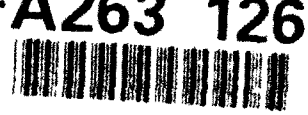
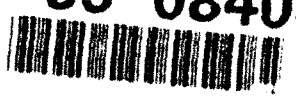


ARPA/ONR Quarterly Report

CONTRACT NUMBER:	N00014-91-J-1985
CONTRACTOR:	Duke University
ARPA ORDER NO:	4331714-01/4-6-88
PROGRAM NAME:	N00014
CONTRACT AMOUNT:	$ 696,899.00
EFFECTIVE DATE OF CONTRACT:	July 1, 1991
EXPIRATION DATE OF CONTRACT:	June 30, 1994
PRINCIPAL INVESTIGATOR:	John H. Reif
TELEPHONE NUMBER:	(919) 660-6568
SHORT TITLE OF WORK:	Implementation of Parallel Algorithms
REPORTING PERIOD:	1 Jan 93 — 31 Mar 1993

The views and conclusions contained in this document are those of the author(s) and should not be interpreted as necessarily representing the official policies, either expressed or implied of the Advanced Research Projects Agency or the U.S. Government.



DESCRIPTION OF PROGRESS

Investigations of several subproblems in the area of derivation of parallel programs were continued during the current quarter. These investigations include:

(1) Michael Landis (graduate student), John Reif (PI), and Robert Wagner (Duke faculty): Intermediate Representation for Parallel Implementation

Our collaboration with Carnegie Mellon and work on the Cray multiprocessor implementation of CVL is continuing. This work will be completed by the end of April.

These research efforts are focused on the possibility of extending a high-level data-parallel language with constructs for process parallelism. Our goal is to begin with a data-parallel language like NESL, which is under development by Guy Blelloch at Carnegie Mellon University. This language provides nested data-parallelism. We believe that by extending it with process parallel primitives, the language will have wider applicability, but yet will still be able to be implemented efficiently.

This work has evolved directly out of research over the past year in trying to develop extensions to a low-level data-parallel intermediate representation to accommodate asynchronous processes. After extensive research we have decided that parallel process extensions to a low-level intermediate representation are not practical because of fundamental differences in primitives provided by different hardware vendors. Instead, we are focusing our efforts currently on the extension of a run-time library for implementing data-parallel languages. This library will provide the support for high-level language development while maintaining portability and efficiency through the use of the C language. As an example, one possibility which we are investigating is the integration of the POSIX thread package with CVL, the C Vector Library under development at CMU. In order to gain experience with parallel systems and with the implementation of CVL, Mike Landis is collaborating with Blelloch's team in order to develop a multiprocessor CRAY implementation of CVL. This work should be done around the end of October, at which time we will focus on the extensions to this library.

(2) Michael Landis (graduate student), John Reif (PI), and Robert Wagner (Duke faculty): Data Movement on Processor Arrays

We have completed our study of developing ways of evaluating uniform expressions in near minimum parallel time on higher-dimensional processor arrays. A paper describing the solution on two-dimensional arrays is ready for submission to a journal publication. (This paper is a follow-up to Robert Wagner's paper, "Evaluating Uniform Expressions Within Two Steps of Minimum Parallel Time", Which solved the problem for two-dimensional arrays only.)

(3) John Reif (PI): Data-Parallel Implementations of Fast Multipole Algorithms for N-Body Interaction

Summary:
We are exploring data-parallel implementations of Fast Multipole Algorithms (FMA) for computing N-body interaction. Several algorithmic variants of FMA, such as adaptive FMA and other fastest known improvements [Reif,Tate92] are being expressed in a data-parallel fashion using the languages NESL (Nested Sequence Language, by Blelloch at CMU) and Proteus (at Duke and UNC). The data-parallel model provides a succinct high-level expression which exposes parallelism in a scalable fashion, and facilitates exploration and comparison of the parallel time complexity of algorithmic variants. Implementations are realized by transformation of the data-parallel programs to a lower-level widely portable vector model (VCODE), for example targeting the CM-5.

Details:
Many-body simulation is the key computational component in many challenging problems such as fluid mechanics and molecular dynamics simulation; the potential benefits of the latter include computer aided drug design and protein structure determination. In N-body simulation the goal is to simulate for a collection of N particles distributed in space the motion over time due to gravitational or electrostatic interaction between the particles. The naive solution requires N^2 comparisons to compute forces arising from pairwise interaction. More sophisticated algorithms reduce this complexity by relying on approximation of the lesser effects of far-away clusters of particles (perhaps modeling them by a few large particles), and on multigrid techniques which exploit this approximation by hierarchically decomposing the particle space into near and far-away points in order to isolate these far-field interactions.

The Fast Multipole Algorithm (FMA) [Greengard87] is a linear-time algorithm for calculating N-body interactions which uses multipole expansions to approximate the potential field created by a collection of bodies outside the region that contains the bodies. The Adaptive FMA (AFMA) improves on the FMA for non-uniform distributions. We have expressed the AFMA in a data-parallel manner using the Proteus programming language with two objectives. First, to prototype a complex adaptive grid computation quickly and correctly in a high-level language with the goal of expressing available parallelism in a succinct manner, and second, to explore the feasibility of recently developed transformation and translation techniques that translate data-parallel Proteus expressions to a standard set of vector operations. The AFMA was written and the nested parallelism can be translated to yield a nested sequence representation of the problem. The result is a work-efficient implementation of the program on a large class of parallel machines.

We are also pursuing the data-parallel implementation of several algorithmic variants of the FMA [eg, Reif-Tate 92] using NESL (developed by Blelloch at CMU). The implementations, relying on NESL's transformation to an underlying vector model (VCODE), will be widely portable and scalable, and so our efforts will have broad impact. Parallel work and parallel time complexity for both the VRAM and PRAM models are easily derived from the data-parallel versions, facilitating comparative evaluation of algorithmic improvements.

(4) Peter Mills (Research Associate) with John Reif: Rate Control in Parallel Algorithms

Summary:
Recent work has focused on extending high-level parallel computation paradigms with constructs for expressing relative rates of progress. The introduction of rate control supports a succinct specification of intended resource allocation, and is a first step in extending models of parallel computation with real-time properties, such as processor rates, in order to support timing analysis. We are currently pursuing implementation of the rate construct on a sequential interpreter for the Proteus language to use in experiments with algorithmic variations of adaptive N-body simulation.

Details:
We have developed a new parallel programming construct, the rate construct, which specifies constraints on the relative rates of progress of tasks executing in parallel, where progress is the amount of computational work as measured by elapsed ticks on a local logical clock. By prescribing

expected work, the rate construct constrains the allocation of processor-time to tasks needed to achieve that work; in a parallel setting this constrains the distribution of tasks to processors and multiprocessing ratios, effected for example by load balancing. The utility of the rate construct has been evidenced for a variety of problems, including weighted parallel search for a goal, adaptive many-body simulation in which rates abstract the requirements for load-balancing, and multiple time-stepped computations in which the use of rates can alter the frequency of asynchronous iterations.

One promising application of rate control is in algorithms for N-body interactions which rely on an optimization in which, for a given particle, interactions with far-away points are computed less frequently since their effects fall off rapidly with distance. Such a technique is used for example in the Generalized Verlet Algorithm [Grubmuller91], where particles are separated into "distance classes" and interactions with far-away particles are computed less frequently. The rate construct can be used to control this iteration frequency for clusters which may be running on asynchronous processes. Another application of rate control is to effect higher frequencies of iteration for well-separated clusters which have high densities and thus must have small motion integration steps to accommodate higher acceleration.

A paper describing the rate construct and various applications will appear in the1993 IEEE Workshop on Real-Time Parallel and Distributed Systems. We are currently pursuing sequential implementation of the rate construct, and are also investigating means of transforming rate primitives in a parallel setting to lower-level real-time and scheduling constructs.

(5) Peter Mills (Research Associate) with John Reif: Implementing Asynchronous Parallelism using Tagged-Memory

Summary:
Recent efforts have concentrated on extending high-level parallel computation models with abstractions for asynchronous concurrency which roughly mimic tagged memory. A novel construct, guarded communication using linear operators, has been introduced and methods of extending parallel functional languages such as NESL (CMU) and Concurrent ML (Bell Labs) with linear operators are under investigation. A scalable extension for asynchronism in a functional style promises to have large impact in expressing and implementing parallel algorithms for machines such as CM-5 and KSR-1.

Detail:
We are developing high-level mechanisms for asynchronous concurrency which include a variant of synchronization variables and a novel construct we call linear variables. Synchronization variables are a synchronization mechanism found in coordination languages such as PCN and CC++ as well as in Id's I-structures. Linear variables are a further extension which model resource consumption, and prove valuable in succinctly modeling channel and rendezvous operations within a shared-memory framework. Linear variables prove particularly advantageous in that they can be readily ported to many architectures, and promise to be amenable to optimization techniques which transform the program to decrease non-local references.

We are investigating extending an existing widely portable data-parallel language, CMU's NESL (supporting nested data parallelism) with a wrapper for asynchronous parallelism built on linear variables (similar to Id's M-structures). The intent is to extend and thus capitalize on existing techniques for transforming nested data parallelism to vector models, i.e. the transformation of NESL to VCODE. (Such an implementation strategy will most likely rely on run-time library extensions rather than extensions to a low-level intermediate representation, as mentioned by Landis and Wagner above).

Ongoing work:
- Extending NESL (CMU's nested data-parallel language) with mechanisms for asynchronous parallelism.
- Development of refinement techniques for transforming extended NESL to threads of vector code, targeting such machines as CM-5.
- Demonstration of viability of these techniques through concrete implementations of N-body algorithms, specifically clustering and Fast Multipole Methods, targeting MSPMD machines (e.g., CM-5).

(6) Peter Su (postdoc) and John Reif: Implementations of Parallel Algorithms in Computational Geometry

We have been working on the implementational aspects of parallel algorithms. Specifically, we have been studying parallel algorithms for constructing Voronoi Diagrams and related problems. Our interest in this study is not only to build effective algorithms for these problems, but also to consider the kinds of tools that make such work easier and more effective.

Our work has been broken up into three stages:

(1) Study the theory and practice of conventional algorithms for this problem.
(2) Study the current body of theoretical work on parallel algorithms for this problem.
(3) Using the knowledge gained in (a) and (b), design and implement parallel algorithms for this problem on several machines. Then study the performance of these algorithm and how well the theoretical results match the behavior of the implementation.

We have been actively working on stages (1) and (2) for the last few months and we are now ready to move on to stage (3). The study of practical sequential algorithms has been especially helpful in the pursuit of simple and efficient parallel algorithms, since they provide a good set of ideas to extend and refine in a parallel setting.

Using the experience that we gain from this work, we are also investigating and planning tools that could aid the programmer in implementing effective parallel algorithms. Since many parallel algorithms, especially in computational geometry, have similar structure, one could imagine a tool for reasoning about abstract classes of algorithms. In particular, such a system could aid the programmer in tuning performance parameters for specific machines based on architectural characteristics such as global memory bandwidth and latency, processor speed, local memory size, and so on. Also, more basic tools for doing visualization and performance analysis are needed to help the programmer to effective experimental analysis of his implementations. Tools for profiling, animation, simulation and data analysis would all be extremely useful in these settings. At this point, there are no such tools widely available to the research community.

Initial implementations of the ideas in this work has begun and has been successful. I presented a paper at the DAGS conference this summer that describes Cray algorithms for basic proximity problems. I have also begun to explore implementations of the other ideas on various machines, including the MasPar MP-1, the CM-5, and the KSR-1. This development work will make up a large part of my PhD thesis, which should be finished by this spring.

In addition, we have designed an efficient algorithm for constructing Delaunay triangulations which we are in the process of implementing on the KSR-1. It uses a novel 'transactional' method of constructing the diagram in incremental phases. Each phase attempts to add as many

points as possible in parallel, but if two insertions conflict, then one or the other must back off. We structure the insertion phases in a way that is reminiscent of transaction processing systems so that the current diagram is guaranteed to be unique. In addition, we randomize the insertion order of the points to guarantee that the algorithm can achieve sufficient parallelism.

(7) Shenfeng Chen with John Reif: Parallel Sort Implementation

Summary:
The fastest known sort is a parallel implementation of radix sort in a CRAY, due to CMU's Guy Blelloch. The current sorting algorithms on parallel machines like Cray and CM-2 use radix and bucket sort. But they are not taking advantage of possible distribution of the input keys. We are developing an algorithm using data compression to achieve a fast parallel algorithm which takes this advantage. We expect the new algorithm to beat the previous fastest sort by a few factors. We are working to implement this new parallel sorting algorithm on various parallel machines.

Details:
Radix sort is very efficient when the input keys can be viewed as bits. But the basic radix sort is not distribution based so it needs to look up all digits.

Our approach is to find the structure (distribution) of the input. This is achieved by sampling from the original set. Then a hash table is build from those sample keys. All keys are indexed to buckets separated by consecutive sample keys. A probability analysis shows that the largest set can be bounded within a constant of the average size.

The indexing step is made faster by binary searching the hash table for match. From previous result, each hash function computation needs only constant time.

Our algorithm needs O(nloglogn) time in sequential given that the compression ratio of the given input set is not too big. In parallel, our algorithm works well in chain-sorting. In list ranking sorting, the total work is also reduced.

We have implemented this algorithm on Sparc II and compared its performance with the system routine quicksort. It turns out that our algorithm outwins the quicksort() for sufficiently large number of keys

(32M). Thus, it may find its place in sorting large database operations (e.g., required by joint operations). In these applications the keys are many words long so our algorithm is even more advantageous in this case where the cutoff is much lower.

Ongoing work:
We are currently implementing the algorithm on Cray Y-MP. Due to the larger main memory, we expect better performance over Space II. We are also comparing our algorithm to the radix sort implemented by Blelloch on Cray.

(8) Deganit Armon (A.B.D.) with John Reif: Dynamic Graph Separator Algorithms.

Summary:
We continued work on dynamic graph problems, using the techniques we developed when studying the dynamic separator problem. These are techniques for converting a fixed input randomized algorithm into a randomized algorithm that accepts changes to the input. In addition we showed a method for converting an expected time randomized algorithms to randomized algorithms with high likelihood time bounds. We attempted to apply these techniques to other dynamic graph problems, in particular dynamic nested dissection and planar graph algorithms.

Details:
Randomized algorithms that use sampling select a small sample of the input, apply an "expensive" algorithm to the sample, and then extrapolate the result to the entire dataset. The solution will not necessarily be "exact", but the error can usually be bounded. Examples of such algorithms range from the version of quicksort in which a pivot is found by taking the mean of a small sample, to complex algorithms for finding graph separators, to implementations in computational geometry. We described a technique for transforming such algorithms so that they can deal with dynamically changing input, and applied this method to the problem of finding a sphere separator for a set of points. We showed that while the static algorithm takes linear time, computing a separator after adding or deleting a point from the input set requires only a logarithmic number of steps. We also showed that maintaining a more complex separator structure could also be done dynamically in polylog time.

Another characteristic of randomized algorithms is that while we can determine the expected time to completion, the actual running times may vary considerably. We showed a technique which, through the use of

multiple processes (called replicants) which are performing the same computations, we can guarantee the expected time bounds (with some slowdown) with high likelihood. This technique is particularly useful when in addition to changing the input the algorithm is also presented with queries about the input. We can thus guarantee timely processing of a query by one or more of the replicants. We showed how this method can be applied to the problem of maintaining graph separators with only a log^2 slowdown. This method can be applied to other randomized algorithms that involve maintaining a data structure and answering queries, such as arise in computational geometry.

A paper describing these techniques and their application to the dynamic sphere separator problem has been submitted to WADS 93. Currently we are working on finding randomized algorithms which can be dynamized using these techniques.

(9) Prokash Sinha with John Reif: Randomized Parallel Algorithms for Min Cost Paths

Summary:
We have completed our initial investigation to derive randomized parallel algorithms for Min Cost Paths in a Graph of High Diameter. Our present accomplishment is a randomized sequential algorithm with an order of magnitude performance gain for some dense graphs.

We also found a similar result for PRAM computational model which meets the work we proposed to do in our paper "A Randomized Algorithm for Min Cost Paths in a Graph of High Diameter: Extended Abstract" (J. Reif and P. Sinha). Currently we are in the process of submitting our findings to technical journals and conferences. Our next phase of work would include similar derivations of randomized parallel algorithms for a wide variety of discrete structures which arises naturally in the area of Graph Theory and Combinatorics. Our current research effort is to extend the techniques of Flajolet and Karp to develop techniques and tools for timing analysis of algorithms. This effort is to derive tools for semiautomatic randomized analysis.

(10) Hongyan Wang with John Reif: Social Potential Fields: A Molecular Dynamics Approach for Distributed Control of Multiple Robots.

Summary:
Much of the early research in robotic planning and control has considered the case of only a single robot. There is now a number of robot systems which include a small number of autonomous robots and consequently there is a quickly growing literature on the planning and cooperative control of systems of small numbers of robots. Our work is concerned with Very Large Scale Robotic (VLSR) systems consisting of at least hundreds to perhaps tens of thousands or more autonomous robots. Our molecular dynamics approach is distributed and robust and flexible.

Details:
We view our VLSR systems as a molecular dynamics system, with predefined force laws between each ordered pair of components (robots, obstacles, objectives and other configurations). These force laws are similar to those found in molecular dynamics, incorporating both attraction and repulsion in the form of inverse power laws. However these laws may differ from molecular systems in that we allow the controller to arbitrarily define distinct laws of attraction and repulsion for separate pairs and groups of robots to reflect their social relations or to achieve some goals. For example, we define a pair-wise force law of attraction and repulsion for a group of identical robots. The repulsion will prevent collision among robots and the attraction will keep them in a cluster. This simulates the phenomena called "individual distance" in sociobiology.

Once the force laws are set up (they can be modified by the global controller), each individual's movement is computed locally according to the local environment sensed by individual robots and the force laws. Thus the control is distributed and robust. Each robots obeys Newton's Law and makes movement complying to the total force on it from the other components.

We give concrete examples to show that this distributed autonomous control will have lots of applications in industry, military and other areas in the future when costs for individual robots drop and robots can be made much more compact and more capable and flexible.

We did computer simulations involving large numbers of robots. Some interesting and useful patterns can be achieved by defining proper force laws for the system, e.g. forming a more or less evenly distributed single cluster, forming a circle to guard a static point particle standing for castle. We are doing more simulations showing more complex patterns.

We also discuss about spring laws similar to molecular bondings to robotic control. Theories of graph rigidity support that we can design a VLSR system which has a rigid structure. This has also applications where assemblies are needed to finish some job efficiently.

(11) Hongyan Wang with John Reif: A Constant Time Algorithm for N-body Simulation with Smooth Distributions.

Summary:
N-body simulation problem is as follows: Given N points that have pair-wise interactions, compute the equilibrium configuration of the N points. This problem is central to a large body of work in theoretical physics, chemistry, and scientific computing, including: cosmology, plasma simulation, molecular dynamics, and fluid mechanics. The fastest N-body simulation algorithm due to Greengard has time complexity of O(N) for one step simulation. We propose to use the concept of density function to describe the configuration of the large particle system and a method to compute the equilibrium density function iteratively when given the initial density function in constant time with the time complexity depending only on the potential function and the required precision.

Details:
In a system of large number, say millions of particles, we are interested more in the structure of the system, especially the structure under equilibrium conditions than in the exact positions of all particles. Observations from many fields, such as cosmology, plasma simulations, molecular dynamics, and fluid mechanics, suggest that the distribution of particles is homogeneous and can be described by smooth functions. Thus we propose to use density function to describe the configuration of particle systems.

Based on the fact that under equilibrium conditions, the total force on each particle should equal to 0, we derive an iterative procedure IMPROVE for improving the density function, which is of the form $\Phi^{(n+1)}(x)=IMPROVE(\Phi^{n}(x))$, where x is a position in the domain of interest. Computing the total force on one robot by summing up discretely all the forces from other robots will require Omega(n) time.

Instead, we only sum up forces from a constant number of nearby particles. For particles far away, we do an integration of force function multiplied by density function to approximate the resultant force. This reduces the time complexity to constant. Thus each improvement procedure requires constant time and the number of iterations depends on the required precision, and thus can be constant.

Simulations showed that the iterative improvement procedures converges. The results showed that in 1-d the density function has a bell-shaped curve and in 2-d has a vault-shaped surface in the domain of interest and outside the domain has 0 value.

(12) Akitoshi Yoshida with John Reif: Image and Video Compression

We considered several compression techniques using optical systems. Optics can offer an alternative approach to overcome the limitations of current compression schemes. We gave a simple optical system for the cosine transform. We designed a new optical vector quantizer system using holographic associative matching and discussed the issues concerning the system.

Optical computing has recently become a very active research field. The advantage of optics is its capability of providing highly parallel operations in a three dimensional space. Image compression suffers from large computational requirements. We propose optical architectures to execute various image compression techniques, utilizing the inherent massive parallelism of optics.

In our paper[RY2], we optically implemented the following compression and corresponding decompression techniques:

- o transform coding
- o vector quantization
- o interframe coding for video

We showed many generally used transform coding methods, for example, the cosine transform, can be implemented by a simple optical system. The transform coding can be carried out in constant time.

Most of this paper is concerned with an innovative optical system for vector quantization using holographic associative matching. Limitations of conventional vector quantization schemes are caused by a large number of sequential searches through a large vector space. Holographic associative

matching provided by multiple exposure holograms can offer advantageous techniques for vector quantization based compression schemes. Photo-refractive crystals, which provide high density recording in real time, are used as our holographic media. The reconstruction alphabet can be dynamically constructed through training or stored in the photorefractive crystal in advance. Encoding a new vector can be carried out by holographic associative matching in constant time.

We also discussed an extension of this optical system to interframe coding.

On going work:
We are investigating optical algorithms for video compression.

(1) Computational Geometry by Optical Computers
Some problems require inherently high degrees of interconnections which may not be provided by any conventional electrical computers. The advantage of optical computers is their apparent parallelism in a three dimensional space. Several computational models have been already proposed and constructed by various research groups. As the progress of optical computers continues, there is a great demand in designing and investigating various algorithms that are efficient and appropriate for the proposed models. This situation resembles to the one a decade ago, when various algorithms were investigated for the theoretical VLSI model. Thus, we understand that the investigation on optical computing algorithms will be essential to the development of optical or hybrid massively parallel computers.

Optical techniques are particularly suited for processing images. This leads us to believe that many problems found in computational geometry may be efficiently solved by optical computers. Some researchers have recently started to investigate some basic problems. We have been investigating these and some other problems. We have obtained some new results.

(2) Optical Interconnection
Among processing units placed on a plane, various space-invariant interconnections can be holographically established in constant time. We are investigating appropriate interconnections and efficient algorithms for several problems.

(3) Efficient computation for optical scattering
An efficient algorithm to solve the Helmholtz equations was developed by Rokhlin at Yale. We have been studying his algorithm.

(4) Simulation of optical computing algorithms
We implemented a software simulator for optical computing algorithms. The simulator is written in C on the X-window environment. It has a lisp-like user interface, and images, which are the basic data structures in the optical computing algorithms, are treated as lisp objects. We simulated some algorithms designed for computational geometry problems.

We are improving the simulator and planning to implement it on a parallel machine.

(13) Researchers supported (other than PI):

Mike Landis, graduate student
Peter Mills, post-doc
Peter Su, visiting graduate student
Robert Wagner, professor
Akitoshi Yoshida, graduate student

(14) Papers

(1) Optical Expanders with Applications in Optical Computing (with A. Yoshida), *Applied Optics* , Vol. 32, 159-165,1993.

(2) Memory-Shared Parallel Architectures for Vector Quantization Algorithms (with T. Markas), 1992. Accepted for the *Picture Coding Symposium*, Lusanne Switzerland, Mar 93. Also submitted for journal publication.

(3) Parallel and Output Sensitive Algorithms for Combinatorial and Linear Algebra Problems (with J. Cheriyan), 1992. Accepted at *Symposium on Parallel Algorithms* (SPAA'93), Velon, Germany, July 1993.

(4) The Complexity of N-body Simulation (with S.R. Tate), 1992. Accepted at *20th Annual Colloquium on Automata, Languages and Programming* (ICALP'93), Lund, Sweden, July, 1993.

(5) Multispectral Image Compression Algorithms (with A. Markas) accepted at *Data Compression Conference* (DCC'93), Snowbird, UT, March 1993.

(6) Rate Control as a Language Construct for Parallel and Distributed Programming (with P. Mills and J. Prins), 1992. Accepted at the *IEEE Workshop on Parallel and Distributed Real-Time Systems* (IPPS'93), Newport Beach, CA, April 1993.

(7) Algebraic Methods for Testing the k-Vertex Connectivity of Directed Graphs, (with J. Cheriyan), *3rd Annual ACM-SIAM Symposium on Discrete Algorithms*, 1992. Accepted for publication as "Directed s-t Numberings, Rubber Bands, and Testing Digraph k-Vertex Connectivity," in *Combinatorica*.

(8) Searching in an Unknown Environment (with M. Kao and S. Tate), *4th Annual ACM-SIAM Symposium on Discrete Algorithms* (SODA92), San Diego, CA, 1992. To appear in *Information and Computation.*

(9) Optical Techniques for Image Compression (with A. Yoshida). 2nd Annual IEEE Data Compression Conference (DCC 92), Snowbird, UT, March 1992, pp. 32–41. Also to appear in Image and Text Compression, edited by James A. Storer, Kluwer Academic Publishers, 1992.

(10) Fast and Efficient Parallel Solution of Sparse Linear Systems (with V. Pan). Accepted for publication in *SIAM Journal on Computing*, 1992.

(11) Nested Annealing: A Provable Improvement to Simulated Annealing (with S. Rajasekaran). Accepted for publication in *Journal of Theoretical Computer Science*, November 1992.

(12) On Threshold Circuits and Efficient, Constant Depth Polynomial Computation (with S. Tate). Accepted for publication in *SIAM Journal of Computing*, 1992.

(13) Planarity Testing in Parallel (with V. Ramachandran), University of Texas at Austin Technical Report TR-90-15, June 1990. Invited to special issue of *Journal of Algorithms*, 1992.

(14) The Computability and Complexity of Ray Tracing (with D. Tygar and A. Yoshida). To appear in *Discrete & Computational Geometry*, 1992.

(15) Randomized Algorithms for Binary Search and Load Balancing on Fixed Connection Networks with Geometric Applications (with S. Sen). *2nd Annual ACM Symposium on Parallel Algorithms and Architectures*, Crete, Greece, July 1990, pp. 327-337. To appear in *SIAM Journal of Computing*, 1992.

(16) Strong k-connectivity in Digraphs and Random Digraphs (with P. Spirakis). Accepted for publication in *Algorithmica*, 1992.

(17) Probabilistic Parallel Prefix Computation. Accepted for publication in *Computers and Mathematics with Applications*, 1992.

(18) Efficient VLSI Fault Simulation. To appear in *Computers and Mathematics with Applications*, 1992.

(19) Continuous Alternation (with S. Tate). To appear in a special issue of *Algorithmica*, edited by B. Donald, 1992.

(20) Quad Tree Structures for Image Compression Applications (with T. Markas). Special issue of *Journal of Information Processing and Management*, 1992.

(21) The Power of Combining the Techniques of Algebraic and Numerical Computing: Improved Approximate Multipoint Polynomial Evaluation and Improved Multipole Algorithms (with V.Y. Pan and S.R. Tate), *33rd Symposium on Foundations of Computer Science*, October 1992. Also submitted for journal publication as "The Complexity of Trummer's Problem, Zeta Function Evaluation, and N-body Simulation" (with S. Tate).

(22) Towards Randomized Strongly Polynomial Algorithms for Linear Programming (with S. Krishan). Duke University Technical Report CS-1991-18. Submitted for publication to *Operations Research Letters*, Dec 92.

(23) Method for Deriving Systolic Algorithms (by R.A. Wagner and M.D. Landis), 1992. Submitted for journal publication.

(24) Evaluating Uniform Expressions Within Two Steps of Minimum Parallel Time (by R.A. Wagner), 1992. Submitted for journal publication.

(25) Shortest Paths in Euclidean Space with Polyhedral Obstacles (with J.A. Storer). *Symposium on Mathematical Foundations of Computer Science*, Czechoslovakia, August 1988. Revised as "Shortest Paths in the plane with polygonal obstacles", 1992. Submitted for journal publication.

(26) Error Resilient One-Way Dynamic Communication, (with J.A. Storer). Revised as "Error Resilient Optimal Data Compression," Submitted for journal publication.

(27) On Applications of Crypto-complexity to Analyzing Efficiency of Capital Markets (with S. Azhar), 1992. Submitted for journal publication.

(28) Fully Dynamic Graph Connectivity in Logarithmic Expected Time (with P. Spirakis and M. Yung), 1992. Submitted for journal publication.

(29) On Parallel Implementations and Experimentations of Lossless Data Compression Algorithms (with T. Markas), 1992. Submitted for publication.

(30) A Randomized Algorithm for Min Cost Paths in a Graph of High Diameter: Extended Abstract (with P. Sinha), 1992. Submitted for publication.

(31) Fast Algorithms for Closest Point Problems: Practice and Theory (by Peter Su), 1992. Submitted for publication.

(32) A Fast Sort and Priority Queue for Entropy Bounded Inputs (with Shenfeng Chen), 1992. Submitted for publication.

(33) Social Potential Fields: A Molecular Dynamics Approach for Distributed Control of Social Behavior in Robots (with Hongyan Wang), 1992.

(34) Dynamic Algebraic Algorithms (with S. R. Tate), 1992. Submitted for publication.

(35) Dynamic Parallel Tree Contraction (with S. R. Tate), 1992. Submitted for publication.

(36) A Dynamic Separator Algorithm with Applications to Computational Geometry and Nested Dissection" (with D. Armon), 1992. Submitted for publication.

(37) Using Learning and Difficulty of Prediction to Decrease Computation: A Fast Sort and Priority Queue on Entropy Bounded Inputs (with S. Chen), 1992. Submitted for publication.

(38) Re-Randomization and Average Case Analysis of Fully Dynamic Graph Algorithms (with P.G. Spirakis and M. Yung), 1992. Submitted for publication.

(39) Strictly Polylog Time, Linear Space Algorithms for Nearest Neighbor Search and Dynamic Separators in d-Dimensions (with D. Armon) 1992. Submitted for publication.

(40) Diffraction realization of an optical expander (with R. Barakat), 1993. Submitted for publication.

(41) Reduction on Multidimensional Processor Arrays (R. Wagner and M. Landis), 1993. Submitted for publication.

(42) A Data-Parallel Implementation of the Adaptive Fast Multipole Algorithm (with L. Nyland and J. Prins), 1993. Submitted for publication.

(43) Improving viewing condition for reduced information holographic display (with A. Yoshida), 1993. Submitted for publication.