1988
ANNUAL REPORT

International Computer Science Institute

February 15, 1989

Contents

1 Introduction

The **International Computer Science Institute** is an independent, non–profit basic research institute located near the University of California campus in Berkeley, California. The ICSI was started in 1986 as a joint project of the Computer Science Division of UC Berkeley and the German Federal Computer Science Laboratory (GMD). German support of the Institute is now provided by a government–industry consortium. ICSI also receives support from a variety of U.S. sources and participation by other countries is in progress. The German initiative was motivated by a recognition of the lack of an international facility for fundamental research in the field of computer science. The GMD and the supporting industrial consortium expect the participation of foreign scientists to do for fundamental computer science what such international collaboration has done for the field of physics. They also expect to benefit from the experience that young scientists from Germany and other countries will gain from participation in the work of the Institute and ultimately enhance international recognition of the work which goes on in their own countries. The selection of Berkeley for the Institute was based on the outstanding reputation of the UCB Computer Science Faculty and on their enthusiasm for the project. Following some preliminary organizational effort, 1988 represents the first year that ICSI has functioned at an operating level. This report summarizes the research objectives developed, the progress achieved in the last year, and plans for the near future.

The core of the Institute's program is the intramural research effort. The International Computer Science Institute strives to maintain ongoing basic research projects of the highest standard in selected areas of computer science and engineering. Only by maintaining research projects at the forefront of technology can ICSI have the strength to meet its other goals in international cooperation. The particular areas of concentration are chosen for their fundamental importance and their compatibility with the strengths of the Institute and UC Berkeley staff. The current emphasis is on distributed and parallel computation with particular attention to massive parallelism, but a variety of topics in basic computer science are being explored.

In addition to its intramural research, ICSI maintains a number of other programs in support of international cooperation in advanced computer science and engineering. These include a post–doctoral program, exchange visits, summer jobs for graduate students and partial support of selected working conferences. One of the principal advantages of ICSI is its ability to act rapidly and flexibly when opportunities arise and thus gain maximum leverage from its resources. As a basic research organization, ICSI does not engage in product development. The Institute does help in the practical realization of its research results and grants non–exclusive, royalty–free licenses to supporting institutions.

All of the ICSI programs place special attention on cooperation with sponsor nations. The post–doctoral program is fairly conventional. A committee of sponsor-nation and ICSI scientists select among applicants on the basis of overall scientific potential and the availability of an appropriate project at ICSI or UCB. Salaries are competitive and all expenses are paid by ICSI. Exchange visits by more senior investigators can be of any length and are arranged at the mutual convenience of the host and visitor. These are often supported by other funds, but ICSI can provide full support where this is warranted. Several of these visits have given rise to new or strengthened international collaborations.

The summer program for graduate students has concentrated on placing advanced United States students in sponsor–nation laboratories. ICSI advertises in the U.S. for interested candidates, who are required to be beyond the qualifying exam and are asked to provide references. In parallel with this, sponsor–nation committees identify laboratories which might be interested in U.S. students. The ICSI staff reviews the applicants and makes tentative matches between openings and applicants. The appropriate files are forwarded to host laboratories and all further negotiations are between the employing lab and the student applicant. The salary is paid by the employer. ICSI pays for the travel costs to the host laboratory and offers assistance with arrangements. The idea, again, is that a small expenditure of Institute funds and effort can have a significant effect.

The ICSI intramural research program is designed to concentrate international and multi–faceted research teams on computing problems of the greatest scientific and practical potential. The focus is on the middle range future, five to twenty years out. The narrow gap between theory and practice in computer science makes it an exceptionally interesting field, but could create conflict-of-interest problems for an international research institute. By concentrating on pre–competitive but result–oriented projects, ICSI can contribute to the technology base as well as the general scientific knowledge of its sponsors.

The initial research plan of ICSI is centered on distributed and parallel computation with a special concern for massively parallel systems. It has been clear for some years that parallel computation is the key to many desirable applications. Distributed and parallel systems of moderate scale are now fairly well understood and are widely employed. But the problems involved with millions or billions of independent computations appear to be different and to require new techniques. ICSI is currently addressing issues in four key areas: theory of computation, realization of massively parallel systems, applications of such systems, and very large distributed networks.

Section 2 of this report outlines our efforts and plans in the area of applications of massively parallel systems. The most promising applications are to problems in artificial intelligence including vision, language and speech, and knowledge representation and inference. Section 3 discusses the Institute's research in the theory of computation and its relation with other topics. The realization of massively parallel systems in hardware and in software is the subject of Section 4. Section 5 outlines our work on the supporting systems theory and software for the very large international networks of the near future. Finally, Section 6 describes the ICSI research environment and some of the other work being carried out at the Institute.

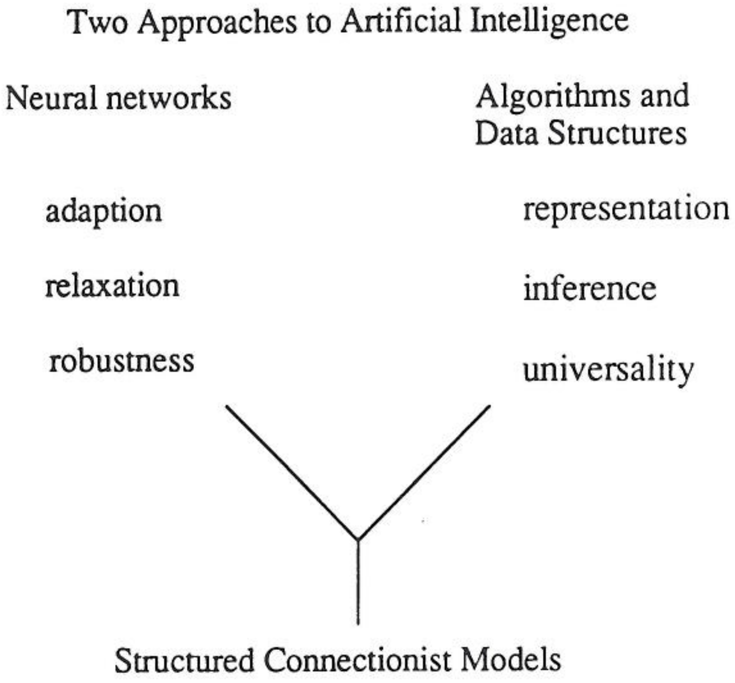

Two Approaches to Artificial Intelligence
Neural networks
Algorithms and Data Structures
adaption
relaxation
robustness
representation
inference
universality
Structured Connectionist Models

Figure 1

2 Applications of massive parallelism

(J. Feldman, H. Bourlard, J. Diederich, M. Fanty, S. Omohundro)
(1988 visitors: D. Baggi, A. Kobsa, P. Munro, B. Neumann, L. Shastri)

2.1 *Introduction*

There is wide–spread and increasing interest in massively parallel computational models and their applications. Much current work examines the bulk properties of unstructured masses of units and the learning capabilities of such systems. One major appeal of neural–style computation is the hope that it will "eliminate programming" and that a single general system can be taught any required task. There is essentially no chance that this will come about, although interesting lessons are being learned in the attempt. Our approach to massively parallel computing is quite different and explicitly utilizes specific structures for different functions. This is certainly less newsworthy than a universal learning machine, but a great deal of evidence from several disciplines suggests that there will be no way to avoid analysis and design in the development of complex systems. One way to view our work on structured connectionist models is as a synthesis of the most significant ideas from two apparently disparate approaches to building intelligent systems (cf. Fig. 1).

As [Newell, 1983] points out, the early attempts at developing artificial intelligence pursued two distinct paths. One group of investigators started with the structure of animal brains and tried to build toward complex systems. The powerful computational ideas from neural network theory are *error-tolerance, parallel interaction (relaxation)*, and *learning*. The idea behind structured connectionist modelling is that these advantages can be married to the ideas of representation and inference from mainstream AI to yield a powerful and robust computational paradigm. Figure 2 (after [Shastri, 1985]) presents a vastly oversimplified version of how a system might encode and exploit conceptual knowledge. The memory network is a category-based hierarchy with each concept and property-name represented by a rectangular unit. The triangular nodes stand for intermediate units which become active when two of their three input lines are active. Suppose the system has a routine that retrieves its knowledge of food tastes as an aid to ordering wine, such as that cartooned in the lower half of the figure. If activation is spread simultaneously to the "main course" of the meal and to the desired property "has–taste," exactly one triangular evidence node, b , will receive two active inputs and this will lead to the activation of the concept "salty." This is the required answer, but for technical reasons an intervening clean–up network is needed where the answer is actually put to use. One interesting feature of Shastri's model is that the same memory network is able to classify a salty, pink food as ham – the triangular evidence nodes work in both directions.

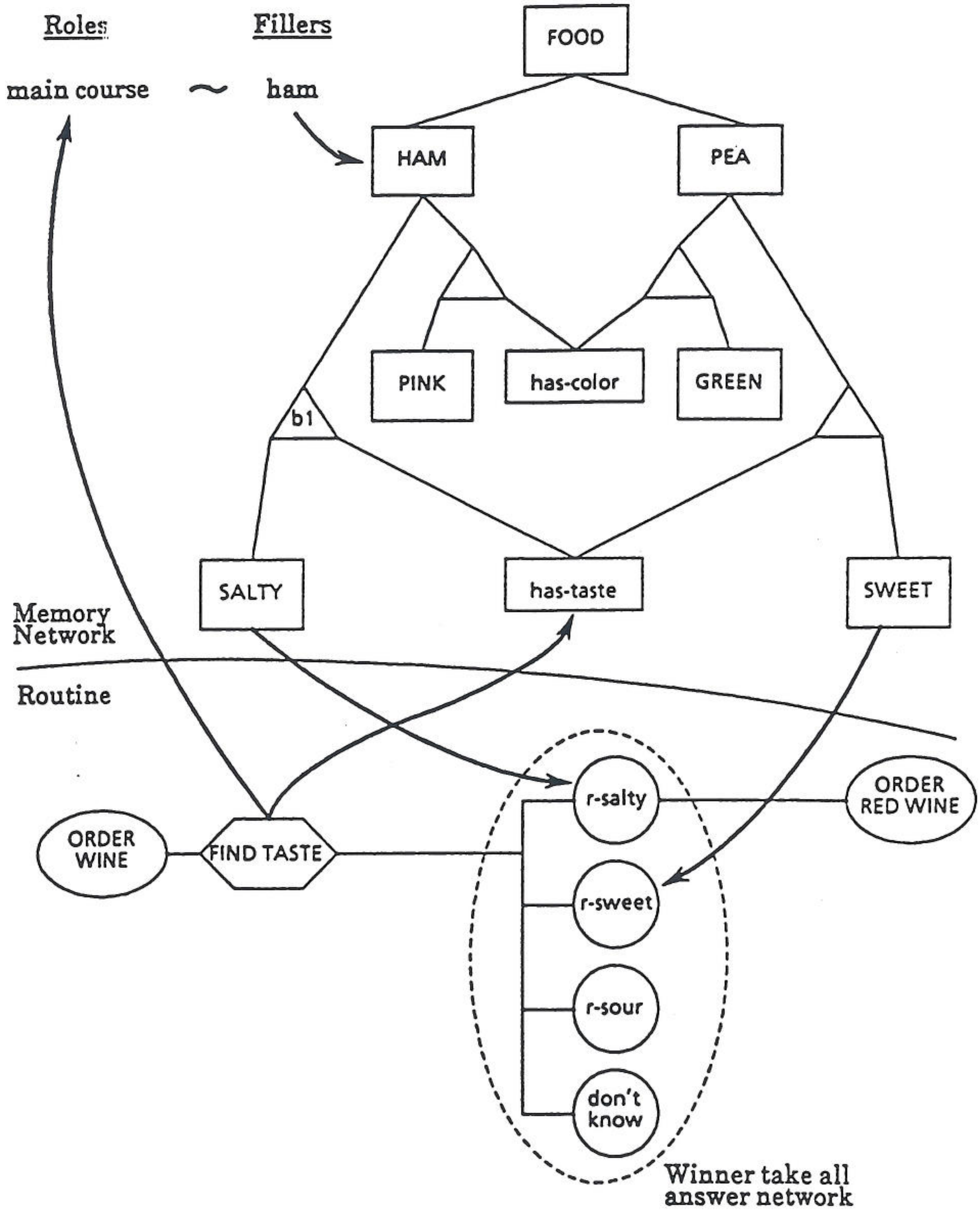
Shastri's work covered much more complicated problems including some that have not been treated adequately in logic–based AI research. For example, the system is able to use evidence combination to assess the likelihood that someone who is both a Quaker and a Republican might be a pacifist. It also included a principled treatment of inheritance hierarchies with exceptions. Notice that the domain structure (semantic networks) came from AI while the realization as a connectionist network adds important capabilities. We are exploring the properties of structured connectionist networks, their realization and their applications particularly in Artificial Intelligence.

The core of our work remains the application of connectionist models to problems in Artificial Intelligence. Our current and planned efforts focus on knowledge representation, learning and perception with most studies exploring more than one issue. The work of Shastri described above (cf. Fig. 2) was an early connectionist knowledge system. Current work is concentrating on how conceptual combinations, e.g. green apple, would be treated in such networks and especially how appropriate inferences from modifiers can be drawn. A related study on learning is concerned with how a connectionist system could recruit structures such as Fig. 2 and more complex relational structures.

2.2 *Knowledge Representation and Learning*

The subproject on connectionist knowledge representation is part of the "Knowledge–Guided Learning of Language" project at ICSI. The task is to define and implement a connectionist knowledge representation system (CKRS) which encodes information from various input sites, such as natural language input and visual input (in a restricted sense). CKRS should allow massively parallel reasoning about a domain and is permanently enlarged by connectionist learning techniques such as recruitment learning.

It is the objective of CKRS to support learning processes by using generic knowledge



Connectionist Retrieval System

Figure 2

about a domain. It was the objective for 1988 to build a first experimental version of CKRS and to do various experiments with connectionist learning methods such as recruitment and reinforcement learning. There is a continuing effort on general connectionist learning techniques and on their uses. [Porat, Feldman, 1988]. Also the applications group is working with the theory group on understanding and extending complexity theoretic approaches to learning [Blum, Rivest, 1988; Valiant, 1988; Baum, Haussler, 1988]. Several experimental versions of connectionist representation and learning systems were built. A representation system was implemented which allows the use of taxonomies of generic concepts as well as taxonomies of generic relations. Dependencies between features of relations (e.g. number and type of arguments) and properties of objects in a simple blocks worlds were learned by backpropagation methods.

The emphasis of the work in 1988 was on "recruitment learning." This technique is used to allocate new units for conceptual representations and to integrate these units in existing networks.

An experimental version of a "knowledge–intensive recruitment learning" [Diederich, 1988 d,f,i,j,k] method was implemented, which uses built–in knowledge for analytical generalization. The method requires only the presentation of a single example to build a new concept representation. On the connectionist network level, the central process is the recruitment of new units and the assembly of units to represent new conceptual information. Free, uncommitted subnetworks are connected to the built–in knowledge network during learning. The goal of knowledge–intensive connectionist learning is to improve the operationality of the knowledge representation: Mediated inferences, i.e. complex inferences which require several inference steps, are transformed into immediate inferences; in other words, recognition is based on the immediate excitation from features directly associated with a concept.

A first version of a connectionist "learning by being told" [Diederich, 1989a,c] method was realized, which allows the user to express a "statement" in a symbolic description language or in a system–guided interview. This statement is compiled into an instruction to the connectionist network, and recruitment learning is triggered to integrate knowledge expressed in the instruction in the connectionist representation. This is considered to be an important extension to "learning by example" methods in neural networks.

The objective of this research is to demonstrate how high–level learning techniques such as "learning by being told" (or "learning by instruction") can be realized in a connectionist system. The instruction has two parts: a goal–concept which is already part of the connectionist representation system, and a training example which consists of a set of attribute/value pairs, represented by units in the network. The presentation of the input triggers recruitment learning, i.e. the acquisition of a network unit from the pool of free units and the integration of this unit in the network of concept units. The recruitment learning process guarantees that the newly recruited unit is integrated immediately below the goal–concept; a new concept unit is generated this way. The newly recruited unit is the immediate sub–concept of the goal–concept and the immediate super–concept of former sub–concepts of the goal–concept. The new concept unit owns attributes and features (i.e. values of attributes) from the training example, the goal–concept, and attributes and features inferred from knowledge already part of the network.

It is important to note that the integration process requires changes in the network of concept units, i.e. knowledge is not only added to the network, but the network of knowledge units is actually changed in order to allow real integration.

2.3 *Vision*

Knowledge representation networks like Figure 2 play a crucial role in the higher levels of all language and vision tasks. One significant effort completed in 1988 was a comprehensive connectionist model of form and motion perception [Feldman, TR–88–011]. This extends the Four Frames model of [Feldman, 1985] to deal with the relevant temporal issues.

The central problem of vision is linking visual–feature information to the knowledge of how objects in the world can appear. The problem of going from a set of visual features to the description of a situation is called the indexing problem, by analogy to looking up something in an index. Obviously enough, it is more effective to index with invariant, real–world features than with their retinal manifestations and facilitating this is the primary function of the stable feature frame. Recognition of an object or situation is modeled as a mutually reinforcing coalition of active nodes in world knowledge. The mutual excitation of feature and model networks also involves top–down, context, links from visual elements to the feature units that are appropriate. We note in passing that the indexing process involves de–spacing the feature information; there could not be separate recognition networks for an object in each position in space.

The basic idea is that each visual element of a complex object is represented by a node that corresponds to a particular set of feature values as computed in the feature frame. Since indexing from features to elements occurs in parallel, there will usually be several simultaneously active element nodes for a complex object currently in view. This simultaneous activation of subparts will tend to cause the correct complex objects to be activated, independent of the details of how the relationships among the subparts are modelled. When the details of complex object representations are considered, a number of difficult technical problems arise. Recall that the visual–appearance models are far from complete – they are more like verbal descriptions of something not currently in view.

Obviously enough, the side and bottom views of a horse have relatively little in common. Even within the side view, the horse could appear in a variety of orientations and scale configurations and the relative positions of its subparts could also differ considerably. One must also account for the fact that there could be several distinguishable horses in a scene and that some of these may be partially occluded. Our current solution, depicted in Figure 3, involves instance nodes, separate sub–networks for different views and cross–referenced structural descriptions. The prototype horse has a general hierarchical description in which, e.g., the trunk is composed of a body, legs, and a tail. What visual elements might be involved in recognizing a horse will depend on whether it is a front, side or other view. Thus the matching process would select together a prototype and a view which best matched the active visual elements. As always, there is assumed to be mutual inhibition among competing object descriptions and view nodes. There is a good deal of ongoing work on

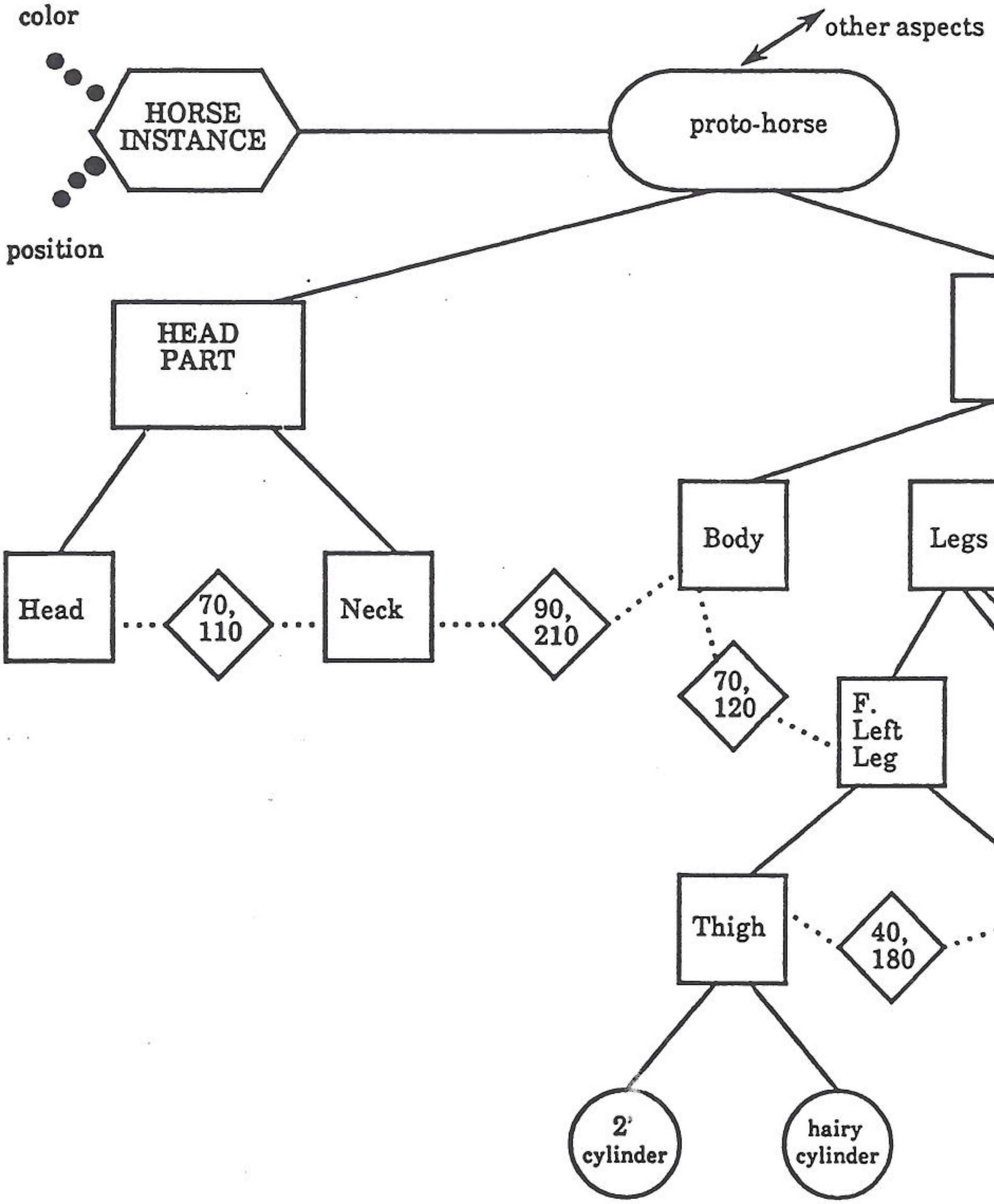

color
other aspects
HORSE INSTANCE
proto-horse
position
HEAD PART
TRUNK
Head
70, 110
Neck
90, 210
Body
Legs
Tail
70, 120
F. Left Leg
F. Right Leg
Thigh
40, 180
Calf
2' cylinder
hairy cylinder

Figure 3

this kind of recognition network [Cooper, Hollbach, 1987; Plaut, 1984; Cooper, 1988] and it does appear to be computationally reasonable. A major goal of this work is to extend these ideas to recognition of motions, such as a horse's canter. This involves reworking all levels of the four–frames model and confronting several important issues elided in the atemporal treatment.

One reason that it has been difficult to isolate the temporal properties of the visual system is that the internal time scale of the computing elements is of the same order as the events they are trying to compute. Nature has been constrained to elements with millisecond operation times and every aspect of the system exhibits signs of this constraint. It is true that there are special adaptations to detect binaural differences that translate to a much smaller (microsecond) time scale [Knudsen et al. 1987], but these are not well understood and there is no indication that they are used in normal vision. The rise time of photo–receptor potentials, the transmission times of axons and the firing rate of neurons all lie in the range of a few to a few tens of milliseconds. The performance of the visual system in resolving time differences, speeds, flicker rates, etc. fits nicely into the same range of times, but we need to understand how it is done. Pulfrich illusions caused by reducing the light to one eye show that notion of relative time of visual events can be easily confounded. An additional conceptual problem arises because much of the information in the system is transmitted by a temporal code (spike rate) which requires more time than the events it is describing. It does not seem likely that we will be able to treat temporal change as just another visual property like color or spatial scale. The fundamental complexity of temporal issues in form vision extends through all conceptual and anatomical levels. A large body of evidence suggests that there are two parallel pathways extending through many anatomical levels and characterized, as least in part, by different temporal characteristic [Maunsell, 1987]. Taking temporal change as an organizing principle provides a new perspective on several classical vision problems. As in [Feldman, 1985], it is claimed that no significant issue has been overlooked and that the answers provided are consistent with each other and with the relevant behavioral, biological and computational findings. Also, unfortunately, it still appears that there is no alternative model in the literature.

Another observation is that temporal change in visual information has many possible real–world causes and is detected by several mechanisms of the visual system. All of these interact in complex ways and it is difficult to isolate one mechanism experimentally or theoretically. The inputs represent various ways that the visual system obtains information pertaining to temporal change. The outputs depict the kinds of real-world events to which the system can attribute the changes. These include a perception of self-motion, the movement of articulated objects within a scene and non-rigid shape changes. We are also able to detect other sources of image change such as changes in illumination. One important point is that there is no simple relation between the kind of real–world change and its manifestations. The situation is exactly analogous to the problems in spatial vision where the brightness at a point is a joint function of illumination, distance, albedo and orientation. The visual system apparently solves these inverse problems by best–fit in a parameter space embodied in neural networks. Since the solution is normally rapid and robust, the networks must embody much of the solution in their structure.

The main inputs to the change processing network come from three sources: Pursuit eye movements, local (short–range) slip detection, and the matching (correspondence) of features displaced in space and time. In addition, estimates of depth (from stereopsis, etc.) and top–down contextual expectations play a central role in the interpretation of visual change. But, as in [Feldman, 1985], constancy feature calculations form just the base of our concerns. The additional major issue is how the change processing network could fit into the visual system model. Basically, the motion and change network is an elaboration of the motion stable feature frame of [Feldman, 1985]. It is the computational detail of this elaboration that the work addresses. One question concerns how the invariant temporal computations might interact with the other invariant features computed in the stable feature frame and some related questions involving eye–movements and other changes.

The indexing process that links the feature frame with world knowledge retains its central role. A specific motion primitive, the trajectory, is hypothesized as the key link between features and objects. The way in which trajectories are computed and used is the basis for the higher levels of the model.

The static situations of the earlier model are extended to multi–temporal scenarios and many of the previous mechanisms are extended. Finally, an attempt is made to make computational sense of the hypothesized what and where dichotomy between the ventral and dorsal branches of visual cortex. There are ambitious efforts to build detailed connectionist models of trajectory calculation (with T. Olson at Rochester) and moving object recognition (with N. Goddard at Hughes). A closely related ICSI project focusses on the complete problem of visual recognition from raw input images.

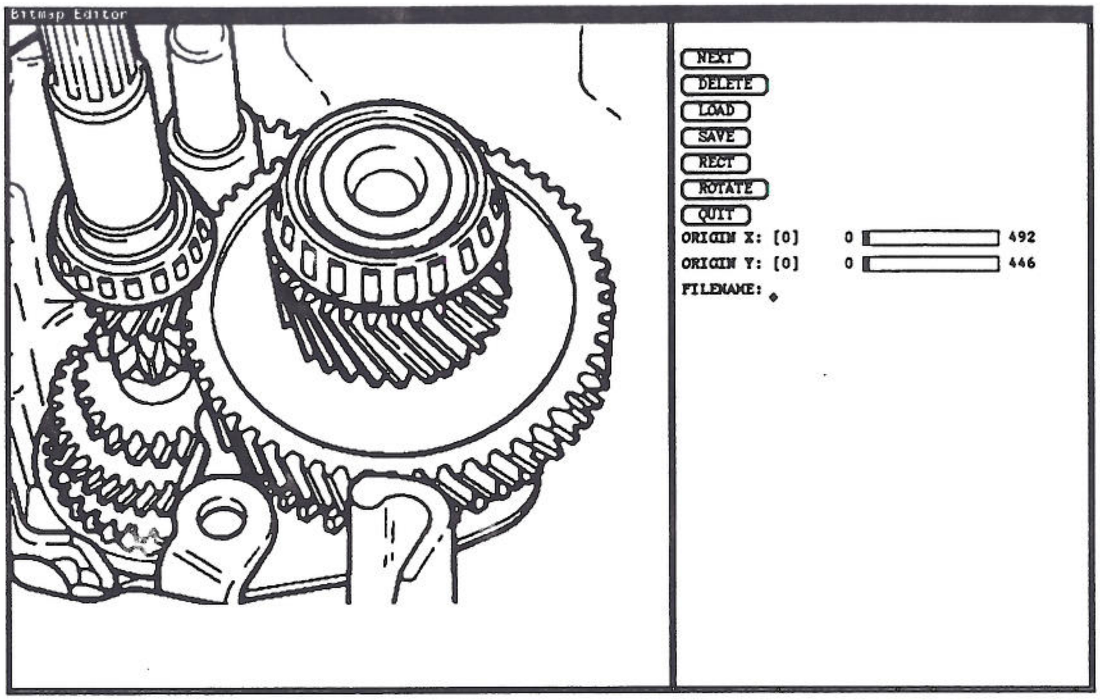
Perhaps the most fundamental problem in this domain is the recognition of known three–dimensional objects from images. We would like to both develop a systematic theory for the computations involved in this class of problem and to build a working system with present day hardware. Our view is that vision is highly knowledge intensive and much of the research will be concerned with the tasks of effectively obtaining, representing and using visual knowledge. At a low level, statistical decision theory and differential geometry provide tools for developing a rigorous theoretical foundation. In more complex situations, these basic tools break down in practice because of a combinatorial explosion in the number of possibilities. Representing and computing with structured objects seems to require the use of artificial intelligence techniques such as default hierarchies and we are developing a systematic framework within which different representational techniques may be evaluated. Because we want to build working systems, we are quite concerned with algorithmic efficiency. We have developed a variety of algorithms based on recent ideas in computational geometry which implement such tasks as finding best matches very efficiently. As hardware architectures improve we would like to develop massively parallel systems and so are investigating connectionist as well as algorithmic implementations of the basic operations.

We are building a recognition system both for intrinsic interest and to evaluate techniques in the context of real problems. We want to study domains which are rich enough to be non–trivial and yet constrained enough to be tractable. Initially we are focussing on line drawings of faces and figures, particularly cartoons. Working with line drawings eliminates a large number of the computationally expensive early vision tasks and requires us to immediately deal with the essential recognition tasks. Work on polyhedral scenes has not generalized well to more complex domains. Faces and figures are a much richer domain and force us to confront both the complex geometry and symbolic structures characteristic of real vision. There are a number of clearly defined goal tasks we would like our system to be able to accomplish. Once knowledge about a particular cartoon figure has been obtained, it should be able to identify the face of that character in a new drawing. More refined goals include identifying facial expressions and various character types.

There are several questions of fundamental importance that we hope this research will address. The task domain is at the boundary between the geometric domain of images and the symbolic domain of visual semantics. One of our major interests is interfacing the statistical and geometric techniques that are characteristic of pattern recognition and much connectionist learning with symbolic techniques which are more characteristic of traditional artificial intelligence. There is a close tie between this work and efforts at ICSI on connectionist knowledge representation and inference. We also want to explore the usefulness of various kinds of learning in building up visual knowledge structures. We are exploring low level algorithms for learning geometric relations and constraints, intermediate level information theoretic approaches to category formation and high level learning approaches like explanation based learning for constructing symbolic structures. Initially we plan to use these techniques as tools for assisting the hand construction of a visual knowledge base. We will use these results to study the extent to which visual knowledge acquisition functions can be automated. We are also working with Prof. George Lakoff (UCB) and his students to relate the visual primitives needed for recognition to the image schemas that arise in linguistics and with Prof. Jitendra Malik (UCB) on relating our work to his efforts on aspect graphs.

The work on this project since its inception in the fall of 1988 has centered on edge extraction and geometric algorithms. A small computer vision laboratory with a state-of-the-art image scanner and color workstation was established. We have developed software to extract edges from scanned images, to find good approximations of curves by segments and to identify plausible vertices. We have also developed a collection of efficient algorithms based on ideas from computational geometry to represent and learn smooth nonlinear geometric constraints. We are beginning the integration of these geometric structures with symbolic ones.

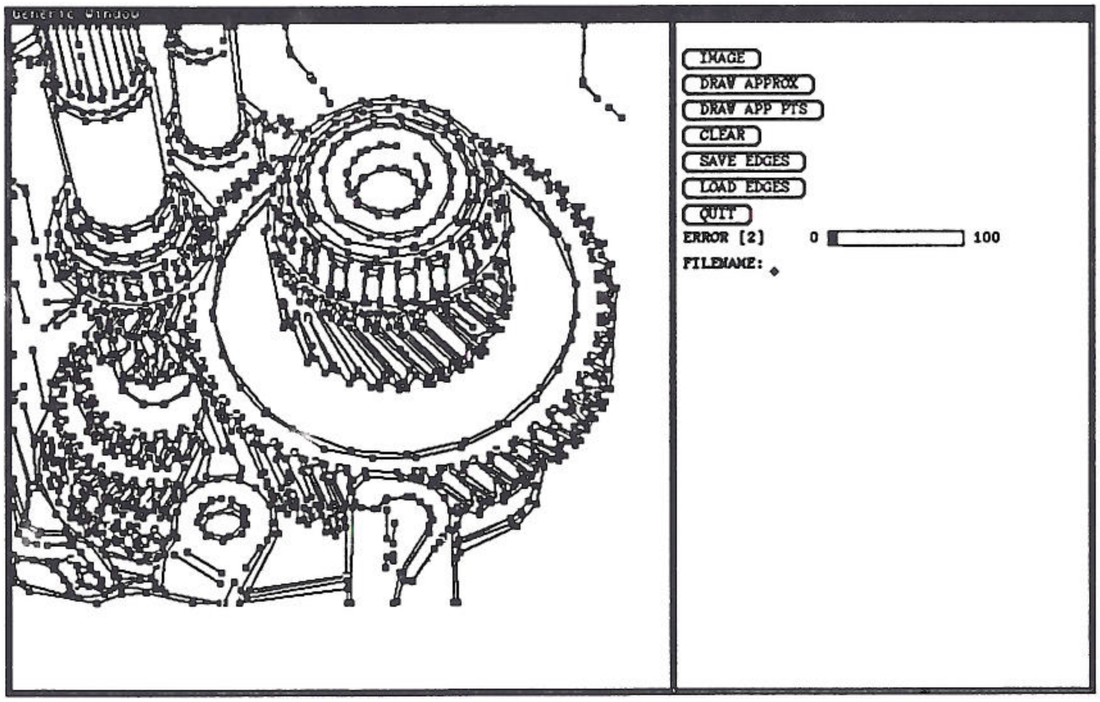
We have developed software to manipulate scanned bitmap images in a variety of ways including cleaning, thinning and edge extraction. We have developed a curve approximation algorithm for choosing the best representation of curves by straight segments and tested its invariance under rotation. This is applied to bitmap edges to produce segment lists for further processing. Figure 4 shows an image and the extracted edges and vertices. For fast access to particular regions of the image with particular properties, we have developed the octbox tree. The image is mapped into a higher dimensional space including the image coordinates x and y and their sum $x + y$ and difference $x - y$ and other features such as angle as coordinates. Hyper–rectangular boxes in this space correspond to octagons in the image with particular ranges of feature values. The image edges are represented in a hierarchical tree supporting fast access. Figure 5 shows the projection into the image of two levels of such a tree for the bitmap above. We have developed software to use such a representation to find plausible candidates for parallel edge segments in real time on a Sun workstation.



ORIGIN X: [0]

ORIGIN Y: [0]

FILENAME:

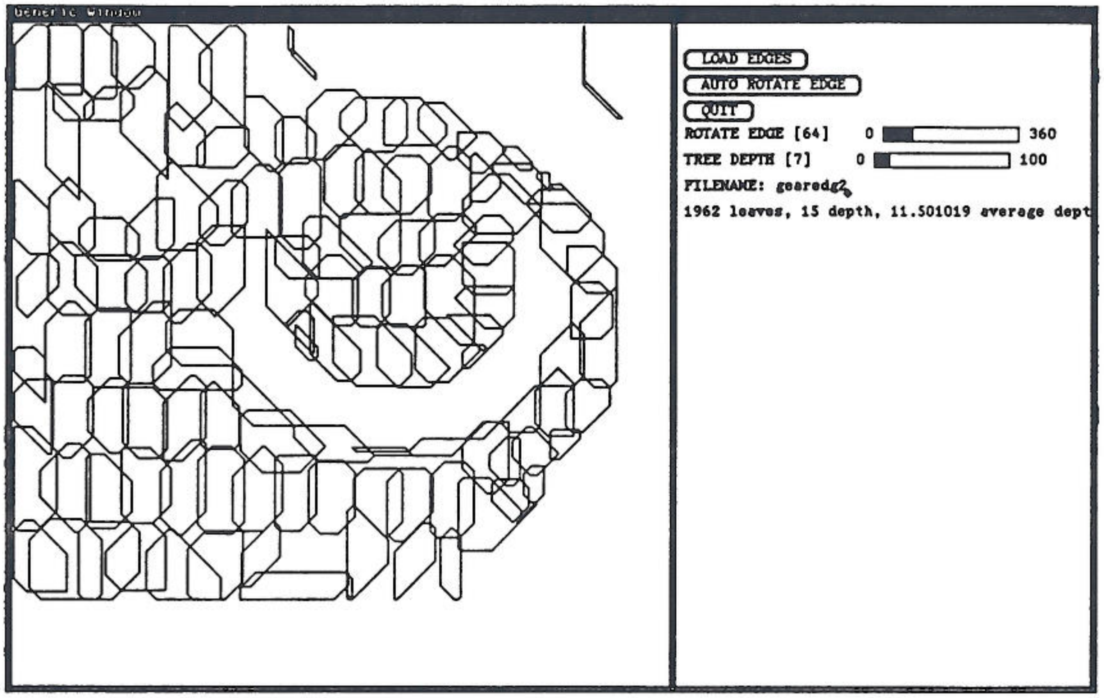


ERROR [2]

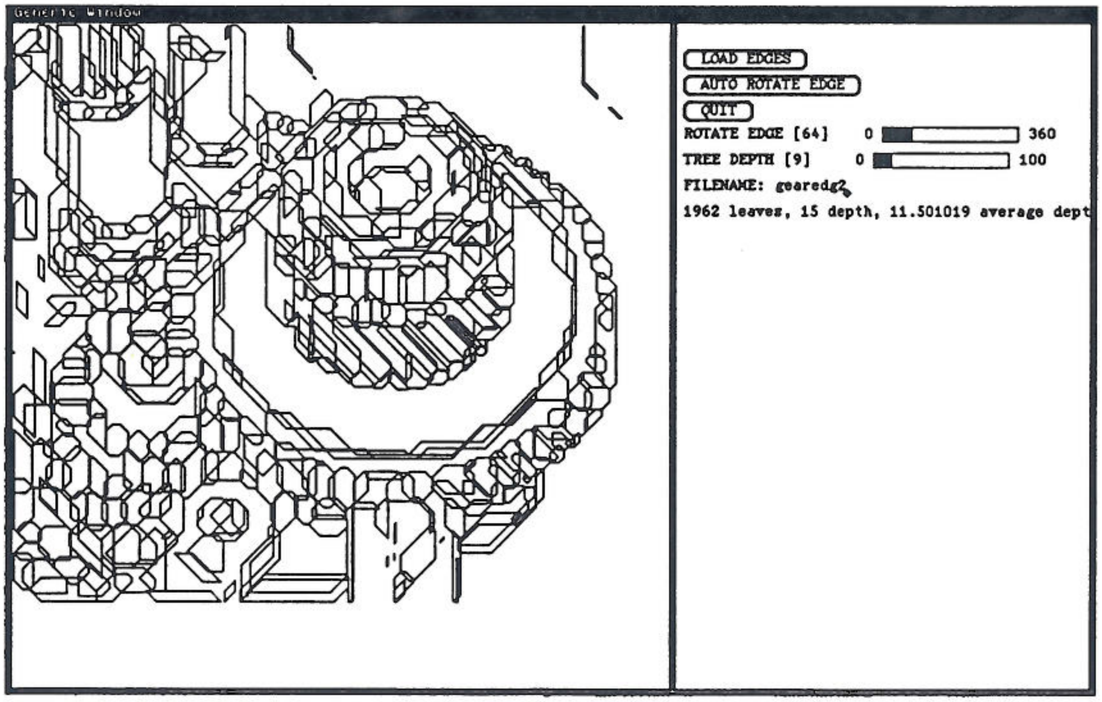
FILENAME:

A line drawing bitmap$_2$and extracted edges and vertices.

Figure 4



```
ROTATE EDGE [64]    0            360
TREE DEPTH [7]     0            100
FILENAME: gearedg2
1962 leaves, 15 depth, 11.501019 average dept
```



```
ROTATE EDGE [64]    0            360
TREE DEPTH [9]     0            100
FILENAME: gearedg2
1962 leaves, 15 depth, 11.501019 average dept
```

Levels 7 and 9 of the image tree.

Figure 5

Ideas from nonparametric statistics allow one to develop algorithms with provably good average behavior with respect to an arbitrary underlying probability distribution. For example, a multi–dimensional data structure known as a k–d tree allows one to structure a set of N points drawn from a fairly arbitrary underlying probability distribution on a k–dimensional space in such a way that the nearest neighbor to a new point (a common pattern recognition task) may be found in a time only logarithmic in the number of stored samples on average. We have developed a variety of similar structures for representing the smooth nonlinear maps and submanifolds which commonly arise in parameterized models such as the space of facial expressions. Nonlinear mappings are built up from samples by piecewise linear approximation over a Delaunay triangulation of the sample inputs. The multi–dimensional data structure allows one to find the approximate image of a new input in a time only logarithmic in the number of stored samples. The representation allows efficient computation of the inverse of such a map as well as a least squares pseudo inverse for singular cases. Similar structures can be used to represent constraint surfaces and to efficiently do nearest parameterized point queries and completion of partial matches.

We have proven a number of theorems showing the theoretical usefulness of the data structures based on the Delaunay triangulation. In practice, however, more heuristic techniques appear to give nearly the same statistical performance while being simpler and faster. In particular, we have developed a data structure which we call a "boxtree" which is useful for a variety of geometric learning tasks. It approximates smooth function and surface interpolation very cheaply and can also be used for density estimation and region representation. We have implemented test versions of this structure and are currently developing a coherent package of these routines.

When we recognize complex structured objects we must use more sophisticated knowledge representation techniques. Information will be stored in structures like semantic networks and will be accessed through techniques of evidential reasoning. We expect significant overlap with the connectionist knowledge representation work at ICSI and will be working to extend it to domains with continuous parameters. A variety of important algorithmic questions arise in this domain as well.

Our current efforts are now focussing on building the underlying symbolic system to support these geometric structures and perform recognition. We will begin with a simple priority queue based agenda for pending tasks ordered by likelihood possibly adding backtracking if necessary. Once a minimal symbolic recognizer is functioning, we will use it to study several different approaches to visual knowledge acquisition. We will use scanned images of facial parts from cartoonists manuals to build up a database of visual elements. Their properties will drive the feature selection and category formation processes. Work will also proceed on the development of the geometric learning algorithms and on making explicit connections with connectionist approaches.

2.4 *Language, Speech, and Sound*

The third major focus of the applications group is on the related problems of natural language, speech, and sound. The Institute's long range plan calls for a major effort on modelling language acquisiton as a testbed for ideas on knowledge representation, vision, language, and learning. This is still in the formative stages, but is already having an influence on our work in related areas. There have also been productive meetings with potential collaborators on the UCB faculty and elsewhere. Some valuable expertise in connectionist natural language processing was gained when Andreas Stolcke from the Technical University of Munich joined the group.

The ICSI effort on massively parallel approaches to speech recognition is rapidly becoming recognized as one of the leaders in this field. The work is a joint project with the realization group and is described in detail in Section 4.4. The bulk of the effort this year has been focussed on the intermediate level problem of mapping acoustic feature vectors to possible phonemes. Future research will attack both the lower level task of selecting appropriate features and the higher level problems of expoliting regularities of syntax, semantics and context to enhance recognition of continuous speech. The higher levels of speech recognitions should have much in common with visual recognition and we expect to be able to exploit structures like those of Figure 3.

All of our applications efforts have two distinct goals: performance on the target task and a better understanding of the required computational primitives. One application that is exploring temporal aspects of connectionist models involves rhythm synthesis for jazz music.

The introduction of computer technology to problems of music and musicology is as old as computer science and goes back to Charles Babbage and Ada Lovelace. Starting with the Illiac Suite in 1956, considerable progress has been achieved both in computer assisted sound synthesis and in algorithms for symbol manipulation in composition and music analysis. Several centers – Bell Labs., CCRMA at Stanford, IRCAM in Paris, the Experimental Music Studio at MIT – have been dedicated to these problems.

With less than half-a-dozen exceptions, however, no results of applications of computer techniques have been obtained for the psychoacoustical, perceptual problem of Swing in Afro-American Jazz. Properly, and holistically, defined, Swing is the medium of the Jazz message – as space is the medium of sculpting. Technically, Swing consists of patterns of rhythmic accentuations and tension-release acoustical devices which are almost instinctively perceived by the listener – though an analytic description, let alone a teaching methodology, is elusive.

This project consists of the construction of a connectionist model for the synthesis of Swing. Using the Rochester Connectionist Simulaton, a system is being developed which accepts as input the harmonic grid of a Jazz standard and which constructs the lines played by a rhythm section consisting of piano, bass and drums. The main program is built around a connectionist network consisting of several automata running in parallel such that control passes from one to the other. The output drives a Kurzweil 250 RMX digital sound synthesis machine. A hand simulation of the output of the network can be played by the sound synthesizer. The lines of piano, bass and drums exhibit some realism, but the overall style, and thus the swing, is sometimes poor. A program exists which, given the changes of the tune, constructs a connectionist network with units representing basic harmony, notes of the piano, bass and drums, and displays its data while running thanks to a recently constructed graphic interface. This basic program is fully deterministic, thus it plays always the same thing on each chorus. It performs like a primitive rhythm section.

Current plans call for expansion of the effort in several ways. First, the interface between the software running on the SUN and the Kurzweil has to be built. Probably a Roland MPU401 MIDI Processing Unit will be used. On the software side, the different degrees of choice have to be built in the network so that each chorus will be different. Choices exist at the level: (1) basic harmony, or chord substitution; (2) notes of the chord played by the piano – e.g., more or less dissonant – and (3) in what position – voice leading; (4) notes played by the bass – e.g., walking bass; rythmic patterns by the drums, that is, (5) basic beat – several patterns – and (6) other accents; time shifts for (7) piano and (8) bass in respect to the basic beat. The network could then switch between all those choices at run time, for example in a random, or filtered random, way.

To insure stylistic consistency, however, the choices could be dictated, or at least biased, by an upper connectionist layer, which can be, at run time, supplied by the user with parameters representing consonance vs. dissonance, sparseness vs. aggressivity, old–fashionedness vs. modernity, one parameter for each of the eight basic choices of above. It would not be difficult to store sequences for replay, some of which could be used to train the network to play in a given style.

3 Theory of Computation

(R. Karp, L. Blum, M. Luby. R. Cleve, M. Blum,
G. Frederickson, S. Hambrusch, M. Karpinski, S. Floyd, P. Dagum)

3.1 *Introduction*

In the theory group, significant progress has been made on the major themes outlined in the research plan. These include: parallel computation, randomized algorithms and pseudo–random number generation, foundations of complexity theory for numerical analysis, and computational learning theory. In addition, a promising new research direction concerning systematic program checking has been initiated. Although the topics are diverse, a number of common threads run through this work. As stated in our research plan: "All of us are concerned with classifying problems according to their computational complexity and determining the influence of the model of computation on the difficulty of solving a problem. The study of parallel computation is a central theme that arises both in the context of connectionist computing and in the study of more conventional architectures and algorithms. Another recurrent theme is the study of computational systems that organize themselves through a process of learning from examples, rather than being programmed to execute specific algorithms. Finally, much of our research focuses on the power of randomization to simplify algorithms and reduce their time and space requirements."

3.2 *Parallel Computation*

3.2.1 *Parallel Algorithms*

The problem of min-max tree search and the related problem of and-or tree search arise in the design of game-playing algorithms and in the construction of strategies for backward-chaining deduction. The famous alpha-beta pruning algorithm is the most widely used sequential algorithm for searching game trees, and a simplified form of that algorithm can be used for and-or tree search. Results by Tarsi and Saks and Wigderson establish that these sequential algorithms are optimal in various senses. Richard Karp and Yanjun Zhang [Karp, Zhang, 1989] have devised a parallel algorithm for min-max tree search and and-or tree search and have shown that, when the number of processors is equal to the height of the tree being searched, the algorithm achieves a linear speed-up over alpha-beta search on every problem instance. They are currently trying to demonstrate a linear speed-up when the number of processors is substantially greater than the height of the tree.

Philip Gibbons has introduced a new model of parallel computation called the asynchronous PRAM [Gibbons, 1989]. This model is intended to capture within a shared-memory framework the cost of data transfers between processors. This is done by allowing interprocessor communication to occur only at certain synchronization points, and to assess a charge for synchronization. Within this model Gibbons has devised optimal algorithms for basic problems such as sorting and the computation of prefix sums. These algorithms are less fine-grained, and therefore better suited for practical use, than the conventional PRAM algorithms for the same problems.

Gibbons and Karp are coauthors of recent papers giving fast parallel algorithms for the problems of constructing the transitive compaction of a digraph [Gibbons, Karp, Ramachandran, Soroker, Tarjan, 1988] and for testing whether one tree is isomorphic to a subtree of another given tree [Gibbons, Karp, Miller, Soroker, 1988]. Karp and Ramachandran have made the final revisions of an extensive survey article on parallel algorithms for shared-memory machines[Karp, Ramachandran, 1989].

The paper [Beame, Luby, 1989] examines the parallel complexity of finding a maximal independent set of vertices in a hypergraph (this is a natural generalization of finding a maximal independent set in an undirected graph). When the maximum number of vertices incident to a hyperedge is restricted to be a constant, they give a randomized algorithm that has expected running time polynomial in the log of the input size. They conjecture that a similar algorithm runs in expected time polynomial in the log of the input size for any hypergraph.

The paper [Luby, 2-1988] introduces some general techniques for converting randomized parallel algorithms into deterministic parallel algorithms without increasing the running time or the number of processors. These techniques are apply to $\Delta + 1$ coloring, the maximal independent set and other problems. This work has since been actively pursued and extended by researchers both at M.I.T. and at Stanford.

Extending their earlier work, Marek Karpinski and D. Grigoriev constructed a deterministic NC^2 reduction from the problem of deciding whether a bipartite graph has a perfect matching to the problem of interpolation of univariate determinants over finite fields. Also, the applications of interpolation techniques of [Grigoriev, Karpinski, Singer, 1988] to the case of Boolean Circuits was worked out in detail [Karpinski, 1988]. This paper enables certain subclasses of randomized NC to be placed in deterministic polynomial time (as well as boolean NC). Two other related results are:

1. A near-optimal parallel algorithm for perfect matching and Hamiltonian cycle on dense graphs [Dahlhaus, Hajnal, Karpinski, 1988]; and

2. An efficient parallel algorithm for finding maximum matchings in planar bipartite graphs. It runs in $0((n/2 - \ell + \sqrt{n}) \log^7 n)$ parallel time and $0(n^{1.5} \log^3 n)$ processors on a CRCW PRAM [Dalhaus, Karpinski, Lingas, 1989].

3.2.2 *Parallel Complexity Classes*

One of the goals of computer science research is to better understand the relative strengths of different parallel complexity classes. The computational power of some fundamental parallel complexity classes is related to the depth complexity of *IMP*, the problem of multiplying together n $n \times n$ matrices (over some ring R). For example, when the ring is $GF(2)$, *IMP* is NC^1-hard (uniformity conditions aside) for the Boolean complexity class *LOGSPACE* and, when the ring is Z_{2^n}, *IMP* is NC^1-hard for *NLOGSPACE*. Also, over more general rings (such as the real or complex numbers), *IMP* is NC^1-complete for the "algebraic" complexity class *DET*.

The function *IMP* is a possible candidate function for separating algebraic NC^1 from *DET*. Also, *IMP* over the specific ring $GF(2)$ might be a candidate for separating Boolean NC^1 from *LOGSPACE*. In either case, the problem is to prove a super-logarithmic lower bound on the depth of any circuit computing *IMP* over the appropriate ring. In addition to the regular algebraic structure of *IMP*, there exist some positive results in related contexts. [Shamir, Snir, 1980] obtained a lower bound of $\Omega(\log^2 n)$ for the the depth complexity of computing *IMP* over the *semi-ring* of positive reals, and [Karchmer, Wigderson 1988] have obtained the same lower bound over the Boolean semi-ring (where multiplication is $\wedge$ and addition is $\vee$). Although these results rely heavily on "monotonicity" properties that do not exist in rings, there are some approaches that Cleve is exploring that appear to be worth investigating further.

3.3 *Algorithms*

3.3.1 *Enumeration Algorithms*

Many interesting problems in science and engineering reduce to enumerating structures within certain combinatorial objects. Researchers encountering these problems are naturally interested in efficient algorithms for these enumeration problems. Often, however, one can prove that, unless $P = NP$ (conjectured to be false), no efficient enumeration algorithms exist. This is the case when enumerating the number of perfect matchings in a bipartite graph. Thus, the best one can hope for is an efficient algorithm to approximate the number of perfect matchings. [Dagum, Luby, 1989a] (also [Dagum, Luby, Mihail, Vazirani, 1988a]) have constructed an efficient randomized approximation algorithm for the matching problem when the underlying bipartite graph contains large factors. Such an algorithm, on input parameters ϵ and δ, outputs an approximation to the number of perfect matchings with a relative error of less than ϵ occurring with a probability greater than $1 - \delta$. The algorithm runs in time polynomial in the size of the problem and in ϵ and δ. Dagum and Luby also give complexity results indicating that a randomized approximation algorithm to enumerate matchings in the class of d-regular bipartite graph on $2n$ vertices, for any $d \in o(n)$, is as hard as the general problem.

Efficient enumeration algorithms for the matching problem have been given for special classes of graphs. The enumeration of perfect matchings in planar graphs has been solved by Kasteleyn (1961). His original motivation was to enumerate the number of "dimer" configurations in liquids and solids. [Dagum, 1988b] has constructed a linear-time enumeration algorithm for bipartite graphs which have the "skew" property. Applications of this result include linear-time algorithms for enumerating perfect matchings in outerplanar graphs and for enumerating Kekule structures in various conjugated systems of organic compounds. (The latter is of interest to chemists who are concerned with certain stability issues that arise in the study of conjugated systems.)

A basic open problem is to determine whether the problem of computing the number of linear extensions of a partially ordered set (poset) is solvable in polynomial time. A result of Dyer, Frieze and Kannan (1988) implies that there is a polynomial time randomized algorithm that computes a close approximation to the number of linear extensions of a poset. Because their algorithm does not solve the poset problem directly, it does not seem to be optimal for this problem. [Dagum, 1988c] has shown that the 0-1 polytope obtained as the convex hull of the incidence vectors of the linear extensions of a poset is magnifying. It then follows that a random walk on the 1-skeleton of the polytope converges rapidly to a uniform distribution. For technical reasons, however, such a random walk cannot be explicitly constructed. Dagum is currently working on efficiently simulating this random walk, which, if possible, would imply a direct and efficient randomized approximation algorithm for enumerating linear extensions.

3.3.2 *Random Digraphs*

Although there is an extensive literature on random undirected graphs, virtually nothing has been written about the properties of random directed graphs. Karp has studied the structure of random digraphs on n vertices in which each directed edge is present independently with probability p. He has shown that, when np, the expected number of edges directed out of a vertex, is equal to a constant c greater than 1, and n tends to infinity, the size of the largest strongly connected component is almost surely close to a certain constant (dependent on c) times n, and nearly all the other vertices lie in strongly connected components of size 1. He has used this structural information to devise an algorithm for constructing the transitive closure of a digraph. On random digraphs the algorithm almost surely runs in linear time.

3.4 *Competitive Algorithms*

There are many situations in which an algorithm or data structure must process a sequence of queries, or an operating system must process a sequence of requests for service. In such situations one can distinguish between on–line algorithms, in which each request must be processed without knowing the requests that will occur later, and off–line algorithms, in which the entire sequence of requests is known in advance. An on–line algorithm is called c–competitive if, on every sequence of requests, its cost of execution is not more than c times greater than the cost of executing an optimal off–line algorithm on the same sequence of requests. In [Fiat, Karp, Luby, McGeoch, Sleator, Young, 1988] we consider randomized on-line algorithms for processing a sequence of requests for pages from memory; the object is to minimize the number of page faults that occur. It is shown that, if main memory has space for K pages, then the best competitiveness ratio c that a randomized on–line algorithm can achieve is approximately log K. This is in contrast to the deterministic case, where the best ratio that can be obtained is K. The concept of a competitive algorithm is applicable in a wide variety of situations, and we are exploring a number of open questions in this area.

3.4.1 *Feasibly Constructive Proofs*

Several upper and lower bounds on the Ramsey number $R(s,t)$ for various values of s and t are known. Many of the lower bounds involve nonconstructive methods (introduced by [Erdös, 1947]) that establish the existence of a graph with specific properties without providing any feasible method for constructing such a graph. Constructive lower bounds (that is, methods that establish the existence of a desired graph by feasibly constructing one) appear to be much more difficult to obtain. Almost all constructive lower bounds that are currently known for Ramsey numbers are considerably weaker than their nonconstructive counterparts. [Cleve, Dagum, 1989] have obtained a constructive lower bound of $R(3,t) \in \Omega(t^{\frac{\log 4}{\log 3}})$. This appears to be the first constructive super–linear lower bound for $R(3,t)$. The proof consists of a simple recursive procedure for constructing a triangle–free graph of size $\Omega(t^{\frac{\log 4}{\log 3}})$ whose independence number is t. This result also translates into a feasible method for constructing triangle–free k–chromatic graphs of size $O(k^{\frac{\log 4}{\log 4 - \log 3}})$. The previously obtained constructions yield graphs of exponential size.

3.5 *Pseudo–Randomness*

3.5.1 *Pseudo–random Number Generation*

The main emphasis of this work has been to construct a pseudo-random number generator from any one–way function. The importance of this goal can be explained as follows. There are many natural problems that are conjectured to be one–way functions, whereas it is very hard to think of a natural example of a conjectured pseudo–random number generator. On the other hand, one of the most basic primitives in cryptgraphy and in the study of removing randomness from probabilistic algorithms is a pseudo–random number generator. Thus, the goal is to provide the crucial link to convert what seems to exist naturally (one–way functions) into an extremely valuable commodity (a pseudo–random number generator).

Intuitively, a function f is *one–way* if it is easy to compute but hard to invert, i.e. given x the value of $f(x)$ can be computed in polynomial time but there is no polynomial time algorithm that receives as input $f(x)$ (when x is a randomly chosen string of length n) and outputs a y such that $f(y) = f(x)$ with probability exceeding $\frac{1}{n^c}$ for any constant $c > 0$. It has not yet been proven that one–way functions exist (if $P = NP$ then they certainly do not exist, but even if $P \neq NP$ it is not clear if they exist), but there are many examples of functions that seem to be one–way in practice and that are conjectured to be provably one–way. Examples of conjectured one–way functions are to be found in number theory, coding theory and combinatorial theory.

A polynomial time computable function f is *pseudo–random* if $f(x)$ is strictly longer than x and if no polynomial time algorithm can distinguish $f(x)$ from a truly random string of the same length as $f(x)$ (when x is chosen randomly) with any non–negligible probability. Intuitively, $f(x)$ "looks" just like a random string to any polynomial time computable function, even though it is generated from a random string x that is strictly shorter.

There has been a flurry of activity trying to make the connection between one–way functions and pseudo–random number generators in recent years. Previous results show how to construct pseudo–random number generators based on one–way functions with special properties [Blum, Micali, 1982; Blum, Blum, Shub, 1982; Yao, 1982; Levin, 1985]. With these results, pseudo–random number generators based on the intractability of some natural number theory problems have been constructed, but the constructions do not apply to other natural conjectured intractable problems. [Goldreich, Krawczyk, Luby, 1988] show how to construct pseudo-random number generators based on one–way functions that are "regular," and then go on to provide natural examples of combinatorial and coding theory problems, conjectured to be intractable, that are regular. [Goldreich, Levin, 1989] show that every one–way function hides an easily computable bit of the input. This result radically simplifies the previous constructions of pseudo–random number generators from one-way functions, and in addition makes them extremely more efficient. Finally, [Impagliazzo, Levin, Luby, 1989] remove all restrictions, showing that a pseudo–random number generator can be constructed from any one–way function. The construction is substantially different in nature from previous constructions, and in the case when the one–way function is regular an implementation of a cryptosystem based on the construction might be run–time competitive with cryptosystems used in practice. An important theoretical result of this work is that the existence of one–way functions, pseudo-random number generators, private key cryptography and bit commitment are all equivalent questions.

We would like to develop constructions that are even more efficient than the existing ones, with the goal being a private key cryptosystem based on the intractability of some natural problem that is as fast as any cryptosystem used in practice. We also propose to explore even weaker conditions than one–way functions that can be used to construct pseudo–random number generators.

3.5.2 *Block Ciphers and DES*

Cleve has completed a Ph.D. thesis (University of Toronto, 1989) that contains an investigation of the cryptographic security attainable in block ciphers whose structure is strongly related to that of the Data Encryption Standard (D.E.S.).

3.6 *Foundations of Complexity Theory for Numerical Analysis*

Complexity theory, as has been developed during the past 25 years, has had tremendous impact on our understanding of algorithms for discrete problems. However, it is not as naturally suited to studying algorithms for the more inherently continuous problems arising in numerical analysis, scientific computation, optimization theory, and more recently, in robotics and computational geometry. For example, it does not address issues such as the condition of a problem, the stability of algorithms, the dynamics of a computation or the topology of algorithms, that are directly relevant here. The goal of our work in this area is to develop a theory of computation and complexity appropriate to study these issues.

Lenore Blum together with Mike Shub (at IBM, Watson Research Center) and Steve Smale (at U.C. Berkeley) have developed the foundations for such a theory of computation over the reals or complex numbers (or an arbitrary ring). See [Blum, Shub, Smale, 1988a; 1988b; 1989]. The theory combines, and builds on, ideas from classical recursive function theory and discrete complexity theory, as well as from algebra, dynamical systems and numerical analysis. Analogues to pivotal concepts and results of the discrete theory abound, but are subtly different. For example, the Julia sets of complex dynamical systems theory provide examples of undecidable sets in this setting. The analogue to Cook's Theorem for computing over the reals is the NP–completeness of the 4–Feasibility Problem, i.e. the problem of deciding whether or not a real degree 4 polynomial in n variables has a real solution. In the discrete case, a simple counting argument shows that NP problems are solvable in exponential time. This is far from obvious over the reals, since now there are a continuum number of possible guesses. However, since 4–Feasibility is solvable in singly exponential time (by [Renegar, 1988] and [Canny, 1988]), and 4-Feasibility is NP-complete, all NP problems over the reals are solvable in exponential time. Thus, one gets the same result as in the discrete case, but for much deeper reasons.

By looking at complexity issues in the domain of the reals, one can more readily employ tools from mainstream mathematics such as algebra, geometry, topology and dynamical systems. For example, the question, "Does P=NP over the reals?" reduces to the question "Is 4-Feasibility in P?" And this points to the question, "When does a complex variety have a real point?" Techniques of algebraic geometry and cohomology theory are applicable here and are currently being studied to attack this problem.

Blum has presented these results at the 29th Annual Symposium on Foundations of Computer Science (October, 1988) and as an invited address at the Annual Meeting of American Mathematical Society (January, 1989).

3.7 *Computational Learning Theory*

Sally Floyd is working on problems in computational learning theory within the framework of probably approximately correct learning (pac–learning) introduced in [Valiant]. Algorithms in this model learn an approximation of a target concept taken from a concept class C of subsets of the instance space X. The learning algorithm draws examples from the space X, where each example is taken from a fixed but unknown distribution P on X. Each example is labeled "1" or "0", indicating whether or not that example is a member of the target concept. The algorithm draws a fixed number of examples of the target concept, and the goal is to output a hypothesis that is close to the target concept, with "closeness" defined with respect to the distribution P. A key parameter for determining the number of examples needed to learn a concept class C in this model is the Vapnik–Chervonenkis (VC) dimension of the class. Floyd has devised a space–bounded algorithm that learns a concept from certain concept classes of Vapnik–Chervonenkis dimension d by saving at most d+1 examples in memory at one time.

3.8 *Designing Programs to Check Their Work*

Manuel Blum has initiated a new approach to reliable computation that he calls program correctness checking. A program correctness checker is an algorithm that checks the output of a given program on a given input. The idea is related to but different from the classical approaches of Verification and Testing:

Verification is concerned with mathematical theorem proving, specifically with proving that a program will return the correct output on all its inputs. A correctness checker, by comparison, only gives evidence that the program gives the correct output on the particular input in question. Testing is concerned with generating experimental evidence that a program will work correctly on all inputs. Generally, a tester for a program is run at the time the program is written and before it is distributed. A program correctness checker, by contrast, is run each and every time that the program is run.

Correctness checking is different from verification and testing: it (only) gives evidence that the given program works correctly on the given input. The correctness checker idea arises from the observation that scientists, mathematicians, engineers ... all learn to check their work as a way of ensuring its reliability. It is therefore surprising that programs are not written to check theirs. There are several reasons for this:

1. Computer hardware almost never makes errors – but that fails to recognize that programmers unfortunately do!

2. Programs are hard enough to write without having to also write program checkers for them – but that is the price of increased confidence!

3. There is no clear notion what constitutes a good checker. Indeed, the same students who are cautioned to check their work are rarely informed what it is that makes for a good procedure to do so – but that is just the sort of problem that computer scientists should be able to solve!

Blum argues that the lack of correctness checks in programs is an oversight. Programs have bugs that could perfectly well be caught by such checks. The papers [Blum, Raghavan, 1988], [Blum, 1988], and [Blum, Kannan, 1989] define a "program correctness checker" for a computational problem π. (Informally, it is a probabilistic algorithm for checking the correctness of any given program for π on any given input I). These papers give examples of checkers for numerous computational problems such as EXTENDED GCD, SORTING, MATRIX MULTIPLICATION, NETWORK FLOW, etc.

One interesting consequence of this work is the observation that randomness is particularly important for designing fast checkers. Another is the observation that if a computational problem does not appear to have a fast checker (one that runs in little oh of the time required to solve the problem), then it may be the wrong computational problem to solve. An appropriate extension π_1 of the given computational problem π may may be just as easy to solve (as π) and also have a fast checker. For example, no fast (little oh) method is known to check a GCD computation, but an EXTENDED GCD computation can be quickly checked. Similarly, no fast method is known to check a program for computing DETERMINANT, but a program that computes the LU DECOMPOSITION and DETERMINANT of a matrix (as most DETERMINANT programs do) can again be quickly checked.

Finally, Blum observes that the interactive probabilistic proofs arising in cryptography are closely related to checkers. His observation suggests a 1–2 approach to the design of checkers via the design of interactive proofs. In his current work with graduate student Sampath Kannan, he seeks to develop this as an approach to checker design for computational group problems.

4 ICSI Realization Group

(N. Morgan, J. Beer, J. Beck, E. Allman, J. Bilmes, P. Sinha, M. Holm)

The ultimate goal of the group is to answer fundamental questions about the use of architectures and technology to implement massively parallel systems. Solutions to these questions may play an important part in the establishment of new groundrules for computer science in the 21st century. Some of the specific questions that we hope to address:

A. What are the best technologies, architectures, and design concepts for a generic massively parallel machine which would be useful for a variety of target application areas (e.g., speech and vision)?

B. How can these machines be dynamically organized into application–specific subsystems without greatly degrading performance over a completely special–purpose system? In other words, what is the best trade–off between programmability and hard–wired special purpose performance?

C. How can structured connectionist models, (which are being designed to attack traditional AI problems such as knowledge representation), be integrated with pattern classification systems to make use of the best features of both? In other words, does connectionism provide a reasonable platform for the unification of pattern recognition and AI?

We are establishing a system design research facility to investigate architectures and implementations for massively parallel computing systems. Experimental components and systems are being designed and implemented which may also prove useful for the verification of concepts developed elsewhere at ICSI. In particular, members of this group are collaborating closely with members of the Applications group to design systems with a demonstrable capability for speaker–independent continuous speech recognition. Later collaborations are likely to include work in the application of connectionist algorithms to vision, natural language understanding, locomotion, music, and biomedical signal processing. Initial systems are only "massive" in a virtual sense, but the conceptual issues of larger systems are a major consideration.

A significant part of the effort will be grounded in specific applications, so that the relevance of theoretical work can be periodically assessed. However, there will be a parallel effort to examine the deeper conceptual issues.

Beginning in the fall of 1988, the Realization Group has built an infrastructure of equipment, software, and people. Laboratory equipment was selected and purchased, such as: a 400 MHz oscilloscope, an expandable logic analyzer, a VME card cage with a single–board computer running a real–time operating system, and miscellaneous test equipment. CAD software for IC design was ported from UC Berkeley systems, and Joachim Beer's architectural simulator was ported to the Sun–4. Engineering professionals Jim Beck and Eric Allman were hired to design hardware and software systems, respectively. Two bright young students from the Computer Science Department were drafted to assist: Pawan Sinha for hardware design, and Jeff Bilmes for programming. The group has just begun to meet on a regular basis, and initial plans and goals have been agreed upon. Faculty ties include R.W. Brodersen from the Electrical Engineering Dept., who is collaborating to provide VLSI design tools, and C. Sequin, who will be spending his sabbatical here next academic year.

The Realization Group has engineering goals that are too ambitious to be "realized" with this small a staff. The intention is to leverage group capabilities with a number of strategic alliances or collaborations with other research groups, both within and outside of ICSI. For instance, making significant progress in speech recognition requires an effort by researchers actually working in speech, with prerequisite efforts of many man–years to build a practical system. To this end, we are incorporating the speech recognition system built up by researchers at SRI, and are actively collaborating to improve what is already a state–of–the–art speaker–independent system. Rather than developing all of our own simulation systems, we are importing IC design tools from UC Berkeley, and connectionist simulators from U. of Rochester (although the latter may be redesigned at some point at ICSI). Rather than try to do all possible connectionist IC designs here, we are splitting the task with interested colleagues at UC Davis (and hopefully elsewhere).

4.1 *Architectural studies*

We are evaluating existing computer architectures for use in connectionist applications. Connectionist networks potentially provide a large degree of computational parallelism. We are investigating how this parallelism can best be mapped onto existing architectures and what constraints and requirements are imposed on the architecture by the connectionist paradigm - possibly leading to new architectural designs. One focal point in the investigation of parallel architectures is the comparison between MIMD and SIMD machines. SIMD connection machines consist of thousands of very simple processing elements, while MIMD machines currently consist of only hundreds of nodes. However, MIMD machines are based on much more powerful processing elements. SIMD connection machines might therefore allow for the possibility of assigning individual units to individual processors and to establish a 1–1 correspondence between the architecture and a connectionist network, thus allowing for a fine grained parallelism. MIMD machines, on the other hand, require the network to be partitioned into larger logical units to be executed on the nodes of the MIMD machine. This structuring might considerably simplify the inter–process communication.

In the fall of 1988 we tried to identify some of the inherent problems associated with different network models and their amenability to parallel implementations. Because of the heterogeneity of artificial neural net models, it is difficult to evaluate architectures for use in connectionist applications. Hopfield nets and multilayer feedforward perceptrons (MLPs) can, due to their very regular connectivity, be effectively mapped on a large range of parallel architectures such as systolic arrays, pipelines, SIMD array processors, etc. This is due to the fact that in many cases the computational structure of these nets can be shown to correspond to vector and matrix operations. Even when the nets are not densely connected one might still be able to utilize this approach as long as the nets exhibit a regular structure which from a computational point of view can be represented as 'banded' matrices which again can be processed in parallel. Currently most connectionist architectural research is concerned with these models (partially because of the perceived importance of these network models for signal processing, pattern recognition, and classification problems).

The structured connectionist models of the Rochester school present a different situation. These networks are problem–structured in the sense that all network nodes represent specific semantic entities (e.g. conceptual structures, inheritance hierachies, etc.) of some particular application domain. The direct realization of domain and application specific semantic constraints in structured connectionist models results in a network which is highly irregular (the term 'structured connectionist model' refers to the logical and semantical structure of the network rather than the regularity of the physical network) and the individual processing units - while still computationally very simple – are not homogeneous as in other types of network models. Typically, there are a small number of different units reflecting specific semantic entities. Furthermore, these units possess a small number of different input sites and site functions, thus making it very difficult to effectively map complex heterogeneous nets onto parallel architectures. At this point, we have not been able to find a uniform computational process that corresponds to the computational structure of such a 'structured' network the way matrix multiplication corresponds to Hopfield nets or MLPs.

A problem that also has to be addressed is network learning or, in implementation terminology, weight modification. This problem applies to all types of network models under active investigation. No standard algorithm has emerged yet, even for particular classes of network models. Any architectural study can easily be invalidated by the emergence of completely different learning algorithms that might impose different computational requirements.

The goal is to try to find underlying computational principles or communication requirements that are common to all (or at least a large subset of) neural network models. One problem is that for structured connectionist models the network topology is domain dependent and one has to identify common connection patterns that are general enough to encompass most arising topologies. Special attention must be paid to the communication requirements in parallel implementations. This is easily overlooked, because on the abstract network level all that is seen is a formal model of a neuron and a given interconnection topology. On the implementation level, the processing granularity and the topology of the net impose severe constraints on the communication requirements.

4.2 *Systems design*

For difficult problems in speech, vision, and natural language understanding, it is unlikely that either parallel systems of general–purpose microprocessors or small systems with high-speed special-purpose ICs will provide sufficient computational power. One of the basic questions that the Institute is researching is how large systems of heterogeneous, or (in some sense) special–purpose elements can be designed and used for these applications. In order to experiment with such hardware implementations, there must be a consistent and well–defined hardware and software environment. Newly designed chips must be testable and usable without the need for a new board–level superstructure or low–level operating system for each design. While the ideal of simply plugging in each new chip without any software or hardware changes is unrealistic, we are designing strategies to minimize such changes. The use of a standard host interface (Ethernet) and a standard host system (UNIX) are good first steps. While neither of these choices are optimum for high–speed real–time operation, they provide a good general interface consistent with available hardware and software. We are also running a real–time system (VxWorks) on a single-board VME bus computer in a card cage with 19 slots for experimental hardware. This is interfaced to our Suns via Ethernet, and VxWorks applications can be developed under the UNIX environment. These choices were made as part of an ongoing collaboration with researchers in the EE department at UCB; they have exactly the same equipment, and we will be sharing and trading software and hardware. In general, interface circuitry and software are being designed in a modular fashion to isolate details specific to the architecture and choice of individual processing elements.

Our current focus is to put together all the software and hardware tools necessary to design ICs and boards for our various interests. We are currently analyzing the cost tradeoffs for schematic capture systems at the board design level.

4.3 *IC Design*

We are establishing a limited capability for application–specific integrated circuit (ASIC) design. Semi–automatic IC design and layout tools have been imported from UCB, and silicon foundries (via MOSIS) provide the manufacturing capability. In recent years, this design and manufacture path has progressed from being useful for pedagogic purposes to becoming a reliable vehicle for electronics research [Ruetz, Brodersen, 1986; Richards, et al, 1987; Azim, et al, 1988]. Processes currently available from MOSIS include a 1.2–micron CMOS digital process, with 2 layers of metal to facilitate speed and connection routing, and a CMOS analog process with an extra layer of polysilicon available for capacitors. Special–purpose computational elements designed with these tools have frequently shown speed advantages of from 1 to 3 orders of magnitude over general–purpose processors using similar technology, for applications which are the focus of the design. Realization group members are learning to use the CAD tools, and are preparing to design some small test chips.

Computationally intensive "inner loops" in parallel algorithms for applications of interest are targeted for implementation as function blocks in VLSI. Both digital and analog ICs are being considered. The former benefit from the availability of semi–automatic CAD technology, while the latter have a performance advantage for some cases. Digital designs are the primary focus in our design tools, because of the availability of the Berkeley digital design tools, as well as the scalability of digital designs to smaller geometries. Our collaboration with UC Davis will emphasize analog designs, which is the major area of strength of our co–researchers there.

The major focus of the above studies is the efficient hardware implementation of non–homogeneous connectionist networks. Software implementations of such networks have been used for recognition of pre–segmented voiced stop consonants [Waibel, et al, 1988], feature extraction for the analysis of speech and brain waves [Gevins, Morgan, 1984, 1988], and, in the form of "structured" networks, may prove to be useful for traditional AI tasks [Feldman, et al, 1988]. The fast CAD tools mentioned above permit rapid modification of basic computational blocks, so that new computational elements can be designed and employed in heterogeneous parallel networks.

4.4 *Applications Work - Speech Recognition*

Hidden Markov Models (HMM) are widely used for automatic isolated word and connected speech recognition [Murveit, Weintraub 1988]. Recent work in the use of connectionist networks for feature selection [Gevins, Morgan, 1984, 1986, 1988] and discriminant HMMs [Bourlard, Wellekens, 1988a, 1988b] suggest that judicious use of these systems may improve the performance of current speech recognition systems. ICSI researchers working with Markov Models and Multilayer Perceptrons (MLPs), in collaboration with speech researchers at SRI and the VLSI design research group at UC Berkeley, are working together to build an experimental connectionist speech recognition system. In particular, a speaker–independent continuous speech recognizer based on HMM methods developed at SRI is being modified to employ Multilayer Perceptrons to do feature selection and to estimate the local probabilities for a discriminant HMM. Fast hardware will facilitate experimentation with these algorithms, because of the long training time required. A test system will be built using a special–purpose IC to implement the MLPs. This IC will be designed using fast CAD tools (Lager IV) from UC Berkeley.

To initiate the ICSI–SRI collaboration, we have transferred phonetically segmented data (in the form of labeled feature vector indices) from SRI to ICSI and trained context-independent and context-dependent MLPs. SRI prepared a segmented and labeled set of over 1000 sentences described by a basis of 60 phonemes and from which 4 different features had been extracted and vector quantized. Thus, each time slot of speech was described by a vector of 4 integers. These new features and their corresponding phonetic labels were transferred from SRI to ICSI, where programs were written to transform them into a form expected by our existing Multilayer Perceptron programs. Two networks were then trained with input vectors employing multiple frames of context (e.g., 7 frames into the past and future). More precisely, a 2-layered network (which is known to be enough for binary inputs) was trained for mapping the input vectors to phonemes. In the MLP, each input frame (associated with a particular acoustic vector, i.e. a particular set of 4 quantized acoustic vectors) was represented by 4 groups of binary units, each of them coding a particular feature. The number of hidden and output units was fixed to the number of phonemes (i.e. 60).

While the trained MLPs have not yet been integrated into the SRI HMM system, applying the trained MLP to the training set produced plausible phonetic classification. We are currently concluding this pilot study which is, we believe, the first test of connectionist methods for a large, nationally distributed multi-speaker data base of continuous English speech. Preliminary results seem to indicate that our context-sensitive artificial neural networks can at identify frames of continuous speech significantly better than the maximum likelihood methods used at the frame level in HMM systems (38 % versus 28 % frame correct). The pilot study, when completed, will also demonstrate the trade-off between network complexity and generalization capability for a real problem. This topic has been examined in recent theoretical studies [Baum, Haussler, 1988] and is a critical point in the applicability of connectionist ideas to real-world problems.

4.5 *Future Directions*

Our ultimate goal is to build high speed massively parallel systems with heterogeneous dedicated subsystems which can do useful work in artificial intelligence and signal processing applications, such as machine understanding of natural language, visual scenes, and speech. The combination of theoretical work on such systems with the practical experience of designing and building smaller parallel systems are important steps toward these lofty goals. A more concrete goal is to provide a service for researchers in which the path from algorithm to working hardware is streamlined so that they can actually experiment with an electronic implementation of their ideas fairly quickly (in months rather than years). Our current work is the first step in this direction.

Initial designs only use a mature technology (CMOS) to implement advanced circuit, architectural, and algorithmic design ideas. This strategy permits a relatively fast turn-around on experiments in these areas of system design. However, it is also necessary to keep track of evolving technologies such as gallium arsenide, non-standard MOS processes, charge-transfer devices, and optical processing methods. It is our current opinion that these novel technologies are not mature enough to be used for a reliable (and versatile) outlet for our various algorithmic research ideas. However, many of the properties of these technologies make them very interesting for the implementation of parallel systems. While it might be premature to undertake system development efforts using these technologies, ICSI is continuing to study their properties.

In the near term (1989), major goals are:

A. Design tools, including VME "testbed" – Now that we have a staff, we must set up the design and simulation systems. While these tasks are conceptually straightforward, they can be expected to amount to a significant amount of work to even reach a reasonable working set-up. Improvements can be expected to occupy much of the staff's time well on into the year.

B. Architectural study – Dr. Beer should have substantially completed his study in this period. An expansion will be required to include technological considerations as well as architectural. Collaborations with Professor Sequin and his students will tie in with this.

C. Fast hardware for numerical computation – we will integrate a fast computational board into the VxWorks testbed. Hopefully, this board will be one currently being built on campus by the Brodersen group, but we may decide it better either to build our own or to purchase some commercial board. This will permit faster turnaround on large training and recognition experiments, particularly for speech.

D. Speech recognition – Given the design tools and fast hardware, speech algorithms designed during the year should be testable on the VxWorks system by the end of the summer. Nelson Morgan will have, together with Herve Bourlard, produced a technical report describing the joint research. Tests should show a demonstrable improvement in speech recognition accuracy using the Bourlard probability estimation techniques.

E. Experimental connectionist hardware – There will be a succession of small test chips submitted to MOSIS this year. This will be done partly to test out the design methodologies, and partly to test out specific design ideas. In addition to testing out specific components (such as multiplier elements), another goal will be to examine the feasibility of building up standard cells which are connectionist elements. Conceptually, one would like to be able to transform the problems of connectionist design into standard VLSI design techniques (gate array, standard cell), in which the fundamental atoms of design are ones which are meaningful to a connectionist algorithm (units, sites, and links). We will keep such goals in mind as we design the first series of test chips. Additionally, we will also keep in mind the speech recognition training application, which will be the target for some of the complex chips which will be built in the following year.

By the end of 1989, we will have a system design facility with a tested development path. We will have conducted architectural and technological studies, both conceptual and empirical, which will give us a clearer view of our goals. Finally, our pilot studies in speech recognition should have demonstrated the potential of our methods in an application area with a clear criterion for performance evaluation.

By three to five years from the ICSI inaugural, we expect to have significant initial results in the areas described above. Additionally, the Realization Group should have made a significant contribution to the machine recognition of continuous speech, including fast training hardware. Exploratory work in the stitching together of AI and pattern recognition through the medium of structured connectionist models should also have taken place. We will also have demonstrated a smooth design path for connectionist software and hardware during this time.

5 Very Large Distributed Systems

(D. Ferrari, D. Anderson, D. Haban)
(1988 visitors A. Barak, K.G. Shin)

5.1 *Introduction*

Much current research in operating systems is focused on "high-level mechanisms" such as distributed transactions, support for replicated data, object-oriented programming systems, user interfaces, and facilities for parallel distributed computation. "Low-level mechanisms" such as virtual memory, process control, kernel structure, naming, local IPC, and network communication have not kept pace with the progress in high-level mechanisms. Many of the current research projects are based on outdated operating systems (such as UNIX)[1] and are crippled by the inappropriate low-level mechanisms provided by these systems. The main objective of this project is the development of optimal low-level mechanisms for the next generation of distributed computer systems, to support the massive parallel distributed computations such systems will be used for in the future.

User demand for high-performance computation and support for applications influences computer technology, since computer engineers design systems in accordance with application needs and build faster computers and faster networks. But also vice versa, technology has an effect on user demand since it opens doors for more and more applications in various areas. Based on investigations of future technology, the project extrapolates trends in computer and communication technology into the middle-to-distant future (5-20 years). The dominant host type will be the workstation; many will be shared-memory multiprocessors with a small number (10-100) of processors (see for example [Gajski and Peir, 1985] and [Hill et al., 1986]). Communication networks based on fiber optics will allow low-delay (30 to 50 milliseconds coast-to-coast) and high-bandwidth (.01 to 1 Gigabit/second) communication among very large numbers of hosts in the U.S., and eventually in the world (see [Turner, 1986] [Shimada et al., 1986]).

We use the term "*very large distributed system*" (VLDS) to refer to a hypothetical system running on the hardware base described above. A VLDS will link thousands or millions of hosts under diverse ownership. Its main function will be to provide secure, well-integrated access to logical *services*. These services might provide access to public databases (such as encyclopedias and archives), news media, sales, advertising, banking, interpersonal communication (mail, telephone, facsimile, and video conferencing), and entertainment (including distribution of audio and video). The processing power of VLDS hosts (and perhaps of

[1] UNIX is a trademark of AT&T Bell Laboratories.

specialized compute servers) is another type of remotely-accessible resource. A VLDS will allow access to supercomputers, which is currently a serious problem, and will connect the resources of a company which is spread over an entire continent, or may be the whole world. In addition, a VLDS can be seen as a massively parallel computing facility, which might involve thousands or millions of processors in a single computation. This loosely coupled connection of computers into a network should easily scale up, i.e., allow more and more computers to be added to it. Massively parallel computing raises new challenges for distributed operating systems, such as those in the area of VLDS management (e.g., load balancing and reconfiguration).

Because of the synergy that may arise from combining communication, service access, and processing in a single unified system, VLDS design is an important direction in computer systems research. The use of a VLDS for high-performance computing will augment (and often replace) the use of specialized hardware, general-purpose parallel hardware, and supercomputers to address the processing requirements of graphics, artificial intelligence, simulation and scientific applications.

This report is divided into three major parts: objectives, progress, and future plans.

5.2 *Objectives*

The DASH project is developing a design for a VLDS. The current design includes facilities for high-speed movement of large amounts of data across address spaces, a highly parallelizable kernel structure, real-time scheduling and real-time communication, efficient mechanisms for communication security [Anderson, Ferrari, 1988].

To support these functions and facilities, the basic system-level requirements fall into four main groups: 1) interprocess communication (IPC) performance, 2) global system architecture, 3) local system architecture, and 4) system support for managing a VLDS. We will focus in the reminder on these four issues.

5.2.1 *Interprocess Communication*

Local interprocess communication (IPC) in DASH makes use of the message passing (MP) system. The MP system relies on virtual memory remapping instead of software copying for moving large amounts of data between the address spaces of the processes that need to communicate. This design is expected to eliminate a major bottleneck in high-performance communication, and to increase the feasibility of moving services such as file systems from kernel space to user space. This expectation is based on the care with which the per-operation overhead (which negatively affects the latency of small messages) has been kept low and the maximum transfer rate of large messages has been made as high as possible [Tzou, Anderson, 1988].

Remote IPC involves network communication. Future distributed systems will use high-performance large-scale networks and will support a range of communication-intensive applications. These applications will be characterized by very different reliability, security, and performance requirements. For instance, many of them will involve the transmission of digital audio, digital video or their various combinations with each other as well as with data, text, and images. In most cases, such applications will require guaranteed communication performance, e.g., limited message loss rate and bounded message delay.

In most existing distributed systems, the interface to the network level typically provides a simple abstraction such as unreliable, insecure datagrams. Upper layers then use this facility to provide higher–level abstractions such as reliable request/reply message-passing [Cheriton, Zwaenepol, 1983], reliable and secure typed message streams [Sansom, et al., 1986], or reliable byte streams [Postel, 1981a]. Because the bottom–level abstraction (e.g., datagrams) is simple, it is easy to port the system to different network types. However, this approach suffers from several basic problems, stemming from the simplicity of the abstraction:

(a) Communication clients cannot express their performance, reliability and security needs to the communication provider. This results in wasted work. For example, data integrity is often a mandatory part of communication primitives, and is provided by software checksumming. This work is wasted for applications that do not require data integrity. Conversely, network interfaces may do data checksumming in hardware, but if this is concealed from upper layers, then checksumming may be redundantly done in software.

(b) Simple abstractions do not allow the communication provider to impose static limits on client behavior, such as the amount of client data outstanding within the network. The problem of congestion must then be attacked by methods (such as ICMP's source–quench messages [Postel, 1981b]) that are often ineffective.

(c) No provisions are made for real-time performance guarantees. Such guarantees are needed for interactive high–bandwidth traffic such as digitized audio [Bastian, 1986] and video.

In an attempt to solve these problems, the DASH network communication system is based on an abstraction called a *parameterized message channel*, or simply a *channel*. Current networks and protocol architectures do not directly support channels. However, our approach is capable of exploiting future advances in communication technology.

5.2.2 *Global System Architecture*

The global system architecture of a distributed system is centered in its naming mechanisms. There often are naming mechanisms at multiple levels; they differ in the nature of the named entries, and in the means of assigning and resolving names. Some of these mechanisms may also involve authentication and security [Anderson, Ferrari, 1987]. The DASH system must support a global naming system for a VLDS on which security functions are based. The naming system must support organizational autonomy in the senses of hierarchical delegation of authority for name assignment, and lack of central trusted agents in name resolution. Naming must be source– and target–location independent. This reduces the location-dependence of program execution, thus simplifying large-scale distributed programming. The naming system must be scalable so that performance does not decline with increasing system size, even when remote references are frequent.

In order to meet these requirements, DASH defines a structure for global naming of permanent entities (hosts, owners, services), and for local naming of temporary services (service tokens).

5.2.3 *Local System Architecture*

A VLDS local system architecture is concerned with support of user-level services, support of the inherent parallelism on shared-memory multiprocessors, and incorporation of modern software engineering ideas in order to make the kernel maintainable, extensible and portable.

We exploit shared-memory multiprocessors by generating "fine-grained" kernel parallelism, i.e., parallelism within single kernel operations as well as between concurrent operations. At the same time, we seek to avoid problems due to increased software complexity, and the performance overhead of process creation, scheduling, synchronization, and IPC.

We intend that user-level services, such as file services, window services, and transaction managers be run at the user level (i.e., in protected virtual address spaces). This has several advantages, including flexibility and reliability. The kernel design attempts to minimize the performance impact of user-level services by providing fast local IPC and support for user-level caching in the management of physical memory.

5.2.4 *System Support for Managing a VLDS*

Monitoring distributed systems is a very difficult task; on the other hand, understanding the highly parallel execution of these systems is crucial for their efficiency, performance and correct operation. A distributed operating system for a VLDS must be equipped with a monitoring tool which provides, besides monitoring and debugging support [Haban, 1987] and [Haban, Weigel, 1988], decision data for several control algorithms, such as scheduling and load balancing. Monitoring system behavior and measuring system performance during operation are mandatory tasks for any computer system; to offer insights into incorrect behavior, identify bottlenecks, and extract data for control algorithms residing within the operating system. For instance, monitoring in DASH may be useful in channel establishment to give the channel provider load information that will allow it to meet the real-time requirements of a channel.

The main requirements are that continuous monitoring activities must not change the behavior and must not slow the performance of the system to be monitored while presenting flexible application-oriented information. In order to allow the host system to dynamically respond to the monitored results, the monitoring system must be able to funnel its results back to the host system. This provides the host system with up-to-date information about its own activities, and can be used for fine performance tuning, scheduling decisions and reliability issues. The monitoring system must be able to combine locally monitored data with remote information and knowledge about other host nodes, evaluate this information and make it available to the operating system of each node for load balancing and allocating resources. The monitoring system must also be able to execute routine, low–level, operating system tasks, such as local load or network management, thus reducing the overhead of the host system.

These tasks become important in a system with a large number of nodes where efficient management requires knowledge of the entire host system or parts thereof, and frequent evaluation of local and remote parameters.

5.3 *Progress in 1988*

As a first step, we extrapolated trends in computer and communication technology into the middle–to–distant future. Based on this technology, we identified functions and facilities of future computer systems and determined the basic system-level requirements of these functions and facilities, as well as designs and mechanisms for satisfying these requirements. In order to study these mechanisms, the project is currently building an operating system kernel that implements our distributed system design and will be used to evaluate and refine it. The kernel is being implemented on Sun 3 workstations, and partly on a Sequent Symmetry shared–memory multiprocessor.

A large software project like DASH tends to alternate between expansion and consolidation phases, and 1988 was spent largely in consolidation. This consolidation had two major parts. First, we completed a draft of the design of the entire system (including network protocols, VM system, and kernel structure) at the level of C++ interfaces. The design was documented in a series of three technical reports. Second, we improved our software engineering environment by designing a system that allows multiple programmers to do kernel development for multiple target machines, while still maintaining consistent version management.

In addition to this consolidation, progress was made on many fronts, which is summarized in the following four sections. In addition, a UNIX–based DASH file server is currently being built; this will support program loading and paged VM. We also planned and began implementing a demonstration of the DASH real–time network communication facility.

5.3.1 *Interprocess Communication*

Local Interprocess Communication

We evaluated the performance of the DASH message–passing system, and demonstrated that virtual memory remapping can significantly improve the performance of local interprocess communication (IPC) [Tzou, Anderson, 1988].

One goal of DASH is to support high performance communication. At very high bandwidths, virtual memory remapping is important because an extra copy forced by the operating system may easily double the transfer time for a packet between applications [Lei, 1988]. However, other systems that have used this technique for message transfer have suffered from high per–operation overhead, limiting the use of the technique. The DASH design is intended to reduce this overhead.

On a SUN 3/50 workstation, we measured the message–passing performance between processes in two user address spaces. The performance results showed that large messages can be moved between virtual address spaces at a rate of more than 37 MB/sec, an order of magnitude higher than with software copying. On the other hand, the performance of small messages is not sacrificed.

We also did a "microscopic" performance analysis of the message–passing system. We broke a message–passing operation into short code segments and timed them with microsecond precision. The results show the relative costs of data movement and the other components of message–passing operations, and allow us to evaluate several specific design decisions.

Remote Interprocess Communication

In an attempt to solve the problems of high–performance large–scale networks, the DASH network communication system is based on an abstraction called a *parameterized message channel*, or simply a *channel*. A channel is a simplex (unidirectional) stream with several performance and security parameters. These parameters express 1) the needs of channel clients (user programs and communication protocols), and 2) the capabilities of the channel provider (network and higher layers). This information can be used in two ways. First, channel providers can eliminate unnecessary or redundant work, and can optimally schedule resources such as network bandwidth and CPU. Second, the channel client can use the parameters to select optimal methods for achieving whatever reliability and flow control are needed [Anderson, 1988].

A simplified network layer and support facilities (e.g. a mechanism for naming objects on remote hosts) have been implemented using the Sun3/50 DASH kernel. The network layer supports best–effort channel establishment and data transfer. The necessary machine–dependent network drivers are also in place. DASH has defined the *DASH Message Language* (DML) for the specification of message structures. A message representation standard which dictates how messages with DML specifications are to be represented has also been defined. A preprocessor that converts a limited set of DML type definitions into a set of macros that facilitate building and accessing messages of those types has been built.

Network Architecture

Much of the work relating to the network architecture was spent on design [Anderson, Wahbe, 1988]. The DASH communication architecture can be implemented on multiple networks. Each network to which a DASH host is connected is represented in its kernel by a software module with a prescribed channel–based interface. These network objects provide host–to–host *network channels*. They encapsulate network–specific protocols for channel creation, deletion, and transmission, and other tasks such as routing and network management. The DASH *network layer* is the collection of these networks and the network objects.

The *subtransport layer* (ST) provides the basic form of inter–host process–level communication. All higher–level network communication in DASH passes through ST, and ST is the only direct client of the network layer. ST supports *ST channels*, which are "value–added" versions of network channels. ST provides communication security, does fragmentation and reassembly, multiplexes ST channels onto network channels, and arranges for "fast acknowledgement" of messages sent on ST channels.

In most existing protocol architectures, the functions of ST are done at higher levels (e.g., security is provided at the transport level). The DASH architecture has the advantage that these functions are consolidated into a single per host module. For example, we argue in [Anderson, Ferrari, Rangan, 1987] that a single secure channel between hosts is sufficient for authentication and privacy, rather than one per operation, session, or network.

The *Remote Operation Facility* (ROF) supports request/reply communication. It uses a set of ST channels called the *ROF connection.* Processes can also directly establish ST channels for stream–mode communication. The design of channel–based stream transport protocols is an area of future study.

Channel Simulation System

An algorithm to establish the parameterized message channels has been proposed in [Ferrari, 1989]. For the study of this algorithm and its refinements, as well as any other establishment algorithms that may be developed, we implemented a simulator for wide area networks. The simulator has been written in CSIM, a simulation language based on C. The simulator expects as input the adjacency list representation of a point–to–point wide–area network. Attributes of the parameterized channel streams and the request arrival process can be specified by choosing among a number of available options. Because of the modularity of the code, the simulator can be easily modified to mimic any extra code that one desires to add over a virtual circuit establishment protocol, and measure relevant properties of the algorithm. This facilitates easy comparison of two algorithms for channel establishment. We believe that it will be possible to use a large portion of the simulator code in an actual implementation of the channel establishment algorithm.

Establishing Channels

With the aid of the simulator, we have investigated the problem of establishing channels that will provide deterministic, statistical, and best–effort delay bound guarantees in store–and–forward wide–area networks. An algorithm that achieves this objective in a single round–trip between the source and the destination of the channel to be established has been devised [Ferrari, 1989], and is being tested by simulation. The algorithm is based on worst–case arguments for the deterministic case, and on statistical worst–case ones for the establishment of statistical channels. Together with the establishment algorithm, we have proposed a novel deadline–based multiple–class scheduling algorithm, which chooses the packet to be shipped next in each node on the basis of its deadline (i.e., its delay bound in the node) and of its type (deterministic packets have priority over statistical ones, which have priority over best-effort ones, which have priority over any other activities of the node). A study of this scheduling algorithm will be performed soon.

A distributed, rate–based flow control mechanism has also been created for real–time channels. The mechanism is necessary if performance is to be guaranteed even in presence of machines that, due to malice or failures, would tend to overload the network and is quite inexpensive and fast.

5.3.2 *Global System Architecture*

In order to meet owthe requirements of a global naming system for a VLDS, DASH defined a structure for global naming of permanent entities (hosts, owners, services), and for local naming of temporary services (service tokens). Certain remotely–accessible logical resources in a DASH system are called *services.* They can be accessed by clients through the *service access mechanism* (SAM). SAM provides replication transparency, location transparency, and failure transparency. It allows clients to use temporary capabilities ("service tokens") to reduce name–resolution overhead. The system uses a tree–structured name space. Names are location independent in that 1) they do not imply the location of the named entity, and 2) they are the same regardless of the location or identity of the entity using the name.

The use of service tokens and distributed name caching are vertically integrated mechanisms in the sense that they involve both distributed and local architectures. The DASH global architecture is an open system in that it facilitates the addition of new services by users. At a lower level, it is an open system in that it provides an open framework for stream–oriented services protocols, it supports inter–service naming, and authentication is factored from authorization.

5.3.3 *Local System Architecture*

We finished a detailed design of the kernel which can be found in two reports: *The DASH Local Kernel Structure* [Anderson, Tzou, 1988] and *The DASH Virtual Memory System* [Anderson, Tzou, Graham, 1988].

We built a prototype of the kernel on SUN 3/50 workstations, and are porting it to a Sequent Symmetry shared–memory multiprocessor. Although the prototype is still in a early stage, it is stable enough and contains enough basic elements for the purpose of experimentation.

The DASH kernel is structured as a dynamic set of processes that share a single address space, communicating via message–passing and synchronized operations on shared data objects. The kernel provides user virtual address spaces, each occupied by zero or more user processes. The message–passing system handles communication between processes on a single host. It does not handle network communication directly, but provides the interface between the various entities that together support network communication. The scheduler and interprocess communication mechanisms support real–time processing and communication requirements. Process scheduling is deadline driven, and is integrated with the message–passing system. A message may be assigned a deadline, and this deadline will affect the deadline of the process that receives the message.

The DASH kernel encapsulates its machine–dependent part in a "virtual machine" interface. This interface has been designed to encompass features that are unique to shared memory multiprocessors so that the upper levels of the kernel can exploit these features. Moreover, the kernel is designed and implemented in an object–oriented model, using the C++ language.

The port of the DASH kernel to a Sequent Multiprocessor system with currently 6 processors is partly finished. We have a running kernel which will allow us to conduct experiments on the multiprocessor to validate design decisions and measure the performance. Due to the encapsulation of the machine–dependent part into a "virtual machine" interface, the port of the kernel was focused on processor and virtual memory dependent modules. Machine–independent modules remained untouched.

5.3.4 *System Support for Managing a VLDS*

The requirements of monitoring a VLDS listed earlier are met by the hybrid, real–time monitor TMP (Test and Measurement Processor) concept [Wybranietz, Haban 1988]. In the current implementation in the INCAS project [Nehmer, 1987], the monitoring system traces the host system, evaluates and supervises the execution of applications, and displays information in real–time about their progress with minimal overhead (lower than 0.1%) and without interfering with the host system. This service is particularly important when monitoring long–running (concurrent) processes, complex distributed programs with dynamically changing structures, and systems with real–time constraints.

Due to the capability of the TMP to monitor and interact with the host node, it is possible to use it to perform certain operating system tasks, thus reducing the amount of overhead of the host processor [Haban, Wybranietz, Barak, 1988]. We describe four such tasks: the scheduling algorithm, with aids in managing tasks and messages; the load balancing algorithm, which is responsible for dispersing the load among the nodes; the remnant collection algorithm, responsible for removing any remnants due to remote node failures; and the process location algorithm, which helps in re–establishing communication links between processes. All these problems require extensive time overheads which lengthen task response times.

We investigated how the real–time monitor aids in scheduling tasks and messages based on given deadlines [Haban, Shin, 1989]. Ideally, every task should meet all of its timing constraints: start at a required time and produce results before a certain deadline. Particularly, knowledge of task execution times and system loads is a must for real–time system designers. A real–time monitor provides accurate measurements of execution times and timely checks of timing constraints for the scheduler. The remaining computation time and message deadlines can be determined with the aid of the monitor. Using all these timing measurements, we showed in [Shin, Muthuswamy, 1988] that messages can be scheduled on the polled bus to minimize the probability of missing deadlines. Given task timing constraints, monitoring the task execution behavior can be used to detect as early as possible whether or not certain tasks will miss their deadlines. The earlier the system finds these deficiencies, the more time is available for it to counter them.

The following algorithms represent a class of tasks that are performed by a system with a large number of nodes. Each such algorithm consists of several low–level tasks, e.g., monitoring the I/O rate of a process or sending *keep alive* messages, and a higher level tasks, e.g., process migration. The high–level procedures, which are invoked with a relatively low frequency, involve decision making that in many cases depends on some threshold functions. Updating these threshold functions requires that the system carry out the low–level tasks, which in turn are executed with relatively high frequency. It is at this level of activity that the TMP services are needed most. Since the TMP can monitor the node activities with minimal overhead, it can relieve the host processor of performing many of these functions, particularly network-related chores. The TMP performs the monitoring and the initial processing of the results. Then it may decide to initiate certain activities by triggering the operating system of the host processor or simply by updating its threshold functions for future use. The TMP can also send and receive information from other TMPs over the network, containing information about their respective hosts.

The load balancing algorithm consists of four parts [Barak, Shiloh, 1985]. In the first part, each monitor profiles the host system processes (length of the internal queues, CPU usage, rates of I/O and IPC). The second part of the algorithm includes load information exchanges. Using the communication links of the TMP, this load information can be exchanged among the TMPs by using some dissemination algorithm ([Allon, et al., 1987] and [Barak, Litman, 1985]), while at the same time, each TMP makes the information received available to the local operating system. The matchmaking part is done by comparing the information about the local load to that of other nodes, and by using some pre–specified decision (threshold) parameters [Lee, Toweley, 1986]. The role of the TMP at this level is limited to triggering process migrations due to special requirements by individual processes. The fourth part is the migration of the process to another node.

5.4 *Future Plans*

Project goals for 1989 include the following:

We will finish a prototype of the DASH subtransport layer, which will provide a flexible facility for secure real–time communication. We hope to begin using an FDDI network that will provide high-performance real–time network channels; DASH is designed to run on such a network.

We will complete enough of the VM system and service access mechanism to allow execution of programs from a file server. This will make DASH a self–sufficient system, and will facilitate subsequent experiments.

We will complete the Sequent port, and do a number of performance experiments involving kernel parallelism (in particular, TLB consistency on remapping and parallelism in network protocols).

We will tune up the performance of components that we have already implemented.

Many problems remain to be investigated in the area of real–time communication. Those that will be attacked in 1989 include:

(a) the buffer allocation and management policies suitable for a real–time service;
(b) the best ways to guarantee a given bound on delay variance or jitter;
(c) the introduction of security, fault tolerance, accounting, and charging capabilities into the design of a real–time service;
(d) a procedure for fast channel establishment, i.e., for setting up a channel while delivering the first packet of data;
(e) the feasibility of implementing real–time services in a virtual–circuit network consisting of Datakit or Datakit–like nodes.

To test and evaluate parts of the global system architecture, we have access to XUNET (the experimental university network set up by AT&T Bell Laboratories), a T1–speed virtual circuit network based on Datakit nodes, which is being upgraded to T3–speed, and to BERKOM, the Berlin metropolitan–area BISDN (broadband integrated services digital network).

We also plan to assess our designs and implementations by performing experiments that will allow us to:

(a) Evaluate and justify the decision of organizing the kernel as a large number of processes. Measure the overhead of synchronization and context switching, as well as the benefit of kernel parallelism.
(b) Evaluate the effectiveness of deadline–based scheduling on real–time communication.
(c) Evaluate different user process scheduling algorithms, concentrating on the effects of cache footprint phenomena due to process migration.
(d) Study the performance of locking mechanisms. In particular, measure the performance of different implementations of spin locks (e.g., the memory traffic caused by the cache consistency protocol while a processor is spinning). Identify synchronization hot spots, and modify the design the eliminate them.
(e) Measure the performance of the DASH kernel on the Sequent multiprocessor.
(f) Validate the design decisions for the multiprocessor version.

Research on TLB Inconsistency in Shared–memory Multiprocessors

A translation–look–aside buffer (TLB) is a fast storage that keeps frequently used page table entries in a processor. There are multiple TLB's on a shared–memory multiprocessor, but the hardware usually does not ensure consistency among them. In this work, we will develop software algorithms for synchronizing TLB's, and analyze their performance. We conjecture that maintaining full TLB consistency by software is expensive.

We will study various types of consistency semantics, their relationship with correctness, and their relationship with mutual trust between processes. In many situations, TLB inconsistency does not cause nonrecoverable errors. Our goal is to reduce the overhead of synchronizing TLB's by tolerating TLB inconsistency and using weaker consistency semantics.

This work is important to DASH because DASH moves large amounts of data by virtual memory remapping, which changes the page table and causes TLB inconsistency. We will integrate the research results into the design and implementation of the DASH virtual memory system and message-passing system.

6 Research Environment

As part of its commitment to advanced research, ICSI strives to maintain the highest quality environment. The Institute occupies a newly designed 14,000 square foot research facility at 1947 Center Street, just off the central UC campus in downtown Berkeley. There is an expansion underway which will almost double the useable research space. The facility is completely wired for high-speed communications and contains its own library, conference rooms, computer room and hardware laboratory. There are specially equipped laboratories for research in vision, sound and circuit design and synthesis. Researchers are equipped with advanced workstations and the purchase of a large multiprocessor is being studied. The communication link from ICSI to the UC campus has been upgraded to a microwave system with greatly expanded bandwidth. Through its link to UCB, the Institute has access to a wide variety of campus computers and to national and international networks. The campus library system provides another invaluable resource to Institute researchers.

The Berkeley environment is also rich in human resources. ICSI is most closely affiliated with the Computer Science Division, with a large range of joint and split appointments. But because of the nature of its research, the Institute has also developed strong ties with several other departments and programs in engineering and the mathematical, physical, life and cognitive sciences. Most of these programs are quite distinguished and the collaborations are providing a major source of ideas and critiques. Students from several UCB departments are finding ICSI the optimal environment to pursue their research. Students, post-doctoral fellows and visiting scientists from other institutions and countries help add to the lively intellectual atmosphere. The Institute has an active program of seminars (Section 7) and its members have access to the vast array of technical opportunities in the Bay Area. One most important, aspect of the ICSI research environment is the Institute's administrative staff which is extremely efficient at protecting the scientists from unnecessary overhead. Another important part of the ICSI research environment is the inclusion of additional research projects not included in the core program.

The main focus of the Institute's program is centered on the core problems of large scale distributed and parallel computation. The groups described in Sections 2–5 are not rigidly defined and many projects cut across group boundaries. There are also smaller projects which do not fit within the main theme, but are pursued because of special opportunities for international collaboration. One such ongoing effort is in the area of advanced data bases. The presence of these additional projects at the Institute is an important part of the overall research environment.

One effort is a cooperative research program on extended data base systems being pursued jointly by researchers from the University of California, Berkeley and ones from the Integrated Publication and Information Systems Institute (IPSI) of GMD in Darmstadt. Its main points are a program to continue cooperation started in 1988, a program to explore extensions to POSTGRES and a program to explore parallelism in POSTGRES and in object–oriented data bases in general.

During 1988 a cooperative program was begun to explore connecting the VODAK object–oriented data model to the POSTGRES data base management system. In effect, POSTGRES would be required to efficiently simulate the VODAK data model. The purpose of this exploration was to stress test the POSTGRES data model with a difficult object modelling problem as well as to stimulate increased cooperation between IPSI and Berkeley. Under this project, Mr. Wolfgang Klas visited Berkeley for two periods totally 3 months during 1988.

There are many application specific data models that have been proposed in the literature. Most attempt to capture richer semantics than would be appropriate for a general purpose data model. VODAK [KLAS88] is an example of such a data model. Others include SDM [HAMM81] and CAD [BATO85]. It is unlikely that custom data base systems will be constructed to support such data models; rather they will be simulated on top of other more general purpose data models. POSTGRES [STON86, WENS88] is an example next–generation data base management system being constructed at Berkeley. It is intended to be a general purpose DBMS that is applicable to a myriad of applications for which current commercial relational systems are not suitable. The main points that POSTGRES exploits are:

an integrated rules system [STON86b]
a rich type system with inheritance [ROWE87]
a no–overwrite storage manager with versioning [STON87]

Other systems with similar objectives but different approaches include IRIS [FISH87], STARBURST [LIND87], ORION [BANE87] and EXODUS/EXTRA [CARE86]. Of these POSTGRES is by far the most ambitious project.

We now have POSTGRES working well enough to desire users to stress test the system and give us some feedback on the function which we have implemented. During his initial visit, Mr. Klas concluded that it was feasible to implement VODAK on top of POSTGRES [KLAS88b]. During the next phase we plan to move ahead with a real implementation. This will give us feedback on the viability and robustness of our system as well as give IPSI a running system on which to base their application programs which will use VODAK.

Smaller projects such as the database work above as well as seminars and meetings add to the rich scientific life at the Institute. The ICSI scientific staff is well aware of the special quality of its environment. There is a uniform commitment to the highest standard of basic research in areas of great scientific and practical importance. The emphasis is on working together, both within the Institute and with international partners. ICSI remains an experiment in cooperation in a highly competitive world and its scientists are determined to justify the privileges that they have been granted by their contributions to scientific progress and to international understanding.

7 Lectures at ICSI – 1988

Dr. Amnon Barak, The Hebrew University of Jerusalem and ICSI, "MOSIX: An Integrated Multicomputer UNIX System."

Dr. Joachim Beer, International Computer Science Institute, "A Risc Pipeline."

Dr. Avrim Blum, MIT Laboratory for Computer Science, "Training a 3–Node Neural Network is NP–Complete."

Dr. Lenore Blum, International Computer Science Institute, "Most Julia Sets are Undecidable or Some Connections Between Dynamical Systems and the Theory of Computation."

Dr. Benny Chor, Department of Computer Science, Technion, "Towards a Theory of Average Case Complexity: An in depth study."

Dr. Mark Derthick, Department of Computer Science, Carnegie-Mellon University, "Mundane Reasoning."

Dr. Joachim Diederich, International Computer Science Institute, "Connectionist Recruitment Learning."

Dr. H.P. Frei, Swiss Federal Institute of Technology, Zurich, Switzerland, "Concept Spaces in Information Retrieval."

Dr. Feng Gao, University of California, Berkeley, "Communication Complexity of Parallel Computation."

Dr. Oded Goldreich, Israel Institute of Technology, Haifa, Israel "Towards a Theory of Average Case Complexity (a Survey)."

Dr. Shafi Goldwasser, Computer Science Department, Massachusetts Institute of Technology. "Theorems for Non–Cryptographic Fault–Tolerant Distributed Computation."

Dr. Dieter Haban, International Computer Science Institute, "Very Large Distributed Systems."

Prof. David Haussler, University of California, Santa Cruz, "A Computational Learning Theory of Multilayer 'Neural' Networks."

Dr. Geoffrey Hinton, University of Toronto, "How to Make Boltzmann Learning Efficient."

Dr. Marek Karpinski, International Computer Science Institute and University of Bonn, (1) "Parallel Complexity of Algebraic Interpolation Problems," and (2) "Parallel Complexity of Perfect Matching and Density of Roots of Exponential Univariate Determinants over Finite Fields."

Dr. Christof Koch, California Institute of Technology, "Computing Motion in Two Massive Parallel Structures: Resistive Networks and the Primate Visual System."

Dr. Marc Linster, GMD, Sankt Augustin, "A Critical Look at KRITON: Moving from First to Second Generation."

Dr. Mike Luby, International Computer Science Institute, "Fast Algorithms for Parallel RAMs."

Dr. Michael Luby, University of Toronto and ICSI, "P Meets the Permanent (a Survey)."

Dr. H.G. Metzler, Daimler–Benz, West Germany, "Mobile Robot Research at Daimler–Benz."

Dr. John E. Moody, Department of Computer Science, Yale University, "Speedy Alternatives to Back Propagation."

Dr. Nelson Morgan, International Computer Science Institute, "Applications of Neural Networks: Some Real Signal Processing Problems."

Drs. M. Naillon and A. Sirat, Laboratoires d'Electronique et de Physique Appliquee Limeil Brevannes Cedex, France, "Neural TV Image Compression Using Hopfield Type Networks."

Dr. Erich J. Neuhold, INIS, NJIT and IPSI, GMD, "Using Multiple Databases."

Dr. Stephen Omohundro, International Computer Science Institute, "Geometric Algorithms and Networks for Vision."

Dr. Stephen Omohundro, Center for Complex Systems Research, University of Illinois at Urbana-Champaign, "Geometric Learning by Neurons and Algorithms."

Prof. John Reif, Department of Computer Science, Duke University, "Nested Annealing: A Provable Improvement on Simulated Annealing."

Dr. Dietmar Roesner, Baden–Wuertenberg AI Center Ulm, West Germany, "Natural Language Generation."

Dr. Katsunori Shimohara, NTT Human Interface Laboratories, Kangawa, Japan, "Back-Propagation Networks for Event-Driven Temporal Sequence Processing."

Dr. Barbara Simons, IBM Research Center, Almaden, California, "On Pipeline Scheduling."

Dr. Paul Smolensky, Institute of Cognitive Science, University of Colorado, Boulder, "Parallel Distributed Processing of Symbolic Structure."

Dr. Erich J. Smythe, Indiana University, Bloomington, "Syllable Recognition through the Identification of Formant Transitions in a Connectionist Network."

Dr. Gerald J. Tesauro, Center for Complex Systems Research, University of Illinois at Urbana-Champaign, "A Parallel Network that Learns to Play Backgammon: Recent

Results."

Dr. Alex Waibel, Carnegie-Mellon University, "Modularity in Phonetic Neural Networks."

Dr. Christian J. Wellekens, Philips Research Laboratory, Brussels, "Links Between Hidden Markov Models and Multilayer Perceptrons."

8 ICSI Technical Reports: 1987 – 1988

1987 Technical Reports

TR–87–001
Vijaya Ramachandran, "A Minimax Arc Theorem for Reducible Flow Graphs," November 24, 1987, 11 pages, $1.75.

1988 Technical Reports

TR–88-001
Erich Neuhold and Michael Stonebraker, "Future Directions in DBMS Research," February 4, 1988, 10 pages, $1.00.

TR–88-002
Oliver Gunther, "The Cell Tree: An Index for Geometric Databases," June 29, 1988, 20 pages, $1.75.

TR–88–003
Heinz Beilner, "Measuring with Slow Clocks," July 1, 1988, 21 pages, $1.75.

TR–88–004
Amnon Barak and Richard Wheeler, "MOSIX: An Integrated UNIX for Multiprocessor Workstations," October 25, 1988, 16 pages, $1.75.

TR–88–005
Kang Shin and Dar–Tzen Peng, "Static Allocation of Periodic Tasks with Precedence Constraints in Distributed Systems," October 21, 1988, 40 pages, $2.50.

TR–88–006
Kang Shin and Yi–Chieh Chang, "Load Sharing in Distributed Real–Time Systems with Broadcast State Changes," October 31, 1988, 46 pages, $2.50.

TR–88–007
Dieter Haban, Dieter Wybranietz, and Amnon Barak, "Monitoring and Management–Support of Distributed Systems," November, 1988, 29 pages, $1.00.

TR–88–008
Herve Bourlard and C.J. Wellekens, "Links Between Markov Models and Multilayer Perceptrons," November 10, 1988, 27 pages, $1.75.

TR–88–009
Manual Blum, "Designing Computers to Check Their Work," November 15, 1988, 21 pages, $1.75.

TR-88-010

Joachim Diederich, "Knowledge–Intensive Recruitment Learning," November 16, 1988, 32 pages, $2.50.

TR-88-011

Jerome A. Feldman, "Time, Space and Form in Vision," December 1, 1988, 70 pages, $3.00.

TR-88-012

Lenore Blum, Mike Shub, and Steve Smale, "On a Theory of Computation and Complexity Over the Real Numbers; NP Completeness, Recursive Functions and Universal Machines," December 1, 1988, 77 pages, $3.00.

TR-88-013

Manual Blum and Sampath Kannan, "Program Correctness Checking and the Design of Programs that Check Their Work," December 1, 1988, 16 pages, $1.75.

1989 Technical Reports

TR-89-001

Domenico Ferrari, "Guaranteeing Performance for Real–Time Communication in Wide–Area Networks," January 1, 1989, 20 pages, $1.75.

9 References

[Ajtai, Ben-Or, 1984] Ajtai, M. and M. Ben-Or, "A Theorem on Probabilistic Constant Depth Computations," *Proc. 16th Ann. ACM Symp. on Theory of Computing*, 1984, 471-474.

[Ajtai, Wigderson] Ajtai, M. and A. Wigderson, "Deterministic Simulation of Constant Depth Circuits," *Proc. 26th Ann. IEEE Symp. on Foundations of Computer Science*, 1985, 11-19.

[Allen, 1985] Allen, J.F., "Maintaining Knowledge about Temporal Intervals," in *Readings in Knowledge Representation*, R. Brachman and H. Levesque (Eds.), Morgan Kaufman Publishers, Inc. 1985.

[Alon, Barak, Manber, 1987] Alon, N., Barak, A., and Manber, U., "On disseminating information reliably without broadcasting," *Proc. 7th Int. Conf. on Distributed Computing Systems*, pp. 74-81, Berlin, Sept. 1987.

[Anderson, 1988] Anderson, D.P., "A Software Architecture for Network Communication," *Proc. 8th International Conference on Distributed Computing Systems*, San Jose, California, June 1988.

[Anderson, Ferrari, 1987] Anderson, D.P., D. Ferrari, P.V. Rangan, and B. Sartirana, "A Protocol for Secure Communication in Large Distributed Systems," *Proc. 7th International Conference on Distributed Computing Systems*, Berlin, September 1987.

[Anderson, Ferrari, 1988] Anderson, D.P. and D. Ferrari, "The DASH Project: An Overview," Rept. No. UCB/CSD 88/405, U. Berkeley, February 1988.

[Anderson, Ferrari, Rangan, 1987] Anderson, D., D. Ferrari, and P.V. Rangan, "Subtransport level: the right place for end-to-end security mechanisms", *Technical Report No. UCB/Computer Science Dept. 87/346*, Computer Science Division, EECS, UCB, Berkeley, CA, March, 1987.

[Anderson, Tzou, 1988] Anderson, D. and S.-Y. Tzou, "The DASH Local Kernel Structure," *Technical Report No. UCB/Computer Science Dept. 88/463*, Computer Science Division, EECS, UCB, Berkeley, CA, November, 1988.

[Anderson, Tzou, Graham, 1988] Anderson, D., S.-Y. Tzou, and G.S. Graham, "The DASH virtual memory system, *Technical Report No. UCB/Computer Science Dept. 88/461*, Computer Science Division, EECS, UCB, Berkeley, CA, November 1988.

[Anderson, Wahbe, 1988] Anderson, D. and R. Wahbe, "The DASH Network Communication Architecture," *Technical Report No. UCB/Computer Science Dpt. 88/462*, Computer Science Division, EECS, UCB, Berkeley, CA, November 1988.

[Angluin, 1978] Angluin D., "On the Complexity of Minimum Inference of Regular Sets, Information and Control," 39(3): 337-350, 1978.

[Angluin, Smith, 1983] Angluin, D. and C.H. Smith, "Inductive Inference: Theory and Methods," *Computer Surveys*, 15, No. 3, 237-269, Sept. 1983.

[Angluin, 1987] Angluin D., "Learning Regular Sets from Queries and Counterexamples," *Information and Computation*, 75, 87-106, 1987.

[Azim, Shung, Brodersen, 1988] Azim, S.K., C.S. Shung, R.W. Brodersen, "Automatic Generation of a Custom Digital Signal Processor for an Adaptive Robot Arm Controller," *Proc. IEEE Conf. Acoustics, Speech & Signal Processing*, New York, Vol. 4, 2021-2024, 1988.

[Ballard, Gardner, Srinivas, 1987] Ballard, D.H., P.C. Gardner, and M.A. Srinivas, "Graph Problems and Connectionist Architectures," *Technical Report 167* (revised), Computer Science Department, U. Rochester, 1987.

[Barahona, 1982] Barahona, F., "On the Computational Complexity of Ising Spin Glass Models," J. Phys. A: Math. Gen., 15:3241-3253, 1982.

[Barak, Litman, 1985] Barak, A. and A. Litman, "MOS: A Multicomputer Distributed Operating System," *Software Practice & Experience, Vol. 15, No. 8*, August 1985.

[Barak, Shiloh, 1985] Barak, A. and A. Shiloh, "A Distributed Load Balancing Policy for a Multicomputer," *Software Practice & Experience, Vol. 15, No. 9*, September 1985.

[Bastian, 1986] Bastian, M., "Voice-Data Integration: an Architecture Perspective," *IEEE Commun. Mag.*, Vol. 14, No. 7, July, 1986.

[Baum, Haussler, 1988] Baum, E.B., and D. Haussler, "What Size Net Gives Valid Generalization?", submitted *Neural Computation*, November, 1988.

[Beame, Luby, 1989] Beame, P., and M.G. Luby, "Parallel Search for Maximal Independence given Minimal Independence," submitted to *ACM Symposium on Parallel Algorithms and Architectures*, converence to be held in Santa Fe New Mexico, June 1989, to appear as an I.C.S.I. technical report.

[Beer, 1987] Beer, J., "Concepts, Design, and Performance Analysis of a Parallel Prolog Machine," Ph.D. Dissertation, Technical U. Berlin, 1987.

[Ben–Or, Cleve, 1988] Ben–Or, M. and R. Cleve, "Computing Algebraic Formulas Using a Constant Number of Registers," *Proc. 20th Ann. ACM Symp. on Theory of Computing*, 254-257, 1988.

[Ben–David, Chor, Goldreich, Luby, 1989] Ben–David, S., B. Chor, O. Goldreich, M.G. Luby, "Towards a Theory of Average NP," *21st Annual ACM Symposium on Theory of Computing*, 1989, accepted to *1989 Structures of Complexity Theory*, to appear in *ICSI Technical Report.*

[Blum, 1988] Blum, M "Designing Programs to Check Their Work," *ICSI Technical Report TR-88-009*, November 1988, 21 pp. Submitted for publication in CACM.

[Blum, Blum, 1975] Blum, L. and M. Blum, "Towards a Mathematical Theory of Inductive Inference," *Information and Control*, 20, 125-155, June 1975.

[Blum, Blum, Shub, 1982] Blum, L., M. Blum, and M. Shub, "A Simple Secure Unpredictable Pseudo-Random Number Generator," *SIAM J. on Computing*, Vol. 15, 1986, 364-383. A preliminary version appeared in *Proc. Crypto* , 1982.

[Blum, Kannan, 1989] M. Blum and S. Kannan, "Program Correctness Checking... and the design of programs that check their work", ICSI Technical Report TR-88-013 (December 1988), 15 pp. Accepted for *21st Annual Symposium on Theory of Computing*, May 1989.

[Blum, Shub, Smale, 1988a] Blum, L., M. Shub, and S. Smale, "On a Theory of Computation over the Real Numbers; NP Completeness, Recursive Functions and Universal Machines," *FOCS*, 1988.

[Blum, Shub, Smale, 1988b] "On a Theory of Computation and Complexity over the Real Numbers: NP–Completeless, Recursive Functions and Universal Machines," ICSI TR-88-012, December, 1988. To appeal in the Bulletin of the American Mathematical Society, July 1989.

[Blum, Micali, 1982] Blum, M. and S. Micali, "How to Generate Cryptographically Strong Sequences of Pseudo-Random Bits," *SIAM J. on Computing*, Vol. 13, 1984, 850-864. A preliminary version appeared in *Proc. 23rd Ann. IEEE Symp. on Foundations of Computer Science*, 1982, 112-117.

[Blum, Raghavan, 1988] Blum, M. and P. Raghavan, "Program Correctness: Can One Test for It?", IBM T.J. Watson Research Center Technical Report RC 14038 (#62902), (September 1988), 12 pp. Invited to IFIP 89 by Prof. Jozef Gruska, Chairman of the area "Fundamental Tools."

[Blum, Rivest, 1988] Blum, A. and R. Rivest, "Training a 3-Node Neural Network is NP–Complete," COLT, 1988, *Proceedings*, MIT, August 1988.

[Bourlard, Wellekens] Bourlard, H. and C.J. Wellekens, "Speech Pattern Discrimination and Multilayer Perceptrons," *Computer, Speech, and Language*, in press.

[Bourlard, Wellekens] Bourlard, H. and C.J. Wellekens, "Links Between Markov Models and Multilayer Perceptrons," *Proc. Intl. Neural Network Conference*, Boston, 1988.

[Braitenberg, 1984] Braitenberg, V., "Vehicles: Experiments in Synthetic Psychology," MIT Press, Cambridge, MA, 1984.

[Broder, 1986] Broder, A., "How Hard is it to Marry at Random? (On the Approximation of the Permanent," *Proc. 18th Ann. ACM Symp. on Theory of Computing*, 1986, 50-58.

[Brown, 1988] Brown, C.M. (Ed.), "Advances in Computer Vision," Vols. I and II, Lawrence Erlbaum Associated, Publishers, to appear, 1988.

[Bukys, 1986] Bukys, L., "Connected Component Labeling and Border Following on the BBN Butterfly Parallel Processor," *Butterfly Project Report 11*, Computer Science Dept., U. Rochester, October 1986.

[Canny, 1987] Canny, J., *The Complexity of Robot Motion Planning*, Ph.D., Thesis, MIT, 1987.

[Canny, 1988] Canny, J. "Some Algebraic and Geometric Computations in PSPACE," *20th Annual ACM Symposium on Theory of Computing*, 1988.

[Cheriton, Zwaenepol, 1983] Cheriton, D.R. and W. Zwaenepoel, "The Distributed V Kernel and its Performance for Diskless Workstations," *Proc. of the 9th ACM Symp. on Operating System Prin.*, Bretton Woods, New Hampshire, October, 1983.

[Cleve, 1989] Cleve, R., *Methodologies for Designing Block Ciphers and Cryptographic Protocols*, Ph.D., Thesis, University of Toronto.

[Cleve, Dagum, 1989] Cleve, R. and Dagum, P.,"A Constructive $\Omega(t^{1.26})$ Lower Bound on the Ramsey Number $R(3,t)$," *TR–89–009, International Computer Science Institute*, Berkeley, CA, 1989.

[Cooper, Hollbach, 1987] Cooper, P.R. and S.C. Hollbach, "Parallel Recognition of Objects Comprised of Pure Structure," *Proc. DARPA Image Understanding Workshop*, Los Angeles, February 1987.

[Dagum, 1988b], Dagum, P. "A Linear-time Algorithm for Enumerating the Perfect Matchings in Skew Bipartite Graphs", *submitted to J. of Algorithms.*

[Dagum, 1988c], Dagum, P. (1988c), "The Polytope of Linear Extensions is Magnifying", *manuscript.*

[Dagum, Luby, 1989a] Dagum, P. and M. Luby, "Approximating the Permanent of Graphs with Large Factors," *paper in preparation.*

[Dagum, Luby, Mihail, 1988] Dagum, P., M. Luby, M. Mihail, and U. Vazirani, "Polytopes, Permanents and Graphs with Large Factors," to appear in *Proc. 29th Ann. IEEE Symp. on Foundations of Computer Science*, 1988.

[Dahlhaus, Hajnal, Karpinski, 1988] Dahlhaus, E., P. Hajnal, and M. Karpinski, "Optimal Parallel Algorithms for the Hamiltonian Cycle Problem on Dense Graphs, *Proc. 29th IEEE FOCS*, 1988.

[Dahlhaus, Karpinski, Lingas, 1989] Dahlhaus, E., M. Karpinski, and A. Lingas, "A Parallel Algorithm for Maximum Matching in Planar Graphs," *Research Report No. 8532-CS*, University of Bonn, 1989.

[Devroye, 1986] Devroye, Luc, "Lecture Notes on Bucket Algorithms," Birkhauser Boston Inc., Boston, 1986.

[Diederich, 1986] Diederich, J., "Kognitive Parallelverarbeitung," GMD Bericht 159, R. Oldenbourg Verlag, Muenchen Wien, 1986.

[Diederich, 1988] Diederich, J., "Aspekte des Konnektionismus," *GMD Technical Report 319*, Sankt Augustin, July 1988.

[Diederich, 1988] Diederich, J., "Connectionist Recruitment Learning," *European Conference on Artificial Intelligence, ECAI-88*, Munich, August 2–5, 1988.

[Diederich, 1988d] Diederich, J., "Knowledge Intensive Learning in Neural Networks," Poster, *IEEE ICNN-88*, San Diego, 1988.

[Diederich, 1988f] Diederich, J., "Knowledge–Intensive Rectuitment Learning," *TR-88-010, International Computer Science Institute*, Berkeley, November, 1988.

[Diederich, 1988] Diederich, J., "Simulation Schizophrener Sprache," *ISBN 3-531-11908-7*, Westdeutscher Verlag, Opladen, 1988.

[Diederich, 1988] Diederich, J., "Spreading Activation and Renshaw Inhibition in Neural Networks," Poster, *IEEE, ICNN-88*, San Diego, 1988.

[Diederich, 1988i] Diederich, J., "Steps Toward Knowledge–Intensive Connectionist Learning," J. Barnden & J. Pollack, (Eds.), *Advances in Connectionist and Neural Computation*, Ablex. Pub., 1988.

[Diederich, 1988j] Diederich, J., "Knowledge Representation and Learning in a Structured Neural Network," J. Kindermann & C. Lischka, (Eds.), *Workshop Konnektionismus, 47-62, GMD Technical Report 329*, Sankt Augustin, 1988.

[Diederich, 1988k] Diederich, J., "Knowledge Representation and Recruitment Learning in a Connectionist Inheritance Network," to appear in: J. Pollack and J. Barnden (Eds.), *Advances in Connectionist and Neural Computation Theory*, Ablex Publishers, 1988.

[Diederich, 1988] Diederich, J., "Trends im Konnektionismus," *Kuenstliche Intelligenz, Vol. 1, No. 2*, 1988.

[Diederich, 1988] Diederich, J., "Wissenserwerb und Wissensrepresentation in KRITON," Heyer & Krems, (Eds.), *Wissensarten und ihre Darstellung*, Springer Verlag, Berlin Heidelberg, New York, 1988.

[Diederich, 1989a] Diederich, J., "Learning by Being Told," in connectionist systems, submitted to *IJCAI-1989.*

[Diederich, 1989] Diederich, J., "Spreading Activation in Multiple Concept Hierarchies," submitted to *IJCAI-1989.*

[Diederich, 1989c] Diederich, J., "Spreading Activation and Connectionist Systems for Natural Language Processing." *ICSI-TR-89-008, International Computer Science Institute*, Berkeley, February, 1989, to appear: *Theoretical Linguistics*, 1989.

[Diederich, Linster, 1988] Diederich, J. and M. Linster, "Knowledge-Based Knowledge Elicitation," in *Topics in Expert System Design*, G. Guida and C. Tasso (Eds.), North Holland, Amsterdam, 1988.

[Diederich, Ruhmann, May, 1988] Diederich, J., I. Ruhmann, M. May, "KRITON: A Knowledge Acquisition Tool for Expert Systems," J.H. Boose, G. Gaines, (Eds.), *Knowledge Acquisition Tools for Expert Systems. Knowledge-Based Systems, Volume 2, 83-94*, Academic Press, London, San Diego, New York, Berkeley, Boston, Sydney, Tokyo, Toronto, 1988.

[Edelsbrunner, van Leewen, 1983] Edelsbrunner, H. and J. van Leewen, "Multidimensional Data Structures and Algorithms." A bibliography, IIG, Technische U. Graz, Austria, Rep., 104, 1983.

[Erdös, 1947], Erdös, P., "Some Remarks on the Theory of Graphs", *Bull. Amer. Math. Soc.*, 1947.

[Fanty, 1985] Fanty, M.A., "Context-Free Parsing in Connectionist Networks," *TR 174*, Computer Science Dept., U. Rochester, 1985.

[Fanty, 1986] Fanty, M.A., "A Connectionist Simulator for the BBN Butterfly Multiprocessor," *TR 164*, Computer Science Dept., U. Rochester, 1986.

[Fanty, Goddard, 1987] Fanty, M.A. and N. Goddard, "The Connectionist Simulator User's Manual," Computer Science Dept., U. Rochester, 1987.

[Fanty, 1988] Fanty, M.A., "Learning in Structured Connectionist Networks," Ph.D. Thesis and *TR 252*, Computer Science Dept., U. Rochester, 1988.

[Feldman, 1972] Feldman, J.A., "Some Decidability Results on Grammatical Inference and Complexity," *Information and Control*, 20, 244-262, 1972.

[Feldman, 1985] Feldman, J.A., "Connectionist Models and Parallelism in High Level Vision," CVGIP 31, Special Issue on Human and Machine Vision, 178-200, 1985.

[Feldman, 1986] Feldman, J.A., "Neural Representation of Conceptual Knowledge," to appear, forthcoming book, L. Nadel et.al. (Eds.), 1987; also, *TR 189*, Computer Science Dept., U. Rochester, June 1986.

[Feldman, 1-1987] Feldman, J.A., "A Functional Model of Vision and Space," in *Vision, Brain and Cooperative Computation.* M. Arbib and A. Hanson (Eds.), Cambridge, MA: MIT Press/Bradford Books, 1987.

[Feldman, 2-1987] Feldman, J.A., "Connectionist Representations of Concepts," in *Connectionist Models and Their Applications*, D. Waltz and J.A. Feldman (Eds.), Ablex Publishing Corp., 1987.

[Feldman, 3-1987] Feldman, J.A., "Energy Methods in Connectionist Modelling," in *Pattern Recognition, Theory and Applications*, P.A. Devijver (Ed.), *NATO ASI Series in Computer Science*, Springer Verlag, 1987.

[Feldman, 4-1987] Feldman, J.A., "Massively Parallel Computational Models," in forthcoming book, M. Commons (Ed.) Cambridge, MA: Harvard University Press, in press, 1987.

[Feldman, 1988] Feldman, J.A., "Time Space and Form in Vision," *TR-88-011*, International Computer Science Institute, Berkeley, California, November, 1988.

[Feldman, Ballard, Brown, 1985] Feldman, J.A., D.H. Ballard, C.M. Brown, and G.S. Dell, "Rochester Connectionist Papers: 1979-1985," *TR 172*, Computer Science Dept., U. Rochester, December 1985.

[Feldman, Fanty, 1988] Feldman, J.A., M.A. Fanty, N. Goddard, and K. Lynne, "Computing with Structured Connectionist Networks," *Communications of the ACM*, 1988.

[Ferrari, 1989] Ferrari, D., "Guaranteeing Performance for Real–Time Communication in Wide–Area Networks," *Technical Report No. ICSI TR-89-001, International Computer Science Institute*, Berkeley, CA, January 1989.

[Fiat, Karp, Luby, 1988] Fiat, A., R.M. Karp, M.G. Luby, L. McGeoch, D. Sleator, N. Young, "Competitive Paging Algorithms," submitted to *J. Algorithms*, 1988.

[Floyd, 1988] Floyd, S., "Space–bounded Learning and the Vapnik–Chervonenkis Dimension," preprint, 1988.

[Friedman, Bentley, Finkel, 1977] Friedman, J. H., J.L. Bentley, and R. A. Finkel, "An Algorithm for Finding Best Matches in Logarithmic Expected Time," *ACM Transactions on Mathematical Software*, 3:3, 1977, 209-226.

[Gajski, Peir, 1985] Gajski, D.D. and J.K. Peir, "Essential Issues in Multiprocessor Systems," *IEEE Computer*, 2-38, June 1985.

[Gao, 1988] Gao, F., "On the Trade–off between Communication Time and Multiprocessor Connectivity for Parallel Computation," preprint, February 1988.

[Geman, Geman, 1984] Geman, S. and D. Geman, "Stochastic Relaxation, Gibbs Distributions, and the Bayesian Restoration of Images," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6(6): 721-741, 1984.

[Gevins, Morgan, 1984] Gevins, A.S. and N.H. Morgan, "Ignorance-based systems," *Proc. IEEE Conf. Acoustics, Speech & Signal Processing*, San Diego, Vol. 3, 39A.5.1–39A.5.4., 1984.

[Gevins, Morgan, 1986] Gevins, A.S. and N.H. Morgan, "Classifier–Directed Signal Processing in Brain Research," *IEEE BME–33, #12*, December 1986.

[Gevins, Morgan, 1988] Gevins, A.S. and N.H. Morgan, "Applications of Neural–Network (NN) Signal Processing in Brain Research," *IEEE ASSP Trans.*, Vol. 36 (7), 1152-1161, 1988.

[Gibbons, 1989] Gibbons, P.B., "A More Practical PRAM Model," manuscript, January, 1989, submitted to *ACM Symposium on Parallel Algorithms and Architectures.*

[Gibbons, Karp, Miller, 1988] Gibbons, R., R.M. Karp, G.L. Miller, D. Soroker, "Subtree Isomorphism is in Random NC," *Proc. of Aegean Workshop on Computing,* June, 1988; submitted to *J. Algorithms.*

[Gibbons, Karp, Ramachandran, 1988] Gibbons, R., R.M. Karp, V. Ramachandran, D. Soroker, and R. Tarjan, "Transitive Reduction in Parallel via Branchings," *Report No. UILU-ENG-88-2254*, Coordinated Science Laboratory, Applied Computation Theory, U. of Illinois at Champaign–Urbana, October 1988.

[Godbeer, 1987] Godbeer, G. *The Computational Complexity of the Stable Configuration Problem for Connectionist Models*, M.Sc. Thesis, U. Toronto, 1987.

[Godbeer, Lipscomb, Luby, 1988] Godbeer, G.H., J. Lipscomb, M. Luby, "On the Computational Complexity of Finding Stable State Vectors in Connectionist Models (Hopfield Nets)," *Technical Report 208/88*, March 1988.

[Goddard, Lynne, 1987] Goddard, N., K. Lynne, and T. Mintz, "The Rochester Connectionist Simulator," *TR 233*, Computer Science Department, U. of Rochester, 1987.

[Gold, 1967] Gold, E.M., "Language Identification in the Limit," *Information and Control*, 10, 447-474, 1967.

[Gold, 1978] Gold, E.M., "Complexity of Automaton Identification from Given Data," *Information and Control*, 37, 302-320, 1978.

[Goldberg, Spencer, 1987] Goldberg, M. and T. Spencer, "A New Parallel Algorithm for the Maximal Independent Set Problem," *Proc. 29th IEEE FOCS*, 1987, 161-165.

[Goldreich, Krawczyk, Luby, 1988] Goldreich, O., H. Krawczyk, and M. Luby, "On the Existence of Pseudo–random Number Generators," to appear in *Proc. 29th Ann. IEEE Symp. on Foundations of Computer Science*, 1988.

[Goldreich, Levin, 1989] Oldreich, O., and L. Levin, "A Hard–Core Predicate for all One–Way Functions," *21st ACM Symposium on Theory of Computing*, 1989.

[Grigoriev, Karpinski, 1987] Grigoriev, D.Y. and M. Karpinski, "The Matching Problem for Bipartite Graphs with Polynomially Bounded Permanents is in NC," *Proc. 28th Annual IEEE Symp. on Foundations of Computer Science*, 166-172, 1987.

[Grigoriev, Karpinski, 1988] Grigoriev, D.Y. and M. Karpinski, "A Deterministic Algorithm for Rational Functions Interpolation," *Research Report No. 8526-CS*, U. Bonn, 1988.

[Grigoriev, Karpinski, Singer, 1988] Grigoriev, D.Y., M. Karpinski, and M. Singer, "Fast Parallel Algorithms for Sparse Multivariate Polynomial Interpolation Over Finite Fields," *Research Report No. 8523-CS*, U. Bonn, 1988.

[Gschwendtner, 1988] Gschwendtner, A., (Ed.), *DARPA Neural Net Report*, June 1988.

[Haban, 1987] Haban, D., "DTM – A Method for Testing Distributed Systems," *Proc. of the 6th IEEE Symp. on Reliability in Distributed Software and Database Systems*, Williamsburg, Virginia, March 1987.

[Haban, Shin, 1989] Haban, D. and K.G. Shin, "Monitoring Distributed Real–Time Systems and its Applications," *ICSI*, submitted for publication.

[Haban, Weigel, 1988] Haban, D., and W. Weigel "Global Events and Global Breakpoints in Distributed Systems," *Proc. of the 21st. Annual Hawaii Int. Conf. on System Sciences*, Hawaii, January 1988.

[Haban, Wybranietz, Barak, 1988] Haban, D., D. Wybranietz, and A. Barak, "Monitoring and Management-support of Distributed Systems," *Report No. TR-88-007 ICSI*, Berkeley, (submitted for publication), November 1988.

[Haken, Luby, 1988] Haken, A. and M. Luby, "Steepest Descent Can Take Exponential Time for Symmetric Connection Networks," accepted for publication in *Complex Systems*, 1988.

[Hastad, 1986], "Almost Optimal Lower Bounds for Small Depth Circuits", *Proc. 18th Ann. ACM Symp. on Theory of Computing*, 6–20.

[Hill, *et al.*, 1986] Hill, M., *et al.*, "Design Decisions in SPUR," *IEEE Computer*, 8-22, November 1986.

[Hillis, 1985] Hillis, D., "The Connection Machine," MIT Press, 1985.

[Hinton, 1986] Hinton, G.E., "Connectionist Learning Procedures," *AI Journal*, in press, 1986.

[Hodges, 1986] Hodges, D., "An Approach to Fiber Optic Data/Voice/Video LAN," *IEEE INFOCOM* 86, 516-523, 1986.

[Hollbach, 1986] Hollbach, S.C., "Tinker Toy Recognition from 2D Connectivity," *TR 196*, Computer Science Dept., U. Rochester, October 1986.

[Hopfield, 1982] Hopfield, J.J., "Neural Networks and Physical Systems with Emergent Collective Computational Abilities," *Proceedings of the National Academy of Sciences USA*, 79:2554-2558, 1982.

[Hougen, Darrell, Omohundro, 1988] Hougen, Darrell R. and Stephen M. Omohundro, "Fast Texture Recognition Using Information Trees," U. of Illinois at Urbana-Champaign, Department of Computer Science, *TR UILU-ENG-88-1714*, 1988.

[Hutchinson, 1988] Hutchinson, J., et. al., "Computing Motion Using Analog and Binary Resistive Networks," *Computer*, March 1988.

[Impagliazzo, Levin, Luby, 1989] Impagliazzo, R., L. Levin, and M. Luby, "Pseudo–random Function Generators from One–way Functions," accepted to *21st Annual*

ACM Symposium on Theory of Computing, 1989, preliminary version (with R. Impagliazzo) will appear as an ICSI technical report.

[Jerrum, Sinclair, 1988] Jerrum, M. and A. Sinclair, "Conductance and the Rapid Mixing Property for Markov Chains: the Approximation of the Permanent Resolved", *Proc. 20th Ann. ACM Symp. on Theory of Computing*, 1988, 235-244.

[Judd, 1987] Judd, J.S., "Complexity of Connectionist Learning with Various Node Functions," *Proc. IEEE Int'l. Conference on Neural Networks*, San Diego, June 1987.

[Karchmer, Wigderson, 1988] Karchmer, M. and A. Wigderson, "Almost Optimal Lower Bounds for Small Depth Circuits", *Proc. 18th Ann. ACM Symp. on Theory of Computing*, 6–20.

[Karp, 1989] Karp, R.M., "The Transitive Closure of a Random Digraph," manuscript, 1989.

[Karp, Ramachandran, 1989] Karp, R.M. and V. Ramachandran, "Parallel Algorithms for Shared–Memory Machines," to appear as a chapter in Handbook of Theoretical Computer Science, 1989.

[Karp, Zhang, 1989] Karp, R.M., and Y. Zhang, "Parallel Game Tree Search," submitted to *ACM Symposium on Parallel Algorithms and Architectures*, 1989.

[Karpinski, 1987] Karpinski, M., "On the Monte Carlo Space Constructible Functions and Separation Results for Probabilistic Complexity Classes," *Information and Computation*, 75, 178-189, 1987.

[Karpinski, Verbeek, 1987] Karpinski, M. and R. Verbeek, "Randomness, Probability and the Separation of Monte Carlo Time and Space," *LNCS 270*, 189-207, Springer Verlag, 1987.

[Karpinski, Wagner, 1988] Karpinski, M. and K. W. Wagner, "The Computational Complexity of Succinct Multi–Vertex Representation of Graphs," *Zeitschrift fur Operations Research*, 32, 201-211, 1988.

[Kearns, Valiant, 1988] Kearns, M. and L.G. Valiant, "Learning Boolean Formulae or Finite Automata is as Hard as Factoring," *TR 14-88*, Harvard U., 1988.

[Lee, 1988] Lee, K., *Large–Vocabulary Speaker–Independent Continuous Speech Recognition: The SPHINX System*, Ph.D. Thesis, CMU, April, 1988.

[Lee, Towsley, 1986] Lee, K.J. and D. Towsley, "A Comparison of Priority–based Decentralized Load Balancing Policies," *Proc ACM SIGMETRICS Conf.*, 1986.

[Lei, 1988] Lei, B., "Critical Issues in High Bandwidth Networks," (Ed.), *Request for Comments 1077*, November 1988.

[Lepley, Miller, 1983] Lepley, M. and G. Miller, "Computational Power for Networks of Threshold Devices in an Asynchronous Environment." Manuscript written between 1983 and 1986.

[Levin, 1985] Levin, L., "One–way Functions and Pseudo–Random Number Generators," Combinatorica, Vol. 7, No. 4, 357-363, 1987. A preliminary version appeared in *Proc. 17th Ann. ACM Symp. on Theory of Computing*, 1985, 363-365.

[Lippman, 1988] Lippman, R.P., "Neural Net Classifiers for Speech Recognition," *The Lincoln Laboratory Journal*, Vol. 1, No. 1, 1988.

[Lipscomb, 1987] Lipscomb, J., "*On the Computational Complexity of Finding a Connenctionist Model's Stable State Vector*," M.Sc. Thesis, U. Toronto, 1987.

[Lischka, Diederich, 1988] Lischka, C., and J. Diederich, "Von Symbolen zu lernfaehigen Systemen," *Chip Soft Tech, No. 2*, April, May 1988.

[Luby, 1982] Luby, M.G., "Integrity Preserving Cut is NP–complete," NP–completeness column, *Journal of Algorithms*, 185, D.S. Johnson (Ed.), Bell Laboratories, June 1982.

[Luby, 1986] Luby, M.G., "Simple Parallel Algorithm for the Maximal Independent Set Problem," *SIAM J. on Computing, Vol. 15, No. 4*, 1036-1053, 1986.

[Luby, 1-1988] Luby, M.G., "On the Parallel Complexity of Symmetric Connection Networks," *TR*, Computer Science Department, U. Toronto, 1988.

[Luby, 2-1988] Luby, M.G., "Removing Randomness in Parallel Computation without a Processor Penalty," to appear in *Proc. 29th Ann. IEEE Symp. on Foundations of Computer Science*, 1988.

[Luby, Karp, Marchetti–Spaccamela, 1984] Luby, M.G., R.M. Karp and A. Marchetti–Spaccamela, "Probabilistic Analysis of Multi–Dimensional Bin Packing Problems," *16th Annual ACM Symposium on Theory of Computing*, April 30-May 2, 1984.

[McClelland, Rumelhart, 1986] McClelland, J.L. and D.E. Rumelhart, "Parallel Distributed Processing: Explorations In the Microstructure of Cognition," (Eds.), Vol. 2: Applications, MIT Press/Bradford Books, 1986.

[Murveit, Weintraub, 1988] Murveit, H. and M. Weintraub, "1000–Word Speaker–Independent Continuous–Speech Recognition Using Hidden Markov Models," *ICASSP–88, Vol. 1*, 1988.

[Nehmer, et al., 1987] Nehmer, J. et al., "Key Concepts of the INCAS Multicomputer Project," *IEEE Trans. on Software Engineering, Vol. SE-13, No. 8*, 1987.

[Newell, 1983] Newell, A., "Intellectual Issues in the History of Artificial Intelligence," in The Study of Information: Interdisciplary Messages, F. Machlup and U. Mansfield (Eds.), John Wiley & Sons, Inc., 1983.

[Omohundro, 1988] Omohundro, Stephen M., "Fundamentals of Geometric Learning," U. of Illinois at Urbana-Champaign, Department of Computer Science *TR UILU-ENG-88-1713*, 1988.

[**Omohundro, 1987**] Omohundro, Stephen M., "Efficient Algorithms with Neural Network Behavior," *Complex Systems*, 11, 1987, 273-347.

[**Pitt, Warmuth, 1988**] Pitt, L. and M. Warmuth, "Reductions Among Prediction Problems: On the Difficulty of Predicting Automata," *IEEE 3rd Structure in Complexity Conference*, 60-69, 1988.

[**Poljak, Sura, 1983**] Poljak, S. and M. Sura, "On Periodical Behaviour in Societies with Symmetric Influences," *Combinatorica*, 3(1), 119-121, 1983.

[**Porat, 1987**] Porat, S., "Stability and Looping in Connectionist Models with Asymmetric Weights," *TR 210*, Computer Science Dept., U. Rochester, March 1987.

[**Porat, Feldman, 1988**] Porat, S. and J.A. Feldman, "Learning Automata from Ordered Examples," *TR 244*, Computer Science Dept., U. Rochester, March 1988.

[**Postel, 1981a**] Postel, J., "Transmission Control Protocol," *DARPA Internet RFC 793*, September, 1981.

[**Postel, 1981b**] Postel, J., "Internet Control Message Protocol," *DARPA Internet RFC 792*, September, 1981.

[**Rabaey, Pope, Brodersen, 1985**] Rabaey, J.M., S.P. Pope, and R.W. Brodersen, "An Integrated Automated Layout Generation System for DSP Circuits," *IEEE Transactions on Computer–Aided Design*, Vol. CAD–4, No 3., 285-296, July 1985.

[**Ranade, 1987**] Ranade, A., "How to Emulate Shared Memory," *Proc. 28th Annual IEEE Symp. on Foundations of Computer Science*, 185-194, 1987.

[**Renegar, 1988**] Renegar, Jr., "A Faster P-Space Algorithm for Deciding the Existential Theory of the Reals," *Technical Report No. 792*, School of Operations Research and Industrial Engineering, Cornell U., April 1988.

[**Renegar, Shub, 1988**] Renegar, Jr. and M. Shub, "Simplified Complexity Analysis for Newton LP Methods," *Technical Report No. 807*, School of ORIE, Cornell U., June 1988.

[**Richards, Sherstinsky, Brodersen, 1987**] Richards, B., A. Sherstinsky, and R.W. Broderson, "A Parameterized VLSI Video–Rate Histogram Processor," *Proc. IEEE Conf. Acoustics, Speech & Signal Processing*, Dallas, Vol. 1, 13.5.1–13.5.4., 1987.

[**Rome, Uthmann, Diederich, 1988**] Rome, E., T. Uthmann, and J. Diederich, "KI-Workstations," *Ueberblick, Marktsituation, Entwicklungstrends*, Addison-Wesley Verlag (Deutschland) GmbH: Bonn; Reading, Mass; Menlo Park, Ca; New York; Don Millis, Ontario; Wokingham, England; Amsterdam; Sydney; Singapore; Tokyo; Madrid; San Juan, 1988 *ISBN 3-925118-83-7*.

[**Ruetz, Brodersen, 1986**] Ruetz, P. and R.W. Brodersen, "A Realtime Image Processing Chip Set," *ISSCC Digest of Technical Papers*, 148-149, 1986.

[Sansom et al., 1986] Sansom, R.D. et al., "Extending a Capability Based System into a Network Environment," *1986 SIGCOMM Symposium*, 1986.

[Sejnowski, 1988] Sejnowski, T.L., "Analysis of Hidden Units in a Layered Network Trained to Classify Sonar Targets," *Neural Networks, Vol. 1*, 1988.

[Shamir, Snir, 1980], Shamir, E. and M. Snir, "On the Depth Complexity of Formulas," *Math. Systems Theory* **13**, 301–322.

[Shastri, 1985] Shastri, L. "Evidential Reasoning in Semantic Networks: A Formal Theory and its Parallel Implementation," Ph.D. Thesis and *TR 166*, Computer Science Dept., U. Rochester, September 1985.

[Shastri, Feldman, 1986] Shastri, L. and J.A. Feldman, "Neural Nets, Routines and Semantic Networks," in *Advances in Cognitive Science*, N. Sharkey (Ed.) Ellis Horwood Publishers, 1986.

[Shastri, 1988] Shastri, L., "Semantic Nets: An Evidential Formalization and its Connectionist Realization," Morgan Kaufman, Los Altos, Pitman Publishing Company, London, 1988.

[Shimada, Nakagawa, Takeshi, 1986] Shimada, S., K. Nakagawa, I. Takeshi, "Gigabit/s Optical Fiber Transmission Systems – Today and Tomorrow," *IEEE Int. Conference on Comm.*, 1538-1542, June 1986.

[Shin, Muthuswamy, 1988] Shin, K.G. and Y.K. Muthuswamy, "Message Communications in a Distributed Real-time System with a Polled Bus," *Proc. 22nd Annual Hawaii Int'l Conf. on System Sciences, Vol. II*, January 1989.

[Smale, 1986] Smale, S., "Newton's Method Estimates from Data at One Point," *The Merging Disciplines: New Directions In Pure, Applied and Computational Mathematics*, Springer Verlag, 185-196, 1986.

[Smolensky, 1987], Smolensky, R., "Algebraic Methods in the Theory of Lower Bounds for Boolean Circuit Complexity", *Proc. 19th Ann. ACM Symp. on Theory of Computing*, 77–82.

[Tanenhaus, Dell, Carlson] Tanenhaus, M.K., G.S. Dell & G. Carlson, "Context Effects in Lexical Processing: A Connectionist Approach to Modularity," in Modularity in Knowledge Representation and Natural Language Understanding, J. Garfield (Ed.), in press.

[Turner, 1986] Turner, J., "Design of a Broadcast Packet Network," *IEEE INFORCOM 86*, 667-675, 1986.

[Tzou, Anderson, 1988] Tzou, S.-Y. and D.P. Anderson, "Performance Evaluation of the DASH Message Passing System," *Technical Report No. UCB/Computer Science Dpt. 88/452*, Computer Science Division, EECS, UCB, Berkeley, CA, November 1988.

[Valiant, 1979] Valiant, L.G., "The Complexity of Computing the Permanent," *Theoretical Computer Science, Vol. 8*, 1979, 189-201.

[Valiant, 1984] Valiant, L.G., "A Theory of the Learnable, " *CACM 27*, 1134-1142, 1984.

[Valiant, 1988] Valiant, L.G., "Functionality in Neural Networks," COLT, 1988, *Proceedings*, MIT, August 1988.

[Vazirani, 1987] Vazirani, U., "Strong Communication Complexity or Generating Quasi–Random Sequences From Two Communicating Semi–Random Sources," *Combinatorica*, Vol. 7, No. 4, 375-392, 1987.

[Waibel, 1988] Waibel, A., "Modular Construction of Phonetic Neural Networks," *Proc., ATR Workshop on Neural Networks and Paralled Distributed Processing*, Osaka, Japan, July 7-8, 1988.

[Waibel, Hanazawa, Hinton, 1988] Waibel, A., T. Hanazawa, G. Hinton, K. Shikano, and K. Lang, "Phoneme Recognition: Neural Networks vs. Hidden Markov Models," *Proc. IEEE Conf. Acoustics, Speech & Signal Processing*, New York, Vol. 1, 107-110, 1988.

[Waltz, Feldman, 1987] Waltz, D. and J.A. Feldman (Eds.), "Connectionist Models and Their Applications," Ablex Publishing Corp., 1987.

[Watrous, 1988] Watrous, R.L., "Connectionist Speech Recognition Using the Temporal Flow Model," Represented at the 1988 *IEEE Workshop on Speech Recognition*, Harriman, NY, May 31-June 3, 1988.

[Wybranietz, Haban, 1988] Wybranietz, D. and D. Haban "Monitoring and Performance Measuring Distributed Systems during Operation," *ACM SIGMETRICS Conf.*, Santa Fe, New Mexico, May 1988, in *ACM Performance Evaluation Review 16(1)*, May 1988.

[Yao, 1982] Yao, A., "Theory and Applications of Trapdoor Functions," *Proc. 23rd Ann. IEEE Symp. on Foundations of Computer Science*, 1982, 80-91.