of Logic, other Logic related area (specified).

All submitted papers will be reviewed by the scientific committee of the symposium, who will make final decisions on acceptance or rejection. Accepted papers should be presented at the symposium by one of their authors. Each accepted paper will be allocated a thirty−minute period for presentation and questions. Camera−ready extended abstracts not exceeding five pages will be included in a proceedings volume that will be distributed to all participants.

SCIENTIFIC COMMITTEE

- S. Cosmadakis (CTI and U. of Patras)
- C. Dimitracopoulos (U. of Athens)
- A. Kakas (U. of Cyprus)
- A. Kechris (Caltech)
- L. Kirousis (U. of Patras) Chair
- Ph. Kolaitis (UC Santa Cruz)
- G. Koletsos (N.T.U. of Athens)
- E. Kranakis (Carleton U.)
- M. Mytilinaios (Athens U. of Economics)
- Th. Pheidas (U. of Crete)
- A. Sinachopoulos (Archimedia S.A., Athens)
- P. Spirakis (CTI and U. of Patras)
- Th. Tzouvaras (U. of Thessaloniki)
- S. Zachos (N.T.U. of Athens)

ORGANIZING COMMITTEE

- C. Dimitracopoulos (U. of Athens)
- C. Hatzikyriakou (U. of Thessaly)
- A. Kakas (U. of Cyprus)
- C. Kalfa (U. of Thessaloniki)
- A. Sinachopoulos (Archimedia S.A. Athens)
- Th. Tzouvaras (U. of Thessaloniki) Chair

INVITED TALKS

- D. Bjorner (Technical U. of Denmark)
- P. Peppas (U. of Patras)
- K. Sagonas (Uppsala U.)
- I. Soskov (Sofia U.)

IMPORTANT DATES

- Submission: March 28, 2003
- Notification: May 9, 2003
- Camera−ready abstracts: May 30, 2003

TUTORIALS

- Philosophy of Logic: W. Demopoulos (U. of Western Ontario)
- Complexity of Logic−Related Problems: E. Koutsoupias (U. of Athens and UCLA)
- Logic−Based Information Integration: M. Lenzerini (U. di Roma "La Sapienza")
- Set Theory: A. Louveau (U. Paris 6)

WEB ADDRESS:
http://www2.cs.ucy.ac.cy/pls4

ADDRESS FOR SUBMISSION
Electronic submissions in PostScript are strongly encouraged
Lefteris Kirousis Department of Computer Engineering and Informatics
University of Patras, University Campus, GR−265 04 Patras, GREECE
Phone: +30 2610−99 7702
Fax: +30 2610−99 1909
Email: kirousis@ceid.upatras.gr

Cooperation of Universidade Nova de Lisboa and Technische Universit. at Dresden in the field of Computational Logic

Luís Moniz Pereira and Reinhard Kahle
Centro de Inteligência Artificial (CENTRIA), Universiddade Nova de Lisboa

The Departamento de Informatica of the Universidade Nova de Lisboa (UNL) and the Fakult. at f. ur Informatik of the Technische Universtit. at Dresden (TUD) intend to develop a new integrated binational study programme in the eld of Computational Logic. Currently, both universities are coordinating the work−package Education and Training within the European Network of Excellence in Computational Logic, CologNET and members in the International Network of Excellence (IQN) on Rational Mobil Agents funded by the German Academic Exchange Service (DAAD).

It is planned to set up a joint master programme based on the existing Mestrado em Inteligencia Artificial Aplicada at UNL and the Master in Computational Logic at TUD. With respect to plans for the new Mestrado em Engenharia Informatica at UNL it can be integrated as a profile in Computational Logic.

For this purpose the German Academic Exchange Service (DAAD) provides special funds for the first four years plus an additional year at the beginning for preparation. Recently, this rst year funding was awarded by the DAAD for the preparation of a joint master programme at UNL and TUD during the academic year 2003/2004. If the study programme starts in 2004 the funds cover (mutual) exchanges of lectures between Lisbon and Dresden and additional costs. Student grants can be awarded within the Sokrates/Erasmus programme, in which UNL and TUD already take part. In general, the programme Al²an provides special funds for students from Latin America to study in the European Union. There are various programmes funded by the DAAD to attract international students to study in Germany.

To establish a joint master programme, UNL and TUD have to set up study regulations addressing the following points: mutual recognition of courses taken by master students at the partner university; regulation about obligatory stays abroad; waving of tuition fees; double degrees (simultaneously given by both, UNL and TUD).

The language of courses offered to students of both universities should be English. There exists already preparatory work to install a teleteaching system between Lisbon and Dresden, founded by CologNet and IQN. It allows for transmitting lectures from one university to the other, and holding joint seminars.In March 2003 there will be a test of the hard− and software in Lisbon. It is planned to have the first tranmissions between Dresden and Lisbon.

The planned joint master can serve as a basis for an European Master programme within the forthcoming programme Erasmus World installed by the European Union. In this programme, high quality joint master programmes of at least three European universities can be awarded with the quality label EU master programme. It is intended to attract students from third countries to study in the European Union. For this end, special grants for foreign students are available. UNL and TUD intent to apply for such a European masters programme with partner universities from the CologNet network.

In addition to these support opportunities for a master programme, the German Science Foundation (DFG) is supporting PhD programmes within International Research Training Groups (Graduiertenkollegs). It is also planned to establish such a PhD programme which allows for additional funding of PhD students. The programmes runs for up to nine years and comprises also money for exchanging researchers and lectures, for organizing workshops and for coordination costs.

The Graduiertenkolleg has to be organized by at least five professors of each university. These professors should have a strong publication record and experience in PhD supervision. From the Dresden side, presumably, Franz Baader, Bernhard Ganter, Steffen Holldobler, Michael Thielscher and a new professor for Algebraic and Logic Foundations of Computer Science will take part. On the Lisbon side, Jose Alferes, Pedro Barahona, Carlos Dam asio, and Luis Moniz Pereira are already involved in the cooperation, and other professors which comply with the requirements are welcome to consider participating.

The structure of the study programme

The courses of the planned profile in Computational Logic will be split into two groups: mandatory and optional ones. It follows in principle the structure of the existing international master programme in Dresden. The mandatory courses are (cf. appendix for Dresden course description and MIAA page for Lisbon course descriptions):

- Constraint programming
- Declarative programming
- Introduction to computational logic
- Logic programming implementation 1
- Logic programming and artificial intelligence

Except for the course Introduction to Computational Logic all these courses are already established in Lisbon, although the length and the content of some of the courses may vary. The list of optional courses consists of already established courses offered in Lisbon (+) and in Dresden (−)

- Knowledge representation (+)
- Computational reasoning (+)
- Agents (+)
- Computational cognitive science (+)
- Natural language processing (+)
- Theorem proving 1 (−)
- Advanced logics (−)
- Specification and verification (−)
- Theoretical computer science and logic (−)
- Syntax−directed semantics (−)
- Inference techniques (−)

This list is open to extensions. In particular, it is desirable that mandatory courses similar to the Dresden ones are offered in Lisbon, too, and vice−versa. The study regulation will require that a student has to spend at least one or two semester(s) at the partner university.

Dresden is awarding credit points according to the ETCS system (roughly, 1 Lisbon credit is 1.5 ECTS). The existing CL master requires 120 points, 90 for courses, 30 for the thesis.

Committees

The formal university regulations in Germany require that Dresden has three different committees for a study programme:

1. Evaluation committee, which selects the students;
2. Study committee, which sets the curriculum;
3. Examination committee, which oversees.

It is possible and often the case that the members of these three committees are the same. Therefore, the Lisbon side can set up just one committee for all three purposes.

Schedule

- Sept 03 − Aug 04: Preparation of the joint master (with funding of the DAAD)
- Feb 04: First draft for the International Graduiertenkolleg (PhD programme)
- Apr 04: Submission of the application for the International Graduiertenkolleg (PhD programme)
- Sept 04: Start of the joint master programme

Upcoming Events

Start date	End date	Category	Subject	Place
08 Sep 2003	**14 Sep 2003**	Sym	**FME 2003: The 12th International FME Symposium**	Pisa, Italy
09 Aug 2003	**15 Aug 2003**	Conf	**Eighteenth International Joint Conference on Artificial Intelligence (IJCAI 2003)**	Acapulco, Mexico
28 July 2003	**29 July 2003**	WS	**CADE−19** Grand Challenges for Automated Reasoning	Miami, USA
07 July 2003	**10 July 2003**	Sym	4th Panhellenic Logic Symposium, Thessaloniki, Greece,	Thessaloniki, Greece
08 Jun 2003	**14 Jun 2003**	Conf	**RDP 2003 − Federated Conference on Rewriting, Deduction and Programming** **The following conferences, workshops and scientific meetings participate in RDP 2003:**	Valencia, Spain
13 Jun 2003	**14 Jun 2003**	WS	6th International Workshop on Termination (WST'03)	Valencia, Spain
12 Jun 2003	**14 Jun 2003**	WS	4th International Workshop on First order Theorem Proving (FTP'03)	Valencia, Spain
12 Jun 2003	**13 Jun 2003**	WS	12th International Workshop on Functional and (Constraint) Logic Programming (WFLP'03)	Valencia, Spain
12 Jun 2003	**12 Jun 2003**	WS	IFIP Working Group 1.6 on Term Rewriting (WG 1.6)	Valencia, Spain
10 June 2003	**12 Jun 2003**	Conf	6th International Conference on Typed Lambda Calculi and Applications (TLCA'03)	Valencia, Spain
09 Jun 2003	**11 Jun 2003**	Conf	14th International Conference on Rewriting Techniques and Applications (RTA'03)	Valencia, Spain
09 Jun 2003	**09 Jun 2003**	WS	4th International Workshop on Rule−Based Programming (RULE'03)	Valencia, Spain
08 Jun 2003	**09 Jun 2003**	WS	17th International Workshop on Unification (UNIF'03)	Valencia, Spain

08 Jun 2003	**08 Jun 2003**	WS	3rd International Workshop on Reduction Strategies in Rewriting and Programming (WRS'03)	Valencia, Spain
05 May 2003	**08 May 2003**	Conf	Sixth International Conference on Theory and Applications of Satisfiability Testing	S. Margherita Ligure − Portofino (Italy
15 April 2003	**16 April 2003**	WS	10th Workshop on Automated Reasoning	Liverpool, UK

Event types

Sym	Symposium
Conf	Conference
WS	Workshop
SSchool	Summer School