KnowledgeWeb

—

A Design and Feasibility Study

B. Buchberger[2], M. Chakravarty[1], J. Darlington[3],
Y. Guo[3], T. Ida[1], I. Mejuev[1], and W. Schreiner[2]

ISE-TR-98-151

[1] University of Tsukuba, Japan
[2] Research Institute for Symbolic Computation, Linz, Austria
[3] Imperial College, London, United Kingdom

Abstract

Training people on science and engineering as well as providing scientific and engineering solutions are key abilities in the technology-driven development of our society. Recent innovations in global networking and, in particular, the creation of the World-Wide Web and the widespread use of mobile-code technology, put a new generation of software systems within our reach, which promise a new quality of knowledge dissemination and absorption.

Herein we document our design of a combined software technology basis and content creation methodology (tentatively named KNOWLEDGEWEB) that addresses both modern education methods and distributed solution providers in the area of science and engineering. In accordance with the twofold goal, we propose two main technologies: *active books* and *knowledge on demand*. The first component, active books, blurs the distinction between traditional textbooks and educational software by integrating content-dependent software into the static text (like, for example, a computer algebra system into a textbook on mathematics). The result is a new medium, which assists the student in acquiring the desired knowledge. The second component, knowledge on demand, does not aim at teaching knowledge, but at providing knowledge encoded in solvers, i.e., software components that realize solution procedures, such as constraint solvers, data-mining procedures, and so on. In other words, it favours a black-box approach to knowledge dissemination in contrast to the white-box approach of active books. These solvers may be spread over the global network, some of them may require special hardware resources like supercomputers. For many problems a set of solvers has to be combined into concerted operation, and moreover, a convenient user-interface is needed.

Contents

1 Introduction

The technology-driven development of our society requires sophisticated methods for training people on science and engineering as well as for providing scientific and engineering solutions. The current state of global networking and, in particular, the World-Wide Web and the widespread use of mobile-code technology, put new software systems within our reach, which promise a new quality of knowledge dissemination and absorption. In the following, we introduce a software technology basis and content creation methodology, called KNOWLEDGEWEB, that addresses both modern education methods and distributed solution providers in the area of science and engineering.

1.1 The Vision

This report outlines the design of a new generation of software systems that provide a new quality of knowledge dissemination and absorption by combining three key areas in which the involved groups have developed strong experience during the past years:

- distributed software systems,
- advanced programming and problem solving paradigms,
- mathematical education and training.

In our opinion, the combination of these technologies will yield software products for conquering two crucial markets focusing on important social and commercial challenges of the next century:

- training people on science and engineering;
- providing scientific and engineering solutions.

Both markets address the need of people to learn to cope with a constantly changing and increasingly complex environment that is essentially shaped by human technology:

1. The first market refers to a person's ability to understand and to influence the developments in a particular application domain. This requires sound education in order to learn the laws of the domain, to analyze complex problems within this domain, and to provide a functional solution to problems in this domain. The domain thus must become a "white box" whose internal workings the person must know.

2. The second market refers to a person's ability to utilize solutions to problems in unfamiliar application domains. This requires the ability to contact solution providers for these domains, send them problems and get answers in return, both formulated in a suitable language. This domain can be viewed as a "black box" whose internal workings the person need not care of.

Of course both markets are not contradictory but complementary: we need to educate experts in one application domain that can develop solutions for this domain based on the knowledge of experts in other domains.

1.2 The Spearheads

In Section 2, we outline two technologies addressing the markets outlined above. These proposals are based on our convictions that the demands can be met by novel approaches to knowledge presentation and dissemination only.

1. **Educating People: *Active Knowledge***

   Up to today, knowledge transfer mostly relies on the use of paper-based books. The knowledge presented by such books is completely "passive": the reader tries to understand the thoughts of the author without being able to ask questions, to run exercises and get feedback, *there is no possibility for interaction with the book.*

   We propose to integrate existing software components into an authoring environment for the development of active books that incorporate facilities for the kind of interaction sketched above. The book can embed executable contents and it can draw resources from the Internet. Based on this environment, an interactive mathematics book is being developed as a case study.

   Today, a couple of software systems serve the need of developing (CD-ROM based) multi-media products for the infotainment market. Our proposal extends these goals by addressing the special requirements of scientists and engineers to interface with technical software systems and Internet resources and especially emphasizing the interaction between reader and book.

2. **Providing Solutions: *Knowledge on Demand***

   Up to today, there is no good way to provide scientific and engineering solutions over the network. Low-level approaches based on facilities such as Web-servers with CGI interfaces are rather ad-hoc and do not easily allow the multi-disciplinary integration of solution servers from different application domains.

We propose to develop an environment for the creation of servers for scientific and engineering solutions. The core of this environment is a middleware that integrates standard client frontends (Web browsers) on one side with various types of technical software systems (Mathematica, MathLab, ...) on the other side based on a generic format for the exchange of technical data between server and middleware and a generic format for the presentation of results in the client frontend. These formats are based on already ongoing developments in these fields (OpenMath, MathML).

Today, a couple of software systems serve the huge market of SMEs (small and medium enterprises) developing Web servers for online commerce over the Internet (browsing catalogues, ordering products, ...). Our proposal intends to provide similar services for scientists and engineers to enable them to put their services on the network, which requires different kinds of technologies and interfaces.

Of course, the technical basis of both proposals partially overlap. While both activities can be pursued independently and separately, their combination would yield considerable synergy effects.

1.3 The Technologies

The vision formulated in the previous section is based on a couple of key technologies in which our groups have a long standing experience. Various related projects are already being pursued by these groups individually or in collaboration (see Section 3).

Distributed Software Systems Today we experience the increasing demand to *interconnect* software products that previously operated in an isolated mode in order to enable them to *cooperate* to achieve a common goal. As a result we get *distributed software systems* consisting of many components which asynchronously execute in different locations and whose activities have to be appropriately coordinated (see e.g. Projects 3.1.2, 3.1.4, 3.1.3, 3.3.1, and 3.3.2). For this purpose, *middleware* is employed in order to free the components from low-level tasks such as locating other components or selecting appropriate communication protocols (see Project 3.2.1).

Nowadays, the Internet serves as a general-purpose communication backbone with high-level services such as the World-Wide-Web built on top of it; the programming language Java promises to become the language of choice for Internet based systems. Middleware standards such as CORBA or widely available products as DCOM or Java's RMI may save as the glue to combine components to distributed applications. Multicasting features embedded in Web browsers, presentation tools, and groupware start to support distributed collaboration.

The distributed applications outlined in the current report will use these individual technologies as standard components; there is no necessity to develop proprietary counterparts. However, the real challenge lies in their integration into a consistent framework which requires technical expertise as well as knowledge in the application domain as provided by the project partners.

Advanced Programming and Problem Solving Models Many of the scientific and engineering services which the current report addresses can be best formulated and solved using programming and problem solving models that are on a much higher level than conventional imperative programming languages such as C, C++, or Java. Examples of such paradigms are functional programming, logic programming, constraint solving, automated proving in restricted mathematical theories using special methods, as well as any combinations of these methods (see e.g. Projects 3.1.4, 3.2.2, 3.2.3, 3.3.1).

As an example, take the very important engineering problem of modeling control systems and answering questions about properties such as their stability. A very convenient way of solving this problem is to formulate each component of this system as a higher-order predicate (parameterized over predicates describing its sub-systems) that relates certain measurable quantities expressed in real numbers (pressure, temperature, ...); this formulation can be nicely expressed in a functional/logic language with constraints. Plugging together the descriptions of the individual components yields a predicate which can be evaluated by a language interpreter that generates systems of constraints over the reals to be handled by a constraint solver.

A server solving control system problems may therefore consist of a functional/logic language interpreter combined with a constraint solving engine; this engine (one or more, since the constraint solver may be based on a parallel algorithm) may itself represent an autonomous server offering its functionality over the network.

In a similar way, many of the services addressed by the current report may make use of various advanced programming and problem solving methods.

Mathematical Education and Training An important aspect for the preparation of active knowledge in the form of interactive books is a clear understanding of the logical structure of the material being presented and the relationship between its individual components, in order to master the complexity of their interaction and to find appropriate means of presentation. This is much more difficult than in conventional paper-based books since the individual components may depend in various ways on each other, may be used in many contexts and may be invoked by each other and by the reader of the book (executable contents). This raises many logical and software-engineering issues that can be only appropriately addressed if a formal basis (model, theory) of the underlying domain is available according to which the presentation is structured (see Projects 3.1.1, 3.2.2, and 3.3.2).

As an example, take an algebra course presenting knowledge (theory, algorithms, reasoning strategies) about various concrete and abstract mathematical structures such as algebraic numbers, rings, and polynomials. One possibility to organize the material is to structure it in the form of *functors*, i.e. domain parameterized over other domains, where each functor encapsulates all the knowledge about the domain (carrier set, operations, proving methods) that can be appropriately invoked from other functors depending on it. Correspondingly a tower of mathematical structures can be constructed and accordingly presented, supporting demonstration examples can be developed, giving the user the possibility to *experience*, to *exercise* and to *experiment* on all the material covered by the book.

To develop education and training material therefore not only requires the technological basis but also guidelines and techniques how to develop and structure the material; example material has to be developed and to be elaborated as *tutorials* for developers of new material.

2 KnowledgeWeb Proposals

The following proposals outline two concrete possibilities how the combined expertise of the partners can be effectively utilized to create a new generation of software systems. Each activity can be pursued independently and separately, but their combination would yield considerable synergy effects.

2.1 Active Knowledge

The transition from traditional print media to electronic documents significantly increases flexibility and creates new opportunities; but to date, this potential is seldom exploited fully. A printed book is by its very nature a passive object; but an electronic book can be active, in the sense that it can *react* to manipulations by the reader. More precisely, content-depended software components may be integrated, which complement the static text. Input from the reader is processed by this software—the book reacts and adapts in accordance. We call these generalized books *active books.* The supplied content is not just passively read, but reacts to feedback from the reader.

The transition from standalone PCs to networked computing environments provides additional features for active books. It is no longer required that the complete content, including its active components, resides on a single machine. Instead it may be spread over multiple processing elements, some of them may have special features, such as special I/O devices or special performance, needed for certain active components of the book. The result are *distributed* active books.

The main purpose of active books, just as with their passive predecessors, is to convey knowledge, which is acquired by the reader. The learning process can be greatly aided by the active nature of the medium, independent of whether the learned material is the differential calculus or the use of a new software tool. In the case of the differential calculus, the book may contain a computer algebra system, such as Mathematica, to check a students answers to exercises and to provide solutions to problems stated by the student. In the case of the tutorial for new software, the active book will include the software whose use is studied, to provide the hands-on experience, which is so vital in learning how to operate new software.

In other words, traditional books contain passive knowledge, whereas we want to provide *active knowledge.* Active knowledge is more accessible for the learning person as it can adapt and react to the learner. Hence, the new technology is not merely a new medium, but it adds value.

2.1.1 State of the Art

On first sight, it may seem that the whole Web is a huge distributed active book, with all its CGI scripts and Java applets realizing the active components. But this view is too general and does not match our concept of active books. The main differences between active books and the Web in general are twofold: (1) the Web is a mainly self-organizing, collective effort of independent contributors, whereas an active book is the consequence of a co-ordinated effort of a comparatively small group of people; and, partially as a consequence, (2) the Web is highly diverse and fragmented, whereas the content of an active book is monolithic, albeit structured. Nevertheless, active books can, of course, be provided via the Web, i.e., they can be elements of the Web.

As a consequence, the technology of the Web, including HTML, Java, and CGI, forms the basis of the development of active books, but it is too general to be used directly, as it would require substantial coding efforts that distract from providing content. In other words, we require an additional set of tools that allows to provide content without worrying too much about the technical realization of the resulting active book.

It should be clear that our intention is not the development of mere electronic support for displaying text or multimedia documents, such as hypertext. We take the idea of electronic documents one step further and integrate content-dependent software. Such extensions have, to date, received considerable less attention, although they provide many new opportunities.

Nevertheless, there already exist documents that have to be considered as active books by our definition, but we observe that the technology is not fully utilized and, more importantly, a methodology and tool set to write active books is missing. Nevertheless, the existing instances demonstrate the usefulness of the concept, and our main aim is to simplify the production, such that it is accessible to a broader group of authors and is more cost-effective.

2.1.2 Proposal

We propose the development of (1) a methodology and (2) a tool set for the creation of active books. Both are meant to minimize the efforts needed to create active books and to allow the authors to concentrate on creating content. The proposed software makes use of standard formats and languages, in particular, HTML and Java.

The active research that is under way at the Imperial College, RISC, and the University of Tsukuba provides important background for the proposed project. In particular, various activities described in the Appendix are of immediate interest for software supporting active knowledge.

The *Theorema* project (Appendix 3.1.1) investigates the use of an extension of Mathematica to develop textbooks in the area of Mathematics that include an engine to support proofs and mathematical reasoning. Furthermore, the *Distributed Algebraic Geometry* project (Appendix 3.1.2) develops distributed algorithms for classroom presentations. Both projects require expertise not only with regard to educational software, but also with regards to its implementation in distributed environments and the integration of active components into hypertext presentations.

Another project that is concerned with the development of *Middleware* for distributed systems is described in Appendix 3.2.1. It features the integration of state-of-the-art interfaces between different software components and especially the integration of Java-based user interfaces. Finally, our work on *Distributed Goffin* 3.3.1 investigates the high-level specification of distributed systems. The last two projects, demonstrate our efforts to simplify the use of distributed systems, the integration of diverse software components, and generation of modern user interfaces.

Methodology The development of active books to realize active knowledge requires a design methodology on two levels: first, a programming methodology that integrates hypertext viewers with content-dependent software components, and second, a content design-methodology that uses the new technology to maximize the benefit for the reader. The programming methodology defines an interface technology between the different components of the system, using modern protocols, such as Java RMI or CORBA. Furthermore, it guides the development of the active components in an active book. The content design-methodology essentially defines in which way active components should be used in active books in a science and engineering context.

Tools To simplify the development of active books, we propose to develop authoring tools supporting the integration of heterogeneous active and passive components into hypertext documents as well as the distribution of the components over multiple processing elements. Authoring applications provide the possibility to search, browse, and organize distributed content, while creating an front-end representation of an active book in the form of HTML-based documents, which may be enhanced with a scripting language such as JavaScript. HTML and JavaScript define the outline and representation of embedded content originating from external Web sites. Deployment of a HTML-based outline of an active book, generated by means of the authoring application, on a Web server makes it possible to consider a completed active book itself as an active content element and, therefore to embed one active books within each other. The user interface of authoring applications is ideally implemented as a thin Java-applet client, providing a high degree of portability and interoperability.

Content used by active books may include active (Java applets, CGI, ActiveX, VRML virtual reality worlds) as well as passive elements (multimedia, HTML documents)—see also [BN96]. Optionally, to provide a high-performance storage for passive content, third party object-oriented database servers might be applied. Authoring active books correlates closely with content requesting and delivery techniques. To request a context element, an active book interfaces with a middleware component, which provides transparent content referencing, search, and delivery. The middleware functionality from the point of view of active books closely matches the requirements imposed by the *Distributed Solution Warehouse* to the *Knowledge Mediator* middleware (Section 2.2) [BS97, MLDM96]. CORBA/Java ORB and HTTP/CGI models can be nicely applied to implement middleware with integrated functionality. These models also provide a standard interface to middleware components. An integrated middleware enables the use of *Distributed Solution Sites* as a kind of active content available for the creators of active books.

2.2 Knowledge on Demand

2.2.1 Overview

The World Wide Web (WWW) is becoming a low-cost, platform independent standard interface mechanism with which to access the computational resource that are distributed all over the world. Moreover, the powerful idea of executable contents or active information has opened new and exciting avenues which will lead WWW to a new landscape of organising and assimilating scientific and technical knowledge that are distributed all over the world into a global-wide knowledge base. With the efficient deployment of such an ever-growing knowledge repository, the way of working in industry and academic societies will have a revolutionary change.

We consider the research in this new field is of fundamental scientific, commercial and social importance. We therefore propose as a research and development project to build up a prototype of Web based global mathematical knowledge on demand service engine which aims to provide to workers in industry and academy all kinds of mathematical services via the Internet using Web. The system is based on the Web based distributed computing model where the mathematical solutions contributed by the expertise in various scientific decipline are organised into a globally distributed solution warehouse. These solutions can be provided on demand to the user via the Knowledge Gateway middleware to the users all over the world.

2.2.2 System Architecture

The system is based on a distributed computing model. It composed of geographically distributed computation sites, each of which provides its local mathematical solutions. At each site a set of *Solution APIs* are provided to support the solution deployment from local experts and also a *Site Management Agent* which handle the communication between the site with its clients.

The entire site is integrated via Web as a globally distributed knowledge warehouse. This integration is Supported by the *Knowledge Mediator* middleware which provides a centralized control of the distributed sites and acting as a broker for building clients and the distributed servers to deliver knowledge to its required clients. The Math-lounge is provided as a Web based GUI client environment, which provides a uniform view of the distributed knowledge warehouse.

The functionality of those major components is described in detail as follows:

- **Math-Lounge:**

  The Math-lounge is a Web-based client tool for deploying distributed knowledge service. It is composed of an intelligent search engine, which supports an efficient querying to the site management database for required service. The Math-lounge also uses Java Applets technology to provide active information for delivering solutions.

- **Knowledge Mediator:**

  The knowledge Mediator acts as a middleware establishing the connective between clients and the server sites containing the solutions required by the clients. The middleware is also served as a framework for managing the distributed service sites and service provision. The major components of the middlware include:

  - Site Management Database: it consists of a repository recording the detailed information about each registered solution sites. This repository is maintained automatically via a set of site agents dynamically updating the information of each registrated sites. The repository will be used to build up the math-louge for deploying the service and also will be used to monitor the status of the whole system.

  - Service Manager: it is a database recording user information including the accounting information of service provision, statistic information about service usage of each site and the database of dynamic messages of user's feedback. It also provides a set of service help functions including user help, service alias, alternative service for fault tolerance.
  - Solution Broker: it is responsible for establishing connection between a service requesting client and its required service. The broker is implemented using Java RMI object communication mechanism.

- **Distributed Solution Warehouse:**

  The geographically distributed computation sites, each of which provides its local mathematical solutions, integrated based on the Web infrastructure, form a distributed solution warehouse. Each site consists of a set of:

  - *Solution APIs*: which are used to support the solution deployment from local experts. It is implemented using JavaIDL.
  - *Site Management Agent*: which handle the communication between the site with its clients. It also responsible for reporting the updated information of the site.

2.2.3 Research Programme

The research aims to build up a prototype of the knowledge on demand engine in 2 years time. The first year we will focus on develop the key components and their integration technology. We also will at this stage to build up several solution sites for proving concepts. In the second year, we will focus on the development of Math-lounge GUI and Site APIs. Many functions for commercial services such as accounting, security and system maintainance will be investigated and integrated into the system. A full functional demonstration of the KoD engine will be delivered by the end of the second year.

3 Project Survey

KNOWLEDGEWEB draws from a collection of projects underway at the Research Institute for Symbolic Computation (Linz, Austria), the Imperial College London (United Kingdom), and the University of Tsukuba (Japan). The results of these projects build the basis on which KNOWLEDGEWEB is developed, and so, KNOWLEDGEWEB's feasibility critically rests on this previous and ongoing work. The purpose of this section is to survey these projects and put them into relationship with KNOWLEDGEWEB. The projects a grouped by research institutions and preceded by a brief outline of the research group.

3.1 Research Institute for Symbolic Computation

The Research Institute for Symbolic Computation (RISC-Linz) of the Johannes Kepler University Linz has pursued for almost two decades research on computer algebra, computational geometry, computational logic, automated reasoning, constraint solving, functional and logic programming and parallel and distributed processing with applications in geometric modeling, computer-aided engineering, robot simulation, NC-programming, expert systems, and neural networks.

The institute has a faculty of 15 members with doctoral degree. About 50 students from mathematics and computer science (more of half of which are from abroad) currently pursue their diplomas respectively Ph.D. Six doctoral students of RISC-Linz have graduated *sub auspiciis praesidentis.* Prof. Buchberger is chairman of the section of mathematics and computer science of the European Academy of Sciences and chairman of the Hyper-G consortium promoting the hypermedia information system HyperWave.

By an initiative of Prof. Bruno Buchberger, the Software Park Hagenberg was founded, a technological center that currently incorporates about 25 companies and several academic institutes all of which are related to software (i.e. methods, tools, development, application, marketing, and training). The Fachhochschule Hagenberg initiated by Prof. Buchberger offers two curricula on "Software Engineering" and on "Communication- and Media-Design". The institute participated in the Central European Initiative (CEI) and is a member of the Austria Center for Parallel Computation (ACPC).

Members of RISC-Linz have participated in numerous national research projects (supported by Austrian Science Foundation FWF and the Austrian Federal Ministry of Science and Research BMWF), international research programs (e.g. the European ESPRIT III and ESPRIT IV programmes and in the Japanese RWCP Real-World Computing Program) and also collaborated in various development projects with industrial partners. Currently, RISC-Linz is in collaboration with the University of Tskuba pursuing a two years project on "Distributed Constraint Solving for Functional Logic Programming" sponsored by the Japanese Research Institute For Advanced Information Technology (AITEC).

3.1.1 Theorema

This project pursued at RISC-Linz [The98, BJ$^+$97] provides new technology for generating formal proofs for any application field (mathematics, computer science, engineering, but also "soft" sciences like economy, political science etc.). Theorema is based on the Mathematica software system but goes far beyond the present capabilities of Mathematica, which is limited to mathematical problem solving based on known mathematical result but with no reasoning power.

A variety of general and special reasoning techniques is being implemented in Theorema including provers for propositional and first-order predicate logic, inductive domains (natural numbers and lists), non-inductive domains (polynomials), and simplification provers [Buc96]. The proofs generated by the system are presented in natural language [BJV97] and in a structured way that allow the user to control the level of detail displayed. Future versions of the system will provide tutoring components for various groups of users (math and computer science students, engineers, software developers, scientists, managers, etc.) to the system. These tutoring components will introduce the users to formal reasoning techniques and gradually develop their reasoning potential.

We expect that the system will have a wide market. In the future world of rapidly changing technologies, the techniques of mastering clear, precise, fast and complex intellectual analysis and reasoning will prove to be one of the fundamental and unchanging contents of global education. Improving the quality and reliability of formal thinking will have a significant impact on the further development of research, technology, economy, and society. Formal thinking and reasoning techniques can only be trained by intensive interaction with an experienced teacher in a wide variety of examples that adapt to the individual background and educational level of the student. Hence, this type of teaching until now was not available to the masses. The Theorema system will be able to put this individual and interactive tutoring potential right onto the desk of the user be it students in one of the traditional branches of education (high school and academic education) or in the increasingly important areas of adult and recurrent education.

3.1.2 Distributed Algebraic Geometry

In this project, RISC-Linz is developing a distributed version of the library CASA that provides a set of powerful methods for performing computations and reasoning about geometric objects in classical algebraic geometry [TW97]. CASA is implemented as a set of packages in the computer algebra system Maple which provides powerful presentation and visualization facilities (similar to those of Mathematica).

We have developed an application framework for writing parallel programs in Maple which allows to create concurrent task and have them executed by Maple kernels running on different machines of a network. The system called "Distributed Maple" [Dis98b, Sch98] consists of two components:

1. A Java class library which implements a general purpose communication framework and scheduling mechanism for distributed applications.

2. A binding that allows to access the Java scheduler from Maple and implements a parallel programming model on top.

The portable technological basis of the system makes it easy to install it in any environment, to implement distributed algorithms, to exchange them among researchers, and to present the algorithms in classroom. We are now going to develop an “electronic textbook” of such algorithms that can serve as the basis of integrated presentations.

3.1.3 Distributed Algorithms Education

In the frame of this project, we have developed the toolkit DAJ for designing, implementing, testing, simulating, and visualizing distributed algorithms in Java [DAJ97, Sch97]. DAJ consists of a Java class library with a simple programming interface that allows to develop distributed algorithms based on a message passing model. The resulting programs may be executed in standalone mode using a Java interpreter or embedded as applets into HTML pages and executed by Web browsers. The goal of the toolkit is to provide an universally accessible platform for research and education in the area of distributed algorithms. DAJ is freely available over the World Wide Web.

Our motivation for this work stemed from some uneasiness with how to teach distributed algorithms and programming: there are various excellent textbooks on this topic, but they describe distributed algorithms in an abstract notation rather far away from real programs. Furthermore, there is a lack of an easy to use and universally accessible platform for implementing the algorithms taught in class and investigating their dynamical behavior. We therefore have designed and implement the DAJ toolkit that provides an easy way of programming distributed algorithms and visualizing their dynamic behavior based on a programming model that is intuitive but still close to "real" systems; this gives the possibility to develop documents that integrate text and executable code in a single framework.

3.1.4 Distributed Constraint Solving for Functional Logic Programming

This two-years project (1997–1999) is being pursued by RISC-Linz in cooperation with the University of Tsukuba and sponsored by the Japanese Research Institute For Advanced Information Technology (AITEC). Its goal is the development of a distributed software system consisting of

- a functional logic language interpreter on one machine based on an existing implementation developed by the University of Tsukuba on the computer algebra system Mathematica;
- a number of constraint solving engines running on other machines based on various methods created and implemented by RISC-Linz using C++ and the software library Sturm.

The functional logic language is extended in two directions: two directions:

- the possibility to specify (non-linear) constraints over real numbers;

- the possibility to specify parallelism within the program.

We aim to target the language towards hierarchical modeling complex control systems from engineering using the higher-order features of the language and solving the large constraint sets using the computational power available in large computer networks.

A first skeleton prototype of the envisioned system was delivered in March 1998 and is freely available [Dis98a]. The system implements the constraint functional-logic programming calculus CFLP [MS98] which is an extension of the narrowing calculi LCNC and HLNC developed at the University of Tsukuba [MOI96a, MO95, HI97].

CFLP is implemented by an interpreter in the computer algebra system Mathematica. During execution, the interpreter cooperates via the MathLink protocol with external processes that concurrently solve systems of constraints over real numbers. The first prototype runs on a single machine; it is going to be extended to an implementation that handles non-linear constraints over the reals by external solvers running on various machines in a network. The system can be then applied to problems from application domains like physics or electrical engineering.

3.2 Imperial College London

The recent history of centrally provided parallel computing at Imperial College started with the inauguration of the Imperial College / Fujitsu Parallel Computing Research Centre (IFPC) in May 1994. This Centre was formed by a donation from Fujitsu Laboratories of Japan of a 128 processor Fujitsu AP1000 distributed memory parallel machine. The Centre's role has been to run an open programme of parallel applications and research throughout Europe. The subsequent enthusiastic and successful adoption of parallel computing within Imperial College led the College, again in collaboration with Fujitsu, to submit a bid under the 1996 HEFCE JREI for upgraded resources to support a multi-disciplinary programme dedicated to Imperial College users. This bid was successful and led to the award of £1.65M. Subsequent negotiations with Fujitsu enabled the initial proposal (a Fujitsu AP3000 with 80 U200 UltraSPARC processors) to be upgraded to an 84 processor AP3000 with 60 U300 processors, a VX vector machine and 500GBytes of backing store.

The new facility constitutes the Imperial College Parallel Computing Centre (ICPC) which started operation in summer 1997 with the installation of the VX vector processor and the first 16 nodes of the AP3000. The AP3000 was upgraded in January 1998 with the delivery of the first 48 of the U300 processors and is scheduled to be upgraded to the full configuration in June 1998. The IFPC continues to operate in parallel with the ICPC providing an open service internationally.

The ICPC's role is to provide support for multi-disciplinary applications of high performance, parallel computing, across all the constituent Departments of Imperial College. Professor Darlington acts as Director of the Centre and Dr. Guo as Technical Director.

The ICPC serves a practical and an academic role. Its practical role is the physical provision and support of state-of-the-art HPC resources to Imperial College workers. Its academic role is to link this activity with the development and dissemination of appropriate computational techniques in order to achieve maximum effective use of these resources. It achieves this latter purpose by working with the applied groups, gathering and disseminating shred knowledge and experience, and by linking its activities to fundamental computational methods research carried out by research groups directly associated with the Centre in the Department of Computing.

Current research in the Department of Computing directly associated with the Centre includes

- **Data Mining and Networked Computing** Sponsored by FECIT £200,000
- **High-level Parallel Environments** EPSRC £260,000
- **Optimisation and Decision Support** EPSRC £300,000
- **High Performance Financial Computing** ESRC (ROPA) £260,000

3.2.1 Distributed Data Mining and Decision Support System

This project pursued by the Imperial College Parallel Computing research Centre (ICPC) and supported by Fujitsu European Centre for Information technology aims to build a tool-rich, platform-neutral, high performance and dynamically customisable decision support system for the AP3000 Fujitsu parallel server by applying data mining and parallel optimisation technologies and a World Wide Web-based client-server computing model. This system will provide powerful support for decision making in an internet/intranet based enterprise setting by fully embedding the decision making functions into the World Wide Web infrastructure. The overall design implements the "3-tier" architecture for Web-based applications by integrating a Web server and a set of decision support application servers into a middle tier between the client-side decision making tools and the data server that provides uniform access to databases [DMG97, CDG+98a].

The application is presented to the user as a web-based client that integrates visualisation for data and knowledge with a powerful visual programming environment [CDG+98b]. In this programming environment the users interactively explore and analyse data sets and produce data analysis procedures, which can be re-used and modified later. Local and physically remote databases can be accessed, but the distributed nature of the application is transparent to the user.

The client is implemented with Sun's JavaBeans and executes in standard web browsers.

The middle layer of the application provides system services for handling user objects and manages the components of the system. It includes a subsystem for running user sessions, storing persistent objects, transaction handling, and remote database access. It implements security policies that control users' access to data, data analysis procedures and results. The whole system consists of components that interact via APIs (application programming interfaces). Components can be maintained and upgraded without affecting the rest of the system and new components that conform to the system API can be added. The middle layer manages the repository of components that are available at any given time in an installation of the system. Further features include remote database access via JDBC (Java Database Connectivity) and full auditing and accounting logs. The architecture of the middle layer is generic and can be specialised for particular application environments, with instances for data mining and financial modelling currently under development. It is implemented with Sun's Enterprise JavaBeans (EJB).

Server components consist of high-performance data analysis functions that are optimised for parallel machines [DGST97]. These functions are encapsulated as distributed objects and can be executed on dedicated machines which can be physically remote. Any kind of computationally intensive application can be provided in this way, and current functions include parallel data mining and financial modelling. The component integration facility is also used to build interfaces with third-party systems such as statistical packages. The server objects are implemented as CORBA objects, using JavaIDL. The high-performance algorithms are written in C with the parallel MPI library.

The most important development of this project so far is the design and implementation of the Java-based middle layer. This system is designed as a generic middleware to support a wide range of Web-based client-server computing models. It can be extended to support a fully distributed knowledge deployment system as proposed in the KnowledgeWeb project.

3.2.2 Automatic Generation of Parallel Financial Code

This project is pursued by the Imperial College Parallel Computing research Centre (ICPC) and supported by Economical and Social Research Concil of UK. The research activities of the project is centralised by building up a financial modelling environment which generate parallel simulation code from financial models expressed in mathematical terms by the financial engineers.

The basic building blocks of the system will include a model specification system, a knowledge base, a set of libraries of parallel numerical codes and an interactive program synthesis engine. The model specification system will be based on Mathematica enhanced with graphical user interface facilitating modelling and visualisation activities. The specification system will allows modellers to design their models directly at the mathematical level. The underlying symbolic computation system provides a uniform notation for the mathematics of specification and the procedural description of the knowledge necessary for code synthesis. It simplifies incorporating analytic model and lets transformations apply algebraic reasoning such as determining convergence criteria, establishing series approximations and estimating truncation errors.

The knowledge base provides an object oriented representation of domain knowledge and rules that implements many model design choice such as rules for coordinate transformations and discretisation (such as Crank-Nicholson, ADI or Runge-Kutta methods). The knowledge base also provides mechanisms to support interactive model design and implementing various design choices such as the solver to be used with the equational system. The set of parallel numerical codes will be implemented to provide the essential functions for building up various solvers. The interactive program synthesis engine will carry out the program synthesis tasks through a series of levels that progress from mathematical specification to algorithm synthesis and then to computational structuring and then to parallel code generation.

The system will be implemented for any parallel computer system running the MPI standard communication library so that the generated codes will be generally portable. The project will use the 80 node massively parallel Fujitsu AP3000 in the ICPC. It is expected to be widely used by financial engineers. It therefore forms another important site in the KnowledgeWeb system for providing financial modelling solutions.

3.2.3 Applying Constraint Logic Programming Languages for Modelling Multi-objective Decision Making under Uncertainty

The project, supported by AITEC of Japan, involves the application of constraint logic programming technologies to build up a modelling tool for large scale multi-objective decision problems. The distinct advantage of the CLP specification lies in its declarative nature and the flexibility of abstraction, thus enabling operations research to be effectively applied to real world applications. A CLP specified business model can then be translated into constraints involving binary integer variables and the resulting mixed integer linear programming problem solved using parallel constraint solvers.

The major components of the proposed integrated and holistic solution to the modelling problem will comprise —

1. A CLP based framework that allows the natural specification of complex modelling and decision making procedures.

2. A methodology for translating such logic-based specifications into an integer programming formulation.

3. The modelling language will interface with a set of parallel constraint solvers. The generated constraints will be solved using underlying constraint solvers developed at Imperial College.

4. The whole system will be developed to facilitate incremental expansion. An open interface to a wide range of constraint solving algorithms for interrelated integer programming algorithms will be provided using the API (application program interface) technique.

In the project, the user interface will be based on the the WWW technology. Using Java Applets the logic based modelling language will be available via the Web. This will provide a platform independent client tool for decision makers. The solvers will run on a high performance server. Using this Java enabled Web-based client-server model, the power of high level modelling and high performance constraint solving can be integrated and deployed for a wide range of business users to solve large scale multi-objective decision making problems. Therefore, this project provide a concrete site in the KnowledgeWeb system providing particular service for multi-objective decision making.

3.3 University of Tsukuba

The Symbolic Computation Group (SCORE) of the University of Tsukuba is concerned with symbolic computing and has an emphasis on the design, implementation, and application of declarative programming languages. Over the years we were engaged in extensive research on the integration of declarative paradigms—in particular, of functional programming and logic programming. We designed computational models of functional logic programming based on a rewriting method called narrowing and applying our expertise in term rewriting. We defined narrowing calculi with mathematical rigor and proved various properties of the calculi. Based on the narrowing calculi we built several systems, among which are a distributed functional-logic system written in JAVA, a parallel functional-logic system (written in C) running on Fujitsu's AP1000+, and functional-logic system written in Mathematica, which runs on a variety of computers that are supported by Mathematica. Our competence in the design of computation models and theoretical investigation of the models are highly regarded in the international research community.

Nowadays, our efforts are oriented towards parallel and distributed computing. Until recently, we have used a Fujitsu AP1000+ (32 processors) and, in the near future, we will install a high-performance workstation cluster. Among our research in symbolic computation that exploits these facilities is integer programming based on a parallel Buchberger algorithm. This project is conducted in cooperation with Prof. Darlington's group at the Imperial College, London. Furthermore, we are concerned with the parallel implementation of functional languages and their extension to provide a high-level approach to parallel and distributed computing. Our research in this area spans from semantics and language design over optimized implementation techniques to programming methodology. Furthermore, our research naturally incorporates another important research area connected to constraint solving, namely distributed functional-logic programming with constraints. SCORE and RISC are now jointly developing a prototype system that is aimed at the industrial use of functional and logic systems with constraints.

We believe that the conceptual complexity of distributed systems requires strong support from the used programming languages and programming environments, to ease the development and maintenance of complex systems. Languages with a functional and logic programming heritage seem ideal due to their clean semantics, type safety, concise denotation and manipulation of complex data structures, as well as support for modular programming and code reuse.

The SCORE group currently consists of three faculty members (Prof. Dr. Tetsuo Ida, Dr. Aart Middeldorp, Dr. Manuel Chakravarty) and nine graduate students.

3.3.1 Distributed Goffin and Mobile Code

During the last five years, members of the Imperial College and the University of Tsukuba have investigated the high-level description of parallel and distributed systems following the idea of separating computation and co-ordination. The research resulted in the language *Goffin* [CGKL98, CGK98] (also called *Distributed Haskell*). Furthermore, we worked in the direction of a multi-threaded implementation framework [Cha97, Cha98] for this language.

Goffin is considered to be a declarative parallel programming language. It utilizes the functional language Haskell [H^+92] as a computational base language and extends it with a co-ordination layer that has its roots in concurrent constraint programming [Sar93]. The co-ordination layer of the language is used to specify and organize a set of concurrently executing agents, which can be distributed over the processing elements of a parallel computer, the workstations of a local network, or the Internet. While maintaining the clear semantics of its functional base language, Goffin provides mechanisms to express reactive behaviour, explicit agent placement, soft real-time constraints, and dynamically reconfigurable inter-application communication—features needed in parallel and distributed systems and often neglected in declarative languages.

The functional base language with its higher-order programming model and flexible evaluation order together with the strict separation of computation and co-ordination supports structuring software. Furthermore Goffin includes features like strong typing, automatic memory management, and a mature module system, which are expected in a modern programming language. All this serves to enhances programmer productivity and opportunities for code reuse.

Our implementation work for Goffin is based on an extension of the Spineless Tagless G-machine [Pey92], which allows a highly optimized implementation of graph reduction. We extended it to include multi-threading and support for agent distribution and communication. Currently, we are working in the direction of a mobile code framework and the use of runtime code-generation techniques (also known as just-in-time compilation). Mobile code implies a high architecture independence as well as the possibility to include code into documents (in the same way as Java applets), which is important for our work on active books 2.1. Runtime code-generation is important to increase the efficiency of the mobile code, but even in a non-mobile environment, it enables code optimizations that are impossible in a traditional compiler. In contrast to Java's byte code, we use a structured code, which leads to smaller binaries and simplifies runtime code-generation and optimizations.

3.3.2 Tele-Teaching

Active books (Section 2.1) provide an interactive environment for education and training, accessing distributed active and passive elements through the middleware interface. A closely related application that requires an integrated middleware providing transparent content requesting and delivery is the *Virtual Classroom* [SANB95, Har95, Hil95, JJH92, BKW97, PK91]. The virtual classroom creates a distributed educational environment providing means for communication between a teacher and a group of students, while supporting the use of heterogeneuos distributed content. A virtual classroom is based on the same design principles and technologies as an active book, but moreover introduces *teacher* and *student* roles and provides real-time communication facilities, as well as dynamic content composition and delivery. A number of teacher-student interaction patterns for a virtual classroom is currently investigated in the SCORE group and an application prototype is under development.

3.3.3 Computational Models for Functional Logic Languages

There is a growing interest in integrating the functional and logic programming paradigms. The computational mechanism of functional-logic programming languages is narrowing. Since narrowing is a rather complicated operation, involving subterm replacement and unification, it is not easily implemented. Hence calculi consisting of a small number of more elementary operations that simulate the complicated narrowing operation have been proposed. Such narrowing calculi are called lazy. In [MOI96b] and [MO98], we made a fundamental study of aspects related to completeness of one such calculus, called LNC. Completeness is the desirable property that every solution to a given goal can be computed by the underlying computational mechanism.

Typically, lazy narrowing calculi have three kinds of non-determinism: (1)

the choice of the equation in the current goal, (2) the choice of the inference rule of the calculus, and (3) the choice of the program rule. Surprisingly, the first kind of non-determinism cannot be removed without further ado, i.e., lazy narrowing calculi—unlike SLD resolution—are in general not strongly complete. Three major results are reported in [MOI96b]. We established a connection between the strong completeness of lazy narrowing calculi and the completeness of basic narrowing, an efficient variant of narrowing. For the latter several sufficient conditions are known. In [MOI96b], we further showed that completeness is maintained by always selecting the leftmost equation. Finally, we addressed the eager variable elimination problem, a well-known open problem in unification theory. It is known that many redundant derivations can be avoided if the variable elimination rule, one of the inference rules of LNC, is given precedence over the other inference rules. We established the completeness of a restricted variant of eager variable elimination in the case of orthogonal term rewriting systems.

In [MO98], we gave reasonable sufficient syntactic conditions under which the second non-determinism can be completely removed, while retaining completeness. In practice these conditions are usually met. So implementations of lazy narrowing only have to do backtracking over the choice of the program rule, just as in SLD resolution.

The work reported in [MOI96b] and [MO98] deals with unconditional systems only. Since the expressive power of functional-logic programming languages is greatly enhanced by allowing conditions in the program rules, it is important to extend the results concerning the removal of non-determinism in lazy narrowing calculi to conditional rewrite systems. In [HM97], we extended LNC to conditional rewrite systems and we showed that the resulting calculus LCNC is strongly complete whenever basic conditional narrowing is complete. Currently we are investigating whether the other results obtained in [MOI96b] and [MO98] can be extended to the conditional case.

In [SNI97], we extend narrowing to the higher-order case and introduce the higher-order lazy narrowing calculus HLNC.

3.3.4 Other Projects

SCORE pursues a number of other projects whose contents are closely related to the current proposal:

1. **Research on the Principles for Constructing Software with Evolutionary Mechanisms**

   Supported by Grant-in-Aid for Scientific Research on Priority Areas Ministry of Education, Science, Sports and Culture of Japan (SCORE is the member research group of the above project) Period: Fy 1997 - 1999.

2. **Computation Model for Higher-order Functional Logic Programming**

Supported by Grant-in-Aid for Basic Research (B), Ministry of Education, Science, Sports and Culture of Japan Period: Fy 1996-1997.

3. **Application of Conditional Term Rewriting Systems to Declarative Programming**

   Supported by Grant-in-Aid for Scientific Research (C), Ministry of Education, Science, Sports and Culture of Japan Period: Fy 1995-1996.

4. **Construction of Multimedia Programming Environment for Functional Logic Programming**

   Supported by Grant-in-Aid for Developmental Scientific Research (B) Ministry of Education, Science, Sports and Culture of Japan Period: Fy 1995-1997.

5. **Computation Model of Programming Languages based on Term Rewriting**

   Supported by Okawa Foundation for Information and Telecommunications, Fy: 1996.

6. **Declarative programming, databases and human-interface in multimedia computing environment**

   Supported by Tsukuba Advanced Research Alliance (TARA), University of Tsukuba Period: Fy 1994-1996.

7. **Feasibility Study of Declarative Coordination Programming in Open Computing Environment**

   (http://www.score.is.tsukuba.ac.jp/IPA/ipa97.html) Supported by IPA, Period: Fy 1996.

References

[BJ+97] Bruno Buchberger, Tudor Jebelean, et al. A Survey on the Theorema Project. In W. Kchlin, editor, *ISSAC'97 International Symposium on Symbolic and Algebraic Computation*, pages 384–391, Maui, Hawaii, July 21–23, 1997. ACM Press, New York. ftp://ftp.risc.uni-linz.ac.at/pub/techreports/1997/97-15.ps.gz.

[BJV97] Bruno Buchberger, Tudor Jebelean, and Daniela Vasaru. Theorema: A System for Formal Scientific Training in Natural Language Presentation. Technical report, Research Institute for Symbolic Computation (RISC-Linz), Johannes Kepler University, Linz, Austria, October 1997. ftp://ftp.risc.uni-linz.ac.at/pub/techreports/1997/97-34/submitted.nb.html.

[BKW97] David J. Brown, Steven Kerr, and John R. Wilson. Virtual environments in special-needs education. *Communications of the ACM*, 40(8):72–75, 1997.

[BN96] M. H. Brown and M.A. Najork. Collaborative active textbooks: A web-based algorithm animation system. Technical Report 142, SRC, May 1996.

[BS97] Bruno Buchberger and Wolfgang Schreiner. Concert: A software architecture for coordinating educational sessions in distributed environments. Technical report, Research Institute for Symbolic Computation (RISC-Linz), Apr 1997.

[Buc96] Bruno Buchberger. Computer Algebra and Logic. In F. Baader and K.U. Schulz, editors, *Frontiers of Combining Systems*, Applied Logic Series, pages 193–220, Munich, Germanyu, March 26–29, 1996. Kluwer Academic Publishers. Invited paper.

[CDG$^+$98a] J. Chattratichat, J. Darlington, Y. Guo, S. Hedvall, M. Köhler, A. Saleem, J. Sutiwaraphun, and D. Yang. A software architecture for deploying high performance solution on the internet. In *Proc. of HPCN*, 1998.

[CDG$^+$98b] J. Chattratichat, J. Darlington, Y. Guo, S. Hedvall, M. Köhler, A. Saleem, J. Sutiwaraphun, and D. Yang. Deploying enterprise data mining on the internet. In *Proc. of PADD*, 1998.

[CGK98] Manuel M. T. Chakravarty, Yike Guo, , and Martin Köhler. Distributed haskell: Goffin on the internet. In M. Sato and Y. Toyama, editors, *Proceedings of the Third Fuji International Symposium on Functional and Logic Programming*, pages 80–97. World Scientific Publishers, 1998.

[CGKL98] Manuel M. T. Chakravarty, Yike Guo, Martin Köhler, and Hendrik C. R. Lock. Goffin: Higher-order functions meet concurrent constraints. *Science of Computer Programming*, 30(1–2):157–199, 1998.

[Cha97] Manuel M. T. Chakravarty. *On the Massively Parallel Execution of Declarative Programs.* PhD thesis, Technische Universität Berlin, Fachbereich Informatik, 1997.

[Cha98] Manuel M. T. Chakravarty. Lazy thread and task creation in parallel graph-reduction. In Chris Clark, Tony Davie, and Kevin Hammond, editors, *Proceedings of 9th International Workshop on Implementation of Functional Languages '97*, Lecture Notes in Computer Science, Berlin, 1998. Springer Verlag. to appear.

[DAJ97] DAJ – A Toolkit for the Simulation of Distributed Algorithms in Java, November 1997. http://www.risc.uni-linz.ac.at/software/distmaple/.

[DGST97] John Darlington, Yi-ke Guo, Janjao Sutiwaraphun, and Hing Wing To. Parallel induction algorithms for data mining. In *Proc. of IDA*, 1997.

[Dis98a] Distributed Constraint Solving for Functional Logic Programming, May 1998. http://www.risc.uni-linz.ac.at/projects/basic/distcon/.

[Dis98b] Distributed Maple, May 1998. http://www.risc.uni-linz.ac.at/software/distmaple/.

[DMG97] Imperial College Data Mining Group. Software architecture for decisioncentre: A web-based distributed data mining system. In *Proc. of PCW'97*, 1997.

[H$^+$92] P. Hudak et al. Haskell special issue. *ACM SIGPLAN Notices*, 27(5), May 1992.

[Har95] L. Harasim. *Learning Networks: A Field Guide to Teaching and Learning Online.* MIT Press, 1995.

[HI97] Mohamed Hamada and Tetsuo Ida. Implementation of Lazy Narrowing Calculi in Mathematica. Technical Report 97-02, Research Institute for Symbolic Computation (RISC-Linz), Johannes Kepler University, Linz, Austria, January 1997. ftp://ftp.risc.uni-linz.ac.at/pub/techreports/1997/97-02.ps.gz.

[Hil95] S. R. Hiltz. *The Virtual Classroom: Learning Without Limits via Computer Networks.* Ablex Publishing, 1995.

[HM97] M. Hamada and A. Middeldorp. Strong completeness of a lazy conditional narrowing calculus. In *Proceedings of the 2nd Fuji International Workshop on Functional and Logic Programming*, pages 14–32. World Scientific, 1997.

[JJH92] D. Johnson, R. Johnson, and E. Holubec. *Advances Cooperative Learning.* Interaction Book Co., 1992.

[MLDM96] W. Ma, Y. Lee, D. Du, and M. McCahill. Networked hyper quicktime for education-on-demand. Technical Report 96-005, Dept. of Computer Science, U. of Minnesota, 1996.

[MO95] A. Middeldorp and S. Okui. A Deterministic Lazy Narrowing Calculus. In *Fuji Inernational Workshop on Functional and Logic Programming*, pages 104–118, Susuno, Japan, 1995. World Scientific, Singapore. http://www.score.is.tsukuba.ac.jp/ ami/papers/dlnc.dvi.

[MO98] A. Middeldorp and S. Okui. A deterministic lazy narrowing calculus. *Journal of Symbolic Computation*, 1998. To appear (Preliminary version appeared in the Proceedings of the Fuji International Workshop on Functional and Logic Programming, Susuno, World Scientific, 104–118, 1995.).

[MOI96a] A. Middeldorp, S. Okui, and T. Ida. Lazy Narrowing: Strong Completeness and Eager Variable Elimination. *Theoretical Computer Science*, 167(1,2):95–130, 1996. http://www.score.is.tsukuba.ac.jp/ ami/papers/lnc.dvi.

[MOI96b] A. Middeldorp, S. Okui, and T. Ida. Lazy narrowing: Strong completeness and eager variable elimination. *Theoretical Computer Science*, (1&2):95–130, 1996.

[MS98] Mircea Marin and Wolfgang Schreiner. The CFLP Calculus: A Mathematica Implementation (Draft 0.4). Technical report, Research Institute for Symbolic Computation (RISC-Linz), Johannes Kepler University, Linz, Austria, March 1998. http://www.risc.uni-linz.ac.at/projects/basic/distcon/deliverables/research/cflp.ps.gz.

[Pey92] Simon L. Peyton Jones. Implementing lazy functional languages on stock hardware: the Spineless Tagless G-machine. *Journal of Functional Programming*, 2(2), 1992.

[PK91] J. Preece and L. Keller. Teaching the practitioners: developing a distance learning postgraduate hci course. *Communications of the ACM*, 3(1):92–118, 1991.

[SANB95] Ben Shneiderman, Maryam Alavi, Kent Norman, and Ellen Yu Borkowski. Windows of opportunity in electronic classroms. *Communications of the ACM*, 38(11):19–24, 1995.

[Sar93] Vijay A. Saraswat. *Concurrent Constraint Programming*. The MIT Press, 1993.

[Sch97] Wolfgang Schreiner. DAJ – A Toolkit for the Simulation of Distributed Algorithms in Java. Technical Report 97-36, Research Institute for Symbolic Computation (RISC-Linz), Johannes

Kepler University, Linz, Austria, November 1997. ftp://ftp.risc.uni-linz.ac.at/pub/parlab/daj/report/report-main.ps.gz.

[Sch98] Wolfgang Schreiner. Distributed Maple — User and Reference Manual (V1.0). Technical report, Research Institute for Symbolic Computation (RISC-Linz), Johannes Kepler University, Linz, Austria, May 1998. http://www.risc.uni-linz.ac.at/software/distmaple/report/report-main.ps.gz.

[SNI97] T. Suzuki, K. Nakagawa, and T. Ida. Higher-order lazy narrowing calculus: A computation model for a higher-order functional logic language. In *Proceedings of the 6th International Conference on Algebraic and Logic Programming*, number 1298 in Lecture Notes in Computer Science, pages 99–113, Berlin, 1997. Springer-Verlag.

[The98] The Theorema Project, April 1998. http://www.risc.uni-linz.ac.at/software/Theorema/.

[TW97] Quoc-Nam Tran and Franz Winkler. CASA Reference Manual (Version 2.3). Technical Report 97-33, Research Institute for Symbolic Computation (RISC-Linz), Johannes Kepler University, Linz, Austria, October 1997. ftp://ftp.risc.uni-linz.ac.at/pub/techreports/1997/97-33.ps.gz.