1. Overview

The Department of Computer Science was founded in 1967 when the first full professorship in computer science was established at the University of Helsinki. It is within the Faculty of Science, along with the departments of Mathematics, Physics, Chemistry, and others.

Our current teaching faculty (January, 1994) comprises 46 full time teachers. It can be categorized using American academic terminology: 3 full professors (Mannila, Tienari, Ukkonen), 5 associate professors (Mäkelä, Sippu, three open positions), 14 assistant professors (5 research oriented "senior assistants" and 9 teaching oriented "lecturers"), 24 teaching assistants (12 research oriented "assistants" and 12 teaching oriented "instructors"). We also employ approximately 15 of our students in teaching on a part−time basis. Eight senior experts are also associated with the department, these being so−called "docents", who work mainly outside the university but occasionally give courses or supervise theses in the area of their speciality. Approximately 20 research positions are financed from outside sources. We also have a staff of 10 persons.

The department annually admits 200 students to major in computer science. The students are selected according to their standing in a national student examination. The number of completed M.Sc. degrees (5 year degree) was 35 in 1991, 27 in 1992, and 37 in 1993. The study time for a M.Sc. degree ranges from five to eight years. Many of our students work in industry, which slows down or stops the progress of their studies. Fairly frequently our students, after having acquired the basic skills in computer science, redirect their studies by transferring to the Helsinki University of Technology or some other educational institution. Many students study computer science as a minor while pursuing a major in another subject, such as mathematics, physics, economics, psychology, or social sciences. We offer two curricula for students minoring in Computer Science. In 1993 our "approbatur curriculum in Computer Science" (15...25 credit units) was completed by 189 students and our "cum laude curriculum in Computer Science" (35...60 credit units) was completed by 108 students.

There are two graduate degrees in Finland: the Ph.Lic. (3 years) and Ph.D. degree (4 years). The latter has higher quality requirements. Both build upon the M.Sc. degree (5 years). The high demand for our M.Sc. graduates to fill well−paid jobs in industry is a fact which has hampered our Ph.D. education. Our department granted five Ph.D. degrees and eight Ph.Lic. degrees in the three year period 1991−93.

Two principal sources provide funds for computer science research in Finland. The most important one for us is the Academy of Finland under the Ministry of Education and Science. The other important research financier is the Technology Development Centre (TEKES) under the Ministry of Trade and Industry. Lately European ESPRIT financing has also become possible for us. The department is currently participating in four ESPRIT projects or networks (VITAL, TransCoop, NeuroCOLT, NEURONET).

The department maintains jointly with the University Computing Centre a good computer science library. It subscribes to most major international journals in computer science and related fields and acquires a majority of the most important computer science books and conference publications. The library maintains publication exchange with approximately 130 other research institutes in various countries. The library is run by a librarian and a secretary.

The University Computing Centre maintains a communication network and offers UNIX, VAX/VMS and PC services. In addition, our department maintains its own network of approximately 40 SPARCstations and three servers. Classrooms with X−terminals and PCs are available. Each office of the department has a workstation or a terminal.

The department has three informal sections that are used in the planning of the curriculum and in administration. The division is not strict, and several research projects span two sections. The sections cover roughly the following subject areas:

1. *General Computer Science* (professor Esko Ukkonen)
(algorithms and data structures, computational complexity, computational geometry, machine learning, neural networks, computer graphics, numerical and symbolic computation, computers in education)

2. *Computer Software* (professor Martti Tienari)
(programming languages, compilers, formal specification and verification, software engineering, distributed systems, computer networks, operating systems, performance evaluation)

3. *Information Systems* (professor Heikki Mannila)
(databases, human−computer interfaces, computer supported co−operative work, information system design methodology, design of databases, text databases, object−oriented databases, logic databases, database structures and algorithms)

The University of Helsinki has many diverse teaching and research offerings related to computer applications. At the Department of Mathematics there is an active group in mathematical logic, numerical analysis and symbolic computation as well as some interest in theoretical computer science. Our students also benefit from the hardware−oriented teaching (e.g. electronics, digital electronics, microcomputers, interface electronics) given at the Department of Physics. In the Faculty of Social Sciences some teaching and research is devoted to computational statistics, administrative information systems, and the social effects of data processing. In the Faculty of Arts there is a research unit in computational linguistics and a degree program in linguistic theory and cognitive science.

2. Research

2.1. Review

The research at the department has evolved over the years similarly to the international research trends in computer science. Early work in numerical analysis in the 1960's made room for work in programming languages and compilers in the 1970's. Since then the research has diversified and its volume has increased.

Algorithms and data structures is the central research area with longest tradition in the section of General Computer Science. A wide spectrum of algorithmic problems have been studied and several top–quality results have been obtained. The work on string matching algorithms (Ukkonen, Tarhio) has been particularly strong and successful. Structural complexity theory (Orponen) and data structures (Nurmi) are other strong areas. Theoretical work has often been conducted within the framework of systems research providing practical motivation for the problems studied. The motivations and applications of the algorithms developed have come e.g. from Prolog implementation and from the problems in biocomputing.

Neural computing and machine learning are active research directions related to artificial intelligence in the section of General Computer Science. In neural computing the two main research topics have been the design and implementation of a high–level knowledge representation language for neural networks that are capable of performing Bayesian reasoning (Myllymäki, Orponen, Tirri), and the complexity theory of neural associative memories (Floréen, Orponen). The machine learning research group (Elomaa, Kivinen, Mannila, Ukkonen) has studied different learning models and the complexity of learning tasks within these models. The more practice−oriented work has developed, for example, new Occam algorithms for learning decision trees and decision lists, and software tools for testing and comparing various machine learning algorithms. Examples of the newest research themes are the investigation into the foundations of genetic algorithms (Floréen), and experiments in predicting the secondary structure of proteins using machine learning methods.

The good reputation of the research work of the section is reflected by the memberships in ESPRIT Working Group 'NeuroCOLT' and in ESPRIT Network of Excellence 'Neuronet', and by several program committee memberships in well−known international conferences.

In the section of Information Systems the research is mostly algorithmic in character. The longest research project concentrates on data mining and machine learning, and it is a joint effort with the section of General Computer Science (see above). Other research projects include relational database design (Mannila), evaluation of Datalog queries (Sippu), text databases (Kilpeläinen, Mannila), computer−supported cooperative work (Erkiö), and knowledge bases (Tirri, Mannila) and knowledge representation (Grahne).

Also in these areas, the work has lead to several joint papers with foreign authors and program committee memberships; the database design work has been published also as a monograph by Addison−Wesley. Recent results include efficient data mining methods for database re−engineering, new efficiently evaluable query languages for text databases, and methods for automatically forming the grammar describing the structure of a text document. The results are published in high−quality journals and conferences.

In the area of programming languages belonging to the section of Computer Software, the theoretical work has focused on syntax analysis, attribute grammars and logic programming. A monograph on parsing theory is an example of significant outcome in this line of work. Several prototype implementations of various software tools have been developed, including programming environments for Prolog and Mode (an experimental object−oriented language) and translator writing systems HLP84, TOOLS, Metauncle, and PROFIT.

In the area of operating systems and performance evaluation, the research was started with a project on program behavior in hierarchical memories. One of the current research directions concentrates on queueing network modeling, centering on the development of new tools and techniques for modeling of real systems.

The main research areas in distributed systems are formal specification of systems and problems of open distributed processing. In the area of formal specifications computer communication protocols and other concurrent systems have been analyzed using various verification tools. Theories and methods relevant to computer aided verification have been studied. Also tools on this area have been implemented. The recurrent theme of research on specifications is to make analysis and verification feasible for real systems. Thus compositional specifications, verification of concurrent systems and minimization of system specifications for analysis are important.

During the last years the importance of distribution has grown immensely. The first project at the department was of constructive nature: we designed and partially implemented an experimental "midware" platform for distributed applications in a heterogeneous environment. The next project, DRYAD, considers an architecture of abstract services. We also have –– as a realization of the idea –– implemented a trading server. Along with these projects we have participated in the ISO standardization work on Open Distributed Processing. Recently a new research line has been opened with the focus on mobile computing. The area of interest is the new functionality needed for of provided by the wireless and mobile data communication, in this case based on NMT–GSM–telephone systems.

Active projects and research areas as well as individual research of some researchers and graduate students are presented in more detail in Section 2.2. Section 2.3 lists a selection of recent publications.

2.2. Projects

a) General Computer Science

Algorithms and data structures

Algorithms on strings is the main subproject in the area of algorithmics. This work on "stringology" started at the department quite early, in 1981. The initial impulse came from computer applications in molecular genetics where there is a lot of demand for efficient and sophisticated string–manipulation algorithms. The group is now one of the leaders in its special field in the world and has obtained several internationally respected basic results.

The reputation of the group is based on the work done in the 1980's on developing new algorithms for the following problems: edit distance, approximate string matching, DNA sequence assembly, and shortest common superstring. In 1990's, various 'sublinear' approximate string–matching algorithms and new measures for string similarity have been developed. For the important problem of how a text should be preprocessed to speed–up subsequent approximate string searches two solutions have been discovered. One is based on the so–called 'q–grams' and the other on applying dynamic programming over the suffix–tree of the text. For constructing suffix–trees (suffix–tree is the most important data structure used in string matching algorithms) a natural 'on–line' algorithm has been developed.

Most recently, the two–dimensional string matching problem was studied as a new branch of the activities of the group. This problem has received a lot of attention in the field. We were able to obtain the first optimal expected time solutions using simple, practical algorithms. This is in contrast with the many worst–case optimal solutions proposed in the literature which are very complicated and have small practical value. Another new research direction is the study of special databases for strings. We have obtained initial results on reasoning about strings in databases and on extending the relational database model to allow for typical queries on strings. Moreover, animating string matching algorithms has been studied as a student project which has lead to an animation software SALSA.

The on–going work deals with different feature–based schemes for approximate string matching, two and higher dimensional string matching and string databases.

Current members of the group are Prof. Esko Ukkonen (group leader) and Doc. Gösta Grahne (part–time), Doc. Jorma Tarhio (currently visiting University of California, Berkeley), Dr. Niklas Holsti (part–time), Juha Kärkkäinen, Matti Nykänen and Erkki Sutinen. The group gets support from the Academy of Finland.

Publications: [70, 71, 79–89].

Data Structures is another subproject in which we have studied problems arising from the use of data structures in a concurrent environment. Efficiency considerations require that the search paths of a data structure should be short and that operations should not block too large a part of the structure for too long a time. These two requirements conflict, since keeping the paths short after an update can, in a straightforward implementation, block too large a part of the structure. We would prefer that all operations reserve only a small, fixed−size part of a structure at all times.

We started in 1987 by proposing an elegant solution for these problems with dictionary operations. The solution was based on the notion of relaxed balance in AVL and B−trees and the decoupling of updates and rebalancing. In 1991 the idea of relaxing the balance condition was applied to red−black trees, the most promising data structures for implementing a data base in main memory. Our newest results on the area concern augmenting the structures with further operations such as batch updating, and tightening the balance conditions.

The research has been done in co−operation with Prof. D. Wood from University of London, Canada; J. Eckerle from Universität Freiburg, Germany; the group Prof. E. Soisalon−Soininen (leader), L. Malmi, E. Nuutila, and K. Pollari−Malmi from Helsinki University of Technology; and Dr. O. Nurmi from University of Helsinki. The research is partially supported by the Academy of Finland, Deutsche Forschungsgemeinschaft, Natural Sciences and Engineering Research Council of Canada, and Cultural Foundation of Finland.

Publications: [47, 49, 51, 52, 69, 78].

Machine learning

Being able to build computer systems that can learn in some sense is one of the very central problems in artificial intelligence. Inspired by certain theoretical advances in the field such as Valiant's PAC model and Rissanen's MDL principle, the project generally aims, unlike traditionally in AI, to apply to machine learning the approach of theoretical computer science and algorithmics. Our theoretical work is supported by experimenting with the algorithms.

Our results include generalizations and analyses of the RANK algorithm by Ehrenfeucht and Haussler, several theoretical results on the power of the so−called useful and reliable learnability (a variant of the PAC model), and introducing a new family of learning algorithms satisfying the Occam condition for learning decision lists and certain decision trees. The Occam algorithms are suitable for finding common patterns in sets of strings; we have already applied them to synthesize hyphenation rules for Finnish. A new 'geometric approach' to the feature selection in decision tree learning has been developed. In some cases, this method gives extremely accurate learning results. Two extensive tools for experimenting with learning algorithms have been produced: The TELA system for the study of learning algorithms that use traditional form of the data represented as attribute vectors, and the DALI system for the study of learning algorithms that use strings as the basic data.

The current research themes of the group are:

– Development of a decision tree learning algorithm that has a sound theoretical basis and performs well in practice;

– Development of new MDL learning algorithms for various practical learning tasks as well as development of the complexity theory for MDL learning;

– Further development of the Occam algorithms mentioned above and, in particular, their application to the prediction of the folding of protein molecules.

– Further development of the TELA and DALI systems.

The group has good international reputation and is together with nine other European groups a member of the ESPRIT Working Group 'NeuroCOLT'. The group also works in close co–operation with Prof. Heikki Mannila's data mining group.

The members of the group are Prof. Esko Ukkonen (group leader), Dr. Jyrki Kivinen (currently visiting the University of California at Santa Cruz), Tapio Elomaa, and Jaak Vilo. The group gets funding from the Academy of Finland (Project: 'Machine learning and combinatorial pattern matching – theory, algorithms and applications', 1994–96) and from the European ESPRIT Programme.

Publications: [151–161].

Neural Computing

The neural computing activities at the department were initiated by the project NEULOG, funded from 1988 to 1991 by the Finnish Ministry of Trade and Industry (TEKES). The project developed a prototype of a hybrid neural−symbolic programming environment, consisting of a probabilistic high−level knowledge representation language NEULA, and a compiler for translating NEULA declarations into stochastic neural networks capable of performing Bayesian reasoning with respect to the original high−level description. Much of this work is summarized in [7]. The system has later been extended by, e.g., a graphical user interface, and by embedding it as a probabilistic reasoning component in a standard Prolog system [147].

The NEULOG project has spawned two successors. Fundamental problems in the theory of neural computation have been studied since 1992 in project NETO, funded by the Academy of Finland. The results so far include, e.g., complexity analyses of various problems related to the Hamming and Hopfield neural network models [54, 57, 60], and a characterization of the computational power of the discrete Hopfield model [62]. A related development is the investigation into the foundations of genetic algorithms, initiated during Patrik Floréen's visit as a postdoc at Utrecht University.

The knowledge representation and reasoning issues raised in the NEULOG project led in 1992 to the TEKES−funded project REX, studying massively parallel implementations of case−based reasoning techniques. The concrete goal of the project is to implement a generic expert system shell (CAse−Based INferencE Tool, CABINET) for case−based reasoning. The CABINET shell will contain many different computational models for case matching and case adaptation. This allows the user to experiment with various case−based reasoning schemes. The current version of CABINET uses a theoretically justifiable Bayesian method for case matching and case adaptation [23], which can be implemented on a massively parallel feedforward neural network structure [162]. A learning scheme for such networks has also been developed [163].

Current members of the neural computing research group are Ph.D. Patrik Floréen, Mr. Kari Laasonen, Ph.Lic. Petri Myllymäki, Ph.D. Pekka Orponen (project manager for the NETO project), and M.Sc. Henry Tirri (project manager for the REX project). The group has also hosted 1−2 visiting foreign graduate students per year, and participates in two EC−funded networks of neural computing research (the NeuroCOLT working group, and the Network of Excellence in Neural Computing, NEURONET). Besides neural computing, members of the group have worked also on miscellaneous other topics; pointers to this work are given in the full list of publications.

Publications: [2, 7, 8, 23, 54, 57, 60−62, 143, 147, 162, 163].

Individual research

Scientific visualization (Assoc. Prof. M. Mäkelä):
Advanced computer graphics provides means of visualizing huge amounts of scientific data and complicated formal structures and interactions which have earlier been difficult to manage and conceive by human brains. Scientific visualization offers a method to make pictures from the abstract material produced by scientific models and measurements. Thus, the natural human ability to see and think visually are utilized besides the traditional scientific thinking. The most essential problem is to select a proper type of pictures for representing the abstract data and to formalize the transformation from data to pictures. The scope of this research is to find out practical recommendations to attack the problem. Under consideration have been e.g. particle systems, models for flock or herd behavior, similation of solid objects, visualization of large 3−D−nets, visualization of different types of medical data, and the concept and architecture of the visualization system. As the problem area is highly interdisciplinary there are several contacts with the Faculty of Medicine, Helsinki University of Technology, Center for Scientific Computing, and University of Industrial Arts (Helsinki), among others.

b) Computer Software

Open Distributed Processing

The rapid advances in data communication networks have greatly increased the possibilities to make use of the various computational and information services available within reach of a computer network. These services may have different origins, they are implemented on different types of computational platforms, and they can be located on sites far apart. As different services become reachable, interoperability of services must be considered. Distributed computation needs an enhanced functionality of the infrastructure. Furthermore, emerging technologies around wireless communication open up ways for new flexibility.

The department started research activity in the area of open distributed processing at the end of 1980's. In our first project (AHTO) we designed a distributed software environment ("midware") offering for a user a homogeneous interface to the computing services available in a heterogeneous computer network. Since that time the department has participated in the development of the Reference Model for Open Distributed Processing, under the auspices of ISO and ITU−T (former CCITT).

For the time being, there are two projects active in this area: DRYAD, started in 1992, and MOWGLI, started in 1993.

The DRYAD project studies the distributed infrastructure needed for interoperability of services. In this project we have developed an architecture that tolerates heterogeneity. The key concept of the distributed architecture is an abstract service. Clients specify implicitly or explicitly the service behaviour they demand. The infrastructure tries to fulfill these service demands with the concrete tools i.e. concrete services in the network. It activates a service using three steps: 1) it modifies a concrete service request to abstract service description (type management), 2) selects service providers (trading) and 3) invokes the service providers to accomplish the task and to support the needed data communication between the client and the service providers (type and object management). To support this architecture, we have implemented a prototype trading service. Trader performs the task of finding the most suitable service provider available, based on the dynamic state of clients, service providers and network, and the static knowledge about the clients view to abstract services.

The members of the DRYAD project group are prof. Martti Tienari, Dr. Timo Alanko, Lea Kutvonen, Petri Kutvonen and Liisa Marttinen. The work is partially funded by the Technology Development Centre TEKES, ICL Personal Systems, Telecom Finland, Helsinki Telephone Company and Finnish State Computing Centre.

The goal of the MOWGLI project is to study, design and test a data communication architecture for a pan−European GSM−based mobile data service and to develop a prototype based on that architecture. The environment of an application consists of mobile PC's, which can be connected, through a mobile phone, to any part of a fixed data communication network.

The work in the project will concentrate on the architectural aspects which support the mobility of the client, allow parts of the application to operate in a disconnected mode and hide the problems of the wireless connection.

The members of the MOWGLI project group are prof. Martti Tienari, Dr. Timo Alanko, Heimo Laamanen and Markku Kojo. The industrial partners and financial supporters of the project are Digital Equipment Corporation, Nokia Mobile Phones and Telecom Finland. The project is also partially financed by the Ministry of Education.

Publications: [13, 20–22, 40, 41].

Modelling of concurrency

Distributed systems and parallel programs are notoriously prone to errors caused by subtle differences in the relative execution orderings of the components of the system. In order to avoid such errors systems should be carefully specified and analyzed. The international research in this area, often called simply "concurrency", has been active in the 1980's and has resulted in numerous specification methods and models for this purpose, e.g. state transition systems, Petri nets, process algebras and temporal logics. Elegant and rich theories supporting these models have made the models even more useful. Results in these theories can be employed in computer based analysis tools for checking and verifying concurrent real life software.

An important class of distributed algorithms consists of computer communication protocols.We have worked with the problems of protocol analysis since 1984. We begun by constructing a reachability analysis tool PROTAN88 [19] designed for protocols specified with an extended state transition model (ESTELLE specification language). In the 1980's this tool has been tried out by us in analyzing several well–known real–life protocols like X.25, FTAM and X.411/P1. Some of our improvement suggestions were included in the FTAM state tables published by CCITT and ISO.

Lately our work has been more conceptual and theoretical. Various equivalence concepts have been studied in process algebra contexts (Milners CCS and ISO LOTOS) and the equivalence preserving minimization of labeled state transition systems has been studied [90–92, 97, 98]. We have been especially interested in divergence preserving behavioural equivalences and preorders. These equivalences allow neat comparisons between bisimulation and failures based equivalences and congruences. If an equivalence does not preserve divergences, it is difficult to analyze liveness properties after equivalence preserving transformations of a transition system. Many aspects of temporal logic are closely related with equivalences and preorders [93, 94].

A recurrent theme in our research is to make analysis and verification feasible for concurrent systems of realistic size. That is why the dominant goal in our research is to understand and support compositional (or modular) specification and verification of concurrent systems. We have been e.g. analyzing and applying the weakest equivalence (CFFD−equivalence) preserving deadlocks and linear next−timeless linear temporal formulae. Our research employs case analyses (often computer communication protocols) to ensure the practical usefulness of the theoretical concepts.

At the university of Helsinki we have been developing a computer−based verification tool BIDMIN allowing us to minimize labeled transition systems with respect to divergence preserving bisimilarity and branching bisimilarity as well as the related congruences. We also often use a LOTOS oriented verification tool ARA TOOLS developed at the Technical Research Centre of Finland. ARA TOOLS is a prototype aiming at industrial quality.

Concurrency research team is lead by professor Martti Tienari. The team comprises a postdoctoral research fellow Dr. Jaana Eloranta and three graduate research students Ph.Lic Roope Kaivola (Currently at the University of Edinburgh), Ph.Lic. Timo Karvi and M.Sc. Päivi Kuuppelomäki. As an associate member of our team we have professor Antti Valmari (Tampere University of Technology, Software Systems Laboratory) who has an external docent appointment in our department. Professor Valmari is the chief designer of ARA TOOLS. Our concurrency group is participating (1994−96) in European COST247−project "Verification and Validation Methods for Formal Descriptions".

Publications: [18, 19, 90−98].

Performance analysis

The general goal of the research group is to develop tools and techniques for modelling, measurement and analysis of computing systems, especially using empirically oriented methods. The main research areas have been simulation methods and interactive solution environments of queueing networks. New areas of concern will be the performance of data communication systems and distributed systems; especially of interest are the techniques used in telecommunications and the effects of mobility of workstations in a network.

Researchers: Dr. Kimmo Raatikainen, Dr. Teemu Kerola, Dr. Timo Alanko, Dr. Inkeri Verkamo.

Publications: [42−46].

Individual research

Language design and implementation for multiple paradigms (Ph.Lic. Juha Vihavainen):
Multiparadigm programming integrates different programming perspectives in order to simultaneously utilize concepts and mechanisms related to alternative programming paradigms. The different programming paradigms (the main ones being procedural, object−oriented, functional, and logic) are considered to be complementary rather than conflicting computational models. The constructive goals are to develop a general uniform implementation model for multiple paradigms and to produce software (a class library) to implement this model. The implementation approach is based on meta−programming methodology in which a reflective object−oriented architecture is extended by redefining and specializing meta objects. This software addresses traditional translation issues (such as scanning, parsing, semantic analysis, and code generation) but also special problems related to class hierarchies (inheritance), dynamic and static polymorphism (generics), and higher−order values. The implementation is done in C++.

Publications: [24].

Correctable computing (Dr. Niklas Holsti):
A correctable computation is a computation that reacts to changed inputs by updating its outputs to match. As an example, we have developed user interface tools that work in a correctable way: an interactive transcript editor lets the user edit any input, and cooperates with the application program to undo and redo the computation and update the output. Reversible parsing and interpretation methods were developed and a number of applications implemented. Future plans include the design of protocols for propagating corrections between programs.

Publications: [139, 140].

c) Information Systems

Design−By−Example: A tool for database design (DBE)

Relational database design has been studied thoroughly. Unfortunately, use of the design theory in practical database design requires a good knowledge of the theory. Besides being hard to learn, another problem of the design methods is their batch oriented nature. The design data (typically integrity constraints) is given to a design algorithm that produces a schema with the required properties. If the resulting schema is not satisfactory (a sign of omissions or errors in the design data), the process must be repeated in its entirety. The standard design theory does not lend itself easily for incremental database design.

This research project contains strongly interacting theoretical and systems work. The project is constructing a database design tool called Design−By−Example (DBE). The tool will make it easy to design good schemas; design theory will be hidden from the users of the tool. Theoretical research on database design has been carried out at the Universities of Helsinki and Tampere during the last five years; the work is still continuing. Many of the theoretical research problems have been found from the practical work. The following properties set DBE apart from existing tools.

– DBE gives extensive feedback to the designer. The system supports the checking of the completeness of the design data by the use of examples. Usually it is easy for the designer to check the correctness of given design data. It is much more difficult to make sure that no relevant design data has been overlooked. DBE generates automatically an example database on the basis of the design data. All situations not prohibited by the integrity constraints are exhibited in this database.

– DBE is incremental. The schema can be designed little by little; small changes in the design data cause minimal changes in the produced database schema.

– DBE is interactive. It is often possible to continue the design using several routes, all of which are equally good in a formal sense. DBE does not choose its actions arbitrarily, but asks for the designer's assistance.

– DBE works on two levels: it supports design both on the conceptual level (ER schema) and the logical level (relational schema). A full correspondence between the levels is automatically maintained at all times. In particular, changes done on the relational level are automatically reflected back to the ER level.

A prototype implementation of the relational level was completed in

1988. Currently the full system is being implemented for the Macintosh in C in the University of Tampere. It will support the design of SQL–schemas, and provides a data dictionary in the form of an SQL–database. Besides the Macintosh, the system will be ported into the OS/2 environment. The implementation work provides still new problems for more theoretical work.

The principal researchers of the project are Prof. Heikki Mannila and Prof. Kari–Jouko Räihä (University of Tampere). The theoretical work has been funded by the Academy of Finland. The implementation work is supported by the Technology Development Centre (TEKES). The industrial partners of the project include Nokia Data Systems, Oracle Finland, and United Paper Mills.

Publications: [108, 110, 112–115].

Structured text databases (RATI)

A structured text database system is a tool for managing text documents which have some kind of internal structure and in which dependencies between different parts of the document occur. Examples of such documents are manuals for software and equipment, encyclopedias, and dictionaries. Structured text databases attempt to combine the advantages of text processing systems and database systems.

The RATI project studies methods for manipulation of structured documents. The project has devised a data model and a query language for such documents. The data model is based on the use of context–free grammars for defining the structures of documents.

The project has implemented a prototype database system, where this data model and query language are in use. The user manipulates documents in the familiar way with a text editor. The user interface supports multiple representations. For each document type the user or document designer can define several appearances. The document can be edited in any representation. The definition of a representation, called a view, is defined by annotating the grammar defining the structure of the document. A possible representation is, e.g., a SGML document.

The prototype is implemented in C using the X Window System. It uses either a relational database system or the Unix file system for the storage of the documents.

The prototype is being extended by hypertext features and by support for very large volumes of static text. Research topics related to static texts are computer–aided tools for the extraction of grammars out of existing texts, and incrementalization of queries and updates of structured text.

Other ongoing research includes developing high–level query languages for structured text, increasing the power of the view definition mechanism, and studying the relationships and applicability of object–oriented data models to manipulation of documents.

The principal researcher of the project is Prof. Heikki Mannila. M.Sc. Helena Ahonen, Dr. Pekka Kilpeläinen, Ph.Lic. Greger Lindén, and Ph.Lic. Erja Nikunen (Research Center of Domestic Languages) are working in the project. The project has been funded by TEKES and by the Academy of Finland.

Publications: [72–74, 111, 117, 123, 150, 164].

Data mining

Data mining (or knowledge discovery in databases) aims at semiautomatic tools for extracting interesting information from knowledge bases. Typical application areas are analysis of customers databases or error logs.

We have developed applied data mining methods in several application areas. The research on data mining began in connection with the DBE project by developing tools for finding integrity constraints that hold in a database instance. The group has also given a complete analysis of the complexity of this problem. We have also considered finding document structures from marked documents in connection with the RATI project.

The group has also studied the power of sampling in knowledge discovery and given the first general bounds on the sample sizes needed for reliable identification of useful rules from samples. Starting from late 1993, the focus of the group is in the discovery and analysis of association rules. We are looking for efficient methods for finding interconnected elements from ordinary and time–dependent data.

Data mining has strong connections to machine learning, and the project works in close cooperation with Prof. Ukkonen's machine learning group. The group has good international connections.

The members of the group are Prof. Heikki Mannila (group leader), Prof. Kari–Jouko Räihä (University of Tampere), M.Sc. Helena Ahonen, M.Sc. Pirjo Ronkainen, M.Sc. Hannu Toivonen, and Dr. Inkeri Verkamo.

Publications: [112, 114, 136–138, 150].

Software engineering for knowledge−based systems (VITAL)

There is a growing need to make building knowledge−based systems (KBS) more robust, structured and rigorous if the technology is to fulfill its promise. VITAL is a ESPRIT II (P5365) research and development project with nine participating organizations in five European countries. The project aims at building an industrial−strength software engineering methodology together with a software workbench (the VITAL Tower) for building large−scale KBS. VITAL approach is based on KADS methodology, which advocates handling of the complexity of KBS development by using multiple models. Unlike regular software engineering projects, KBS projects emphasize explicit knowledge representation and manipulation during application execution.

VITAL has centered its main development around the notion of process products. Besides these process products, the VITAL methodology has three components:

– the knowledge engineering methodology (how to handle production of process products for a particular application),

– life−cycle model (how to control and monitor the entire project),

– life−cycle configuration (how each process product is developed within the context of the overall project).

The members of the VITAL group at our department have focused on the knowledge transformation issues. In this context a general purpose transformation toolkit ALCHEMIST has been developed. This toolkit allows rapid development of transformation software by providing graphical tools for specifying the transformation and then providing code generation facilities (C++). This software is being used to build transformations needed within the VITAL workbench as well as to external software. The tool is not restricted to VITAL context, it can be used to build any well−defined transformations between persistent representations. In addition to the transformation work, the project group has also participated into the software engineering life−cycle model development.

The principal investigators of the group are Prof. Heikki Mannila (group leader), Ph.Lic. Greger Lindén, M.Sc. Henry Tirri and Dr. Inkeri Verkamo. The Finnish participation into the project is funded by TEKES.

Publications: [39].

Computer Supported Cooperative Work (CSCW)

Work in this area is still in its early stages. During recent years we have participated in two COST 14 working groups. One group studied how group knowledge building could be supported by various forms of information technology [141]. Knowledge is here interpreted widely, meaning practically any representations that aid a cooperative group to perform better than the individuals alone. The other group has been developing an architectural framework supporting CSCW systems.

The other activities in the CSCW area include various student projects where experiments in implementing semantically advanced communication based on e−mail have been carried out. A newer point of interest is work flow management where some experiments have been performed using Lotus Notes. We have planned to continue with studying also conceptually how organizational work flow computing should be supported. One part of this work is concerned with the transaction mechanisms needed.

Researchers: Dr. Hannu Erkiö, M.Sc. Henry Tirri.

Publications: [141].

Transaction management

Concepts such as transaction and a serializable history were introduced when only centralized database systems existed. As distributed systems developed, these concepts were extended to distributed systems. However, an important difference between a distributed system and a centralized system are the relatively large communication delays between the sites and the autonomy of local systems. As a result, the traditional concepts of the transaction and the serializable history, though well suited for traditional centralized systems, may not be suitable in distributed systems. Moreover, as the database concepts have been applied to new areas such as computer−aided design, office information systems and cooperative work, refined and generalized transaction models are needed.

In this research area at our department new transaction models together with protocols for transaction management in heterogeneous distributed database systems have been developed. One approach exploits the information of distributed consistency constraints. This approach has resulted in the introduction of the notions of "partial serializability" and "combined transactions". The transaction research at our department is partly performed in cooperation with INRIA (France) and Purdue University in the United States. This research has resulted in the notions of "value dates" and "fragmented composite objects". The principal researchers in the transaction field are Ph.Lic. Juha Puustjärvi, M.Sc. Henry Tirri and Dr. Heikki Tuuri.

Beginning of the year 1994 our department anticipates involvement in a related ESPRIT III Basic Research Action (TRANSCOOP). The other participants of this project are the Technical Research Centre of Finland (VTT), GMD IPSI (Germany) and the University of Twente (Netherlands). The goal of this project is to develope a transaction model suitable for cooperative work together with mechanisms for controlling cooperative transactions.

Publications: [127, 128, 133−135].

Logic databases and database algorithms

The general goal of this research is to develop efficient data structures and algorithms for database and knowledge base systems. The main concern is the design and analysis of recursive query processing methods for logic databases. The problem of recursion is inherent in knowledge base systems, and recursively defined queries are needed by many new database applications, e.g. for querying complex database objects that are composed of subobjects to an arbitrary number of levels.

New algorithms have been designed for the evaluation and optimization of recursive Datalog queries. A prototype for an experimental logic database system has been designed and implemented. The system is running in the Sun/Unix environment and it has been used e.g. for the analysis of different iteration strategies for bottom−up query evaluation and for testing main−memory database structures.

The members of the research group are Assoc.Prof. Seppo Sippu, Prof. Eljas Soisalon−Soininen (Helsinki Univ. of Technology), Ass.Prof. Otto Nurmi, and M.Sc. Juhani Kuittinen.

Publications: [49, 122, 124, 129−132].

Knowledge representation and reasoning (doc. Gösta Grahne)

As soon as data becomes more complex than a collection of records, there is a need to address knowledge representation and reasoning issues. These issues require solving semantic, algorithmic, and complexity theoretic subproblems. The research has focused on problems stemming from incomplete information in relational databases [107, 119, 120], on recursion [122], on the relationship between updates to logical theories and counterfactual reasoning [116, 144], and between updates and belief revision [145], on database updates and generalized datalog [146]. Current research involves the reasoning about strings (such as DNA sequences) in databases [121].

Publications: [107, 116, 119, 120−122, 144−146].

2.3. Publications

Below is a selected list of the most important publications of the Department since 1988. Publications are arranged according to the ACM Computing Reviews (CR) classification system.

(Ph.D. and Ph.Lic. theses are also listed separately in Ch. 3.3.)

Introductory and survey (A.1)

1. H. Lokki, I. Haikala, S. Linnainmaa, K. Rönkä, and O. Susiluoto: *Data processing.* 3rd ed. Tietotekniikan liitto 1992. 366 pp. /In Finnish/.

Design styles (B.3.2)

2. P. Floréen: A short introduction to neural associative memories. *Bulletin of the European Association for Theoretical Computer Science*, no. 51(1993): 236–245.

Processor architectures – other architecture styles – neural nets (C.1.3)

3. P. Floréen, P. Myllymäki, P. Orponen, and H. Tirri: Compiling object declarations into connectionist networks. *AI Communications* 3(1990):4, 172−183.

4. P. Floréen, P. Myllymäki, P. Orponen, and H. Tirri: Neural representation of concepts for robust inference. In: *Computational Intelligence, II. Proc. International Symposium*, Milan, Italy, Sept. 25−27, 1989. Ed. by F. Gardin et al. North−Holland, 1990, pp. 89−98.

5. R. Morris, L. Rubin, and H. Tirri: A comparison of feedforward and self−organizing approaches to the font orientation problem. In: *Proc. International Joint Conference on Neural Networks, IJCNN '89*, Washington, DC, 1989. Vol. 2, pp. 291−298.

6. R. Morris, L. Rubin, and H. Tirri: Neural network techniques for object orientation detection: solution by optimal feedforward network and learning vector quantization approaches. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 12(1990):11, 1107−1115.

7. P. Myllymäki: Bayesian Reasoning by Stochastic Neural Networks. Ph.Lic. Thesis, Report C−1993−67, Department of Computer Science, University of Helsinki, 1993. 78 pp.

8. P. Myllymäki: Using Bayesian Networks for incorporating probabilistic a priori knowledge into Boltzmann machines. To appear in *Proceedings of SOUTHCON'94* (Orlando, March 1994).

9. P. Myllymäki, H. Tirri, P. Floréen, and P. Orponen: Compiling high−level specifications into neural networks. In: *International Joint Conference on Neural Networks*, Washington, D.C., Jan. 15−19, 1990. Vol. 2, pp. 475−478.

10. P. Myllymäki and P. Orponen: Programming the Harmonium. In: *International Joint Conference on Neural Networks, IJCNN−91*, Singapore, Nov. 18−21, 1991. Vol. 1, pp. 671−677.

11. P. Orponen, P. Floréen, P. Myllymäki, and H. Tirri: A neural implementation of conceptual hierachies with Bayesian reasoning. In: *International Joint Conference on Neural Networks*, San Diego, CA, June 17−21, 1990. Vol. 1, pp. 297−303.

12. H. Tirri: Implementing expert system rule conditions by neural networks. *New Generation Computing* 10(1991):1, 55−71.

Computer−communication networks (C.2)

13. M. Tienari and D. Khakhar (eds.): *Information Network and Data Communication. IV. Proc. IFIP TC6 International Conference*, Espoo, Finland, 16−19 March, 1992. North−Holland, 1992. 482 pp. (IFIP Transactions C: Communication Systems vol. C−6).

Network architecture and design (C.2.1)

14. K.E.E. Raatikainen: A framework for evaluating the performance of IN services. In: *Proc. International Workshop on Intelligent Networks*, Lappeenranta, 1993, pp. 121−129.

Network protocols (C.2.2)

15. S. Kesti and J. Paakki: Revised ASN.1: type compatibility, assignment compatibility, subtype compatibility, new syntactic notation. Nokia Research Center, 1991. 31 pp. (TEKES/COMPET report no. 11).

16. K. Lehtinen: Experiences in using Estelle in X.411/P1 protocol specification. Report A−1988−10, Department of Computer Science, University of Helsinki. 37+67 pp.

17. J. Paakki, K. Granö, A. Ahtiainen and S. Kesti: An implementation of ASN.1 (Abstract Syntax Notation One). In: *Proc. 3rd Symposium on Programming Languages and Software Tools*, Kääriku, Estonia, August 23−24, 1993. Ed. by M. Tombak. University of Tartu, Department of Computer Science, 1993, pp. 95−108.

18. M. Tienari (with 14 other members of COST 11 ter group on FDT/ABM): A framework for the taxonomy of synthesis and analysis activities in distributed system design. In: *Research into Networks and Distributed Applications. European Teleinformatics Conference − EUTECO '88*, Vienna, Austria, April 20−22, 1988. North−Holland, 1988, pp. 859−871.

19. M. Tienari et al.: PROTAN88 − a software tool for verifying communication protocols specified with an extended state transition model.

Report A−1988−5, Department of Computer Science, University of Helsinki. 65 pp.

Distributed systems (C.2.4)

20. T. Alanko, J. Keskinen, P. Kutvonen, M. Mutka, and M. Tienari: The AHTO project − software technology for open distributed processing. Report A−1989−4, Department of Computer Science, University of Helsinki. 53+3 pp.

21. M. Hiltunen: A study of distributed applications and the AHTO system. Report C−1990−31, Department of Computer Science, University of Helsinki. 32 pp.

22. L. Kutvonen and P. Kutvonen: Broadening the user environment with implicit trading. In: *Proceedings of the Second IFIP TC6/WG6.1 International Conference on Open Distributed Processing*, Berlin, Germany, 1993, pp. 157−168.

Parallel programming (D.1.3)

23. P. Myllymäki and H. Tirri: Massively parallel case−based reasoning with probabilistic similarity metrics. In: *Proc. EWCBR−93, First European Workshop on Case−Based Reasoning*, Univ. Kaiserslautern, Nov. 1−5, 1993. Ed. by M.M. Richter et al. University of Kaiserslautern, 1993. (SEKI Report SR−93−12) (SFB 314). Vol. 1, pp. 48−53.

Object−oriented programming (D.1.5)

24. K. Koskimies and J. Vihavainen: The problem of unexpected subclasses. *Journal of Object−Oriented Programming* 5(1992):6, 53−59.

Software engineering − requirements/specifications (D.2.1)

25. J. Paakki: Design engineering case study with COLD − partial results.

Nokia Research Center, 1990. 18 pp. (ESPRIT/ATMOSPHERE report R.2.1.2.2.2.1.1:1).

Software engineering – testing and debugging (D.2.5)

26. P. Kilpeläinen and H. Mannila: Generation of test cases for simple Prolog programs. *Acta Cybernetica* 9(1990):3, 235–246.

Software engineering – miscellaneous (D.2.m)

27. J. Paakki: A note on the speed of Prolog. *SIGPLAN Notices* 23(1988):8, 73–82.

Programming languages – language classifications (D.3.2)

28. J. Paakki: PROFIT: a system integrating logic programming and attribute grammars. In: *Programming Language Implementation and Logic Programming. 3rd Int. Symp. PLILP'91*, Passau, Germany, August 26–28, 1991. Ed. by J. Maluszynski et al. Springer–Verlag, Berlin, 1991, pp. 243–254. (Lecture Notes in Computer Science 528).

Language constructs and features (D.3.3)

29. J. Paakki, A. Karhinen, and T. Silander: Orthogonal type extensions and reductions. *SIGPLAN Notices* 25(1990):7, 28–38.

Processors – compilers (D.3.4)

30. P. Kilpeläinen, H. Mannila, J. Riivari, and E. Ukkonen: Prolog in Ada: an implementation and an embedding. In: *Proc. AIDA'89, 5th Annual Conference on Artificial Intelligence & Ada*. Ed. by J. Diaz–Herrera et al. George Mason University, USA, 1989, pp. 96–107.

31. K. Koskimies and J. Paakki: *Automating language implementation: a pragmatic approach*. Ellis Horwood, 1990. 253 pp.

32. K. Koskimies, O. Nurmi, J. Paakki, and S. Sippu: The design of a language processor generator. *Software − Practice & Experience* 18(1988):2, 107−135.

33. K. Koskimies and J. Paakki: High−level tools for language implementation. *Journal of Systems and Software* 15(1991):2, 115−131.

34. J. Paakki and K. Toppola: An error−recovering form of DCGs. *Acta Cybernetica* 9(1990):3, 211−221.

35. J. Paakki: A practical implementation of DCGs. In: *Proc. of the 3rd Workshop on Compiler Compilers, Institute of Informatics and Computing Technique, Academy of Sciences of the GDR*, Berlin, pp. 249−257. (Abstract in: Compiler Compilers. 3rd Int. Workshop, CC'90, Schwerin, FRG, October 1990. Ed. by D. Hammer. Springer−Verlag, Berlin, 1990. Lecture Notes in Computer Science 477, pp. 224−225).

36. J. Paakki: Prolog in practical compiler writing. *Computer Journal* 34(1991):1, 64−72.

37. J. Tarhio: A compiler generator for attribute evaluation during LR parsing. In: *Compiler Compilers and High Speed Compilation. 2nd CCHSC Workshop*, Berlin, GDR, Oct. 10−14, 1988. Ed. by D. Hammer. Springer−Verlag, Berlin, 1989, pp. 146−159. (Lecture Notes in Computer Science 371).

38. M. Tienari: Compiler compilers. In: *Concise Encyclopedia of Software Engineering*. Ed. by D. Morris et al. Pergamon Press, Oxford, 1992, pp. 57−59.

39. H. Tirri and G. Lindén: ALCHEMIST − an object−oriented tool to build transformations between heterogenous data representations. In: *27th Annual Hawaii International Conference on System Sciences (HICSS '94).* Vol. 2, pp. 226−235.

File systems management (D.4.3)

40. T. Alanko and P. Kutvonen: The AHTO directory: a distributed directory designed for distributed usage. *IEEE Distributed Processing Technical Committee Newsletter* 14(1992):1, 13−20.

Reliability (D.4.5)

41. M. Hiltunen and R.D. Schlichting: An approach to constructing modular fault−tolerant protocols. Report TR−93−10, Department of Computer Science, University of Arizona. 17 pp.

Performance (D.4.8)

42. K.E.E. Raatikainen: Approximating response time distributions. *Performance Evaluation Review* 17(1989):1, 190−199. (1989 ACM SIGMETRICS and PERFORMANCE '89 International Conference on Measurement and Modeling of Computer Systems, May 23−26, 1989, Berkeley, CA, USA).

43. K.E.E. Raatikainen: Cluster analysis and workload classification. *Performance Evaluation Review* 20(1993):4, 24−30.

44. K.E.E. Raatikainen: Experiences of hierachical workload modeling in capacity planning. In: *Proc. CMG '89, International Conference on Management and Performance Evaluation of Computer Systems*, Reno, Nevada, Dec. 11−15, 1989, pp. 284−293.

45. A.I. Verkamo: External quicksort. *Performance Evaluation* 8(1988):4, 271−288.

46. A.I. Verkamo: Performance comparison of distributive and mergesort as external sorting algorithms. *Journal of Systems and Software* 10(1989):3, 187−200.

Data structures (E.1)

47. O. Nurmi, E. Soisalon−Soininen, and D. Wood: Relaxed AVL trees, main−memory databases, and concurrency. Technical Report no. 351, Department of Computer Science, University of Western Ontario, 1993. 22 pp.

Data storage representations (E.2)

48. N. Holsti: Using formal procedure parameters to represent and transmit complex data structures. *SIGPLAN Notices* 23(1988):3, 83−92.

49. O. Nurmi and E. Soisalon−Soininen: Uncoupling updating and rebalancing in chromatic binary search trees. In: *Proc. 10th ACM SIGACT−SIGMOD−SIGART Symposium on Principles of Database Systems*, Denver, CO, May 29−31, 1991, pp. 192−198.

Coding and information theory (E.4)

50. H. Peltola and J. Tarhio: On syntactical data compression. In: *Proc. 2nd Symposium on Programming Languages and Software Tools*, Pirkkala, Finland, Aug. 21−23, 1991. Ed. by K. Koskimies et al. University of Tampere, pp. 205−214. (Report A−1991−5, Department of Computer Science, University of Tampere).

Files (E.5)

51. R. Klein, O. Nurmi, Th. Ottmann, D. Wood: A dynamic fixed windowing problem. *Algorithmica* 4(1989):4, 535−550.

Theory of computation − general (F.0)

52. O. Nurmi and E. Ukkonen (eds.): *Algorithm Theory − SWAT '92, 3rd Scandinavian Workshop on Algorithm Theory*, Helsinki, Finland, July 8−10, 1992. Springer−Verlag, Berlin, 1993. 433 pp. (Lecture Notes in Computer Science 621).

Models of computation (F.1.1)

53. P. Floréen: An analysis of the convergence time of Hamming memory networks. In: *IJCNN, International Joint Conference on Neural Networks*, San Diego, CA, June 17−21, 1990. Vol. 1, pp. 867−872.

54. P. Floréen and P. Orponen: Attraction radii in binary Hopfield nets are hard to compute. *Neural Computation* 5(1993):5, 812−821.

55. P. Floréen: The convergence of Hamming memory networks. *IEEE Transactions on Neural Networks* 2(1991):4, 449−457.

56. P. Floréen and P. Orponen: Counting stable states and sizes of attraction domains in Hopfield nets is hard. In: *International Joint Conference on Neural Networks*, Washington, D.C., June 18−22, 1989. Vol. 1, pp. 395−399.

57. P. Floréen: A new neural associative memory model. *International Journal of Intelligent Systems* 7(1992):5, 455−467.

58. P. Floréen and P. Orponen: On the computational complexity of analyzing Hopfield nets. *Complex Systems* 3(1989):6, 577−587.

59. P. Floréen: Worst−case convergence times for Hopfield memories. *IEEE Transactions on Neural Networks* 2(1991):5, 533−535.

60. P. Floréen: Computational complexity problems in neural associative memories. Ph.D. Thesis, Report A−1992−5, Department of Computer Science, University of Helsinki, 1992, 128+8 pp.

61. P. Orponen: Neural networks and complexity theory. In: *Mathematical Foundations of Computer Science 1992. 17th International Symposium*, Prague, Czechoslavia, August 1992. Ed. by I.M. Havel et al. Springer−Verlag, Berlin, 1992, pp. 50−61. (Lecture Notes in Computer Science 629).

62. P. Orponen: On the computational power of discrete Hopfield nets. In: *20th Int. Colloq. on Automata, Languages, and Programming*, Lund, Sweden, July 1993. Ed. by A. Lingas et al. Springer−Verlag, Berlin, 1993, pp. 215−226. (Lecture Notes in Computer Science 700).

Complexity classes (F.1.3)

63. R. Book, P. Orponen, D. Russo, and O. Watanabe: Lowness properties of sets in the exponential−time hierarchy. *SIAM Journal on Computing* 17(1988):3, 504−516.

64. K. Ko, P. Orponen, U. Schöning, and O. Watanabe: Instance complexity. Report A−1990−6, Department of Computer Science, University of Helsinki. 24 pp. (To appear in: *Journal of the Association for Computing Machinery*).

65. P. Orponen: On the instance complexity of NP−hard problems. In: *Proc. 5th Annual Structure in Complexity Theory Conference*, July 8−11, 1990, Barcelona, pp. 20−27.

Nonnumerical algorithms and problems (F.2.2)

66. Th. Eiter, G. Gottlob, H. Mannila: Adding disjunction to Datalog. *Proc. of the 13th ACM SIGACT−SIGMOD Symposium on Principles of Database Systems*. Minneapolis, Mn. 1994, to appear.

67. Th. Eiter, P. Kilpeläinen and H. Mannila: Recognizing renamable generalized propositional Horn formulas is NP−complete. Report A−1992−3, Department of Computer Science, University of Helsinki. 11 pp.

68. V. Estivill−Castro, H. Mannila and D. Wood: Right invariant metrics and measures of presortedness. *Discrete Applied Mathematics* 42(1993):1, 1−16.

69. R.−H. Güting, O. Nurmi, and Th. Ottmann: Fast algorithms for direct enclosures and direct dominances. *Journal of Algorithms* 10(1989):2, 170−186.

70. P. Jokinen, J. Tarhio and E. Ukkonen: A comparison of approximate string matching algorithms. Report A−1991−7, Department of Computer Science, University of Helsinki. 23 pp.

71. P. Jokinen and E. Ukkonen: Two algorithms for approximate string matching in static texts. In: *Mathematical Foundations of Computer Science 1991, 16th International Symposium*, Kazimierz Dolny, Poland, Sept. 9−13, 1991. Ed. by A. Tarlecki. Springer−Verlag, Berlin, 1991, pp. 240−248. (Lecture Notes in Computer Science 520).

72. P. Kilpeläinen and H. Mannila: Grammatical tree matching. In: *Combinatorial Pattern Matching, 3rd Annual Symposium*, Tucson, Arizona, USA, April 29 − May 1, 1992. Ed. by A. Apostolico et al. Springer−Verlag, Berlin, 1992, pp. 162−174. (Lecture Notes in Computer Science 644).

73. P. Kilpeläinen and H. Mannila: Ordered and unordered tree inclusion. Report A−1991−4, Department of Computer Science, University of Helsinki. 22 pp. To appear in *SIAM Journal on Computing*.

74. P. Kilpeläinen and H. Mannila: The tree inclusion problem. In: *TAPSOFT'91. Proc. Int. Joint Conf. on Theory and Practice of Software Development,* Brighton, UK, April 8−12, 1991. Vol. 1: Colloquium on Trees in Algebra and Programming (CAAP'91). Ed. by S. Abramsky et al. Springer−Verlag, Berlin, 1991, pp. 202−214. (Lecture Notes in Computer Science 493).

75. H. Mannila and E. Ukkonen: Time parameter and arbitrary deunions in the set union problem. In: *SWAT 88, 1st Scandinavian Workshop on Algorithm Theory*, Halmstad, 1988. Ed. by R. Karlsson et al. Springer−Verlag, Berlin, 1988, pp. 34−42. (Lecture Notes in Computer Science 318).

76. H. Mannila and E. Ukkonen: Unifications, deunifications, and their complexity. *BIT* 30(1990):4, 599−619.

77. H. Mannila and D. Wood: A note on the largest empty rectangle problem. *BIT* 28(1988):1, 179−183.

78. O. Nurmi and J.−R. Sack: Separating a polyhedron by one translation from a set of obstacles. In: *Graph−Theoretic Concepts in Computer Science. International Workshop WG '88*, Amsterdam, The Netherlands, June 15−17, 1988. Ed. by J. van Leeuwen. Springer−Verlag, Berlin, 1989, pp. 202−212. (Lecture Notes in Computer Science 344).

79. J. Tarhio and E. Ukkonen: Approximate Boyer−Moore string matching. *SIAM Journal on Computing* 22(1993):2, 243−260.

80. J. Tarhio: A Boyer−Moore approach for two−dimensional matching. Report UCB/UCD 93/784, Computer Science Division, University of

California, Berkeley, 1993.

81. J. Tarhio and E. Ukkonen: Boyer−Moore approach to approximate string matching. In: *SWAT 90, 2nd Scandinavian Workshop on Algorithm Theory*, Bergen, July 11−14, 1990. Ed. by J.R. Gilbert et al. Springer−Verlag, Berlin, 1990, pp. 348−359. (Lecture Notes in Computer Science 447).

82. J. Tarhio and E. Ukkonen: A greedy approximation algorithm for constructing shortest common superstrings. *Theoretical Computer Science* 57(1988):1, 131−145.

83. E. Ukkonen: A linear−time algorithm for finding approximate shortest common superstrings. *Algorithmica* 5(1990):3, 313−323.

84. E. Ukkonen: Approximate string−matching with q−grams and maximal matches. *Theoretical Computer Science* 92(1992):1, 191−211.

85. E. Ukkonen: Constructing suffix trees on−line in linear time. In: *Information Processing 92. Proc. IFIP 12th World Computer Congress,* Madrid, Spain, 7−11 Sept. 1992. Vol. 1: Algorithms, software, architecture. Ed. by J. van Leeuwen. North−Holland 1992, pp. 484−492. (IFIP Transactions A: Computer Science and Technology vol. A−12).

86. E. Ukkonen: Approximate string−matching and the q−gram distance. In: *Sequences II. Methods in Communication, Security, and Computer Science*. Ed. by R. Capocelli et al. Springer−Verlag, New York, NY, 1993, pp. 300−312.

87. E. Ukkonen: Approximate string−matching over suffix trees. In: *Combinatorial Pattern Matching. 4th Annual Symposium, CPM 93*, Padova, Italy, June 1993. Ed. by A. Apostolico et al. Springer−Verlag, Berlin, 1993, pp. 228−242. (Lecture Notes in Computer Science 684).

88. E. Ukkonen: On−line construction of suffix−trees. Report A−1993−1, Department of Computer Science, University of Helsinki. 15 pp.

89. E. Ukkonen and D. Wood: Approximate string matching with suffix automata. *Algorithmica* 10(1993):5, 353−364.

Specifying and verifying and reasoning about programs (F.3.1)

90. J. Eloranta: Minimizing the number of transitions with respect to observation equivalence. *BIT* 31(1991):4, 576–590.

91. J. Eloranta: Equivalence concepts and algorithms for CCS–like languages. Ph.Lic. Thesis, Report C–1991–2, Department of Computer Science, University of Helsinki.

92. J. Eloranta: Minimal transition systems with respect to divergence preserving behavioural equivalences. Ph.D. Thesis, Report A–1994–1, Department of Computer Science, University of Helsinki.

93. R. Kaivola and A. Valmari: Using truth–preserving reductions to improve the clarity of Kripke–models. In: *CONCUR '91, 2nd International Conference on Concurrency Theory*, Amsterdam, The Netherlands, August 26–29, 1991. Ed. by J.C.M. Baeten et al. Springer–Verlag, Berlin, 1991, pp. 361–375. (Lecture Notes in Computer Science 527).

94. R. Kaivola and A. Valmari: The weakest compositional semantic equivalence preserving nexttime–less linear temporal logic. In: *CONCUR '92, 3rd Int. Conf. on Concurrency Theory,* Stony Brook, NY, USA, August 24–27, 1992. Ed. by W.R. Cleaveland. Springer–Verlag, Berlin, 1992, pp. 207–221. (Lecture Notes in Computer Science 630).

95. A. Valmari: Compositional state space generation. Report A–1991–5, Department of Computer Science, University of Helsinki. 30 pp.

96. A. Valmari: Alleviating state explosion during verification of behavioural equivalence. Report A–1992–4, Department of Computer Science, University of Helsinki. 57 pp.

97. A. Valmari and M. Tienari: An improved failures equivalence for finite–state systems with a reduction algorithm. In: *Protocol Specification, Testing, and Verification, XI. Proc. IFIP WG 6.1 11th International Symposium*, Stockholm, Sweden, June 18–20, 1991. Ed. by B. Jonsson et al. North–Holland, Amsterdam, 1991, pp. 1–18.

98. A. Valmari and M. Tienari: Compositional failure−based semantic models for Basic LOTOS. Internal Report 16, Tampere University of Technology, Software Systems Laboratory, July 1993. 25 pp.

Grammars and other rewriting systems (F.4.2)

99. R. op den Akker, B. Melichar, and J. Tarhio: The hierarchy of LR−attributed grammars. In: *Attribute Grammars and their Applications. International Conference WAGA*, Paris, France, Sept. 19−21, 1990. Ed. by P. Deransart et al. Springer−Verlag, Berlin, 1990, pp. 13−28. (Lecture Notes in Computer Science 461).

100. R. op den Akker, B. Melichar, and J. Tarhio: Attribute evaluation and parsing. In: *Attribute Grammars, Applications and Systems. International Summer School SAGA*, Prague, Czechoslovakia, June 4−13, 1991. Ed. by H. Alblas et al. Springer−Verlag, Berlin, 1991, pp. 187−214. (Lecture Notes in Computer Science 545).

101. J. Paakki: A logic−based modification of attribute grammars for practical compiler writing. In: *Logic Programming. Proc. 7th International Conference*, Jerusalem, Israel, June 18−20, 1990. Ed. by D.H.D. Warren et al. The MIT Press, 1990, pp. 203−217.

102. S. Sippu and E. Soisalon−Soininen: *Parsing theory*. Vol. 1: *Languages and parsing*. Springer−Verlag, Berlin, 1988. 228 pp. (EATCS Monographs on Theoretical Computer Science vol. 15).

103. S. Sippu and E. Soisalon−Soininen: *Parsing theory*. Vol. 2: *LR(k) and LL(k) parsing*. Springer−Verlag, Berlin, 1990. 426 pp. (EATCS Monographs on Theoretical Computer Science vol. 20).

104. E. Soisalon−Soininen and J. Tarhio: Looping LR parsers. *Information Processing Letters* 26(1988):5, 251−253.

105. J. Tarhio: ATTE, editor for attribute grammars. In: *Symposium on Programming Languages and Software Tools. Proc. 1st Finnish−Hungarian Workshop*, 8−11 August, 1989, Szeged, Hungary. Ed. by T. Gyimothy. Hungarian Academy of Sciences, 1989, pp. 58−64.

106. J. Tarhio: Uncle−attributed grammars. *BIT* 30(1990):2, 437−449.

Database management − logical design (H.2.1)

107. G. Grahne: Horn tables − an efficient tool for handling incomplete information in databases. In: *Proc. 8th ACM SIGACT−SIGMOD−SIGART Symposium on Principles of Database Systems*, Philadelphia, PA, 1989, pp. 75−82.

108. M. Kantola, H. Mannila, K.–J. Räihä, and H. Siirtola: Discovering functional and inclusion dependencies in relational databases. *International Journal on Intelligent Systems* 7(1992), 591−607.

109. H. Laine: YYY − a database design tool. *Acta Cybernetica* 9(1990):3, 270–280.

110. H. Mannila and K.–J. Räihä: Practical algorithms for finding prime attributes and testing normal forms. In: *Proc. 8th ACM SIGACT−SIGMOD−SIGART Symposium on Principles of Database Systems*, Philadelphia, PA, 1989, pp. 128−133.

111. H. Mannila and K.–J. Räihä: On query languages for the p−string data model. In: *Information Modelling and Knowledge Bases*. Ed. by H. Kangassalo et al. IOS Press, 1990, pp. 469−482.

112. H. Mannila and K.–J. Räihä: Algorithms for inferring functional dependencies. *Data & Knowledge Engineering* 12(1994), 83−99.

113. H. Mannila and K.–J. Räihä: *The design of relational databases*. Addison−Wesley, 1992. 318 pp.

114. H. Mannila and K.–J. Räihä: On the complexity of inferring functional dependencies. *Discrete Applied Mathematics* 40(1992):2, 237−243.

115. H. Mannila: View updates and multiple representations. In: *Symposium on Programming Languages and Software Tools. Proc. 1st Finnish−Hungarian Workshop*, 8−11 August, 1989, Szeged, Hungary. Ed. by T. Gyimothy. Hungarian Academy of Sciences, 1989, pp. 168−176.

Database management – languages (H.2.3)

116. G. Grahne and A. Mendelzon: Updates and subjunctive queries. To appear in *Information and Computation*.

117. P. Kilpeläinen and H. Mannila: Retrieval from hierarchical texts by partial patterns. In: *SIGIR '93. Proc. 16th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, Pittsburgh, PA, June 27 – July 1, 1993. Ed. by R. Korfhage et al. ACM Press, 1993, pp. 214–222.

118. H. Mannila and K.–J. Räihä: Automatic generation of test data for relational queries. *Journal of Computer and System Sciences* 38(1989):2, 240–258.

Database management – systems (H.2.4)

119. S. Abiteboul, P.C. Kanellakis, and G. Grahne: On the representation and querying of sets of possible worlds. *Theoretical Computer Science* 78(1991):1, 159–187.

120. G. Grahne: *The problem of incomplete information in relational databases*. Springer–Verlag, Berlin, 1991. 156 pp. (Lecture Notes in Computer Science 554).

121. G. Grahne, M. Nykänen and E. Ukkonen: Reasoning about strings in datadases. *Proc. of the 13th ACM SIGACT–SIGMOD Symposium on Principles of Database Systems*. Minneapolis, Mn. 1994, to appear.

122. G. Grahne, S. Sippu, and E. Soisalon–Soininen: Efficient evaluation for a subset of recursive queries. *Journal of Logic Programming* 10(1991):3/4, 301–332.

123. P. Kilpeläinen, G. Lindén, H. Mannila, and E. Nikunen: A structured document database system. In: *EP90, Proc. Int. Conf. Electronic Publishing, Document Manipulation & Typography*. Ed. by R. Furuta. Cambridge University Press, 1990, pp. 139–151.

124. J. Kuittinen, O. Nurmi, S. Sippu, and E. Soisalon−Soininen: Efficient implementation of loops in bottom−up evaluation of logic queries. In: *Proc. 16th International Conference on Very Large Data Bases*, August 13−16, 1990, Brisbane, Australia (VLDB '90). Morgan Kaufmann (distr.), 1990, pp. 372−379.

125. G. Lausen and E. Soisalon−Soininen: Locking policies and predeclared transactions. In: *MFDBS 89, 2nd Symposium on Mathematical Fundamentals of Database Systems*, Visegrad, Hungary, June 26−30, 1989. Ed. by J. Demetrovics et al. Springer−Verlag, Berlin, 1989, pp. 317−336. (Lecture Notes in Computer Science 364).

126. G. Lausen, E. Soisalon−Soininen, and P. Widmayer: On the power of safe locking. *Journal of Computer and System Sciences* 40(1990):2, 269−288.

127. W. Litwin and H. Tirri: Flexible concurrency control using value dates. *IEEE Distributed Processing Technical Committee Newsletter* 10(1988):2, 42−49. (Also in: Integration of information systems: bridging heterogeneous information. Ed. by A. Gupta. IEEE Press, 1989).

128. J. Puustjärvi: Distributed management of transactions in heterogeneous distributed database systems. *BIT* 31(1991):3, 406−420.

129. S. Sippu and E. Soisalon−Soininen: An optimization strategy for recursive queries in logic databases. In: *Proc. 4th International Conference on Data Engineering*, Feb. 1−5, 1988, Los Angeles, CA, pp. 470−477.

130. S. Sippu and E. Soisalon−Soininen: A generalized transitive closure for relational queries. In: *Proc. 7th ACM SIGACT−SIGMOD−SIGART Symposium on Principles of Database Systems*, March 21−23, 1988, Austin, TX, pp. 325−332.

131. S. Sippu and E. Soisalon−Soininen: Multiple SIP strategies and bottom−up adorning in logic query optimization. In: *ICDT'90, 3rd International Conference on Database Theory*, Paris, France. Dec. 12−14, 1990. Ed. by S. Abiteboul et al. Springer−Verlag, Berlin, 1990, pp. 485−498. (Lecture Notes in Computer Science 470).

132. S. Sippu and E. Soisalon−Soininen: An analysis of magic sets and related optimization strategies for logic queries. Technical report B82, Laboratory of Information Processing Science, Helsinki University of Technology, 1992.

133. H. Tirri (ed.): Interoperability of heterogenous information systems: final report of the COST 11 ter project. Report A−1989−2, Department of Computer Science, University of Helsinki. 110 pp.

134. H. Tirri, J. Srinivasan, and B. Bhargava: Integrating data sources using federated objects. In: *International Workshop on Distributed Object Management*, Edmonton, CA, 1992. (To appear).

135. H. Tirri, J. Srinivasan, and B. Bhargava: Transactions for fragmented composite objects. Report CSD−TR−91−083, Department of Computer Sciences, Purdue University. 23 pp. (Submitted to: Very Large Database Journal, 1992).

Content analysis and indexing (H.3.1)

136. J. Kivinen and H. Mannila: Approximate dependency inference from relations. In: *Database Theory − ICDT '92, 4th International Conference*, Berlin, Germany, Oct. 14–16, 1992. Ed. by J. Biskup et al. Springer−Verlag, Berlin, 1992, pp. 86–98. (Lecture Notes in Computer Science 646).

Information search and retrieval (H.3.3)

137. J. Kivinen and H. Mannila: The power of sampling in knowledge discovery. Report C−1993−66, Department of Computer Science, University of Helsinki. 18 pp. *Proc. of the 13th ACM SIGACT−SIGMOD Symposium on Principles of Database Systems*. Minneapolis, Mn. 1994, to appear.

138. H. Mannila, H. Toivonen and A.I. Verkamo: Improved methods for finding association rules. Report C−1993−65, Department of Computer Science, University of Helsinki. 16 pp.

User interfaces (H.5.2)

139. N. Holsti: A session editor with incremental execution functions. *Software − Practice and Experience* 19(1989):4, 329−350.

140. N. Holsti: Transcript editing, a simple user interface tool. In: *Programming Environments for High−Level Scientific Problem Solving. Proc. of the IFIP TC2/WG2.5 Working Conference*, Karlsruhe, Germany, 23−27 Sept. 1991. Ed. by P.W. Gaffney et al. North−Holland, Amsterdam, 1992, pp. 321−329. (IFIP Transactions A: Computer Science and Technology vol. A−2).

Group and organization interfaces (H.5.3)

141. S. Eherer, H. Erkiö, H. Lewe, B. Ludwig, S. Myrseth, R. Popping, X. Tong: Information technology support for group knowledge development. Working document of COST 14 Co−Tech working group 1, Sept. 1992. 56 pp.

Deduction and theorem proving (I.2.3)

142. P. Orponen: Dempster's rule of combination is #P−complete. *Artificial Intelligence* 44(1990):1/2, 245−253.

Knowledge representation formalisms and methods (I.2.4)

143. P. Floréen, P. Myllymäki, P. Orponen, and H. Tirri: NEULA − a hybrid neural−symbolic expert system shell. *Tietojenkäsittelytiede* 3(1992): 11−18.

144. G. Grahne: Updates and counterfactuals. In: *Principles of Knowledge Representation and Reasoning. Proc. 2nd Int. Conf. KR91*. Ed. by J. Allen et al. Morgan Kaufmann, 1991, pp. 269−276.

145. G. Grahne, A.O. Mendelzon, and R. Reiter: On the semantics of belief revision systems. In: *Theoretical Aspects of Reasoning About Knowledge. Proc. 4th Conf. (TARK 1992)*, March 22−25, 1992, Monterey, CA. Ed. by Y.

Moses et al. Morgan Kaufmann, San Mateo, CA, 1992, pp. 132−142.

146. G. Grahne, A.O. Mendelzon, and P.Z. Revesz: Knowledgebase transformations. In: *Proc. 11th ACM SIGACT−SIGMOD−SIGART Symposium on Principles of Database Systems*, June 2–4, 1992, San Diego, CA, pp. 246−260.

147. P. Myllymäki, P. Orponen, T. Silander: Integrating symbolic reasoning with neurally represented background knowledge. In: *Proc. of the Finnish AI Conference* (Espoo, Finland, June 1992), Vol. 2. Finnish AI Society, Helsinki, 1992, pp. 231−240.

148. H. Tirri: Applying neural computing to expert system design: coping with complex sensory data and attribute selection. In: *Proc. 3rd International Conference on Foundations of Data Organization and Algorithms (FODO '89)*, Paris, France, pp. 474–489.

149. H. Toivonen: High level integration of rule−based and procedural programming through unification of frames and classes. In: *Proc. 5th Florida Artificial Intelligence Research Symposium*, Ft. Lauderdale, Florida, April 7−10, 1992. Ed. by Mark B. Fishman. Florida AI Research Society, St. Petersburg, Florida, 1992, pp. 129−133.

Learning (I.2.6)

150. H. Ahonen, H. Mannila, and E. Nikunen: Forming grammars for structured documents. In: *Proc. AAAI−93 Workshop on Knowledge Discovery in Databases*, Washington, D.C., July 11−12, 1993. Ed. by G. Piatetsky−Shapiro. AAAI, pp. 314−325.

151. T. Elomaa: Embedding automatically acquired concept descriptions into a production system. In: *Proc. 4th Int. Symp. Knowledge Engineering*. Rank Xerox, 1990. 8 pp.

152. T. Elomaa: Extending the learnability of decision trees. In: *3rd Int. Conference on Tools for Artificial Intelligence (TAI '91)*, Nov. 5−8, 1991, San Jose, CA, pp. 504−505.

153. T. Elomaa and N. Holsti: An experimental comparison of inducing decision trees and decision lists in noisy domains. In: *EWSL−89. Proc. 4th European Working Session on Learning*, 4−6 December 1989, Montpellier. Ed. by K. Morik. Pitman/Morgan Kaufmann, 1990, pp. 59−69.

154. T. Elomaa and J. Kivinen: On inducing topologically minimal decision trees. In: *Proc. 2nd Int. IEEE Conference on Tools for Artificial Intelligence (TAI 90)*, Herndon, VA, Nov. 6–9, 1990, pp. 746–752.

155. T. Elomaa and J. Kivinen: Learning decision trees from noisy examples. Report A−1991−3, Department of Computer Science, University of Helsinki. 15 pp.

156. R. Kankkunen, H. Mannila, M. Rantamäki, and E. Ukkonen: Experience in inductive inference of a hyphenation algorithm for Finnish. In: *STeP−90 Finnish Artificial Intelligence Symposium*, University of Oulu, June 11−14, 1990. Ed. by M. Djupsund et al. Finnish Artificial Intelligence Society, 1990, pp. 183−193.

157. J. Kivinen: Reliable and useful learning. In: *Proc. 2nd Annual Workshop on Computational Learning Theory*, University of California, Santa Cruz, July 31 – August 2, 1989. Ed. by R. Rivest et al. Morgan Kaufmann, San Mateo, CA, 1989, pp. 365−380.

158. J. Kivinen: Reliable and useful learning with uniform probability distributions. In: *Proc. 1st International Workshop on Algorithmic Learning Theory*. Ed. by S. Arikawa et al. Japanese Society for Artificial Intelligence, 1990, pp. 209−222.

159. J. Kivinen, H. Mannila, and E. Ukkonen: Learning hierachical rule sets. In: *Proc. 5th ACM Workshop on Computational Learning Theory*, 1992, pp. 37−44.

160. J. Kivinen, H. Mannila, and E. Ukkonen: Learning rules with local exceptions. Pres. at: *European Conference on Computational Learning Theory* 1993. (To appear).

161. J. Kivinen and M.K. Warmuth: Using experts for predicting continuous outcomes. Pres. at: *European Conference on Computational Learning Theory*

1993. (To appear.)

162. P. Myllymäki and H. Tirri: Bayesian case−based reasoning with neural networks. In: *1993 IEEE International Conference on Neural Networks*, San Francisco, CA, March 28 − April 1, 1993. Vol. 1, pp. 422−427.

163. P. Myllymäki and H. Tirri: Learning in neural networks with Bayesian prototypes. To appear in *Proceedings of SOUTHCON'94* (Orlando, March 1994).

164. E. Nikunen and H. Mannila: Defining and inverting textual views of structured documents. In: *Symposium on Programming Languages and Software Tools. Proc. 1st Finnish−Hungarian Workshop*, 8−11 August, 1989, Szeged, Hungary. Ed. by T. Gyimothy. Hungarian Academy of Sciences, 1989, pp. 108−126.

165. H. Tirri: Concept randomness and neural networks. In: *Proc. 1991 International Conference on Artificial Neural Networks (ICANN−91)*, Espoo, Finland, 24−28 June, 1991. Ed. by T. Kohonen et al. North−Holland, Amsterdam, 1991. Vol. 2, pp. 1367−1370.

166. H. Tirri: Learning with instance based encodings. Pres. at the *2nd Annual Workshop on Computational Learning Theory and Natural Learning Systems: Constraints and Prospects*, Berkeley, CA, 1991. (To appear.)

Problem solving, control methods, and search (I.2.8)

167. R. Greiner and P. Orponen: Probably approximately optimal derivation strategies. In: *Principles of Knowledge Representation and Reasoning. Proc. 2nd Int. Conf. (KR91)*, Cambridge, MA, April 22−25, 1991. Ed. by J. Allen et al. Morgan Kaufmann, 1991, pp. 277−288.

168. P. Orponen and R. Greiner: On the sample complexity of finding good search strategies. In: *Proc. 3rd Annual Workshop on Computational Learning Theory*, Rochester, NY, August 6−8, 1990. Ed. by M. Fulk et al. Morgan Kaufmann, 1990, pp. 352−358.

Simulation and modeling – general (I.6.0)

169. T. Kerola: Qsolver − a modular environment for solving queueing network models. In: *Computer Performance Evaluation: Modelling Techniques and Tools. Proc. 5th Int. Conference*, Torino, Italy, Feb. 13−15, 1991. Ed. by G. Balbo et al. Elsevier, 1992, pp. 427−438.

170. T. Kerola, P. Purovesi, and E. Hietalahti: Qsolver user guide. Report C−1990−32, Department of Computer Science, University of Helsinki. 44 pp.

Model validation and analysis (I.6.4)

171. K.E.E. Raatikainen: Validating percentiles of response times in queueing network simulation. In: *Proc. 1988 Summer Computer Simulation Conference*, Seattle, WA, pp. 134−141.

172. K.E.E. Raatikainen: Sequential procedure for simultaneous estimation of several percentiles. *Transactions of the Society of Computer Simulation* 7(1990):1, 21−44.

173. K.E.E. Raatikainen: Modeling service−time distributions for queueing network simulation. *Communications in Statistics. B, Simulation and Computation* 20(1991):1, 375−390.

174. K.E.E. Raatikainen: Modeling service distributions in queueing network simulation. *Simulation* 59(1992):2, 116−126.

Simulation output analysis (I.6.6)

175. K.E.E. Raatikainen: A sequential procedure for simultaneous estimation of several means. *ACM Transactions on Modeling and Computer Simulation* 3(1993):2, 108−133.

176. K.E.E. Raatikainen: Simulation estimation of dynamic properties in queueing systems. Report C−1993−26, University of Helsinki, Department of Computer Science. 19 pp.

Life and medical sciences (J.3)

177. J. Rinne, H. Lokki, and P. Saurola: Survival estimates of nestling recoveries: forbidden fruits of ringing? *The Ring* 13(1990):1/2, 255−270.

178. J. Rinne, H. Lokki, and P. Saurola: Extensive parameterisation of survival models for recovery data analysis. In: *Marked Individuals in the Study of Bird Population.* Ed. by J.D. Lebreton et al. Birkhäuser, 1993, pp. 65−75.

History of computing (K.2)

179. M. Tienari (ed.): *The First Years of Computer Technology in Finland. A collection of articles*. Suomen atk−kustannus Oy, Espoo, 1993. 518 pp. /In Finnish./

Computer uses in education (K.3.1)

180. L. Huovinen, L. Häkkinen, M. Mäkelä, and P. Nummelin: Using CAD software as a teacher's tool. In: *EURIT 90, A European Conference on Technology and Education*, 1990. (Full paper on disk, textfile: E018).

181. M. Mäkelä, L. Huovinen, and P. Nummelin: Computer uses in the university classroom. *Education & Computing* 6(1990):1/2, 191−197.

182. P. Nummelin, L. Huovinen, and M. Mäkelä: Experiences of the course "Computer Uses in Education". In: *EURIT 90, A European Conference on Technology and Education*, 1990. (Full paper on disk, textfile: E012).

3. Education

3.1. Educational program

The students of the department normally start their university studies at the age of 19. Their goal is to receive a B.Sc. (Bachelor of Science) or M.Sc. (Master of Science) degree in computer science requiring three to four or five years of full time study. Beyond the first degree there are two alternative graduate degrees: Ph.Lic. (Licentiate of Philosophy) degree and Ph.D. (Doctor of Philosophy).

The academic year has two semesters: the fall semester lasts from September 1 to December 20 (classes from September 11 to December 10), while the spring semester lasts from January 16 to May 31 (classes from January 16 to May 10, excluding one week of Easter vacation). It is also possible to study in summer. Intensive courses of 4–5 weeks covering introductory topics are offered in June and August. Graduate courses are also organized in cooperation with other Finnish universities during the summer. These courses typically last for one week and are intended for Ph.Lic. and Ph.D. students. These are often given in English by foreign visitors.

In order to obtain a B.Sc. degree a student must earn 120 units of academic credit. For M.Sc. degree 160 units of credit as well as a thesis is required. One credit should normally correspond to roughly one week (40 hours) of study. Our students typically register for 12 credits ("study weeks") in the fall semester and 15 credits in the spring semester. During the summer sessions a student can earn an additional 8−10 credits. Most students, however, work in industry during the summer to gain practical experience in data processing and earn money. This is actually what the department recommends. Thus, a normal student should earn 27 credits a year, an exceptionally diligent full−year student 40 credits. Nevertheless, there is a considerable variation in study efficiency among students.

Our typical course consists of from 50 to 60 lectures (a lecture lasts 45 minutes) and of from 20 to 30 hours of problem solving, discussion and repetition sessions in small groups of from 10 to 20 students. Each course is examined individually with grades: 3/3 = excellent, 2/3 = good, 1/3 = satisfactory. A typical course is worth 4 or 5 credits. The computer laboratory is supervised in small groups of 6 to 12 students. Students also attend seminar courses, the enrollment of which ranges from 5 to 15 students. In these seminars current literature is read, essays are written and oral presentations are given. A seminar group normally meets 2 hours per week yielding 2 credits per a semester.

In order to receive a M.Sc. degree in computer science, students are required to earn their credits as follows:

Computer science	>= 95 c
Mathematics	>= 25 c
Physics	>= 15 c
General studies	10 c
total	>= 160 c

In mathematics the obligatory courses are calculus (15 c), algebra (5 c), and probability (5 c). Physics can be replaced with almost any other subject, such as economics, administration, statistics, or psychology. For a B.Sc. degree 55 credit units of computer science is sufficient.

The computer science studies for a M.Sc. degree can be subdivided as follows:

Obligatory courses and laboratories	40 c
Elective courses	>= 21 c
Seminars	>= 4 c
Project work	10 c
M.Sc. Thesis, Scientific writing	20 c
total	>= 95 c

The obligatory computer science courses and laboratory work currently cover (academic year 1993–94) the following areas:

Pascal programming	6 c
Data structures	8 c
Operating systems and hardware architecture	10 c
Information systems and databases	11 c
Theory of computation	5 c
total	40 c

In principle, students are fairly free to choose any elective courses. They normally follow the recommendation of the department by building up a specialized background knowledge for a successful thesis in one of our research groups. Thus, a student might orient himself according to his study goals, interests and talents towards, e.g., theoretical computer science, information systems, distributed systems, operating systems, artificial intelligence, or programming languages.

To start studies for the graduate degrees Ph.Lic. (Licentiate of Philosophy) and Ph.D. (Doctor of Philosophy) in Computer Science, a student having shown good academic standing in his M.Sc. studies contacts a professor of the department. At first, a personal study program is designed for the student. It outlines the field of specialization of the studies, the topic for the thesis, and the content and the schedule of the coursework. Each student is assigned a personal advisor.

The requirements for the Ph.Lic. degree can be summarized as follows:

Elective courses and seminars	
in computer science	20 c
in mathematics	20 c
Ph.Lic. thesis	>= 50 c
total	>= 90 c

The elective courses in mathematics can be replaced with coursework in other subjects such as physics, economy, psychology etc. The Ph.Lic. thesis can be written in Finnish, Swedish (the two official languages of our university) or English. The allocation of credits for thesis research indicates that after the required coursework it should take 1–2 years to prepare a

Ph.Lic. thesis.

It is important that the student takes the courses and the seminars early enough to obtain sufficient background for writing the thesis. Active participation in seminar courses is particularly useful as is attending international schools and specialized research courses. Such courses are also regularly given at the department.

The requirements for the Ph.D. degree are otherwise the same as for the Ph.Lic. degree, but a Ph.D. thesis demands more work, from 2 to 3 years, roughly one year more than a Ph.Lic. thesis. The Ph.D. degree can be achieved directly, although we often recommend that our students take the Ph.Lic. degree first, and then by improving and extending their Ph.Lic. research, achieve the Ph.D. level.

The Ph.D. theses are written in English. A thesis should include a scientific contribution which is significant enough to be publishable internationally. A Ph.D. thesis (as well as a Ph.Lic. thesis) can also be assembled from a number of published articles or congress papers, possibly written jointly with other authors. A dissertation of this type, which is actually fairly common, consists of an introductory survey written by the candidate alone, with the individual articles as appendices.

Preparing the thesis is clearly the most demanding part of the Ph.D. and Ph.Lic. studies. To succeed with the thesis it is recommendable that a student works within a research group at the department. The support and the criticism given by the group is often essential for making progress in the work.

3.2. Course descriptions

a) Undergraduate courses

Fundamentals of ADP (2 cu)
Introduction to computers and data processing. Algorithm. Computer hardware. Operating systems. Applications software. Programming languages. Database systems. Communication networks. System analysis and design.

Programming (Pascal) (4 cu)
The course provides the student with the knowledge of the principles of an algorithmic language. The student will be able to program in Pascal and implement those programs in a microcomputer environment.

Programming (Fortran) (3 cu)
The course provides the student with the knowledge of the principles of an algorithmic language. The student will be able to program in Fortran and implement those programs in a computer environment.

Computer Systems Organization (5 cu)
Introduction. Data presentation, error detection and correction. Computer organization. Conventional machine level. Assembly language. Compilation, linking, loading. Input/Output. Secondary storage. Operating system. Data communication equipment and software.

Information systems (4 cu)
Data systems and systems development. Data flow techniques. Conceptual Modelling. Transaction Analysis. Data base design. Relational data model. SQL. User Interfaces. CASE tools.

Programming Project (2 cu)
The student designs, documents and programs a complete, nearly realistic program. In the course of the development she/he also gives small lectures and demonstrations about the project.

Data Structures (5 cu)
Basic data structures. Applications to algorithms. Analysis of algorithms. Implementations of data structures and algorithms in Pascal. Memory management.

Operating Systems (5 cu)
Operating system concepts. Processes. Input/Output. Memory management. File systems.

Database Systems I (5 cu)
Databases and database management systems. Entity–relationship model. Relational data model and relational algebra. SQL language. Relational calculus. QUEL and QBE languages. Hierarchical and network data models.

Functional dependencies and normalization. Database design. Transaction processing.

Theory of Computation (5 cu)
Finite automata and regular languages. Context−free grammars and languages. Rudiments of parsing theory and attribute grammars. Context−sensitive and type−0 grammars. Turing machines. Recursive and recursively enumerable sets. Computability and computational complexity.

Data Structures Project (3 cu)
A simulator or some other rather large program is designed, programmed, tested and documented in a mainframe environment.

Information Systems Project (2 cu)
A small ADP−system is designed and programmed either in a microcomputer or in a mainframe environment.

Artificial Intelligence (4 cu)
A general view of the potential of AI in various kinds of problem solving is given. Main results in AI research and applications are represented. Also basic ability to construct AI based software is provided. The course consists of an overview of AI, LISP, knowledge representation, constraint propagation, exploring alternatives, problem solving, language understanding, image understanding and learning.

Computer Graphics (4 cu)
Overview of graphics systems. Output primitives and their attributes. Two−dimensional transformations. Windowing and clipping. Segments. Interactive input methods. Three−dimensional concepts, representations, transformations, viewing. Hidden−surface and hidden−line removal. Shading and color models. Modeling methods. Design of the user interface. Individual practical work.

Computer Uses in Education (4 cu)
Fundamentals of computer applications in education. Computer as a tutor, tool, and tutee. Computer assisted instruction (CAI) systems. Courseware design, development, and evaluation. Authoring systems and languages. Multimedia CAI. Intelligent CAI. Applications and research. Practical courseware designing in small groups.

Principles of Programming Languages (4 cu)
History. Basic concepts of Ada. Type systems (Algol68, Pascal, Ada). Blocks, subroutines and parameter passing. Modules (Clu, Modula–2, Euclid, Ada). Classes and objects (Simula, C++, Smalltalk, Oberon). Exception handling (PL/I, Clu, Ada). Concurrency (Concurrent Pascal, Ada). Generics. Programming environments.

Data Communications (4 cu)
Data transmission. Data encoding. Digital data communication techniques. Data link control. Multiplexing. Data communication networking techniques. Circuit switching. Packet switching. Radio and satellite networks. Local area networks. Computer communications architecture. Network access protocols. Internetworking. Transport protocols. Integrated services digital network.

Programming in C (2 cu)
Definition of ANSI–C. The UNIX system interface. Programming utilities and libraries. General programming principles. Maintaining program groups. Programming in the large.

UNIX Platform (2 cu)
Insight of a UNIX system for programmers: processes, file systems, peripherals, tools for interprocess communication.

UNIX Networking (4 cu)
Data communication protocols typically used in a UNIX environment are discussed including networking services provided to application programs, as well as design and implementation principles of these protocols. The focus is in the practical aspects of designing and implementing distributed applications using these protocols.

Window Systems (2 cu)
Open Look. Xview. X. CUA.

Implementation of Database Applications (4 cu)
Course introduces students to the principles and practice of implementation of database applications using application development systems and embedded database languages. During the course students implement a small application.

Social Role of Automatic Data Processing (2 cu)
Information society. Information technology policy of Finland. Economic effects. Effects on employment. Quality of work. Educational impacts. Privacy legislation and other juridical issues. Effects of new telecommunication services.

Scientific Writing (4 cu)
Sources of scientific information. Use of libraries and scientific data bases. The structure and details of a scientific publication. Examples of scientific Finnish. Three individual survey writing exercises.

b) Graduate courses

Design and Analysis of Algorithms (5 cu)
Analysis techniques. Design techniques. Models of computation and lower bounds. Algorithms on sets. Graph algorithms. Approximation algorithms for NP–complete problems. Probabilistic algorithms. Parallel algorithms.

Compiler Construction (5 cu)
Examples of industrial compiler projects. Introduction to compiling. Lexical analysis: regular expressions, scanning. Syntax analysis: context–free grammars, recursive descent parsing, parsing conflicts and their solving. Semantic analysis: symbol tables, attribute grammars. Code generation: intermediate languages, optimization.

Database Systems II (5 cu)
Physical data organization in databases. Sequential, hashed and indexed file structures. Secondary indices. Query processing and optimization for relational database systems. Join algorithms. Hypergraphs. Semijoin programs. Logic query processing. Datalog data model. Naive and seminaive evaluation of least fixed points. Optimization by magic sets.

String processing algorithms (5 cu)
Exact string matching. Approximate string matching. Pattern matching in static strings. Text databases and hypertext. Algorithm implementation and comparison project.

Machine learning (4 cu)
Inductive inference. Concept learning. Computational complexity of learning. Learning in uncountable domains. Learning functions. Learning finite automata. Learning by neural networks.

Computational complexity theory (3 cu)
Review of Turing machines and complexity classes. Space complexity. Alternating Turing machines and the polynomial time hierarchy. Oracle Turing machines and relativization. Structure theorems for NP–complete sets. Structure within NP. Probabilistic Turing machines and complexity. Nonuniform complexity measures. Kolmogorov complexity.

Advanced Computer Graphics (4 cu)
Selection of advanced topics such as tray tracing, radiosity, solid modeling, illumination and color, scientific visualization, etc. are taken as a theme of the course. Individual and group work, report writing and oral presentations by the participants.

Neural Networks (3cu)
The emphasis of the course is on providing students with an intuitive understanding of the common neural network models and the related algorithms. The course contents include discussion on the principles of neural computing and some application areas, followed by a more detailed discussion on the basic network models such as feedforward networks and self–organizing maps. The course has several implementation projects and hands–on experiments on various simulator software packages.

Logic Programming (4 cu)
Introduction to logic programming and Prolog. Abstract and Prolog interpreter. Unification. Semantics of logic programs. Backtracking, cut, negation. Concurrent languages. General programming techniques. Meta–interpreters. Definite clause grammars.

Semantics of Programs (3 cu)
Axiomatic semantics of programs. Weakest precondition calculus for the guarded command language of Dijkstra.

Distributed Operating Systems (5 cu)
Interprocess communication, naming facility, process management, resource

allocation, file service, ODP standardization.

Performance Analysis (4 cu)
Measuring techniques. Workload modelling. Simulation methods. Queueing models. Queueing network models. Case studies.

Computer Networks (5 cu)
Specification and analysis of communication protocols in an extended state transition model. OSI upper layer protocols. Security in computer networks.

Concurrency Theory (4 cu)
Process algebra (Milner's CCS) and temporal logic with applications to distributed systems, process equivalences.

Simulation (4 cu)
Modelling and simulation. Data collection and analysis. Random numbers and random−variate generation. Model verification and validation. Output analysis. Experimentation and optimization.

Principles of Software Engineering (4 cu)
Software engineering as a process. Software process evaluation. Software quality assurance, measurement and evaluation. Software quality factors: correctness, testability, reliability, performance, security, safety, changeability, reusability, portability, usability.

Principles of Concurrent Programming (3 cu)
Basic abstractions in concurrent programming, distributed algorithms, Ada−rendezvous, implementation issues.

Computer Architecture (4 cu)
Basics of computer architecture from instruction set design to I/O subsystems. The emphasis is on uniprocessor systems but parallel and distributed architectures are also discussed.

Object−Oriented Programming (4 cu)
Introduction to basic concepts: object philosophy, object, class, inheritance, and polymorphism. Principles of object−oriented programming: data abstraction, encapsulation, information hiding, representation independence, subclass, super class, abstract class, static and dynamic binding; values vs.

objects. History and development: the first object−oriented language Simula, Smalltalk approach: message, method, and protocol. Object−oriented software construction: object−oriented analysis and design; building abstraction hierarchies. Object−oriented programming in C++: classes as types, protection levels, object construction and desctruction, templates, exceptions; examples; C++ vs. ANSI C. Comparison of object−oriented languages: concepts and terms; classes vs. modules.

Relational database design (3 cu)
ER model and relational model. Object−oriented models. Integrity constraints and dependencies. Goals of database design. Axiomatizations of functional and inclusion dependencies. Algorithms for manipulating integrity constraints. Transformations between models. Database design by using examples. Generation of example databases. Dependency inference. Performance issues.

Transaction processing (4 cu)
Serializability theory. Locking and non−locking schedulers. Multiversion concurrency control. Centralized and distributed recovery. Management of replicated data. Multidatabase transaction management. Cooperative transaction management. Prototype systems.

User Interfaces (4 cu)
Psychological foundations. Types of users, user modelling. Design guidelines. Basic interaction styles and techniques. User support. Formal description of interaction. Interfaces to database systems. Hypertext. Multi−user interfaces.

Computer−Supported Cooperative Work (3 cu)
Concepts and history of CSCW. Meeting and decision support. Multi−user interfaces. Groupware. Social and organizational aspects.

Knowledge bases (3 cu)
Knowledge representation. Advanced data models and new database systems, including rule−based, object−oriented, and structured text databases. Knowledge−base systems.

Object−Oriented Databases (3 cu)
New applications areas for databases. Object−oriented data models. Example

systems. Query languages. Implementation principles. Query optimization. Clustering.

Project work (10 cu)
Requirements analysis. Design. Implementation techniques. Quality assurance. Project management. Each student takes part in a software project, where a group of students analyzes the requirements for a software product, designs and implements the product, using systematic software engineering methods.

c) Seminar Courses in 1991–93

- Algebraic Specification
- Analysis of Algorithms
- Bioinformatics
- Computational Geometry
- Computer Uses in Education
- Image Processing Methods
- Knowledge Engineering
- Machine Learning
- Research Seminar on Neural Computation
- Scientific Visualization
- History of Computing

- Distributed Systems
- Fault–tolerant Systems
- Formal Specification of Distributed Systems
- Managing the Software Process
- Mobile Workstation Architecture
- Object–Oriented Analysis and Design
- ODP Reference Model
- Paradigms in Programming
- Performance Analysis of Database Applications
- Performance of Database Systems
- Performance of Transaction Processing Systems
- Quality Factors in Software Engineering
- Research Seminar on Computer Networks
- Research Seminar on Object–Oriented Programming
- Software Engineering Methodologies

– Software Reliability Analysis
– Software Reuse
– Software Specification Methodologies
– Thesis Seminar (Distributed Systems)

– Database Structures
– Hypertext Systems
– Logic Databases (I and II)
– Research Seminar on Databases
– Research Seminar on User Interfaces
– Thesis Seminar Information Systems

3.3. Accepted theses

Theses for Doctor of Philosophy

Seppo Linnainmaa: *Analysis of some known methods of improving the accuracy of floating−point sums*. BIT 14(1974): 167−202.

Eljas Soisalon−Soininen: *Characterization of LL(k) languages by restricted LR(k) grammars.* (A−1977−3).

Esko Ukkonen: *On the effect of rounding errors on the flow of control in numerical processes.* (A−1977−7).

Ralph−Johan Back: *On the correctness of refinement steps in program development.* (A−1978−4).

V.–E. Juhani Virkkunen: *A unified approach to floating−point rounding with applications to multiple−precision summation.* (A−1980−1).

Hannu Erkiö: *Studies on the efficiency of certain internal sort algorithms.* (A−1980−4).

Seppo Sippu: *Syntax error handling in compilers.* (A−1981−1).

Kari−Jouko Räihä: *A space management technique for multi−pass attribute evaluators*. (A−1981−4).

Timo O. Alanko: *Empirical studies of program behaviour in virtual memory*. (A−1983−3).

Kai Koskimies: *Extensions of one−pass attribute grammars*. (A−1983−4).

Heikki Mannila: *Instance complexity for sorting and NP−complete problems*. (A−1985−1).

Ilkka J. Haikala: *Program behaviour in memory hierarchies*. (A−1986−2).

Pekka Orponen: *The structure of polynomial complexity cores*. (A−1986−3).

A. Inkeri Verkamo: *Sorting in hierarchical memories*. (A−1988−1).

Jorma Tarhio: *Attribute grammars for one−pass compilation*. (A−1988−11).

Gösta Grahne: *The problem of incomplete information in relational databases*. (A−1989−1).

Niklas Holsti: *Script editing for recovery and reversal in textual user interfaces*. (A−1989−5).

Kimmo E.E. Raatikainen: *Modelling and analysis techniques for capacity planning*. (A−1989−6).

Jukka Paakki: *Paradigms for attribute−grammar−based language implementation*. (A−1991−1).

Jyrki Kivinen: *Problems in computational learning theory*. (A−1992−1).

Patrik Floréen: *Computational complexity problems in neural associative memories*. (A−1992−5).

Pekka Kilpeläinen: *Tree matching problems with applications to structured text databases*. (A−1992−6).

Jaana Eloranta: *Minimal transition systems with respect to divergence preserving behavioural equivalences*. (A−1994−1).

Selection of theses for Licentiate of Philosophy

Kari−Jouko Räihä: *On attribute grammars and their use in compiler writing systems*. (A−1977−4).

Kai Koskimies: *A study on the programming language Euclid*. (A−1980−2).

Harri Laine: *Semantic integrity and data base update in the grammatical data base model*. (C−1981−48).

Jorma Tarhio: *Attribute evaluation during LR parsing*. (A−1982−4).

Juha Vihavainen: *Design of the simulation language Mode*. (C−1987−42). (In Finnish).

Jukka Paakki: *Generating one−pass semantic analysis for a compiler*. (A−1988−8).

Eeva Hartikainen: *Specification and design of distributed programs that use broadcasting*. (A−1988−13).

Jukka Mutikainen: *An experimental study of attribute selection criteria in decision tree induction*. (C−1989−67).

Erja Nikunen: *Views in structured text databases*. (C−1990−60).

Jaana Eloranta: *Equivalence concepts and algorithms for transition systems and CCS−like languages*. (C−1991−2).

Liisa Räihä: *Sequence comparison: computation of the edit distance*. (C−1991−59).

Juha Puustjärvi: *Management of transactions in heterogeneous distributed database system*. (C−1992−31).

Tapio Elomaa: *A hybrid approach to decision tree learning*. (C−1992−61).

Roope Kaivola: *Compositional linear temporal logic model−checking for concurrent systems*. (C−1993−1).

Pasi Tapanainen: *Finite−state parsing for natural languages*. (C−1993−7). (In Finnish).

Greger Lindén: *Incremental updates in structured documents*. (C−1993−19).

Petri Myllymäki: *Bayesian reasoning by stochastic neural networks.* (C−1993−67).

Selection of master theses in 1991−1993 (written in Finnish or in Swedish)

a) General Computer Science

Matti Gröhn: *Using sound in data analysis*. (C−1993−06).

Antero Lindholm: *Suffix tree algorithms for string matching*. (C−1993−64).

Tapani Järvinen: *Implementation of visual languages*. (C−1992−58).

Marko Myllymäki: *Computer graphics standards*. (C−1992−51).

Petri Myllymäki: *Teaching multilayer neural networks with optimized backpropagation methods*. (C−1991−10).

Ismo Mäkelä: *Triangularization of parametric surfaces*. (C−1991−14).

Matti Nykänen: *Representing temporal information*. (C−1991−30).

Tomi Silander: *Hybrid neural−symbolic knowledge representation schemes*. (C−1993−38).

Marko Teittinen: *Visualization of 4D objects*. (C−1992−39).

Erkki Vuori: *Two dimensional pattern matching*. (C−1992−10).

b) Computer Software

Olli Finni: *Interoperability testing of OSI implementations*. (C−1991−65).

Kari Granö: *ASN.1 extensions and their implementation in the CASN compiler*. (C−1992−53).

Erkki Hietalahti: *Graphical user interfaces − a survey and application to a queueing network package*. (C−1991−51).

Mikko Kolehmainen: *Concurrent programming in an ANSA−like distributed environment*. (C−1992−55).

Juha Korpi: *A performance study of some parallel simulation methods*. (C−1991−54).

Kari Lehtinen: *Performance of the basic services in the AHTO−system*. (C−1992−23).

Pia Ollila: *Connectionless and connection−oriented network layer in the OSI reference model*. (C−1992−09).

Matti Oosi: *The impact of some fault−tolerance techniques on the performance of the system*. (C−1993−03).

Kyösti Rantinoja: *Formal definition of an object oriented programming language*. (C−1991−52).

Juha Sahala: *Management of the OSI stack*. (C−1993−09).

Jyrki Soini: *Extension of local area networks*. (C−1991−29).

Pasi Tapanainen: *Finite−state grammar rules in a parser for natural languages*. (C−1992−05).

c) Information Systems

Helena Ahonen: *Object−oriented document management*. (C−1991−22).

Pentti Elolampi: *Static filtering in logic query optimization*. (C−1993−04).

Virpi Hassinen: *Task analysis methods and their application in designing a user interface*. (C−1993−54).

Heikki Heinaro: *Object−oriented software development in a banking environment*. (C−1992−29).

Kari Hurtta: *Magic sets in logic query optimization*. (C−1993−12).

Kimmo Hätönen: *Knowledge representation for text retrieval*. (C−1992−62).

Jari Jokiniemi: *Transaction management in interoperable database systems*. (C−1991−20).

Anna Kalimo: *User interface standards and their effects on designing user interfaces*. (C−1992−28).

Teija Kujala: *Modern information retrieval techniques*. (C−1991−66).

Maarit Liimatainen: *Information processing solutions for real−time groupware*. (C−1993−10).

Matti Makkonen: *Using semantic nets and default logic in representing multiple inheritance and exceptions*. (C−1991−13).

Kari Nummelin: *Management of exceptional data in databases*. (C−1993−31).

Minna Oja: *Object−oriented design of information systems*. (C−1992−13).

Tarmo Rahikainen: *Database management systems based on semantic models*. (C−1991−23).

Heikki Ristimäki: *Compression of texts written in Finnish*. (C−1993−56).

Pirjo Ronkainen: *Information retrieval and hypertext*. (C−1993−59).

Veli−Matti Suoranta: *Top−down evaluation of logic queries*. (C−1993−16).

Juha Taina: *Hypertext systems and the support for an individual user*.

(C–1992–14).

Markku Tamminen: *Design and analysis of screen layouts*. (C–1993–29).

Paula Turkki: *Query size estimation by sampling in relational databases*. (C–1993–05).

Timo Ukkonen: *Query processing in main memory database systems*. (C–1991–45).

d) Teacher Training

Jari Vanha–Eskola: *Computer science at university level: analysis of the first–year teaching*. (C–1993–58).

3.4. Abstracts of recent Ph.D. theses

Jukka Paakki: *Paradigms for attribute–grammar–based language implementation*. (A–1991–1).

Attribute grammars are a formalism for specifying and implementing programming languages. Methods and techniques are presented expressing attribute grammars themselves as a language. These methods are based on relating attribute grammars with programming paradigms. The presented formalisms are classified as *structured, object–oriented, functional, logic,* and *concurrent* attribute grammars. The characteristics of these attribute grammar paradigms are reviewed and analyzed. The central results of seven self–standing papers are summarized. These papers discuss empirically and theoretically the practical significance of relating the *nonterminal* concept of attribute grammars with the *block* concept of programming languages, the nonterminal concept with the *class* concept, and the *attribute* concept with the *logical variable* concept. Accordingly, structured, object–oriented, and logic attribute grammars are emphasized.

Jyrki Kivinen: *Problems in computational learning theory.* (A−1992−1).

This thesis summarizes results on some computational learning problems within the probabilistic framework proposed by Valiant. We give a polynomial time learning algorithm for concepts that can be represented as hierarchical rule sets. These representations consist of an arbitrary number of rules '*if c then l*' organized into a fixed number of levels. The rule means that an instance satisfying the condition *c* belongs to the class *l*. However, the rule can be overridden by a rule at a preceding level. If the instances are strings, the condition could be a test for the presence of a given substring in the instance. Instances could also be value assignments for a potentially infinite number of Boolean attributes, in which case the condition could be a Boolean conjunction of constant length. We also study reliable and probably useful learning, proposed by Rivest and Sloan. We derive upper and lower bounds for the sample complexity of reliable and probably useful learning in terms of combinatorial measures of the class of concepts to be learned. These measures resemble the Vapnik−Chervonenkis dimension. The results imply that monotone Boolean monomials cannot be learned reliably and probably usefully with a sample size that is polynomial in the number of variables. Rectangles cannot be learned reliably and probably usefully with any finite sample size. The sample complexity of reliable and probably useful learning remains high even if the examples are assumed to be drawn according to the uniform probability measure.

Patrik Floréen: *Computational complexity problems in neural associative memories.* (A−1992−5).

Neural associative memories are memory constructs consisting of a weighted network with a large number of simple nodes, whose computations collectively result in retrieving stored information when partial or distorted information is given as input. We study two types of neural associative memories: the Hopfield and the Hamming memories.

For the Hopfield memory, we first study worst−case bounds on the convergence times in both asynchronous and synchronous updating. However, the main theme in our study of Hopfield networks is the computational complexity of assessing the memory quality of a given network. We show that (assuming $P \neq NP$) it is hard to count the number of stable vectors, to compute the size of attraction domains (in synchronous updating), and to compute the attraction radii (in synchronous updating). It is hard even to approximate the attraction radii, unless $P = NP$.

For the Hamming memory, we study which conditions the parameters of the network must satisfy in order to ensure convergence and we obtain a tight bound on the convergence time. With an optimal choice of parameter values, the worst–case convergence time is $\Theta(p \log (pn))$, where p is the memory capacity and n is the vector length. Restricting the instances to those with a unique nearest stored vector to the input vector, the worst–case convergence time is $\Theta(p \log n)$. For random vectors, the probability for such instances to occur approaches 1 as n grows, if the number of stored vectors grows subexponentially in n. By dynamically changing the connection weights and by taking powers of the activity levels, the convergence is speeded up considerably. Finally, we present a new memory model based on the Hamming memory, but with a feed–forward network structure made possible by the use of more complicated node functions. In simulations, we show that the new memory model behaves as expected.

Throughout the study, we also explore the consequences of allowing "don't know" elements in both the input and the output. Experiments show that "don't know" elements are effective especially in the input.

Pekka Kilpeläinen: *Tree matching problems with applications to structured text databases*. (A–1992–6).

Tree matching is concerned with finding the instances, or matches, of a given pattern tree in a given target tree. We introduce ten interrelated matching problems called tree inclusion problems. A specific tree inclusion problem is defined by specifying the trees that are instances of the patterns. The problems differ from each other in the amount of similarity required between the patterns and their instances. We present and analyze algorithms for solving these problems, and show that the computational complexities of the problems range from linear to NP–complete.

The problems are motivated by the study of query languages for structured text databases. The structure of a text document can be described by a context–free grammar, and text collections can be represented as collections of parse trees. Matching–based operations are an intuitive basis for accessing the contents of structured text databases. In "*G–grammatical*" tree inclusion problems the target tree is a parse tree over a context–free grammar *G*. We show that a certain natural class of grammars allows solving some of the grammatical inclusion problems in linear time.

Tree inclusion problems are extended by introducing logical variables in the patterns. These variables allow extracting substructures of the pattern matches and posing equality constraints on them. We show that most of the tree inclusion problems with logical variables are NP−hard, and also consider solving their polynomial versions. As an application of these problems we finally show how tree inclusion with logical variables can be used for querying structured text databases, and discuss how the inclusion queries should be evaluated in practice.

Jaana Eloranta: *Minimal transition systems with respect to divergence preserving behavioural equivalences.* (A−1994−1).

Three divergence preserving behavioural equivalences for labelled transition systems (lts), namely *D*−bisimilarity, branching *D*−bisimilarity and CFFD−equivalence, are analyzed and compared. For an image−finite lts, which does not contain an infinite sequence of distinct τ−connected states, it is possible to find the unique lts which is state− and transition−minimal with respect to *D*−bisimilarity or branching *D*−bisimilarity. It is well known how to find an equivalent state−minimal lts. We show how the state and transition−minimal lts can be found, and moreover give a transition minimization algorithm in a finite case. In the case of CFFD−equivalence, which is a modification of failures−equivalence, a unique state− and transition−minimal lts cannot always be found.

For rooted versions of *D*−bisimilarity and branching *D*−bisimilarity the uniqueness result holds only for root−unwound lts's. Related equivalences, *D*−congruence and branching *D*−congruence, handle the first transitions from a root, rather than a root itself, in a special way. The relation between rooted equivalences and corresponding congruences is fully analyzed. For *D*−congruence and branching *D*−congruence a unique root−unwound minimal lts is found, for an image−finite (without an infinite sequence of distinct τ−connected states). Moreover, *D*−congruence and branching *D*−congruence are compositional with respect to Basic LOTOS operations. Since the same result holds for CFFD−equivalence, *D*−congruence, branching *D*−congruence and CFFD−equivalence are compatible with compositional reduction.

In order to apply and demonstrate the theoretical results, we specify and verify several concurrent systems. The specifications are reduced with respect to several equivalences, including *D*−congruence, branching

D−congruence and CFFD−equivalence. The analysis shows that it is possible to find errors, as well as reasons for divergences by studying reduced lts's. Moreover, these case studies are used to test the effect of the transition−minimization algorithm, to compare divergence preserving and divergence ignoring equivalences, and to compare branching and weak versions of equivalences. In some studies compositional reduction is used and the results are analyzed. Furthermore, based on the results of these applications, we propose some modifications to the reduction algorithm with respect to CFFD−equivalence.

4. International relations

Student exchange

The Department of Computer Science accepts annually some five new foreign students to start their undergraduate studies. These students are either beginning their computer science studies or complementing their studies. The department offers only occasionally courses lectured in English and thus a good knowledge of Finnish is needed for the successful completion of undergraduate studies.

Each year usually a few Finnish students from the department complement their studies at other European universities. The Department of Computer Science is involved in the ERASMUS Inter–University Coordination Program, financed by the Commission of the European Communities. Through this program it is possible for students of the department to study at the Technical University of Darmstadt. The department also accepts visiting students from the Technical University of Darmstadt. Exchange of students within the Nordic countries is also easy with the financial support of NORDPLUS, a program coordinated by the Nordic Council of Ministers.

Post–graduate studies at the Department of Computer Science can usually be accomplished in English. The department may admit foreign post–graduate students who have completed their Bachelor or Master of Science degree to work on any research areas connected with ongoing projects. The Academy of Finland and the Ministry of Education and Science offer grants for foreign post–graduate students within the international exchange programs.

Research cooperation

The department has active contacts with many European and American Computer Science Departments. Faculty members have joint research efforts with individual researchers from abroad. The results of this international cooperation are partly presented in the publication list. Many faculty members have spent long periods of time abroad as visiting researchers or guest professors. In turn, apart from more than 20 short term visitors every year, the department has hosted several visiting foreign scholars. A listing of long term visitors in 1991–93 is given below.

The department has actively taken part in the European research cooperation. COST 14 Programme "CSCW: Computer Supported Cooperation Work" has faculty members in the working groups "WG: Collaborative Information Spaces" (Hannu Erkiö, 1990–) and "WG7: Communication and Distributed System Support for CSCW" (Henry Tirri, 1990–). The department will be participating in COST 247 project Formal methods in Communication Protocol Design. The department also participates in the ESPRIT Programme in several projects. The largest of these are an industrial project "VITAL – A methodology based workbench for KBS Life Cycle Support (ESPRIT II P5365)" (Heikki Mannila, 1991–1995) and "TransCoop – Transaction Management Support for Cooperative Applications (ESPRIT III P8012)" (Henry Tirri, 1994–1997). In addition the department participates in cooperative research activities as a member institution in the working group "NEUROCOLT –Neural and Computational Learning (ESPRIT III P8556)" (Esko Ukkonen 1994–1996) and in a network of excellence "NEURONET – Network of excellence in Neural Networks (ESPRIT III P8961)" (Henry Tirri, 1994–1996).

Faculty members abroad in 1991–93

Patrik Floréen, Rijksuniversiteit Utrecht, The Netherlands
August 1993 – January 1994

Gösta Grahne, University of Toronto, Canada
August 1989 – July 1992

Matti Hiltunen, University of Arizona, Tucson, U.S.A.
from August 1990

Roope Kaivola, University of Edinburgh, Great Britain
from October 1992

Teemu Kerola, IBM, Austin, Texas, U.S.A.
August 1990 – August 1991

Pekka Kilpeläinen, University of Waterloo, Canada
from August 1993

Jyrki Kivinen, University of California at Santa Cruz, U.S.A.
from August 1993

Heikki Lokki, University of Missouri, St. Louis, and Webster University, St. Louis, Missouri, U.S.A.
from July 1993

Heikki Mannila, Technische Universität Wien, Austria
March – June 1993

Otto Nurmi, Universität Freiburg, Germany
October 1992 – February 1993

Pekka Orponen, Centrum voor Wiskunde en Informatica, The Netherlands
May – June 1993

Jorma Tarhio, University of California at Berkeley, U.S.A.
from August 1993

Henry Tirri, Purdue University, U.S.A.
July 1990 – July 1992

Esko Ukkonen, Universität Freiburg, Germany
August 1990 – July 1991

Foreign visitors in 1991–93

M.Sc. Andrejs Auzins, University of Riga, Latvia
October 1990 – June 1991

M.Sc. Bruce Barker, U.S.A.
January – July 1993

Magistère Patrice Calegari, Ecole Normale Supérieure de Lyon, France
June – August 1993

Dr. Thomas Eiter, Technische Universität Wien,
March – June 1991

Magistère Anne Fouilloux, Ecole Normale Supérieure de Lyon, France
June – September 1992

M.Sc. Jürgen Eckerle,
July 1993

M.Sc. Quan Gu, Suzhou University, China
January – December 1991

Dr.rer.nat. Alois Heinz, Universität Freiburg, Germany
August – October 1993

Dr. Shmuel Katz, Technion, Israel Institute of Technology, Israel
September – December 1992

B.Sc. David Nespoli, U.S.A.
August – December 1993

M.Sc. Cheng Qi, China
from October 1993

Serge Santos, Eidgenössische Technische Hochschule Zürich, Switzerland
from November 1993

Pascal de Vink, Rijksuniversiteit Utrecht, The Netherlands
January – March 1992

B.Sc. Curt Taylor, U.S.A.
August 1990 – December 1992

Other activities

Faculty members regularly serve as referees for international computer science journals and conferences. Many faculty members have also reviewers for various review publications, such as Computing Reviews and Mathematical Reviews. Prof. Martti Tienari has been a member of the editorial board of the scientific journal BIT from 1972 to 1993. Prof. Esko Ukkonen is the editor–in–chief of the Nordic Journal of Computing. Associate prof. Matti Mäkelä is an international coordinator of ACM SIGNUM Newsletter.

The department organized the Third Scandinavian Workshop on Algorithm Theory (SWAT'92), held on July 8–10, 1992, in Helsinki. The conference proceedings was edited by Otto Nurmi and Esko Ukkonen and it was published by Springer–Verlag in the Lecture Notes in Computer Science Series.

Other professional activities of the faculty members during 1991–93 include the following:

Gösta Grahne

Member of the program committee,
Second International Conference on Deductive and Object–Oriented Databases (DOOD '91),
Munich, Germany, 1991

Member of the Ph.D. examination committee for Mary–Anne Williams,
University of Sydney, Australia, 1994.

Member of the Program Committee,
Sixth Conference of Artificial Intelligence Research in Finland (STeP–94),
Turku, Finland, 1994.

Teemu Kerola

Member of the Program Committee,
Seventh International Conference on Modelling Techniques and Tools for Computer Performance Evaluation,
Vienna, Austria, 1994.

Heikki Mannila

Member of the Program Committee,
Third International Symposium on Programming Language Implementation and Logic Programming,
Passau, Germany, 1991.

Member of the Program Committee,
Fourth International Symposium on Programming Language Implementation and Logic Programming,
Leuven, Belgium, 1992.

Member of the Program Committee,
18th International Conference on Very Large Databases (VLDB '92),
Vancouver, Canada, 1992.

Official Opponent for the Ph.D. thesis of Viggo Kann, Kungliga Tekniska Högskolan,
May 1992.

External Reviewer for TfR,
(Sweden), 1992, 1993.

Otto Nurmi

Member of the Program Committee,
Workshop on Algorithms and Data Structures (WADS'93),
Montreal, Canada, 1993.

Pekka Orponen

External reviewer for the position of Lecturer in Computer Science,
Royal Institute of Technology, Stockholm, Sweden, September 1993.

Member of the Ph.D. examination committee for Jingsen Chen,
Lund university, Sweden, June 1993.

Member of the Program Committee,
21st Internat. Colloquium on Automata, Languages and Programming,
Jerusalem, July 1994.

Martti Tienari

Activities in IFIP (International Federation of Information Processing): General Assembly since 1987 , IFIP Trustee since 1989 , Cognizant Officer of TC2(Software:Theory and Practice) and TC12(Artificial Intelligence), Member of Publication Committee, and Chairman of the Activity Management Board since 1992.

External reviewer of two professorships at the Royal Institute of Technology,
Stockholm, Sweden, 1992.

Member of Program Committee,
PSTV'94 – The 14th International Symposium on Protocol Specification, Testing and Verification,
Vancouver, Canada, June 1994.

Henry Tirri

Member of 1993 European Community Cost 14 Project:
Computer Supported Cooperative Work, WG 7: Communication and Distributed System Support for CSCW.

Member of the Program Committee,
IEEE First International Workshop on Interoperability in Multidatabase Systems (IMS–91),
Kyoto, Japan,1991.

Member of the Program Committee,
18th International Conference on Very Large Databases (VLDB '92),
Vancouver, Canada, 1992.

Member of the Program Committee,
Research Issues in Data Engineering: Interoperability in Multidatabase Systems (RIDE–IMS '93),
Vienna, Austria, 1993.

Member of the Program Committee,
Applications of Databases (ADP–94),
Vadstena, Sweden, 1994.

Member of the Program Committee,
Second European Conference on Case–Based Reasoning (EWCBR '94),
Chambery, France, 1994.

Esko Ukkonen

Member of the Steering Committee of the Scandinavian Workshop on Algorithm Theory (SWAT).

Member of the Finnish–Estonian Joint Committee on Informatics.

External reviewer of professorships at Luleå University (1991 and 1994) and at the Royal Institute of Technology, Stockholm (1992).

External reviewer for NSF (Washington D.C., U.S.A.), 1991– ,
and for TFR (Sweden), 1993– .

Chairman of the Program and Organization Committees,
Third Scandinavian Workshop on Algorithm Theory (SWAT'92),
Helsinki, 1992.

Member of the Program Committee,
Third Annual Symposium on Combinatorial Pattern Matching (CPM'92),
Tucson, Arizona, 1992.

Member of the Program Committee,
First European Symposium on Algorithms (ESA'93),
Bad Honnef, Germany, 1993.

Member of the Program Committee,
Fourth International Workshop on Algorithmic Learning Theory (ALT'93),
Tokyo, Japan, 1993.

Member of the Ph.D. examination Committee for Roberto Grossi,
University of Pisa, Italy, 1993.

Member of the Program Committee,
Fifth Annual Symposium on Combinatorial Pattern Matching (CPM'94).
Asilomar, California, U.S.A., 1994.

Member of the Program Committee,
Fourth Scandinavian Workshop on Algorithm Theory (SWAT'94).
Denmark, 1994.

Member of the Program Committee,
Fifth International Workshop on Algorithmic Learning Theory (ALT'94),
Leipzig, Germany, 1994.

5. Library

The department maintains a library with large collections of literature in Computer Science. The library is managed jointly with the University Computing Centre and is mainly used by the staff and advanced students of the department.

Established in 1967, the library now holds more than 40,000 volumes of literature. The annual cumulation is about 2,000 monographic titles (books, reports) and 300 journal subscriptions. In Finland, the collections of this library are probably the best in the field of computer science.

The floor area of the library is 408 sq. meters including a reading room of 60 seats. Admission to the premises is free and collections are freely available to all visitors. Home loans, however, are restricted to the faculty and advanced students.

The library subscribes to most major international journals in computer science and acquires a majority of the most important computer science books and conference publications.

An important part of the collections consists of technical reports that are acquired as exchanges. The department maintains mutual exchange arrangements with about 130 research centers and university departments, in Europe and North America.

To help users search and locate the required literature, the library maintains an on–line database of its holdings. The database includes material acquired since 1982, classified according to the CR Classification System of the ACM. The database now contains nearly 30,000 bibliographic records, representing approximately 75% of the total holdings.

The library staff includes two full–time employees, one librarian and one secretary, who are assisted in their work by a library committee comprising of several members of the faculty.

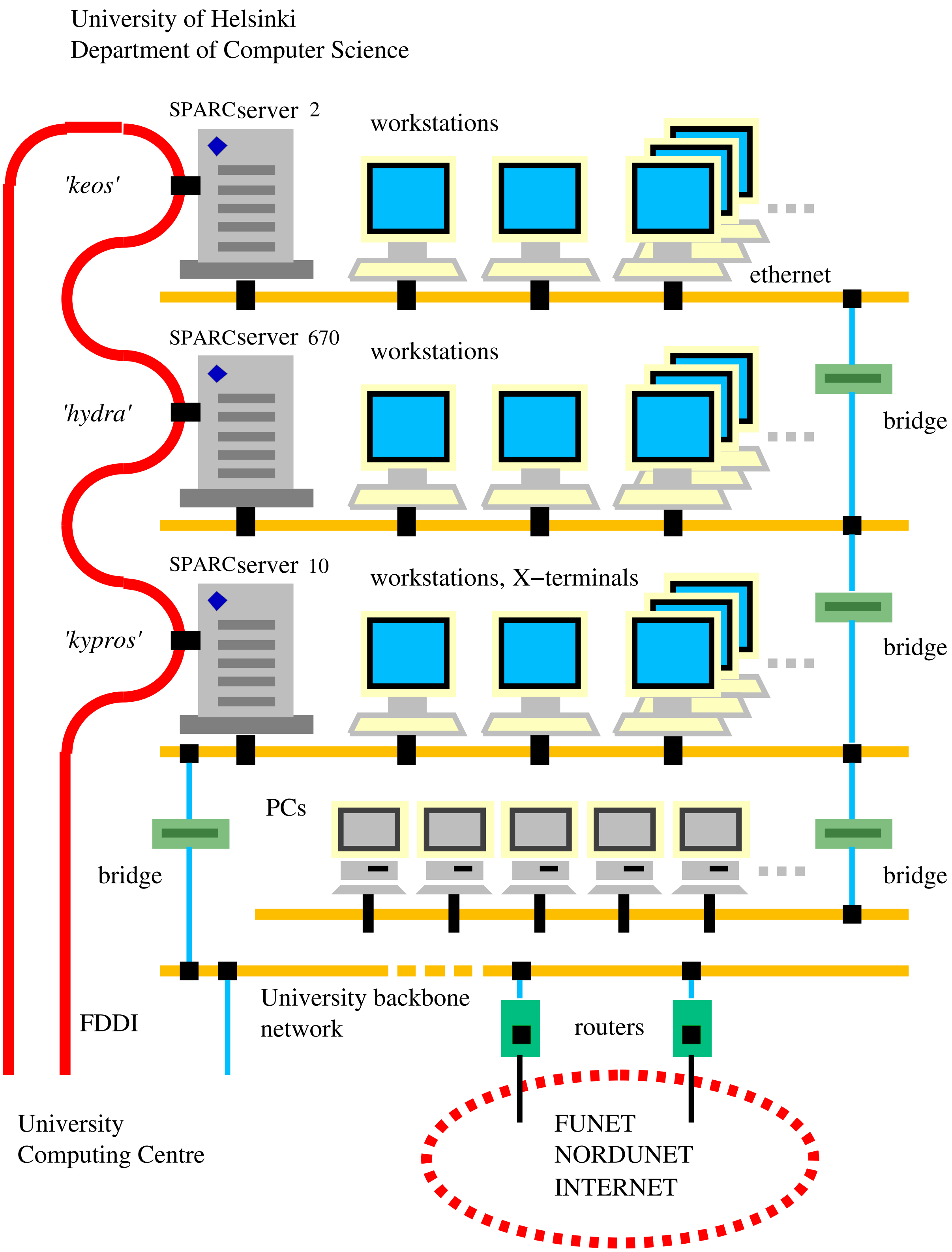
6. Computing facilities

The department offers a wide range of services to support computing activities of the academic staff and students. The policy is to provide access to advanced hardware and software systems.

The computing facilities include two general purpose computers, a dedicated file server, a network of workstations and a number of microcomputers. The departmental general purpose computers are a SPARCserver 670MP/512 and a SPARCserver 10/402; a SPARCserver 2 functions as a secondary file server. The total disk space is currently approximately 17 Gbytes. All these computers and the workstations run SunOS, the Sun version of the UNIX operating system. Together these systems support a wide variety of languages and software tools including electronic mail and news, graphics and visualization tools, several typesetting systems, and relational database systems.

The workstation network consists of about 40 SPARCstation workstations. The department also has a number of X−terminals. Networking is based on ethernet and CDDI (Copper Distributed Data Interface). Both Sun OpenWindows and pure MIT X11 window systems are in active use. The workstations are used as tools for software development, mainly in research and higher level teaching.

About 140 microcomputers are available, ranging from basic old IBM PC machines, through i286 and i386 types up to the heaviest i486 class. As the network solution, PC−NFS is used. The microcomputers are used mainly in introductory and intermediary level courses. Students use the MS−Windows environment, the dBase IV database management system, and Pascal as their primary programming language.



Schematic view of the department network

The network of the department is connected to the university backbone network, giving access to computers at the University Computing Centre as well as to the FUNET wide area network that links Finnish universities and research establishments. The computers operated by the Computing Centre include VAXes running under VMS and Sun, Solbourne and HP machines running under UNIX. Services provided by the Computing Centre include Oracle and Ingres database management systems, SAS statistical analysis package, NAG numerical library, and Pascal, Ada, and Prolog programming environments.

In addition, the department has access to Cray X−MP EA/432, Convex C3840, IBM 9076 SP1, and SGI Onyx VTX computers at the Centre for Scientific Computing within the Finnish State Computing Center.

The national FUNET network is further connected to the Nordic University Network, Nordunet. The Nordunet has a 1.5 Mbps terrestrial connection to the National Science Foundation network in the United States, so the department has a direct Internet connection.

7. Faculty

The members of the faculty are introduced by giving their position, research interests, and a sample of recent publications. E−mail addresses are also given.

All E−mail addresses are in the Internet format. In the domain "cs.Helsinki.FI" there is a general address format containing the first and the last name of a person separated with a period, for example

Hannu.Erkio@cs.Helsinki.FI

for Hannu Erkiö (note the necessary transliteration é−>e, ä−>a and ö−>o).

The publication numbers refer to the classified listing of publications in Section 2.3.

AHONEN, Helena, M.Sc., Research assistant
E−mail: Helena.Ahonen@cs.Helsinki.FI
Interests: text databases, machine learning, data mining, object−oriented databases
Publications: 150

ALANKO, Timo, Ph.D., Assistant professor
E−mail: Timo.Alanko@cs.Helsinki.FI
Interests: distributed operating systems, performance evaluation
Publications: 20, 40

BACK, Ralph−Johan, Ph.D., Docent, Professor (Åbo Akademi)
E−mail: backrj@abo.FI
Interests: program construction methods and tools, formal methods, programming language semantics, program verification, multiprocessor technology

ELOMAA, Tapio, Ph.Lic., Research assistant
E−mail: Tapio.Elomaa@cs.Helsinki.FI
Interests: machine learning, algorithms
Publications: 151−155

ELORANTA, Jaana, Ph.D., Research assistant
E−mail: Jaana.Eloranta@cs.Helsinki.FI
Interests: specification languages, modelling concurrency
Publications: 90–92

ERKIÖ, Hannu, Ph.D., Docent, Assistant professor
E−mail: Hannu.Erkio@cs.Helsinki.FI
Interests: user interfaces, computer−supported cooperative work
Publications: 141

FLORÉEN, Patrik, Ph.D., Research assistant
E−mail: Patrik.Floreen@cs.Helsinki.FI
Interests: neural networks, genetic algorithms, computational complexity
Publications: 2–4, 9, 11, 53–60, 143

GRAHNE, Gösta, Ph.D., Assistant professor
E−mail: Gosta.Grahne@cs.Helsinki.FI
Interests: data and knowledge base systems, relational structures, deduction, knowledge representation, belief revision theory
Publications: 107, 116, 119−122, 144−146

HILTUNEN, Matti, M.Sc., Research assistant
E−mail: Matti.Hiltunen@cs.Helsinki.FI, hiltunen@cs.arizona.edu
Interests: fault−tolerant systems, distributed systems, parallel computation
Publications: 21, 41

HOLSTI, Niklas, Ph.D., Laboratory engineer
E−mail: Niklas.Holsti@cs.Helsinki.FI

Interests: user interfaces, scientific computing, real time systems
Publications: 48, 139, 140, 153

JÄRVINEN, Pertti, Ph.D., Docent, Professor (University of Tampere)
Email: pj@cs.uta.FI
Interests: information systems, research methodologies, human−computer interaction, social effects of computing systems

KEROLA, Teemu, Ph.D. (Purdue), Assistant professor
E−mail: Teemu.Kerola@cs.Helsinki.FI
Interests: performance evaluation, data visualization, operating systems
Publications: 169, 170

KILPELÄINEN, Pekka, Ph.D., Research assistant
E−mail: Pekka.Kilpelainen@cs.Helsinki.FI
Interests: text databases, algorithms, logic programming
Publications: 26, 30, 67, 72–74, 117, 123

KIVINEN, Jyrki, Ph.D., Research assistant
E−mail: Jyrki.Kivinen@cs.Helsinki.FI
Interests: machine learning, computational learning theory
Publications: 136, 137, 154, 155, 157−161

KOSKIMIES, Kai, Ph.D., Docent, Associate professor (Univ. of Tampere)
E−mail: koskimie@cs.uta.FI
Interests: programming languages and environments, object−oriented systems

LAINE, Harri, Ph.Lic., Assistant professor
E−mail: Harri.Laine@cs.Helsinki.FI
Interests: computer aided software engineering (CASE), database design
Publications: 109

LINDÉN, Greger, Ph.Lic., Research assistant
E−mail: Greger.Linden@cs.Helsinki.FI
Interests: text processing (structured documents and dictionaries, incremental updates, transformations, transformation generators), database management
Publications: 39, 123

LINNAINMAA, Seppo, Ph.D., Docent, Research professor (Laboratory for Information Processing, Technical Research Centre of Finland)
E−mail: Seppo.Linnainmaa@vtt.FI
Interests: artificial intelligence, knowledge based techniques, digital image processing

LOKKI, Heikki, Ph. Lic., Assistant professor
E−mail: Heikki.Lokki@cs.Helsinki.FI
Interests: numerical analysis, analytical methods for ringing recovery
Publications: 1, 177, 178

MANNILA, Heikki, Ph.D., Professor
E−mail: Heikki.Mannila@cs.Helsinki.FI
Interests: data mining, database design, text databases, machine learning, algorithms
Publications: 26, 30, 66–68, 72–77, 108, 110–115, 117, 118, 123, 136–138, 150, 156, 159, 160, 164

MARTTINEN, Liisa, M.Sc., Research assistant
E−mail: Liisa.Marttinen@cs.Helsinki.FI
Interests: computer communication, computer networks, network management, open distributed processing

MÄKELÄ, Matti, Dr. Techn., Associate professor
E−mail: Matti.Makela@cs.Helsinki.FI
Interests: numerical methods, mathematical software, computers in education, computer graphics, scientific visualization
Publications: 180–182

NURMI, Otto, Dr.rer.pol., Assistant professor
E−mail: Otto.Nurmi@cs.Helsinki.FI
Interests: databases, algorithms, data structures, computational geometry
Publications: 32, 47, 49, 51, 52, 69, 78, 124

ORPONEN, Pekka, Ph.D., Assistant professor
E−mail: Pekka.Orponen@cs.Helsinki.FI
Interests: computational complexity, complexity in neural networks, programming methods for neural networks
Publications: 3, 4, 9–11, 54, 56, 58, 61–65, 142, 143, 147, 167, 168

PAAKKI, Jukka, Ph.D., Docent, Associate professor (University of Jyväskylä)
E−mail: Jukka.Paakki@cs.Helsinki.FI
Interests: programming languages, language processors, logic programming, program maintenance, protocol engineering
Publications: 15, 17, 25, 27−29, 31−36, 101

PELTOLA, Eero, Ph.D., Docent, Research Professor (Technical Research Centre of Finland, Information Technology, Information Systems)
E−mail: Eero.Peltola@vtt.FI
Interests: software business management, information processing management, information systems, database management, analysis and design of algorithms, searching and sorting

PUUSTJÄRVI, Juha, Ph.Lic., Research assistant
E−mail: Juha.Puustjarvi@cs.Helsinki.FI
Interests: concurrency control, multidatabases
Publications: 128

RAATIKAINEN, Kimmo, Ph.D., Docent, Assistant professor
E−mail: Kimmo.Raatikainen@cs.Helsinki.FI
Interests: performance evaluation of distributed systems, teletraffic, modelling and simulation
Publications: 14, 42−44, 171−176

RÄIHÄ, Kari−Jouko, Ph.D., Docent, Professor (University of Tampere)
E−mail: kjr@cs.uta.FI
Interests: human−computer interaction, databases, algorithms

SIPPU, Seppo, Ph.D., Associate professor
E−mail: Seppo.Sippu@cs.Helsinki.FI
Interests: logic databases, query processing, formal languages and parsing
Publications: 32, 102, 103, 122, 124, 129−132

SOISALON−SOININEN, Eljas, Ph.D., Associate professor (until August 1993), Professor (Helsinki University of Technology)
E−mail: Eljas.Soisalon−Soininen@cs.Helsinki.FI
Interests: database management systems, concurrency control and recovery, logic databases, complex query optimization, algorithms and data structures,

formal languages and parsing
Publications: 47, 49, 102–104, 122, 124–126, 129–132

TARHIO, Jorma, Ph.D., Docent, Assistant professor
E–mail: Tarhio@cs.Helsinki.FI
Interests: algorithms and data structures, string algorithms, data compression, attribute grammars
Publications: 37, 50, 70, 79–82, 99, 100, 104–106

TIENARI, Martti, Ph.D., Professor, Chairman of the Department
E–mail: Martti.Tienari@cs.Helsinki.FI
Interests: concurrency, computer networks and distributed systems, programming languages and compilers, management of computing
Publications: 13, 18–20, 38, 97, 98, 179

TIRRI, Henry, M.Sc., Sr Research Scientist
E–mail: Henry.Tirri@cs.Helsinki.FI
Interests: transaction processing in databases (especially open, heterogeneous environments), mobile computing, database concurrency control, distributed object–oriented systems, neural computing, theory of machine learning, knowledge based systems
Publications: 3–6, 9, 11, 12, 23, 39, 127, 133–135, 143, 148, 162, 163, 165, 166

UKKONEN, Esko, Ph.D., Professor
E–mail: Esko.Ukkonen@cs.Helsinki.FI
Interests: algorithms and data structures, stringology, machine learning, bioinformatics
Publications: 30, 52, 70, 71, 75, 76, 79, 81–89, 121, 156, 159, 160

VALMARI, Antti, Dr.Techn., Docent, Associate professor (Tampere University of Technology)
E–mail: ava@cs.tut.FI
Interests: verification and validation of concurrent and distributed systems

VERKAMO, A. Inkeri, Ph.D., Assistant professor
E–mail: Inkeri.Verkamo@cs.Helsinki.FI
Interests: software performance evaluation, software engineering

Publications: 45−46

VIHAVAINEN, Juha, Ph.Lic., Assistant professor
E−mail: Juha.Vihavainen@cs.Helsinki.FI
Interests: principles and implementation of programming languages, language processors, programming paradigms, multi−paradigm programming, object−oriented programming
Publications: 24

VILO, Jaak, M.Sc. (University of Tartu), Research assistant
E−mail: vilo@cs.Helsinki.FI, Jaak.Vilo@Helsinki.FI
Interests: algorithms and data structures, machine learning, bioinformatics, stringology and text databases

8. Application information for new students

If you are interested in studying at the Department of Computer Science in the University of Helsinki, please contact the Study Office of the Faculty of Science

by mail:

Study Office, Faculty of Science
P.O. Box 3 (Fabianinkatu 33)
FIN−00014 University of Helsinki
Finland

by fax:

+358 0 191 2176

by electronic mail to Timo−Jussi Hämäläinen in the Study Office:

tjhamala@cc.helsinki.fi