Collaboration between Human and Artificial Societies

Julian Padget Carles Sierra
Ulises Cortes Miquel Sànchez
Javier Bejar

Report LSI-97-36-R
Collaboration Between Human and Artificial Societies

The VIM project
Spring Workshop 1997

May 1-3, 1997
Lanjarón (Granada)
Spain
Organizing Committee

Julian Padget, University of Bath
Carles Sierra, Institut d'Investigació en Intel·ligència Artificial
Ulises Cortés, Universitat Politècnica de Catalunya
Miquel Sànchez i Marrè, Universitat Politècnica de Catalunya
Javier Béjar, Universitat Politècnica de Catalunya
The VIM Project
Spring workshop 1997
on
COLLABORATION BETWEEN HUMAN AND ARTIFICIAL SOCIETIES

Programme

May 1: Thursday
- 09:30-10:30 invited talk: Nick Jennings,
  Negotiation and Argumentation in Practical Multi-Agent Systems
- 10:30-11:00 coffee break
- 11:00-13:15 partner presentations
  - Oliver Maerker and Hans Voss (GMD Bonn)
    A computer supported argumentation framework - concepts and practice
  - Ramon Sangüesa (UPC)
    Learning causal networks from data in multi-agent systems
  - Viçenc Torra (URV)
    On integration of information from multiple sources
- 13:30-15:00 lunch
- 15:00-16:30 partner presentations
  - Luc Steels (VUB)
    On the evolution of language (title to be confirmed)
  - Miquel Sanchez, Javier Béjar, Ulises Cortés (UPC)
    Reflective reasoning in a CBR agent
- 16:30-17:00 coffee break
- 17:00-18:30 partner presentations
  - Sabine Geldof (VUB)
    Context Sensitive Hypertext Generation
  - Vincenzo Loia (Salerno)
    Integrating Multiple Knowledge Sources in a Distributed Environment: Experiences in Diagnostic Reasoning
- 21:00 dinner

May 2: Friday
- 09:30-10:30 invited talk: Jean-Pierre Briot
  Issues in the Integration of Object-Oriented Technology with Parallel and Distributed Computing Requirements
- 10:30-11:00 coffee break
- 11:00-13:15 partner presentations
  - Pablo Noriega and Carles Sierra (IIIA)
    Dialogical Frameworks
- Iain Craig (Warwick)
  *From Blackboards to Agents* (full paper in Postscript)
- David DeRoure and Johnathan Dale
  *DIM agents and their frameworks*

- **13:30-15:00 lunch**
- **15:00-16:30** partner presentations
  - Andreas Kind (Bath)
    *Towards Meta-Agent Protocols*
  - Luc Moreau (Southampton) and Christian Queinnec (INRIA/Paris 6)
    *On the Finiteness of Resources in Distributed Computing*

- **16:30-17:00 coffee break**
- **17:00-18:30** partner presentations
  - Wolfgang Goerigk, Friedemann Simon (Kiel)
    *Towards Rigorous Compiler Implementation Verification*
  - Angela Sordan (GMD Berlin)
    *Fuzziness in Compilers and Run-time Systems*

- **21:00 dinner**

**May 3: Saturday**

- **09:30-10:30** invited talk: Henry Prakken
  *Connecting systems for defeasible argumentation with negotiation protocols*

- **10:30-11:00 coffee break**

- **11:00-13:15** partner presentations
  - Iain Craig (Warwick)
    *Converting Declarative into Procedural (And Vice Versa)* (full paper in Postscript)
  - Francisco J. Martin and Juan A. Rodriguez (IIIA)
    *Trading Scenarios for heterogenous agents*
  - Maurizio Giordano (CNR Naples)
    *Experiences with PVM: Communication Issues in Distributed and Cooperative Applications*

- **14:30 lunch** in Granada at La Morayma, afternoon free to enjoy the fiesta de Las Cruces, a tour of the Alhambra is organized for the evening (20:30), followed by a late dinner back at the hotel in Lanjarón
CONTENTS

Negotiation and argumentation in practical multi-agent systems
N. Jennings 1

A computer supported argumentation framework - concepts and practice
O. Maerker & H. Voss 2

POSSCAUSE: Causal learning in distributed environments
R. Sangüesa 4

On integration of information from multiple sources
V. Torra 6

On the evolution of language
L. Steels 7

Reflective reasoning in a CBR agent
M. Sánchez & J. Béjar & U. Cortés 8

Context-sensitive hypertext generation
S. Geldof & W. Van de Velde 10

Integrating multiple knowledge sources in a distributed environment: experiences in diagnostic reasoning
S. A. Cerri, A. Gisolfi & V. Loia 13

Issues in the integration of Object-Oriented technology with parallel and distributed computing requirements
J. P. Briot 27

Dialogical Frameworks
C. Sierra & P. Noriega 28

From blackboards to agents
I. D. Craig 29
Fuzziness in compilers and runtime systems
A. C. Sodan 30

Towards meta-agent protocols
A. Kind 31

On the finiteness of resources in distributed computing
L. Moreau & C. Queinnec 32

Towards rigorous compiler implementation verification
W. Goerigk & F. Simon 33

Experiences with PVM: Communication issues in distributed and cooperative applications
M. Giordano 34

Connecting systems for defeasible argumentation with negotiation protocols
H. Prakken 35

DIM agents and frameworks
D. DeRoure & J. Dale 36

Converting declarative into procedural (and vice versa)
I. D. Craig 39

Trading scenarios for heterogeneous agents
F. Martín & J. A. Rodríguez 40
Negotiation and Argumentation in Practical Multi-Agent Systems

Nick Jennings
A computer supported argumentation framework - concepts and practice

Oliver Maerker and Hans Voss (GMD Bonn)

Abstract: ZENO is a mediating system based on the World-Wide Web. Building on decision theory and a formal model of argumentation, ZENO offers intelligent assistance to human mediators, facilitators, arbitrators and other "trusted third parties" by providing an issue-based discussion forum or conferencing system.

It allows an arbitrary large number of interested parties in various locations to take part in a decision making process extending over a long period of time. ZENO places issues, positions and arguments into a 'picture' which is richer, more precise and focused than the "thread" mechanism typical of news groups. This structure provides a view on the messages which substantially facilitates the browsing and retrieval of relevant past contributions to the discussion. Contributions may include additional HTML hypermedia elements, such as links to other resources on the World-Wide Web, such as links to other resources on the World-Wide Web, images, movies and sound files.

Alternative solutions (i.e. positions) proposed for issues can be prioritized. Using reason maintenance and constraint satisfaction procedures, the preferences expressed in the arguments brought forward by the participants are summarized.

ZENO is being further developed as part of GeoMed (Geographical Mediation), which is a project financed by the European Commission within the Telematics Applications Programme. ZENO provides the basis for the mediation and documentation services of GeoMed.

Pilot applications in real life urban and regional planning processes guide the further evolution of ZENO's functionality. In the City of Bonn, the benefits and potential of ZENO is currently investigated in the context of three real urban and regional planning applications.

This paper briefly describes the argumentation framework of ZENO and its background in general argumentation theory. We proceed by illustrating the application scenarios in the City of Bonn. The goal of the paper is to give a preliminary assessment of the usability or suitability of ZENO with regard to these scenarios. As one basis for such an assessment we can refer to two structured validations of ZENO with students and experts in the field of urban planning.

Addresses of the authors:
GMD & endash; German National Research Center for Information Technology
FIT.KI: Research Division Artificial Intelligence
Codes: Cooperative Design Group
D-53 754 Sankt Augustin, Germany,
Contact: phone: ++49 (2241) 14-2532, fax: - 2072 email: hans.voss@gmd.de

-2
The GEOMED consortium consists of Intecs Sistemi (IT) [Coordinator], GMD (D) [Technical Director], Intrasoft (GR), Vrije Universiteit Brussel (B), TNO-FEL (NL), TNO-Bouw (NL), City of Bonn (D), City of Tilburg (NL), Tuscany Region (IT) and the Technical Chamber of Greece (GR).
POSSCAUSE: Causal Learning in distributed environments

Ramon Sangüesa i Solé
Technical University of Catalonia

April 28, 1997

Abstract

The construction of causal networks from data, the creation of new methods for measuring possibilistic conditional dependence, its estimation from possibilisity distributions derived from data and the application of these concepts to the development of causal network algorithms are the main goals of this work. Possibilistic networks are an extension of the Bayesian network construct. These kind of networks have been used to represent causal associations by conditional independence criteria measuring the dependence among variables and sets of variables. The possibilistic representation of uncertainty allows for working with imprecise data, a problem that is not so easily coped with probabilistic formalisms. One of the main problems of the recovery of possibilistic networks is the need for a test on conditional independence. After some preliminary sections on different methods for graphically representing causation and methods for recovering several types of causal network, the possibilistic directed acyclic graph representation is introduced and its inference procedures discussed. A new measure for detecting possibilistic conditional dependence among variables is defined. Their properties in respect to conditional independence models axiomatization put forth by Pearl and Verma are studied. The measure is graded, so it allows comparison in the degree of association between variables and is well adapted to the comparison of distributions estimated from data. Moreover, the measure exhibits information characteristics that are important for the selection of alternative representations of conditional associations coming from data. This property allows to select from a set of variables those that are more mutually constrained with the confidence that the result network will be more informative.

The definition of causal subgraphs allows for the construction of directed acyclic graphs by stepwise fusion of partial directed acyclic graphs that correspond to causally related clusters of variables. The fact that clusters of causally dependent variables can be detected, up to a certain limit, without dealing with the whole variable domain allows for an efficiency increase respect to other established methods. It also eases the posterior parallelization of the recovery
method. Two new algorithms have been developed, HCS and POSSCAUSE. The first one is an algorithm where dependency information dominates the search procedure. The second one is a parallel extension of this method. Their characteristics in terms of structural quality have been compared with their probabilistic counterparts. Several artificial databases have been used. A discussion on the practical application of these algorithms to the construction of knowledge bases for some domains, specially for monitoring and control is included.
On integration of information from multiple sources

Vicenç Torra i Reventós
Departament d'Enginyeria Informàtica (ETSE), Universitat Rovira i Virgili
Carretera de Salou s/n, E-43006 Tarragona (Spain)
e-mail: vtorra@etse.urv.es

Abstract

Knowledge integration is an essential process in any intelligent system. The development of a system has to take into account that information sources (e.g., sensors or experts) supply partial and erroneous information. Due to this fact, systems are forced to consider (integrate) data provided by several sources.

Consensus theory (data fusion) is the field that studies the aspects related to the combination of the information. It studies the combination functions, the properties that satisfy these functions and how to build a function from a set of imperative properties. It could be said that the goals of this field are to systematize (and formalize) the synthesis process and to characterize the existing methods. Both aspects are needed if it is intended to apply combination functions to new problems.

Data fusion techniques are not only used within artificial intelligence, but there are applications in other fields like mathematics, economy, biology. Within artificial intelligence, they are applied in several frameworks as Knowledge Based Systems (knowledge acquisition, validation, ...), fuzzy control, perception (vision, ...), ... A classical example is to consider a robot provided of sensors to perform some tasks on object recognition, material handling and assembly.

When we restrict to artificial intelligence, information is combined in two cases: (i) to make a decision or (ii) to have a good representation of the environment. In the first case, for instance, it is possible that instead of a single criterion for each alternative, the system has several ones. This case, that corresponds to a multicriteria decision making problem, is usually solved in a two phases process: (i) the aggregation of the degree of satisfaction for all criteria, per decision alternative; (ii) the ranking of the alternatives with respect to the global aggregated degree of satisfaction. On the other hand, to have a good representation of an environment, a system needs that the knowledge supplied by information sources (or the knowledge embedded in it) is reliable and that extends on the whole domain of action. However, the information supplied by a single information source (or by a single expert) is often not reliable enough and/or too narrow in relation to the working domain. In this case, the information provided from several sensors (or experts) can be combined to improve data reliability and accuracy and to include some features that are impossible to perceive with individual sensors.

Data fusion techniques have been applied to several classes of objects\(^1\). So, there have been defined functions on several formalisms, e.g., quantitative and qualitative values, probability and possibility distribution, fuzzy sets, quantitative and qualitative preference relations, classifications (in several formalisms: dendograms, partitions), ...

We will review our contributions to the study of the aggregation of numeric values and fuzzy sets.

In relation to the first case, we will review the definition of the WOWA operator. When the objects to synthesise are numeric values (e.g., numbers in the \([0,1]\) interval) two functions have been studied in the literature with a similar structure: the weighted mean and the OWA operator. Both operators combine the values using a set of weights. However, the meaning of these weights is different in both functions. We will show that in some applications (e.g., in sensor fusion in robotics or in evaluating solutions in Fuzzy CSP) both weights are needed and we will review the WOWA operator. A combination function that generalizes both the weighted mean and the OWA.

---

\(^1\)In fact, for each type of object there is at least a fusion method that can be applied: the plurality function that returns the most voted element.
On the Evolution of Language

Luc Steels
Reflective reasoning in a CBR agent

Miquel Sànchez-Marrè, Javier Béjar and Ulises Cortés
Secció d'Informàtica Artificial. Dept. de Llenguatges i Sistemes Informàtics.
Universitat Politècnica de Catalunya. Campus Nord-Edifici C5.
Carrer Jordi Girona 1-3. 08034 Barcelona, Catalonia.
{miquel|bejar|jia}@lsi.upc.es

Abstract. As a Case-Based Reasoning (CBR) agent evolves over time, and solves new problems based on previous experiences, there are some pitfalls that can appear in its problem solving task. When those troubles arise, is the time to start some reflective reasoning tasks to overcome that problems and improve the CBR performance. Our proposal is to extend the basic CBR reasoning and learning cycle with some new added reflective tasks such as forgetting cases, learning new cases, updating the case library organization, re-exploring the case library, setting meta-cases, etc.

The search time (efficiency in time) in the retrieval step could be increasing due to a bad case library organization. For example, if the case library is organized in a hierarchical way, such as a discrimination tree or network, there could appear some straight paths that end with a leaf node. In these paths, there are no branches at all. If the hierarchical structure can be arranged in such a way that the depth of the structure becomes smaller than the current one (compactness), then the average retrieval time of cases can be reduced.

The size of the case library (efficiency in space) could be enlarging as the CBR agent is learning new cases without an extensive improvement in the performance of the system. A natural human cognitive task appears as the solution to this problem: forgetting. Human beings forget the knowledge they do not use. And hopefully, what it is not used is what is not useful for their goals. Bearing in mind this analogy, we claim that there can be some useless cases stored in the case library that can be removed from it, with a significant increase in the performance. To this end, our proposal is to set a categorization of cases: useful/unuseful cases, normal/exceptional cases, relevant/redundant cases, etc., and an utility measure. Related to that ontology of cases, will appear a relevance measure. This measure will be the guide in the reflective task of learning a new case.
An unsuccessful search in the case library can start other reflective tasks to solve this problem, such as re-exploring the case library with other more flexible partial-matching criterion. This unsatisfying search can state some other reflections, such as the CBR agent reasoning can be improved by splitting the case library into several case libraries with different hierarchical structure. In the retrieval phase, first, the CBR will search within a previously established classification to identify which kind of case it is coping with. For each established class (meta-case) there will be a set of specific discriminating attributes and a different case library. Thus, there will be a partition of the cases by means of its similarity to previous established meta-cases.

The implementation of these techniques in the CBR agent will lead the case library towards an optimal configuration of cases, maximizing competence and minimizing size and response-time (performance) of the system.

**Key words:** Reflective reasoning, Case-based Reasoning.
Context-sensitive hypertext generation

Sabine Geldof

Walter Van de Velde

This paper claims that the role of Natural Language in a hyper-media information system is to provide, at each moment, a context-sensitive navigation point, i.e., a hypertext node in which the relevance of hyperlinks is justified with respect to the context of the interaction. It acts as the primary entry point for the user to the various pages that constitute an information service. We call context the collection of features that determine the desirable content and form of the information.

We describe an experiment based on an existing information server (for Ecran Total, Brussels’ summer movie festival) showing:

- how to capture contextual parameters and
- how to render them in a context-sensitive entry point to information.

The key to our approach is a model of competition for attention between software agents, the outcome of which is reflected in a weighted topic structure, annotated with text templates. The annotated topic structure is the basis for generating a context-sensitive navigation node by a process of template expansion and aggregation.

As a user interacts with the ecran information server, the weighted topic structure is built up according to her clicking behaviour. This application's topic structure contains 3 branches (reflecting 3 possible types of information one could be interested in):

- the programming details: when, where a particular movie will be on show in this festival, and other particularities of this showing (e.g. subtitles..)
- the production details: who is the director of a particular movie, who are the actors, where and when was it produced?
- details about the contents of the movie: the plot, to which category it belongs..

You can try this out by going to the full demonstrator or to a short demo (mock-up). If you want to know more about our work, you can read the paper.

Short demo

For the purpose of quick demonstration we can shortcut the process of capturing the user’s current interest profile which is normally acquired by observing her browsing behaviour for a while. We ask you to indicate yourself the relative importance of the branches in the topic structure, by allocating them a rating. Indicate an arbitrary rating by filling out a (different) number for each of the above mentioned topic branches. Then press the 'proceed' button, you will see a 3-pane window with in the upper right
pane a context sensitive hypertext reflecting these relative weights. The topic that received the highest score will be most elaborated in the context pane.

Live demo

Some hints before you proceed to the on-line demo:

- In order to be able to keep track of your browsing behaviour, you should be identified by the system as an individual user. This means to log on. You can first enter the catalog anonymously, a text in the top frame will prompt you to logon. Logon procedure only requires to specify your name, email adress and to choose a password.
- The metatext only applies to a particular movie description page, so when you are on a time table or an introductory page about a category of movies, no metatext will be shown.
- Your interest profile will be tracked on the basis of 2 types of clicking behaviour in the catalog:
  1. a very direct way to influence the scores of the individual topic branches is to click in the left hand pane, requesting directly for a particular subtopic of information about a movie. E.g., if you click a number of times on programmation related subtopics, your metatext will very quickly present the programmation information first.
  2. your browsing behaviour in the catalog pages (main pane) also determines the metatext, but indirectly, since most pages are weighted for a number of topics (e.g. if they present the director as well as the short plot, they will contribute both to the production and the contents topic), which result in a more indirect effect on the meta-text generation.
- The catalog is bilingual french/flemish (since it’s for Brussels’ public)
- If you have logged on you can also have a look at other (task) agents: one allows to keep track of a personal agenda for the movie festival and another one will advise you on which movie you might like, based on the comparison of your profile (in terms of scores that you would provide to a number of movies) with the closest other profile.

Now you can try it.

These pages are part of an experiment at the Artificial Intelligence laboratory of the VUB, in collaboration with Kris Verhoeye and Joris Van Looveren. This experiment builds further on the 1996 version of the Ecran Total Festival server, realized by Frank Kresin.
This work is supported by the MAGICA project, funded by the European Union program Telematics Engineering (IE2069). Additional funding is provided by the NMS project, funded by IWT, and the project SACEA funded by the Belgian Office of Scientific, Technical and Cultural Affairs. Walter Van de Velde is a senior researcher for the Belgian National Science Foundation.

contact: sabine@arti.vub.ac.be
Integrating Multiple Knowledge Sources in a Distributed Environment: Experiences in Diagnostic Reasoning

S. A. Cerri, A. Gisolfi, and V. Loia

Dipartimento di Scienze dell’Informazione
Università di Milano
via Comelico 39, 20135 Milano, Italy
cerri@dsi.urimi.it

Dipartimento di Informatica ed Applicazioni
Università di Salerno
84080 Baronissi (SA) Italy
[gisolfi,loia]@dia.unisa.it

Abstract

Research in multimodel-based diagnosis stresses the role of a centralized diagnostic agent in a computational framework made of different models. Each of these models provides knowledge according with a given representation of the system to diagnose: structural, behavioral, functional and teleological aspects are each embodied into a different model. According to this approach, various proposals have been formulated in terms of cooperation among different views without addressing in a complete way the issue of concurrency and collaboration, in particular the effect of asynchronous message passing in software design. We present here the use of a distributed and concurrent architecture in which different models collaborate in order to achieve a global diagnosis through a set of independent agents. In two difficult, apparently different application domains (cognitive diagnosis and troubleshooting of a physical device) the same architecture leads to similar components so that it is possible to identify an emerging, rather simple solution for distributed diagnostic reasoning that seems generic enough to be applicable for a wide class of diagnostic problems.

Key-Words: Actor-based Models, Diagnostic Reasoning, Cooperative Problem Solving

1. Introduction

Programming techniques have evolved from the machine-level style dominating in the 50ties up to the agent-level one in the 90ties. This progress has been characterized by important milestones: e.g. structured programming (1970) and object-oriented programming (1980). The current agent-based approach to software design and implementation derives from the development of methods oriented to distribute groups of objects each actively committed to perform independent computations and to exchange asynchronous messages by means of suitable communication protocols.

This evolution has lead to a radical change of the traditional approach of designers involved in software construction: from action-based, algorithmic systems to interaction-based systems. The rationale behind it is not just efficiency but instead the need to capitalize from the opportunities - throughout the entire life cycle of complex systems - offered by the alternative, inherently interactive, behavioural models of the systems, especially when the interactions occur in a distributed, concurrent environment.

Actor-based systems are computational environments in which asynchronous and concurrent events allow the interaction among active objects. An actor is defined in terms of an address, a mail-box containing the accepted requests in their arrival order, and a behaviour that can, at
each computational step, read a message from its mail-box and perform the actions associated. This execution may activate other actions by suspending previous executions or performing them concurrently. The terms new (creation of an actor), send (envoy of a message) and become (definition of behaviour) are known as the fundamental features of the actor model.

Artificial Intelligence research has strongly relied on non algorithmic approaches to the design and implementation of intelligent artifacts. More recently, part of the research interests in AI are focused on alternative approaches to centralized knowledge-based systems, that is new architectures where different knowledge-sources are physically and logically distributed so that separate problem-solving activities cooperate in order to accomplish a common goal. The entities that individually contribute to the overall process are generally named agents. Our research focuses in the adoption of the actor model (with some extensions) as a computational metaphor of distributed problem solving (Gisolfi, and Loia, 1994). In this work we present our research experiences reported in the integration of heterogeneous knowledge sources in distributed environments in order to support problem solving activities. We show how a concurrent architecture designed to model the student behaviour during a learning session (Cerri, and Loia, 1996), has evolved towards a general architecture suitable to perform diagnostic reasoning about physical systems (Loia and Gisolfi, 1996).

The paper is organized as follows. Section two underlines the role of DAI as an organizational paradigm of cooperative problem solving. Section three presents our concurrent model of diagnostic reasoning designed to diagnose mental states of an user attempting to solve a task. Section four describes the extension and specialization of the previous model towards multiple diagnosis of physical systems. Section five closes the paper with conclusive remarks and future research ideas.

2. DAI Approach to Problem Solving

Many efforts in Distributed Artificial Intelligence (DAI) emphasize the role of distributed search on separate knowledge sources (Cerri., Landini, and Leoncini, 1987) (Lesser, 1990) (Fabiano and Cerri, 1996). The state space search is organized into independent sub-domains, each of them explored by local sub-searches: possible collaborative strategies can be useful if there is the need to "share" intermediate results. In general it is possible to distinguish three kinds of distributed search:

Fine-grained decomposition
The partition of the state space involves large collections of small and independent sub-problems. This means that simple computations are enough to support control and data operations. Fine-grain processors realize optimally this situation: processors are numerous, simple and can support limited information and control. This approach is very close to connectionism and is particular useful to treat well-defined problems, in terms of reduced knowledge representation: the prerequisite is that subproblems are in a way independent, or, at least, if they are mutually dependent, a synchronization mechanism is sufficient for communicating data.

Large-grained decomposition
The state space is divided in a relatively small set of independent/dependent sub-states. This means that the role of problem-solving is more significant and thus the possible target architecture becomes more complex: the processors are expected to support robust computational entities (processors as evaluators, such as a LISP processor) and complex
collaboration strategies must be used to organize sophisticated interaction protocols. The more sophisticated is the collaboration among concurrent entities, the more knowledge is required for the protocols to be adequately managed. Actors are one step; probably other ones will be needed. Cooperative Distributed Problem Solving (CDPS) is a particular model of DAI based on a large-grain search space. In this paper we will focus on the link between CDPS and diagnostic reasoning.

In general, the DAI approach to diagnosis may involve hybrid situations, in which the degree of granularity differs depending on the nature of (multiple) models used.

Diagnosis is a difficult task (Davis, 1993). Among the reasons of this difficulty:

- simultaneous presence of multiple disorders;
- the search for a structure able to represent ill-defined problems;
- the difficulty to interconnect badly related concepts;
- the combinatorial explosion of the search space;
- the necessity to achieve practical information.

Model-based reasoning has been one of the most active sub-area within diagnostic reasoning: in particular, multiple-modeling diagnosis is a recent effort devoted to integrate several models of the same system in order to improve the effectiveness and efficiency of the reasoning process. This research, strongly motivated by cognitive arguments, requires to define how multiple models cooperate. In a centralized, synchronous (or blackboard-based) architecture the design and the management of this cooperation does not seem easy. By moving to an appropriate distributed, concurrent and asynchronous architecture we have been able to design and manage the cooperation for diagnosis among multiple knowledge sources in a simple and elegant way. We have operated in two apparently quite different scenarios of diagnostic reasoning, one related to cognitive diagnosis and the other to troubleshooting of a physical device.

User/student reasoning

Student modeling aims at diagnosing what is wrong with a student's method of performing a task and at transferring this information to other modules which compose the tutoring system (Anderson, Boyle, Corbett, and Lewis, 1990). In this way the student can receive an "intelligent" guide since the tutorial module may adapt the initiative of the system with a new balance of didactic actions on the basis of the student's necessity (Self, 1993).

Physical troubleshooting reasoning

In this well studied scenario, the main task is to localize the parts of the technical systems responsible for the abnormal behaviour. In order to optimize the diagnostic search, a multimodeling approach (Chittaro, Guida, Tasso, and Toppano, 1993) may be followed, constituting in an abstract framework in which different knowledge components and representations coexist. Focus-based mechanisms are introduced in order to reduce the size of the complexity.

In both cases, we have achieved the design and implementation of truly concurrent programs able to represent diagnostic reasoning. Our common architecture does not assume any shared memory nor any necessary synchronization among concurrent processes. By distributing knowledge and tasks we obtain a powerful framework for representing the knowledge
underlying a skill. By organizing different categories of intelligent actors, we obtain an efficient medium-grained model able to depict the numerous levels of the diagnostic activity. Our framework of distributed computation is similar to that defined by Agha (Agha, 1986), augmented by more powerful communication schemes not allowable in Agha's model and available as primitives from object-based concurrent languages (Yonezawa, Tokoro, 1987) (Briot, and Gasser: 1992)

3. Concurrent Diagnostic Reasoning of Mental States

The first domain on which our architecture has been used is essentially composed of a student which tries to solve simple high school algebraic problems. This case study is characterized by a remarkable complexity, due to the heterogeneity of the required knowledge and to the richness of problem solving activities (Brown, and Burton, 1978) (Sleeman, 1982).

Our basic scenario consists of:

- a first degree equation, such as 5-4x = 3+6x+7;

- a pupil, which, once received the original equation, generates a transformed equation, derived by applying correct and/or wrong problem solving strategies.

We use the identifier TX to denote the variable terms such as 6x, and the identifier TN to denote the constant terms such as 5.

The system is based on the activities accomplished by four categories of actors, which are designed to exploit the concurrency of the tasks, and to benefit from the knowledge distribution. Hereafter, a brief description of the most important actor classes, with a synthetic explanation of their duties:

Domain-Actors

These actors are used to represent the domain to analyze. In our case study, they depict the terms of the equation.

Rule-Actors

These actors embody problem solving strategies that can be adopted by the pupil. For this reason, they represent "right" reasoning tactics as well as "wrong" actions.

Sewer-Actors

Our approach consists in generating partial diagnostic models, each of them handled by specific actors. In order to link together these single spots of the diagnosis, we use Sewer-Actors to establish a complete collaboration among those actors responsible of the partial analysis.

3.1 Active Search on Active Domain

The Domain-Actors allow us to have in the system a set of intelligent entities which know in what manner it is possible to handle the terms of the equation. This information is important, and its adequate use represents a key issue at the basis of the overall diagnostic system. In
fact, the possible operations that the student can apply on a term depends strongly on its "form" and "position". These features are implemented in terms of constraints among Domain-Actors.

In the following, a few simple Domain-Actors.

**TN-ADD**
This actor represents a TN term which can be added with a similar TN term.

**TN-MULT**
This actor represents a TN term which can be multiplied with a similar TN term.

**TX-ADD**
This actor represents a TX term which can be added with a similar TX term.

**TX-MULT**
This actor represents a TX term which can be multiplied with a similar TX term.

Essentially, the role of these actors is to establish what possible resolution strategies can be chosen - on the terms- in order to solve the equation. These strategies depend strongly from the involved term, and from the remaining elements of the equation. For this reason, it is necessary that the initial knowledge owned by a Domain-Actor evolve in such a way as to acquire a complete understanding about all the remaining terms present in the equation.

### 3.2 Problem Solving Strategies

The Rule-Actors apply the possible resolution strategies on the different terms which compose the equation. The resolution strategies include both the correct mathematical rules and the incorrect ones: we represent with the same formalism both the exact knowledge and the misconceptions of the pupils.

Hereafter some of the actors that represent the correct rules:

**X-ADDSUB**
The execution of this actor allows to add -or subtract- the terms TX

**N-ADDSUB**
The execution of this actor allows to add -or subtract- the terms TN.

**N-L-TO-R**
The execution of this actor corresponds to a displacement of a term TN from the left to the right hand side of the equation.

**X-L-TO-R**
The execution of this actor corresponds to a displacement of a term TX from the left to the right hand side of the equation.

**TERM*TERM**
This actor carries out the multiplication between two terms TN.

**NULL**
The execution of this actor leaves unchanged the state of the equation.

As stated before, we used the Rule-Actors to represent any incorrect technique that may be conceived by the student.

Here some of these rules follow:

*N-L-TO-R-SIGN-ERR*
This actor serves to stress the fact that the student did not change the sign of a term TN after its displacement from the left to the right hand side of the equation.

*PREC-ERR*
The execution of this actor on a designed term of the equation underlines the error made by the student on the application of a priority law on the operators.

*TN-AS-TX*
The student could treat in the same way two different kinds of terms, e.g. a TN term as a TX term (or vice-versa). To handle this error, this actor accomplishes the above-mentioned wrong action.

To optimize the overall diagnostic deduction, the Domain-Actors identify directly the designed Rule-Actors to apply on the terms. After this execution, the Rule-Actors observe the resulting form of the equation. If such a form is more similar to the one provided by the student, then the same Rule-Actor sends such information to its caller, added with more information essential to better drive the remaining diagnosis.

**3.3 Collaborative Diagnosis**
The Rule-Actors, in parallel, try to apply the corresponding solving strategies on the single terms of the equation. Each time that the result is a good candidate in the explanation of the student's cognitive activity, then the Rule-Actor replies to the caller Domain-Actor its personal micro-diagnosis. In such a way, each Domain-Actor collects a set of different solutions, each one devoted to explain a single resolution technique. The organization in a unique diagnostic model of all these portions of knowledge about the student's cognitive process is performed by a designed category of actors, named Sewer-Actors.

In order to better exploit the distribution of the knowledge on the network, we leave disseminated the micro-diagnosis. An extended protocol of communication is established among the Sewer-Actors, in such a way to set up a complete collaborative reasoning which ends by providing a complete explanation about the student's reasoning. Such organization exploits the benefits that the actor computation models provides to this kind of architecture, e.g. a better distribution of knowledge and intelligence over single reasoning activities, in our case actor entities.

To obtain a complete diagnostic model it is necessary to represent any possible, right or not, student's resolution strategies. Due to the complexity of this situation, it could happen that, at the end of the Sewer-Actors' collaboration, we do not obtain a complete explanation model. Several cases can occur, each of them must be treated differently.
4. Concurrent Diagnostic Reasoning of Physical States

Expert reasoning in the domain of physical diagnosis must face a high degree of complexity (Abu-Hanna, Benjamings, and Jansweijer, 1991): faults are detectable through observable symptoms, in general possible faults are very large and several observation may be applied with different standpoints. For this reason many approaches fail to deal efficiently with the complexity (Hamscher, 1991).

The integration of multiple knowledge-based sources represents an important improvement for the completeness and soundness of the diagnostic reasoning (Chittaro, Guida, Tasso, and Toppano, 1993) since this approach fits the natural human approach to use and amalgamate different problem solving tactics and information.

The framework of our architecture is composed by the following agents:

Structural-Agents (StrAs).
Each StrA contains local structural knowledge about its composition (micro-structural model). The complete set of StrAs, thanks to an appropriate collaboration scheme, provides a complete topological view of the system to be diagnosed together with all possible components links. They correspond to the Domain-Actors used to represent the possible terms in an equation to analyze.

Behavioral-Agents (BhvAs).
Each BhvA exerts a specific operating behavior which reflects the possibility of a correct, or faulty behavior for a given device. Analogously with the cognitive approach of the previous section, for each StrA, we may have different behaviors, i.e. different BhvAs, in such a way that the collection of BhvAs gives an active distributed operational model of the physical system.

Proposer-Agents (PrpAs). Each PrpA suggests a diagnostic proposal (micro-diagnosis), analogously to the Sewer actors. A complete diagnostic path (CDP) is built via a collaboration protocol inside a collection of PrpAs.

Teleological-Agents (TelAs) represents, in an active form, a possible goal of the diagnosed system. In practice, each TelA knows what settings must be executed to obtain the represented goal and the operational conditions which favour its realization by means of correct settings.

Focuser-Agent (FcsA).
This agent is useful to speed up the identification of a subset of the Structural-Agents representing the most probable fault components. This subset is called the focus.

Arbiter-Agent (Arbiter).
It guarantees the synchronization between the different agents.

4.1 Collaboration Schemes

In order to stress the collaboration activities we provide only the script-based representation of the agents.

```
(defActor StructA (extends& component ) (slots& ...) (method (simulate ...))

STRUCTURAL-AGENT
```
Figure 1 illustrates the overall diagnostic process by stressing the cooperation among the different agents. For the sake of simplicity, we can divide the overall process in two principal phases. The first phase (depicted by Figure 2), consists in the construction of the focus-of-attention, e.g., a collection of Structural-Agents which are candidates to justify the
malfunctioning(s). This task is an important optimization that allows the reduction of the complexity of the diagnostic search.

The second phase (depicted in Figure 3) explains the collaboration activity that lead to the generation of CDP.

Considering Figure 1, we show below the collaboration control-flow (the symbol) occurring between the two phases.

(1) $\leftrightarrow (1''),(13''')$
(2) $\leftrightarrow (2''),(3''),(4''),(6'')$
(3) $\leftrightarrow (5')$
(4) $\leftrightarrow (7''),(11'')$
(5) $\leftrightarrow (8''),(9'')$
(6) $\leftrightarrow (1''),(7''),(8'')$
(7) $\leftrightarrow (2''),(3''),(4''),(6'')$
(8) $\leftrightarrow (9''),(10'')$
(9) $\leftrightarrow (11''),(12'')$

In the first part of the diagnostic reasoning (Figure 2), after having reported a malfunctioning, the user activates the diagnostic reasoning (\(\leftrightarrow\)) on a given domain. This request is captured by the Arbiter. This agent starts a collaboration (2') with the Teleological-Agents in order to establish the portion of the domain, named teleological-focus, where, most probably, the malfunctioning is present having reported the given symptoms (3'). Thus, the responsible Teleological-Agent interacts with the Arbiter (4') and with the user (5') in order to decide if the anomaly is due to a user's mistake.

If no human error is found then an actual diagnostic search is sent from the Teleological-Agent to the Arbiter (6'). In this case, the Arbiter interacts with the Focuser-Agent (7') in order to obtain a focus of attention (a sub-set of the teleological focus). The cooperation between the Focuser and Structural-Agents (8',9') inside the teleological-focus (see Figure 2) accomplishes the recognition of the focus of attention (10') that is sent from the Focuser to the Arbiter (11') Now the second phase begins (Figure 3): the Arbiter asks the Structural-Agents belonging to the focus of attention to activate (in parallel) their behaviours (1''). The cooperation between these Structural-Agent and each of the corresponding Behavioral-Agents starts (2''): possible micro-diagnosis are deduced (3'',4'') by comparing them with the symptoms. The Behavioral-Agent, for each new local (compatible) local diagnosis, creates a Proposer-Agent identifying the micro-diagnosis (5'') and sends the new diagnostic information to the relative Structural-Agent (6''). When all the Behavioral-Agents have communicated their local diagnoses to the relative Structural-Agent, this can start a new justification; this action is performed in parallel and conducted by the Arbiter (7'') (8''). The global diagnostic reasoning is assured by the cooperation of the Proposer-Agents: each Structural-Agent informs the corresponding Proposer-Agent to pursue the search (9'') within an updated focus of attention. Each Proposer continues the justification considering the symptoms still not justified and the Structural-Agents still not used in the justification (notice the similarity with the cognitive reasoning) (10''). The cooperation strategies (9'') and (10'') are thus repeated until all the symptoms are justified. If the reconstruction of the diagnostic
path is complete, then a CDP is returned; the Proposer-Agents send the CDP to the Arbiter (11’’); otherwise an incomplete diagnosis is returned (12’’).

Conclusions

This paper reports about our experience in designing a generic, cooperative distributed problem solving method suitable for diagnostic reasoning. The main results support the initial intuition that actors exchanging asynchronous messages allow to design relatively simple solutions to a class of complex and (apparently) different problems such as those associated to cognitive diagnosis and troubleshooting of physical devices.

The architecture has been developed for the user/student modeling scenario in ABCL/1 (Yonezawa, 1990) and further extended and tested for the diagnosis of physical systems in Rosette (Tomlinson, Scheevel, and Singh, 1991). Other experiments have been done in the field of adaptive hypertext/hypermedia (Dattolo, and Loia, 1996) where diagnostic deduction is useful to adapt contents and interfaces to new user’s requests.

Some important features are common in user modeling as well as in physical diagnosis, proving the appropriateness of the architecture for diagnostic reasoning in general. Our future research intentions are focused in improving the diagnostic architecture for dealing incompleteness (Gisolfi and Loia, 1995) and in deepening the role of distributed computation in knowledge-based communication (Cerri, and Loia, 1995).

References


Figure 1: The framework and the collaboration among agents.
Figure 2. The construction of the focus-of-attention.
Figure 3: The building of a Complete Diagnostic Path.
Issues in the Integration of Object-Oriented Technology with Parallel and Distributed Computing Requirements

Jean-Pierre Briot
Thème "Représentation et Traitement des Connaissances Distribuées" (RTCD)
Laboratoire d'Informatique de Paris 6 (LIP6)
Université Paris 6 - CNRS
Smail: LIP6, UPMC - Case 169, 4 place Jussieu, 75252 Paris Cedex 05, France
Tel: +33 (1) 44 27 36 67 - Fax: +33 (1) 44 27 70 00
E-mail: briot@laforia.ibp.fr - http://www.laforia.ibp.fr/ briot
April 24, 1997

Abstract

This talk aims at identifying, classifying, and discussing the various ways along which the "object" paradigm is used in parallel and distributed contexts. We distinguish the applicative approach, the integrative approach, and the reflective approach. The applicative approach applies object-oriented concepts, as they are, to structure parallel and distributed systems through libraries. The integrative approach consists in merging concepts such as: object and activity, message passing and transaction. The reflective approach integrates protocol libraries intimately within an object-based programming language. We discuss and illustrate each of these approaches and we point out their complementary levels and goals.

At the end of the talk, we will tentatively raise further issues (and point some directions) for integration of object and agent (multi-agent, software agent, mobile agent) technologies, as well as validation.
Dialogical Frameworks

Carles Sierra and Pablo Noriega

Institut d’Investigacio in Intel.ligencia Artificial
CSIC, Bellaterra

There are many situations where intelligent agents interact among themselves by exchanging illocutions: cooperative problem solving, negotiation, trading.... In these situations agents are usually subject to social conventions -- such as an interaction protocol, shared meaning of language, roles and authority,...-- that may be more or less explicit, more or less severely enforced or more or less flexible. These conventions, however, constitute a common social framework that imposes shared "dialogical" restrictions over the possibly very rich potential individual illocutory behaviour.

In some contexts, as is the case of the fish market, these restrictions are incarnated in an "institution" who constitutes a location and -- through intermediaries -- enforces ontological and behavioural standards. On other occasions -- as is the case in argumentation-based negotiation -- it is a less concrete external environment, composed of hierarchies and commercial relationships for example, what needs to be acknowledged to give meaning to illocutions such as threats. Nevertheless, it seems that even widely differing dialogical contexts such as these two have many structural components in common: illocutory languages, protocols, roles, agent types,...

In this paper we explore the possibility of formally characterizing such structural components as what we have called "dialogical frameworks". We outline the basic definitions and illustrate their usage. We also argue on their significance for providing "accountable" interactions in multi agent applications such as agent-mediated market places.
From Blackboards To Agents

Iain D. Craig

Department of Computer Science, University of Warwick, Coventry CV4 7AL, UK EU

email: idc@dcs.warwick.ac.uk

The blackboard architecture has been successfully used for many years. It has often been associated, incorrectly, with real-time problems and, more correctly, with some notion of co-operative processing. The architecture as typically implemented is based on the structure of the HEARSAY-II system, but this is only one possible interpretation of the underlying blackboard metaphor.

The metaphor describes the problem-solving process in terms of experts who collectively solve problems by writing on a physical blackboard. This is a metaphor for collective and co-operative problem solving which incidentally employs a publicly visible store of intermediate results. When construed in this more abstract fashion, there are a number of aspects which are not included in the HEARSAY-dominated architecture.

In this paper, we will consider an alternative interpretation of the blackboard metaphor, one which emphasises the collective properties which the metaphor describes. By taking the concept of an expert seriously, additional properties become clear, as do the constraints upon communication which the usual interpretation enforces. The result of this new interpretation is not one architecture, but a range of possibilities, each of which permits a different level of autonomy in the components corresponding to the experts of the metaphor.
Fuzziness in Compilers and Runtime Systems

A.C. Sodan

Abstract. We will review some techniques used in compilers for parallel and distributed systems that involve aspects of fuzziness. We consider fuzziness in a general sense being present if the techniques applied classify, rate or otherwise handle information with some tolerance spectrum, operate with similarities, or are not seeking for perfect/optimal solutions (but apply heuristics or approximations). In many cases, fuzziness in this sense is already used — although often theoretically not well-based — and in other cases integrating fuzziness may improve approaches, open new perspectives or will become relevant for future applications as in distributed computing. This offers the challenging task of working towards a theoretical basis for the involved fuzziness, and this basis may be laid by the use of fuzzy theory. Other approaches like statistics or neural networks may offer solutions for some aspects as well, however. We outline mostly the potential application of fuzzy theory and the metrics required in this paper. At some points, the paper is visionary in this respect only, because concrete metrics are not yet in sight. As a side product, the paper investigates — from the view of a user of it — the various benefits and forms of fuzziness. This work evaluates and extends the experience obtained from defining and implementing a fuzzy inference system for a specific compiler configuration task, performed in a joint work with Vicenç Torra.

In compilers much of the information we deal with has to be exact and certain. For example, language syntax and semantics have to be exact and unambiguous. And compilers are required to compile programs in a way such that their semantics are completely preserved despite any transformations and optimizations that may be applied internally. This refers to compilers in general, but in the context of parallelization especially means that sequential code only can be transformed to parallel one, if it can be guaranteed that there are no data-access conflicts. Thus, one on hand, verification is a desirable goal, growing support being provided by theory and formal methods. On the other hand, it means that the basic guideline in compiler construction is to perform certain and safe actions with respect to the application semantics. Furthermore, finally decisions are to be taken from a discrete set of possibilities — i.e. in a crisp way — when applying or not applying a specific transformation or a specific implementation strategy.

However, in many cases nevertheless some fuzziness — which here is meant in a general sense of uncertainty and approximateness — is involved or may at least be used to advantage. In principle, fuzziness can be applied on all aspects that cannot result to faulty behavior. Thus, often fuzziness comes into effect as far as performance is of concern only. So for the parallel speedup, nobody minds if we achieve 0.97% of the optimum only. Furthermore, fuzzy criteria may just be used for a pre-selection of relevant solutions, of which then the best one may be selected by crisp methods. In some cases, solutions may be available only, if not insisting in perfect matching of the criteria. In general, fuzziness often appears related to some kind of discrete optimization problem (search for solutions), relaxing then the criteria being used in it.

Relaxing the criteria in a fuzzy sense, usually pays off by a significantly reduced complexity of the compiler in code and runtime, e.g. if dealing with the basic problem of generating efficient instruction code. Relaxing criteria is — as quite a standard approach — appearing in the form of heuristics, approximations, or implicitly operating dynamic adaptation methods, but this often is not used in a theoretically well-based way. Furthermore, fuzziness may become essential for performing abstracting classifications (e.g. of application-characterizing data), allowing decision making (e.g. in the sense of pre-selection) at a higher and usually more effective level, this also reducing complexity. In some situations, case-based approaches are or may be applied in the compiler for limiting the optimization effort, fuzziness then showing by applying similarity instead of equality for the criterium. Currently, discrete optimization even becomes increasingly relevant by the growth in the number of problems of its kind, e.g. in the framework of task or data mapping to parallel machines, complex transformations for parallelization, or configuration in distributed systems. Relaxing the criteria in order to manage optimization-complexity while aiming nevertheless at a sound theoretical foundation, then becomes a challenging task. From the viewpoint of the compiler, this requires to extract the application points of fuzziness, establish the criteria and define the corresponding metrics that quantify the attached fuzziness.

In this paper, we will give several examples for the presence or application of fuzziness in the compiler and runtime system, and — where possible — outline the potential corresponding metrics (e.g. for application characteristics). Thereby, we discuss also the different ways how fuzziness shows and can be exploited to advantage. Our considerations mostly have fuzzy theory in mind, which is quite promising for offering a good theoretical framework. Fuzzy theory has already very successfully been practically applied in fuzzy control, and currently is — with respect to its theoretical framework — in the process of extension to logic reasoning, being in most cases what is required in the context of compilers and runtime systems. Our experience with a fuzzy inference system (based on fuzzy pre-selection rules) for a specific configuration task — developed together with the fuzzy-theory expert Vicenç Torra — demonstrated that fuzzy theory then can really be highly supportive in the compiler field.
Towards Meta-Agent Protocols

Andreas Kind

School of Mathematical Sciences
University of Bath
www.maths.bath.ac.uk/~ak1

Abstract

Software agents "live" in changing environments. Perception and actions of agents need to adapt dynamically to new situations.

This talk is concerned with meta-agent protocols, an approach to support the implementation of adaptive agents. Meta-agent protocols are derived from meta-object protocols in object-oriented dynamic languages which enable defined access to language design and implementation. In a similar way, a meta level interface to the internal representation of agents can provide support for introspection and adjustment of agents (e.g. communication protocols, behaviour, user preferences). Meta-agent protocols also result in a clear separation between application level and meta level in agent systems.
On the Finiteness of Resources in Distributed Computing

Luc Moreau* and Christian Queinnec†
University of Southampton and Université Paris 6 & INRIA-Rocquencourt

Abstract

Millions of computers are now connected together by the Internet. At a fast pace, applications are taking advantage of these new capabilities, and become parallel and distributed, e.g. applets on the WWW or agent technology. As we live in a world with finite resources, an important challenge is to be able to control computations in such an environment. For instance, a user might like to suspend a computation because another one seems to be more promising. We present a paradigm that allows the programmer to monitor and control computations, whether parallel or distributed, by mastering their resource consumption.

As we live in a world with finite resources, it is of paramount importance for the user to be able to monitor and control computations. This task is all the more complex since computations may be parallel, distributed, and most probably making use of code written by others. This problem is particularly illustrated by the Internet as the user wishes to search information over the WWW, exploits parallelism to improve efficiency, relies on distribution to increase locality, but also wants to concentrate the computing power in the most promising directions to reduce searching time.

There are two types of applications that we particularly wish to program. First, a user that has initiated a computation over the Internet should be able to suspend the computation in order to analyse the results he has already obtained. If these were unsatisfactory, he must be able to resume the computation where it was suspended in order to collect the next results. In this example, computations should be understood as possibly parallel and distributed over the net. Second, service providers offer computing facilities to customers who subscribe to their service by transferring electronic cash; in return, service providers supply them with a handle to submit jobs, create accounts that they debit according to the usage of the facilities, and inform users of the exhaustion of their account. Again, jobs submitted by customers may generate parallel and distributed computations which must be monitored by the service provider.

Our goal is to provide the means by which everybody, customers and service providers, can get the most out of a situation with bounded resources. We believe that parallel computations can be driven by mastering their resource consumption. We present a new language, called Quantum, containing primitives to monitor and control computations. The basic principle of the semantics of Quantum is to keep track of the resources consumed by computations. Resources can be understood as processors cycles, bandwidth and duration of communications, or even printer paper. We shall adopt a more generic view by saying that computations need energy to be performed. Although the notion of energy is part of the semantics of Quantum, the programmer cannot create energy ex nihilo, but can only transfer it between computations via some primitives of the language. As a result, we were able to ensure a general principle for Quantum: given a finite amount of energy, any computation is finite.

Besides its resource-oriented semantics, Quantum is designed to be parallel and multi-user, and to be independent of the memory model (central, distributed, with or without coherence). All these features make it particularly suitable for programming in a distributed environment like the Internet. Furthermore, the language offers the means to write programs of the form “when this computation ends, do this”, which is an essential feature in distributed computing.

*Department of Electronics and Computer Science, University of Southampton, Southampton SO17 1BJ, UK. Email: L.Moreau@soton.ac.uk. This work has been partially funded by EPSRC GR/K30773 and EC project reference ERB 4050 PL 930186
†Université Paris 6 — Pierre et Marie Curie, UFR d'informatique, 4 place Jussieu, 75252 Paris Cedex, France. Email: Christian.queinnec@inria.fr. This work has been partially funded by GDR-PRC de Programmation du CNRS and EC project reference ERB 4050 PL 930186

32
Towards Rigorous Compiler Implementation Verification

Wolfgang Goerigk, Friedemann Simon

{wg|fs}@informatik.uni-kiel.de

April 16, 1997

Abstract

We will give a comprehensive technical overview of our work on rigorous verification of both the compiling specification and the compiler implementation of an initial full correct binary compiler program executable for a subset of Lisp. We will concentrate implementation verification. Machine program correctness will be proved using a special technique of bootstrapping the compiler twice, and of double checking the results. The entire compiler correctness proof crucially depends on this formal technique: we use a compiler as a kind of proof assistant which prints out a formal derivation of the target program from the source program. Afterwards, we check this derivation to be in accordance with the compiling specification.

Although often left out of sight, implementation verification is definitely necessary, not only for compiler programs and not only for safety but also for security reasons for instance in network computing. We will show as a result, that rigorous implementation verification is feasible. The key for success is modularization and a combination of mathematical proof, formal methods and controlled mechanical support.

Our contribution is to actually perform this work for compilers and, hence, to relieve the application programmers burden to prove implementation correctness again and again. Once this work has been done conscientiously and is accepted to reach sufficient mathematical certainty, compilers may be used for full correct program implementation, safely in the sense that whenever they succeed in generating a target program, that implementation is semantically correct w.r.t. the source program. The use of formal methods for program construction and verification, often required for the certification in safety critical application areas, will be sound also w.r.t. the real thing, the machine executable, without further binary machine code inspection.
Experiences with PVM: Communication Issues in Distributed and Cooperative Applications

Maurizio Giordano

Istituto di Cibernetica, CNR Arco Felice, Naples

Distributed Computing generally refers to applications that can be naturally decomposed in several tasks (or activities) that may execute concurrently and exchange data. We present two very different distributed applications: the Parallel Environment for Cellular Automata Network System (PECANS) and the Fishmarket Multiagent System (FM).

In PECANS, task decomposition may be at different levels of task granularity exploiting both control and data parallelism. At low level concurrency is obtained by decomposing automata grid data space in partitions on which the same transition function is applied; data topology and connectiveness are the main issues in implementing communication at this parallelism level.

The FM is a cooperative distributed application: there is no proper task decomposition but different autonomous programs ("agents") performs different roles and interact to make the global system evolving. Dynamic configuration of agent connectiveness, grouping facilities, asynchronous communication are the main issues for this kind of applications.

We used PVM in both PECANS and FM implementations and we show in this presentation some implementation choices we made to cope with the above mentioned communication issues. We also presents some experimental results in order to demonstrate the differences of the two applications.
Connecting systems for defeasible argumentation with negotiation protocols

Henry Prakken

Faculty of Law, Vrije Universiteit, Amsterdam

In this talk I will discuss how logical research on defeasible argumentation can be relevant for research on negotiation protocols. Throughout the talk I will assume that such protocols are to be used by computer systems that mediate negotiations between actors, whether human or artificial. Starting from the observation that negotiation often involves disputation, i.e. arguing about the tenability of a claim, my main point will be that models of defeasible argumentation, as studied in research on nonmonotonic reasoning, should be an element of the disputation part of negotiation protocols.

I will first outline the main features of logical systems for defeasible argumentation. Such systems define the notions of an argument and a counterargument, and they define how, relative to given evaluation criteria, arguments can be assessed as tenable or not. One form that the assessment can take is the dialectical form of a debate between a proponent and an opponent of a claim. The main idea is that the proponent has shown an argument for the claim to be tenable if the opponent eventually runs out of moves in every way of attack.

Then I will compare dialectical systems for defeasible argumentation with disputation protocols. Such protocols are also aimed at testing the tenability of a claim, and they also have a dialectical form, but two main differences are that in disputation protocols the parties are real actors instead of abstract dialectical roles, and that in such protocols the knowledge base is not given in advance, but constructed dynamically during a dispute. For these reasons disputation protocols need a speech act level, defining, for instance, the speech acts of making, denying or conceding a claim, and a 'procedural' level, regulating when these speech acts can be made, and how they affect the commitments of the parties to the various claims.

I will end by briefly discussing how the connection between systems for defeasible argumentation and protocols for dispute could be made, and by listing some open problems, such as the problem of defining an appropriate language in which human users can input arguments into mediating systems.
DIM Agents and Frameworks

David DeRoure and Jonathan Dale

Department of Electronics and Computer Science, University of Southampton, Southampton SO17 1BJ, UK
dder@ecs.soton.ac.uk

Distributed Information Management

Distributed Information Management (DIM) refers to the management of multimedia information across distributed and federated systems and applications. The DIM initiative is promoted by the UK Technology Foresight panel on IT and Electronics to help promote research into the management of change and evolution in distributed systems, particularly in multimedia applications.

Our work has identified four principal DIM activities [DD97b]:

1. Resource discovery
2. Information Integrity
3. Navigation Assistance
4. System Integration

Technologies for DIM

We have conducted a number of projects within the DIM arena by applying the open hypermedia philosophy in a distributed environment. According to this approach, hypermedia links are resolved by a third-party 'link service', and the user may adopt a query-oriented style of interaction as opposed to the traditional use of buttons [Hal94]. This has led to the successful Microcosm TNG (The Next Generation) [GDDH96] and DLS (Distributed Link Service) [CDHH95] systems.

Hypermedia systems are now well established and the engineering issues in creating various types of system have been investigated extensively. The current challenge is not so much the creation of the systems but their integration, and this applies to both the infrastructure and the information: interfaces and protocols can be standardised so that software components interoperate, but there is the additional requirement of useful information exchange across these interfaces.

With the requirements of DIM systems in mind, we have reformulated our approach to distributed open hypermedia from the perspective of an agent system, introducing the notion of DIM Agents [DHDD96]. The term 'DIM' is intended to reflect both the application domain and the fact that intelligence is not necessarily an essential characteristic of these agents. Our previous systems exhibited autonomy and reactivity, but agents provide the necessary proactivity to be adaptive to new situations and take initiative. Furthermore, the concept of mobility is compelling because it captures the notion of a 'visitor' to a resource, which is consistent with the models by which DIM data is managed.
According to the classification of Wooldridge and Jennings [WJ95], our DIM Agents are weak: they are autonomous, reactive, proactive and have social ability. They may also be mobile.

Frameworks

In order to develop solutions to support multiple DIM projects and to maximise interoperability, our first goal has been to design the frameworks for DIM agent systems. Having tested these frameworks, we can then build reuseable software components and establish a methodology for constructing DIM applications.

We are investigating three frameworks:

1. Layered model
   This resembles the seven layer model adopted in telecommunications, with distributed systems primitives at the lowest level building up to DIM applications level at the top (layer five of our model). Software components, at various levels between the machine and the application, provide agreed functionality so that implementors at any level can ‘plug and play’. It is important to identify the layers before too many systems are created which assume different structures.

2. Architecture for DIM agents
   This is the design of the layer which supports DIM applications, and we have investigated an architecture with four classes of agent: the domain agent, resource agent, mobile agent and user interface agent [DD97a]. This architecture has been prototyped in APRIL (Agent Process Interaction Language [MC95]). Every object within the architecture is either an agent or abstracted through an agent.

3. Anatomy of a DIM agent
   Like many agents, DIM agents have components which record, represent, compare and action. The core DIM agents need not incorporate sophisticated knowledge processing; they need the technology to be reactive and proactive. In our prototypes, the behaviour of an agent is specified by a ‘program’ which can be downloaded into an ‘empty’ agent (this permits agent behaviours to be treated as documents in a DIM system, and potentially agent state too).

Summary

Distributed Information Management is an important application area and we believe DIM agents to be an appropriate technology to meet the requirements of this area. Issues of interoperability and scalability must be addressed at all levels, and for the DIM agent technologies this means the need to address frameworks for the construction of DIM systems.

References

CDHH95

**DD97a**

**DD97b**

**DHDD96**

**GDDH96**

**Hal94**

**MC95**

**WJ95**
Converting Declarative into Procedural (And Vice Versa)

Iain D. Craig

Department of Computer Science, University of Warwick, Coventry CV4 7AL, UK EU

email: idc@dcs.warwick.ac.uk

Procedural and declarative dynamics are often seen as irreconcilable alternatives in Computer Science. Declarative semantics are often held as an ideal to which all should aspire. The purpose of this paper is to argue that, in reflective systems such as our ELEKTRA production rule interpreter, the distinctions between procedural and declarative are relatively arbitrary and depend upon where one is in the reflective tower and the use to which an element is put. We argue that higher levels of the tower are able to view objects at lower levels as both procedural and declarative. In addition, the components of rules can be similarly viewed, depending upon perspective and location in the tower.

In order to establish this position, we first describe ELEKTRA. ELEKTRA allows rules to interpret other rules and makes its internal structure available to rules in order to permit them to function in sophisticated ways. As part of the system, there is a pre-processor which converts rules into assertions which describe their structure and content. These assertions are stored in working memory and are used to optimise the process of reflection. It has been shown that ELEKTRA can, itself, be completely interpreted by two meta rules. ELEKTRA can be thought of as an implementation of the reflective tower concept. This concept allows us to change perspectives on objects, as we argue in section three.

At one level in the reflective tower, we have rules that are just executed to produce changes in working memory. At higher levels of the tower, such rules can be handled as if they were data; this is a characteristic of any reflective system, but there is no explicit reification operator in ELEKTRA because it relies upon an implicit quotation mechanism. We also see how the procedures that implement actions in ELEKTRA rules can be given a declarative interpretation; we also see that certain condition elements naturally have a procedural interpretation, but this can also be treated in a declarative fashion when appropriate. What we see is that the interpretation of the components of ELEKTRA rules depends upon how they are used. We finally consider a slight alteration to ELEKTRA in which additional dynamics can be introduced. This is done by making rule syntax considerably cleaner and more uniform and then by attaching rules which determine the way in which rules and their components are to be interpreted. This allows many dynamics to be expressed, and, moreover, expressed in a context-sensitive fashion (this is because rules can inspect the local state and decide what to do). We also make some comments about the possibility of multiple, concurrently active interpretations.
Trading Scenarios for heterogenous agents

Francisco Martín and Joan Antoní Rodríguez

Institut d'Investigació in Intel.ligencia Artificial
CSIC, Bellaterra

In order to test agent capabilities, we propose a framework for defining trading scenarios inspired on the fish market auctions. In these scenarios, agents of arbitrary complexity can participate in electronic auctions under a collection of standardized market conditions and be evaluated against their actual market performance. We argue that the competitive situations we propose constitute convenient problem domains in which to study issues related with agent architectures in general and agent-based trading strategies in particular.

Our proposal involves a set of conventions for the typification of goods, bidding protocol, availability of goods, buyer endowments and performance evaluation criteria. The proposed framework is being implemented on a multi-agent testbed on top of FM96.5 --- a Java-based version of the Fishmarket downward-bidding auction house.

We present illustrative examples of standardized market conditions and discuss how competitive trading scenarios can be designed and activated.
LSI–97–1–R “On the Number of Descendants and Ascendants in Random Search Trees”, Conrado Martínez and Helmut Prodinger.


LSI–97–14–R “NoFun: A Notation to State Non-Functional Specifications at the Product Level”, Xavier Franch.


LSI-97-19-R. "Especificación de restricciones de integridad en el sistema ROSES" (written in Spanish), Maria Amélia Pacheco e Silva and Maria Ribera Sancho i Samsó.


LSI-97-22-R. "Refining Logical Characterizations of Advice Complexity Classes", Albert Atserias and José L. Balcázar.


LSI-97-27-R. "Mean Field Theory of Fluid Neural Networks", Jordi Delgado and Ricard V. Solé.


Hardcopies of reports can be ordered from:

Nuria Sánchez
Departament de Llenguatges i Sistemes Informàtics
Universitat Politècnica de Catalunya
Campus Nord, Mòdul C6
Jordi Girona Salgado, 1-3
08034 Barcelona, Spain
secrelsi@lsi.upc.es

See also the Department WWW pages, http://www-lsi.upc.es/www/