# PART III

# PROJECT SUMMARIES

# Parallel Unification Machine Design and Simulation

**Contact Person:** Professor Fadi N. Sibai

Dept. of Electrical Engineering
University of Akron
Akron, OH 44325-3904
U.S.A.
mail: rlfns@vaxl.cc.uakron.edu

**Keywords**
Parallel Unification, Term Matching, Consistency Check, Shared and Distributed Memory Organizations

**Summary**
A parallel machine architecture for first-order unification was designed by F. N. Sibai. This machine can be used as a hardware unifier by itself, or can be used as a co-processor to a host PROLOG computer. The machine takes two terms to be unified for input, and outputs the most general unifier in case of unification success, or FAIL in case of failure. The machine is composed of an array of match processors (MPs) in charge of matching the arguments of the two terms in parallel and can therefore detect early failures at the match level. If no match failure exists, the MPs send the generated bindings to a consistency check processor (CCP) in charge of checking the consistency of the bindings. The CCP is a powerful unification processor which maintains the bindings in a content-addressable memory for fast search and retrieval. Thus the operations of the array of MPs and the CCP are pipelined in the machine. The performance evaluation of two versions of PUM has been conducted with simulation. The first version is based on a shared-memory organization whereas the second uses a distributed memory organization.

**Reference**
Sibai, Fadi N.,"A Parallel Machine for the Unification Algorithm: Design, Simulation, and Performance Evaluation," Ph.D. Dissertation, Dept. of Electrical Engineering, Texas A&M University, College Station, Texas 77843, U.S.A., December 1989.
Sibai, F.N., Watson, K. L., and Lu, Mi, "A Parallel Unification Machine," IEEE MICRO, Vol. 10, No. 4, August 1990, pp. 21-33.

# Parallel Rule-Firing Production Systems

**Contact Person:** Daniel Neiman

Computer Science Dept.
University of Massachusetts
Amherst, MA 01003
mail: dann@cs.umass.edu

**Keywords**
Parallel Rule-Based Systems, Parallel OPS5

**Summary**
At the University of Massachusetts, we have developed a parallel rule-firing production systems based on OPS5. We are using this system to investigate the control and correctness issues involved in firing rules in parallel. In particular, we have developed an asynchronous rule-firing policy and are studying its implications upon the design and performance of parallel heuristic programs.

# Parallel Closure-Based Automated Reasoning

**Contact Persons:** Ewing Lusk and William McCune*

John Slaney**

*Mathematics and Computer
Science Division, Argonne National Laboratory, Argonne
Illinois 60439, USA

**Automated Reasoning Project, Australian
National University, Canberra ACT, Australia

**Keywords**
Parallel Theorem Proving, OTTER, ROO

**Summary**
This project's goal is to exploit high-performance parallel machines for classical automated theorem proving. We have adapted OTTER, a fast sequential theorem prover developed at Argonne National laboratory, to a class of parallel machines via a parallel closure algorithm. The result is ROO, a parallel theorem prover completely compatible with OTTER that runs on shared-memory multiprocessors such a the Sequent Symmetry. It obtains near-linear speedups on many problems, but performs less well on others. Research is continuing on ways to increase levels of parallelism on such problems, and on algorithms for closure-based distributed-memory algorithms.

# Parallel Completion

**Contact Person:** Prof. Katherine Yelick

505 Evans Hall
Computer Science Division
University of California
Berkeley, CA 94720
internet: yelick@cs.berkeley.edu

**Keywords**
Term Rewriting, Knuth-Bendix Procedure, Completion, Proof Orderings

**Summary**
We have designed and implemented a parallel completion procedure for term rewriting systems, a problem for which the Knuth-Bendix procedure is a well-known sequential solution. Straightforward parallelization of the Knuth-Bendix procedure did not perform well in our experiments, because the outermost loop contains unnecessary serialization. Instead, our parallel procedure is a refinement of inference rules given by Bachmair, Dershowitz and Hsiang. The challenging part of this refinement was the definition of a bookkeeping mechanism that: 1) guaranteed the required liveness property, 2) kept the rewriting system small and inter-normalized, and 3) had low computational overhead and no synchronization bottlenecks. The current implementation runs on the Firefly shared-memory multiprocessor and exhibits nearly linear speedup on some inputs, including problems in group theory. We are currently investigating ports to machines with more processors and physically distributed memory.

# Parallel Logic Programs on Transputers

**Contact Person:** Roman Blasko, PhD.

Institute of Computer Systems
Dubravska cesta 9
842 37 Bratislava
Czechoslovakia

**Keywords**
Logic Programming, Parallel Implementation, Transputer Network

**Summary**
It is a project at the Institute of Computer Systems. The project is planned for 2-3 years. Within the scope of the project the parallel implementation of the logic programming language FCP(:,7) (Flat Concurrent Prolog) is realized. As a first step, we work out a sequential computational model for this language. We implement this model on a transputer. This sequential implementation we want to utilize for the analysis of the needed run-time support for systems. In the next period of work, we want to work out a parallel computational model of FCP(:,7) for distributed memory multiprocessors, i.e. transputer network. This model will utilize OR-parallelism and AND-parallelism among predicates, when the predicates have not any shared variables. We work out the compilation and optimization rules for this model. The last step of this project will be the implementation and performance analysis of the system on a multitransputer network.

# Parallel Distributed Belief Networks

**Contact Person:** Wilson X. Wen

AI Systems
Telecom Research Labs.
770 Blackburn Rd.
Clayton, Vic 3168
Australia
mail: w.wen@trl.oz.au
Tel: (61-3) 541-6273
Fax: (61-3) 541-6173

**Keywords**
Belief Networks, Information Theory, Minimum Cross Entropy, Markov Fields, Boltzmann-Jeffrey Machine Networks, Jeffrey's Rule, Encore Computer

**Summary**
A parallel distributed computational model, Boltzmann-Jeffrey Machine Network (BJMNet), for Minimum Cross Entropy (MCE) reasoning is proposed based on information theory and the theory of Markov fields. Each autonomous component in BJMNet is a hypercube of parallel processors called Boltzmann- Jeffrey Machine (BJM). Each BJM in a BJMNet has its own prior distribution and updates its distribution according to Jeffrey's rule, which is a special case of MCE principle, if the distributions in its intersections with other BJMs are changed. The belief change is propagated throughout the whole BJMNet until an equilibrium is finally reached.

A BJMNet simulator has been implemented for Encore Computer to do parallel reasoning and to simulate the corresponding hardware architecture.

# The IFS Parallel Architectures Group University of Essex

**Contact Person:** Simon Lavington

University of Essex
Department of Computer Science
Colchester CO4 3SQ
mail: lavington@uk.ac.essex
phone:(+44)206 872 677
fax: (+44)206 872 788

**Keywords**
Novel Hardware for Symbolic Computation, Associative Memory, Non-WAM Computational Models

**Summary**
Our main line of research is to investigate ways in which novel parallel hardware can support non-numeric (eg symbolic) computer applications. The overall aim is to speed up run times and reduce software complexity. The principal applications areas are smart information systems, eg knowledge bases, which may draw on a range of techniques from relational DBMS, logic programming, AI paradigms, etc. Our research interests cover a variety of topics from theory to technology, as described below.
The IFS group currently consists fo six academic staff, three senior research officers, two technicians, and two Ph.D. students. Sources of suport include SERC, DTI, and ICL. The group has been continually funded since October 1983.
A tangible result of our research has been the design and implementation of a knowledge-base server, the IFS/1, which was first delivered to an external site in December 1987. The IFS/1 won a British Computer Society Silver Medal for Technical Achievement. We are now working on the successor, the IFS/2. This is a kind of *structure store*, or add-on *active memory* unit, which both stores and manipulates objects such as sets and graphs. The IFS/2 carries out whole-structure operations.
Current research interests include:

1. Knowledge representation formalisms, for example extended semantic networks, with the aim of providing expressive power and the potential for parallel implementation;

2. Non-WAM computational models for logic programming, for example those based on the matrix or connection methods of Prawitz and Bibel, with the aim of providing an alternative basis for exploiting parallelism;

3. Novel representations of production rule systems, for example based on associative tables, with the aim of speeding up the matching phase;

4. Extensions to relational algebra, for example transitive closure and other graph primitives, with the aim of handling linear recursive queries in deductive databases;

5. New strategies for Constraint Satisfaction Systems, with the aim of reducing computational effort;

6. Connectionist models of learning, for example those making use of parallel nearness-matching algorithms, with the aim of reducing run times;

7. SIMD techniques for organising value-comparisons in set and graph operations, with the aim of exploiting natural data parallelism;

8. Memory management in large associative (ie content-addressable) storage hierarchies, for example by semantic caching, with the aim of providing easy protection and manipulation of variable-granularity objects;

9. Use of transputers for distributed control of SIMD data-manipulation, with the aim of providing flexible, modularly-extensible, exploitation of the natural parallelism in AI-related data types.

A recent outgrowth of the IFS Group is PACE: Parallel Associative Combinator Evaluation. PACE is a new computational model for SK combinators, which offers effective modular parallelism for functional languages. The intention is to integrate structure stores such as the IFS/2 into the PACE graph reduction framework.

# METEORs: High Performance Theorem Provers Using Model Elimination

**Contact Person:** Owen Astrachan

Duke University
Dept. of Computer Science
Durham, NC 27706 USA
mail: ola@cs.duke.edu
phone: (919)-660-6522

**Keywords**
Parallel/Distributed Inference, Caching, Model Elimination

**Summary**
Our Model Elimination theorem prover compiles clauses into a data structure that is interpreted at run time by either a sequential machine (currently any UNIX machine with a C compiler; we have tested the machine on Suns, Decstations, and HP workstations), a parallel machine (currently a Butterfly GP1000 or TC2000), or a network of sequential machines (currently sun workstations).
We have investigated methods for reducing redundancy using caching techniques and applied these methods to yield performance gains for a significant class of problems.

# ESCAPE: Expert System Compilation and Parallelization Environment

**Contact Persons:** Robert Chun, Brad Perry, Steve Birminghan

Hughes Aircraft Company
P.O. Box 3310
Fullerton, CA 92634
mail: rchun%atr-2s.hac.com@hac2arpa.hac.com

**Keywords**
Ada, Compilation, Expert System, Parallel Processing, Real Time

**Summary**
ESCAPE represents a synergistic approach utilizing compilation, compaction, and parallelization to achieve real-time computing throughput from rule-based expert systems. The methodology involves synthesizing a set of concurrently executable Ada tasks from a knowledge base of rules. Compaction of code size is accomplished by eliminating the overhead associated with inference engine control constructs not utilized by a particular knowledge base. Heuristics are used to customize the generated Ada code for optimum performance gains given the characteristics of the source knowledge base and target processor. The effectiveness of this approach depends on both the characteristics of the knowledge base and the efficiency of the Ada compiler's task invocation mechanism. A prototype compilation system based on this multifaceted approach has demonstrated speedups in excess of 100X for certain knowledge bases, as well as, additional benefits in terms of increased embeddability and maintainability of the knowledge base.

# Data Parallelism in Logic Programming

**Contact Person:** Giancarlo Succi

DIST, Universita' di Genova
Via Opera Pia 11a
I-16145 Genova, Italia
Tel: +39 10 353 2747 / +39 10 353 2750
Fax: +39 10 353 2948
E-mail: charmi@dist.unige.it

**Keywords**
Data Parallel Implementation of Logic Languages, Set-Oriented Abstract Machine, Abstract Analysis - Object-Size Analysis, Persistency Analysis -

**Keywords**
This project is focussed on the parallel implementation of a set-based logic language (SEL = Subset Equational Language). A data-parallel "approach" (targeted to the CM) is taken, nevertheless a competitive and conservative process parallel implementation is also under design (for a transputer implementation), in order to have, as a long term goal, an integrated framework. SAM (Subset Abstract Machine), the abstract machine under development, belongs to the WAM family, and it is augmented with several abstract analyzers placed at different levels of the "compilation flow". Being SEL a set-based logic language, we think that our data parallel SEL implementation will be a very efficient way of implementing inference systems.

People involved in the project:
Diego Co', Giancarlo Colla, Joy Marino, Sergio Novella, Amedeo Pata, Alex Regoli, Giancarlo Succi, Luca Vigano'.

# Distributed Logic Programming

**Contact Person:** Nissim Francez

Technion
CS department
Haifa 32000
Israel
e-mail: francez@cs.technion.ac.il

**Keywords**
Prolog, Logic Programming, Distributed System, SDP

**Summary**
We develop a notion of sequential and distributed execution of Prolog programs. The setup is a network of nodes interconnected on a broadcast network. Each node has a certain collection of Horn-clauses, and nodes cooperate to answer queries (arriving at ANY node, possibly in overlapped manner) in the same way that a central Prolog interpreter would answer.
The main idea is to use local (finite) failure as a synchronization primitive. NOT allowed to communicate rules/facts, only subqueries. The implementation was on an Ethernet with SUN2s, and is planned to be moved to SUN4s.

**Reference**
Amir Rahat, Nissim Francez, Oded Shmueli: "SDP: Sequential, Distributed Logic Programming", in: Declarative systems (G. David, R.T. Boute, and B.D. Shriver, eds.), North-Holland, 1990 (An IFIP conference).

# Programming Methods for Neural Computing (NEULOG)

**Contact Persons:** Henry Tirri (project manager), Petri Myllymki, Pekka Orponen, Patrik Floreen

University of Helsinki
Department of Computer Science
Teollisuuskatu 23
SF-00510 Helsinki, Finland
mail: tirri@cs.Helsinki.FI
Fax: + 358 - 0 - 708 4441
phone: +358-0-7084226

**Keywords**
Hybrid Systems, Bayesian Reasoning, Connectionism

**Summary**
NEULOG was a 3-year project, which started in 1988 as part of the FINSOFT-program organized by the Finnish Technology Development Centre (TEKES). The purpose of this project was to develop new, truly high level programming methods for neural computing, particularly in view of applications in symbolic computation (expert systems, probabilistic reasoning). The concrete result of the project is a hybrid neural-symbolic programming environment, consisting of a high-level NEULA-language and a compiler that is capable of realizing the symbolic syntax as a neural net. The system is provided with a novel computational mechanism, which efficiently approximates Bayesian reasoning.

# The MUSE Parallel Prolog System

**Contact Persons:** Khayri A.M. Ali and Roland Karlsson

Swedish Institute of
Computer Science, SICS
Box 1263
S - 164 28 Kista, Sweden
mail: khayri@sics.se and roland@sics.se

**Keywords**
Full Prolog, Or-Parallelism, Multiprocessors

**Summary**
Today, multiprocessor machines are commercially available. However, writing programs that effectively utilize processing power of such machines is not an easy task. Ideally, programming of multiprocessors should not be harder than sequential ones. The programmer should consider the machine as a black-box and parallelism in a program should be detected and exploited by the system. Additionally, running a program on multiprocessors should not be slower than on uniprocessor machine. Although this seems to be a difficult task, in the field of symbolic computing and knowledge based applications, we are reaching that goal for Prolog programs. At SICS we have developed a system called MUSE that largely achieves that goal.
MUSE is an Or-parallel implementation of the full Prolog language. It is currently run on a number of shared memory multiprocessor machines, e.g., TP881V from Tadpole Technology, Sequent Symmetry, and the BBN Butterfly I and II. The sequential SICStus Prolog, a fast, portable system, has been adapted to Or-parallel implementation. The extra overhead associated with this adaptation is very low, around 3 - 5%. The speedup factor is very close to the number of processors for a large class of problems.

# Parallel, Concurrent Theorem Proving

**Contact Person:** Robert Johnson

Computer Science Department
Queen Mary and Westfield college
University of London
Mile End Road
London E1 4NS
mail: robj@dcs.qmw.ac.uk
tel: 071-975-5259
fax: 081-980-6533

**Keywords**
Parallel, Concurrency, Tableaux, Connection Method, Theorem-Proving

**Summary**
My research principally involves the parallelisation of the tableaux approach to theorem proving, and the development and evaluation of parallel, concurrent systems. I am currently using Strand over a distributed Sun/Sparc network but aim to utilise other hardware paradigms as Strand is ported to them. I have a concurrent propositional tableaux system completed, and a concurrent connection method system for comparison. Future work will involve the development of parallel, concurrent systems for modal logics.

# ROBIN: Massively Parallel Inferencing and Disambiguation in Structured Connectionist Networks

**Contact Person:** Trent E. Lange

Artificial Intelligence Laboratory
Computer Science Department
University of California, Los Angeles
Los Angeles, CA 90024
e-mail: lange@cs.ucla.edu

**Keywords**
Structured Connectionist Networks, Spreading-Activation Networks, Inferencing, Disambiguation, Natural Language Understanding, ROBIN

**Summary**
Structured connectionist (or spreading-activation) networks offer substantial promise for the problems of reasoning and natural language understanding because of their inherent parallelism and constraint satisfaction abilities. Unfortunately, like most types of connectionist models, they have been limited because of the difficulty they have had in representing variable bindings and performing rule-based reasoning. ROBIN (ROle Binding and Inferencing Network) is an ongoing research project to solve some of these problems. By using uniquely-identifying patterns of activation called "signatures" to represent variable bindings (Lange & Dyer, 1989) and propagating them across the network to perform dynamic inferencing, ROBIN has shown how structured connectionist networks can utilize simple general knowledge rules to infer a plan/goal analysis of the input for short natural language texts. Most importantly, because ROBIN's networks retain the normal constraint satisfaction abilities of spreading-activation networks, they are able to disambiguate between multiple inference paths instantiated in parallel to select the best interpretation in a given context for ambiguous texts that are difficult for traditional symbolic systems (Lange, in press). Research in ROBIN continues to find ways allow its networks to handle more complex rules and hold more dynamic knowledge so that it can reason with and understand longer stories while retaining its massively-parallel inferencing and disambiguation abilities.

**Reference**

Lange, T. & Dyer, M. G. (1989). High-Level Inferencing in a Connectionist Network. Connection Science, 1 (2): 181-217, 1989.

Lange, T. (in press). Lexical and Pragmatic Disambiguation and Reinterpretation in Connectionist Networks. To appear in the International Journal of Man-Machine Studies.

# Parallel Production Systems

**Contact Person:** James G. Schmolze

Dept. of Computer Science
Tufts University
Medford, MA 02155 USA
Phone: 617/381-3681
mail: schmolze@cs.tufts.edu

**Keywords**
Parallel Production Systems, Multiple Rule Firing Systems, Parallel Forward Chaining Systems, Asynchronous Parallel Rule Firing

**Summary**
To speed up production systems, researchers have studied how to execute many rules simultaneously. Unfortunately, such systems can yield results that are impossible for a serial system to produce, leading to erroneous behaviors. We have devised a formal solution to the problem of guaranteeing serializable behavior in parallel production systems that execute many rules simultaneously, and we have a variety of algorithms that implement this solution. Some algorithms are targeted primarily for synchronous execution on a shared-memory multiprocessor, and we have an initial implementation of these algorithms that runs on the Sequent or Encore multiprocessors in Top Level Common Lisp. Another algorithm is targeted for asynchronous execution on a distributed-memory, message-passing machine, and an implementation is under construction for the NCube hypercube.

# ElipSys

**Contact Persons:** Michel Dorochevsky, Kees Schuerman, André Véron, Jiyang Xu

European Computer-Industry Research Centre
Parallel and Distributed Systems Group
Arabellastr. 17
D-8000 München 81
mail: elipsys@ecrc.de
mail: michel@ecrc.de
Tel. + (49) 89-92699-107
Fax. + (49) 89-92699-170

**Keywords**
Logic Programming, Parallelism, Constraints Satisfaction, Database,
Shared/Distributed Memory

**Summary**
ElipSys is a logic programming system being developed at ECRC. It combines the three major technologies required to build large decision support systems: constraint satisfaction, tight database coupling and parallel evaluation.

The project addresses issues at all levels; from language design to implementation. Of particular note is the work to improve the expressiveness and declarativeness of logic programming, and the design of an execution model that is appropriate to a range of hardware platforms, from shared to distributed memory.

A prototype version of ElipSys is running on a 12 processor Sequent Symmetry and a network of SUNs running the MACH operating system. Prototypes of a number of commercial applications have been implemented on ElipSys.

# Project PARIS: Parallelisation of Inference Systems

**Contact Persons:** Wolfgang Ertel and Christian Suttner

Institut für Informatik
Technische Universität München
Augustenstraße 46 Rgb.
D-8000 München 2
Tel.: +49-89/521097
email: <name>@informatik.tu-muenchen.de

**Keywords**
High Performance Parallel Theorem Proving, Or-Parallelism,
Massive Parallelism

**Summary**
The potential for parallelism in automated theorem proving as well as in logic programming is very high. Exploiting this potential efficiently is the main goal of this project.
The starting point for our work was SETHEO, a high performance model elimination theorem prover for first order logic (a free copy of the SETHEO-sources (C,Unix) is available from the contact persons). Based upon SETHEO we developed the parallel theorem prover PARTHEO which runs on transputers as well as on the Intel iPSC/II machine with very good efficiency.
Currently we are aiming at an automated reasoning system with at least four orders of magnitude shorter run-times compared with existing systems. With better proof calculi and heuristic search methods we will achieve drastic search space reductions. Together with scalable parallel implementations on massively parallel architectures this will allow us to solve realistic reasoning problems. We are working on a very flexible easily portable parallel execution mechanism which allows the efficient parallelisation of large classes of arbitrary inference systems with very little effort. This will enable us to exploit the large computational power of heterogeneous networks of powerful computers.

# Experiments with Parallel Software Architectures for Information Filtering: Trellis and FGP

**Contact Persons:** Scott Fertig and David Gelernter

Yale University
Department of Computer Science
New Haven, Connecticut 06520-2158
US mail: Yale University Department of Computer Science
P.O. Box 2158 Yale Station
New Haven, CT 06520-2158
email: Fertig-Scott@cs.yale.edu

**Keywords**
Information Filtering, Realtime Control, Intelligent Databases, Software Architecture, Learning, Linda, Medical Informatics

**Summary**
"Information Filtering" is an area of decisive importance for the future of computer science. The proliferation of electronic data-gathering and data-recording devices has created two related, enormous problem-opportunities. First, data that might be relevant to the realtime control of machinery or organizations, or the "management of situations" in general, emerge too rapidly for humans to respond. Second, data that might contain valuable information in the aggregate collect in pools too gigantic to be manageable. Advances in information-filtering technology are required in both cases. Accordingly, the information filtering problem encompasses in our view (at least) two different but related problems. (1) How to squeeze knowledge out of a collection of realtime data streams—potentially a large and diverse collection of fast-moving streams; (2) how to squeeze knowledge out of a potentially enormous database. "Knowledge" refers to integrated, high-level, problem-specific information—"big-picture" information as opposed to low-level, undigested data. Our group at Yale has developed and implemented two experimental software architectures, one for each of these two sub-problems. Both systems take aim at "high performance" in the same way: by using asynchronous parallelism to run complex, compute-intensive algorithms fast. Both use the Linda[1] coordination language, which is a portable high-level system for expressing asynchronous parallelism; systems built using Linda will run on any platform on which Linda is supported—generally speaking, on any shared- or distributed-memory asynchronous parallel machine, and on local area networks.

[1] a registered trademark of Scientific Computing Associates.

# Parallel Linear & UR-Deduction

**Contact Person:** Geoff Sutcliffe

Department of Computer Science
The University of Western Australia
Stirling Highway
Perth, Western Australia, 6009
Phone : (09) 380 2305
mail: geoff@cs.uwa.oz.au

**Keywords**
Parallel Theorem Proving

**Summary**
The deduction system developed in this project, called GLD–UR, has two deduction components. One component runs a chain format linear deduction system and the other a UR-deduction system. Lemmas created in each of the deduction components are passed to the other component. An extended version of GLD–UR, in which the lemmas created are distributed via a separate 'lemma control' component has also been developed. The speed-ups obtained in GLD–UR are largely due to cross-fertilisation between the deduction components. There is definite potential for superlinear speed-ups. GLD–UR is implemented in Prolog-Linda. Prolog-Linda provides the appropriate data transfer and synchronisation facilities for implementing parallel deduction systems. The implementation of GLD–UR is highly modular, and new deduction or other components can easily be added to the system. A hyperresolution deduction component will be the next component added.

# An Optimally Efficient, Limited Inference, Connectionist Rule-Based Reasoning System with an Included Type Hierarchy

**Contact Persons:** Lokendra Shastri, Venkat Ajjanagadde and D. R. Mani

Department of Computer & Information Science
University of Pennsylvania
Philadelphia, PA 19104, USA

**Keywords**
Connectionism, Knowledge Representation, Reasoning, Type Hierarchy

**Summary**
Humans draw a variety of inferences spontaneously and with remarkable efficiency as though they are a reflex response of their cognitive apparatus. This work is a step toward a computational account of this remarkable 'reflexive' reasoning ability. We develop a connectionist knowledge representation and reasoning system that performs a limited but interesting class of inferences over a restricted class of first-order sentences with optimal efficiency. The proposed system can answer queries in time proportional to the length of the shortest derivation of the query and is independent of the size of the knowledge base. Further, the space complexity of the system is just linear in the size of the knowledge base.

Reasoning requires the dynamic representation and propagation of variable bindings. The system does so by propagating rhythmic patterns of activity wherein bindings are represented as synchronous firing of appropriate nodes in a connectionist network. Rules are encoded by interconnection patterns that propagate and transform these rhythmic patterns of activity while facts act as pattern matchers.

Rule-based reasoning has been combined with reasoning about inheritance and classification by introducing a connectionist Type Hierarchy in the reasoning system. This enables the system to use generic facts ('Cats prey on birds') and qualified rules ('if an animate agent walks into a solid object then the agent gets hurt'), thereby extending the class of rules and facts the reasoning system can handle.

Work in progress concerns extending the system to model abductive reasoning where the plausibility of rules, the feasibility of making certain assumptions and the computational cost are all considered in generating the 'best' explanation for a given set of observed facts.

The project lays emphasis on the use of the connectionist paradigm, which in addition to modeling reflexive reasoning, attempts to explain how such a system may be realized using a network of neuron-like elements operating without a central controller. Neurophysiological evidence suggests that similar mechanisms, which depend on the synchronous firing of appropriate nodes, may in fact be used by the brain to represent and process sensorimotor information.

# Compile-Time Analysis of Concurrent Logic Programs for Multi-processors

**Contact Persons:** Andy King and Paul Soper

Dept. of Electronics and Computer Science
The University of Southampton
Highfield, Southampton, S09 5NH, UK
Phone: +22 (0)703 5930553
FAX: +44 703 593045
JANET: amk@uk.ac.soton.ecs
Bitnet: amk@ecs.soton.ac.uk
uucp: amk@sot-ecs.uucp

**Keywords**
Program Analysis, Concurrent Logic Language, Granularity Analysis, Schedule Analysis, Abstract Interpretation

**Summary**
A concurrent logic program naturally divides into processes which are, in principle, well-suited to distribution on a multi-processor. Performance can be impaired, however, if processes are either too fine-grained or too coarse-grained. An analysis of grain-size enables compile-time decisions to be made as to whether to distribute or coalesce processes, however, incurring unnecessary overheads. An analysis of the data-dependencies between coalesced processes reveals how processes can be scheduled at compile-time. This permits the removal of complex binding operations, a reduction in the enqueuing and dequeuing of processes, and can cut down the generation of garbage.

# qwertz

**Contact Persons:** Joachim Hertzberg, Hans Werner Güsgen

GMD, AI Research Division
Schloss Birlinghoven
D - 5205 St. Augustin

**Keywords**
Massively Parallel Constraint Satisfaction, Connectionist Networks, Boltzmann Machines

**Summary**
The goal of the qwertz project is to evaluate classical methods and techniques for AI planning. A subgoal in the project includes the development of massively parallel constraint satisfaction algorithms, which may be viewed as a special form of algorithms for limited inferences. Currently, two approaches are considered:

- A connectionist network (Feldman-Ballard network) which spreads activation in order to find a solution of a given constraint satisfaction problem.
- Boltzmann machines which view constraint satisfaction as optimization problems, finding "almost" solutions for possibly inconsistent problems.

# Parallel Implementation of Guarded Horn Clauses

**Contact Person:** Handong Wu

Department of Telecommunication and Computer Systems
The Royal Institute of Technology
S-100 44 Stockholm
Sweden
mail: wu@tds.kth.se

**Keywords**
Guarded Horn Clauses, Dataflow, Language Implementation

**Summary**
An execution model for Guarded Horn Clauses (GHC) has been developed based on the principles of Extended Dataflow Architecture (EDA). The execution model is able to explore both Shallow-Or-parallelism and Stream-And-parallelism inherent in GHC programs, which are encoded in the form of EDA graphs. The problem of a nested guard computation can be handled efficiently by the hierarchic structure of control frames at runtime. An EDA graph is of large-grained dataflow graph, and two mechanisms for communication and sychronization are provided: token passing and variable access. The variable operations are classified by local variables or global variables. A local variable belongs to a particular node and cannot be accessed from other nodes; while a global variable can be shared between nodes, and operations on it are restricted. The execution model has been simulated in SICStus Prolog and some preliminary results have been obtained.

**Reference**
Handong Wu (1990): "Extension of Dataflow Principles for Multiprocessing", TRITA-TCS-9004, The Royal Institute of Technology, S-100 44 Stockholm, Sweden, April 1990. (Ph.D thesis)

## A System for Distributed Simplification-Based Theorem Proving

**Contact Persons:** Maria Paola Bonacina and Jieh Hsiang

Department of Computer Science
SUNY at Stony Brook
Stony Brook, NY 11794-4400
email: bonacina,hsiang@sbcs.sunysb.edu

**Keywords**
Parallel Automated Deduction, Simplification-Based Strategies, Unfailing Knuth-Bendix Completion

**Summary**
Our project consists in the design and implementation of *simplification-based* strategies [?] for parallel automated deduction, on a multi-processor with *distributed memory*. For equational logic the basic inference mechanism is an enhanced version of Unfailing Knuth-Bendix completion. Ordered resolution, ordered paramodulation and several contraction inference rules will be featured for first order logic with equality. We plan to develop a first prototype of our system, for equational logic, on an Intel hypercube. The extension to first order logic with equality will follow.
The basic idea in our approach is to have all processors, or *nodes*, *searching in parallel* for a proof of the input theorem. The processors have a high degree of *independence*: they are all peers, working asynchronously and cooperating by exchanging messages. In addition to the basic components of a theorem proving strategy, i.e. the *inference mechanism* and the *search plan*, our strategy features a *distributed allocation algorithm* and several policies for the *routing/broadcasting* of messages. The distributed allocation algorithm distributes among the nodes first the input equations and then those generated during the execution. Thus, at any stage of the derivation, each processor has its own, local data base of equations. The union of these data bases forms the global data base. The processors communicate by sending messages, either *data-messages*, containing equations, or *control-messages*. Each node is responsible for performing inferences by using the equations in its own data base and those received in form of messages. Data-messages are further classified depending on the status of the carried equation: for instance, newly generated equations are treated differently from normalized equations. Different message-handling policies are defined for the different types of messages.
We expect the highly distributed nature of this approach to help significantly in attacking the huge search spaces usually generated by theorem proving problems.

# References

[1] M.P.Bonacina and J.Hsiang, Towards a Foundation of Completion procedures as Semidecision procedures, Technical Report, Dept. of Computer Science, SUNY at Stony Brook, available through ftp.

*Research supported in part by grant CCR-8901322, funded by the National Science Foundation. The first author is also supported by a scholarship of Universitá degli Studi di Milano, Italy.*

# Parallel Reform Computations

**Contact Persons:** Sten-Ake Tarnlund, Hakan Millroth

Computing Science Department
Uppsala University
Box 520
S-75120 Uppsala, Sweden
mail: hakanm@csd.uu.se
Fax: +46 - 18 - 521270

**Keywords**
Logic Programming, Large-Scale Parallelism, Reform

**Summary**
The Reform machine, a parallel abstract machine for logic programming based on the inference system Reform, is designed and implemented in this project. The machine is primarily designed for running the logic programming language Prolog. The basic idea is to retain the sequential left-to-right depth-first backtracking scheme of Prolog with one exception: the recursion levels of a recursive program, including the head unifications at each level, are computed in parallel. The Reform machine is currently being implemented on parallel hardware based on the Inmos Transputer.

# Integrating Rules and Connectionism for Robust Reasoning

**Contact Person:** Ron Sun

Brandeis University
Waltham, MA 02254, USA
rsun@cs.brandeis.edu

**Keywords**
Connectionism, Combining Symbolic and Subsymbolic Reasoning

**Summary**
This work tackles the problem of the problem of modeling commonsense reasoning and alleviating the brittleness of traditional symbolic rule-based models by combining rules with connectionism in an integrated framework. This idea leads to the development of a connectionist architecture with dual representation that combines symbolic and subsymbolic (feature-based) processing for evidential robust reasoning: CONSYDERR.
Psychological protocols are analyzed based on the notions of rules and similarity, and are modeled by the architecture which carries out rule application and similarity matching in a massively parallel fashion throught the interaction of the two levels. In order to understand rule encoding in the architecture, a formal analysis of the model is performed, which shows that it handles a superset of (propositional) Horn Clause logic and Shoham's logic. To further improve the rule-based reasoning capability of the architecture, a solution to the variable binding problem is proposed. This work also explores the notion of causality and shows that commonsense causal knowledge can be well represented by CONSYDERR.
An important aspect of this research is that the architecture utilizes the synergy resulting from the interaction of the two different types of representation and processing, and is thus capable of handling a large number of difficult issues in commonsense reasoning.
The results so far suggest that connectionism coupled with rule-based symbolic processing capabilities can be effective and efficient models of reasoning for both theoretical and practical purposes.