Proceedings of SRS-93[1]

Oregon Graduate Institute of Science & Technology
Department of Computer Science and Engineering
Student Research Symposium

November 5, 1993

[1]OGI-CSE Technical Report 93-020

From the SRS-93 Committee...

The 1993 Student Research Symposium (SRS-93) of the Oregon Graduate Institute's Department of Computer Science and Engineering was held on November 5th, 1993, at the OGI campus. In past years, the Symposium has been coupled with the proficiency paper requirements for advancement to candidacy; this year, it was held as an independent event to showcase the diverse research being pursued within the department by its students. We intended the day to provide opportunities for learning and exchange amongst both new and advanced students as well as our faculty and outside attendees, and it was clearly a success.

We were pleased with the student response to this Symposium, which resulted in a full day of ten presentations. In addition, three papers were accepted for inclusion in the Symposium. We thank all the students who contributed their work, and we congratulate the recipients of the awards:

Best Paper:	Bennet Vance and Scott Daniels,
	On the Semantics of Query Optimization
Runner up:	Eric Stoltz and Michael P. Gerlek,
	Constant Propagation: A Fresh, Demand-Driven Look
Best Talk:	Eric Stoltz and Michael P. Gerlek,
	Constant Propagation: A Fresh, Demand-Driven Look
Runner up:	Bennet Vance,
	On the Application of Partial Evaluation to Database Optimization

The committee would like to thank our anonymous reviewers for their contributions, both faculty and students; their comments will result in improved final papers. We also thank the judges, again both faculty and students, for their time and criticism: Jef Bell, Francoise Bellegarde, Ravi Konoru, Steve Otto, Tim Sheard, Lisa Walton, and Karen Ward.

Our lunchtime panel discussion was fun for all, and something we hope to repeat on a regular basis. We thank the faculty panelists, Jim Hook, Steve Otto, Dick Kieburtz, and Etienne Barnard, and our moderator, Phil Barrett, for providing their views on the topic.

As we would all like this to become an annual department event, your comments and criticisms on the Symposium are welcome; feel free to speak to one of us, or email us at `ogi-srs@cse.ogi.edu`. Thanks again to all who participated in one way or another!

Symposium Committee

Jon Inouye, *Symposium Chair*
Jeff Lewis and Karen Ward, *Program Co-chairs*
Brian Hansen, *Local Arrangements Chair*
Michael P. Gerlek, *Secretary*
Bennet Q. Vance, *Honorary Cookie Czar*

SRS-93 Schedule
November 5th, 1993

9:00	Session 1 (Jeffrey Lewis, chair) Eric Stoltz and Michael P. Gerlek, *Constant Propagation: A Fresh, Demand-Driven Look* Robert Prouty, *Adaptive Execution of Data Parallel Computations on Networks of Heterogeneous Workstations*
10:00	*break*
10:30	Session 2 (Brian Hansen, chair) Bennet Vance, *On the Semantics of Query Optimization* Johan Schalkwyk, *Temporal-Difference Methods and Markov Models* Jeffrey Lewis, *Programming with Algebras*
12:00	Lunch and Faculty Panel Discussion (Phil Barrett, moderator) *Computer Science: Establishing and Maintaining a Core*
1:30	Session 3 (Karen Ward, chair) David Hansen, *Using an Object-Oriented Database to Encapsulate Heterogeneous Scientific Data Sources* Scott Daniels and Bennet Vance, *On the Application of Partial Evaluation to Database Optimization*
2:30	*break*
2:45	Session 4: Short Presentations (Michael P. Gerlek, chair) Daniel Burnett, *Towards Automatic Collection of the U.S. Census* Tito Autrey, *Improving Parallel Heuristic Search in an Expert System* Bill Trost, *Modularity* and *Performance – the Promise of Incremental Specialization*
3:45	Department Colloquium Sara Bly (Xerox Palo Alto Research Center), *Beyond Marks and Meetings: Designing Technologies for Distributed Groups*
4:45	Awards Banquet / Party

Constant Propagation: A Fresh, Demand-Driven Look

Eric Stoltz and Michael P. Gerlek
stotlz@cse.ogi.edu

Constant propagation is a well-known static compiler technique in which values of variables which are determined to be constants can be passed to expressions which use these constants. Code size reduction (preventing unnecessary compilation), bounds propagation, and dead-code elimination are some of the optimizations which benefit from this analysis.

In this paper, we present a new method for detecting constants, based upon an optimistic *demand-driven* recursive solver, as opposed to more traditional iterative solvers. The problem with iterative solvers is that they may evaluate an expression many times, while our technique evaluates each expression only once. To consider conditional code, we augment the standard Static Single Assignment (SSA) form with merge operators called γ-functions, adapted from the interpretable Gated Single Assignment (GSA) model. We show that our approach is fast and finds the same class of constants as other methods.

We also present preliminary experimental results which show the number of intra-procedural constants found in common high-performance Fortran codes.

The full paper will appear in the Proceedings of the 1994 ACM Symposium on Applied Computing (SAC-94).

Adaptive Execution of Data Parallel Computations on Networks of Heterogeneous Workstations

Robert Prouty
prouty@cse.ogi.edu

Parallel environments consisting of a network of heterogeneous workstations introduce an inherently dynamic environment that differs from multicomputers. Workstations are usually considered “shared” resources while multicomputers provide dedicated processing power. The number of workstations available for use is continually changing; the parallel machine presented by the network is in effect continually reconfiguring itself. Application programs must effectively adapt to the changing number of processing nodes while maintaining computational efficiency.

This paper examines methods for adapting to this dynamic environment within the framework of explicit message passing under the data parallel programming model. We present four requirements which we feel a method must satisfy. Several potential methods are examined within the framework and evaluated according to how well they address the defined requirements.

An application-level technique called Application Data Movement (ADM) is described. Although this technique puts much of the responsibility of adaptation on the application programmer, it has the advantage of running on heterogeneous workstations. Related work, such as Dataparallel C and Piranha, is also examined and compared to ADM.

The application of the ADM methodology to a real application, a neural-network classifier based on conjugate-gradient optimization, is outlined and discussed. Preliminary results are presented and analyzed. The computation has been shown to achieve in excess of 70 MFLOPS under quiet conditions on a network of nine heterogeneous machines, two HP 9000/720s, two DEC Alphas, and five Sun SPARCstation 10s, while maintaining an efficiency of nearly 80 percent.

On the Semantics of Query Optimization

Bennet Vance
bennet@cse.ogi.edu

The purpose of optimizers is to improve the efficiency of programs, expressions, or queries. However, in improving a program's efficiency, an optimizer must not change the program's *meaning*; that is, the optimized program must behave in a way consistent with the original, unoptimized program. Most modern database query optimizers are rule-based: They improve queries by applying a succession of transformation rules, each of which ostensibly preserves meaning. In this talk we argue that transformation rules used by existing optimizers actually do *not* preserve meaning in general. Instead, their application is controlled through *ad hoc* auxiliary information to ensure that correct results are obtained. We propose an alternative approach to query transformation in which transformation rules are guaranteed to preserve meaning, and in which no auxiliary information is needed for correctness. Optimizers based on the proposed strategy promise to be simpler and more robust than optimizers built using current techniques.

Temporal-Difference Methods and Markov Models

Johan Schalkwyk
johans@cse.ogi.edu

Reinforcement-learning methods are rapidly establishing themselves as useful approaches to problems that involve development in time, such as the construction of neural-network based controllers. Recent theoretical understanding of these learning methods has concentrated on the relation between reinforcement learning and Markov models, and the consequent relation with Dynamic Programming. One of the results arising from this analysis is that temporal differences (TD) can be viewed as a novel estimator of the eventual termination probabilities (i.e. the probability of ending in each terminal state given any non-terminal state) for absorbing Markov Models. In the current work attention will be limited to temporal differences, although similar results apply for reinforcement-learning methods.

The full paper may be obtained as an OGI-CSE Technical Report, number 93-018.

Programming with Algebras

Jeffrey R. Lewis
jlewis@cse.ogi.edu

One view of programs in a purely functional programming language is that the functions are operators defined by systems of equations. This suggests that programming may be essentially algebraic in nature. This notion is explored in detail in a programming language currently being developed called Algebraic Design Language (ADL). In ADL, datatype declarations are replaced by algebra signatures, recursive function definitions are replaced by algebra homomorphism specifications, and a modular structure is seen thru the use of algebra functors.

In traditional functional programming languages, one declares datatypes for sum-of-products types such as lists and trees. ADL takes a more general perspective, and instead declares signatures for algebras (the operator names and their arities). An algebra signature specifies many algebras. However, one in particular that is of interest is the initial algebra - this corresponds to the usual notion of datatype, with the algebra operators being the data constructors.

The expression language of ADL is much like that of SML, excluding explicitly imperative features such as assignment and exceptions. However, arbitrary recursive function definitions, which can be unstructured and difficult to reason about, are not present in ADL. Instead, algebraic structure is exploited by defining recursive programs as algebra homomorphisms that internalize the recursive structure, requiring only the specification of non-recursive details. ADL distinguishes two categories of such homomorphisms: the initial algebra homomorphisms, called *reduce*s, that correspond roughly to the primitive recursive functions, and non-initial algebra homomorphisms, called *hom*s. Examples of *reduce*s are more familiar to functional programmers, and include the map and fold functions for lists. A characteristic of reduces is that their recursive structure is the same as the data structure they operate on. A good example of a *hom* is quicksort, which operates on lists, but has the recursive structure of a tree. These and other examples are presented in detail.

The structure provided by algebras is found to also provide a good way of structuring program modules. An algebra can be seen as a module, encapsulating the definition of its operators, and abstracted by the definition of its signature. Program modules that build new algebras from other ones then correspond to categorical functors.

Computer Science: Establishing and Maintaining a Core

Faculty Panel Discussion
Jim Hook, Steve Otto, Dick Kieburtz, Etienne Barnard
Phil Barrett, moderator

The field of computer science is both young and remarkably diverse. It includes topics from cognitive psychology to physics, from abstract mathematics to hardware engineering, and from information science to database applications. Is there truly a core to computer science that holds together this broad diversity? Or, will computer science as it matures gradually dissolve into a number of more cohesive disciplines?

Using an Object-Oriented Database to Encapsulate Heterogeneous Scientific Data Sources

David M. Hansen
dhansen@cse.ogi.edu

Scientists are continually faced with the task of integrating data from many computerized data sources in the course of their research. Many of these data sources can be characterized as large, slowly changing databases of chemical and physical property values. Integrating data from these databases is a difficult task primarily because relevant data is available in a variety of heterogeneous data sources. This talk presents an object-oriented heterogeneous database (OOHDB) architecture for accessing large, heterogeneous databases of scientific property data.

While other researchers have addressed the issue of masking heterogeneity with an object-oriented database, none have addressed the problem of efficiently accessing large, external data sources that are not managed by database management systems. Typical heterogeneous database architectures achieve time-efficiency by optimizing and decomposing global queries across external databases, then passing those sub-queries on to the external database managers, where each sub-query is further optimized for efficient local execution.

The OOHDB architecture we are developing provides efficient access to external data that is not managed by a database management system. Efficiency is achieved by using a small amount of space within the OOHDB for representing individual objects drawn from external data sources. This architecture provides opportunities for increasing performance by retaining data read from external data sources at various levels within the OOHDB architecture. Retaining data not only provides much better performance, but also allows performance to be enhanced through traditional mechanisms such as indexing. The design uses the behavioral aspect of the objects in the OOHDB to mask physical and logical heterogeneity among external sources of data. The OOHDB presents a single, uniform object-oriented model of the data of the domain to users and their application programs.

The OOHDB has been implemented using a commercial object-oriented database management system and measurements of space overheads and raw performance have been collected. Preliminary results indicate that careful data retention can provide speedups of better than one order of magnitude. These results suggest that an OOHDB can be used to efficiently access large scientific datasets that are not managed by database management systems.

On the Application of Partial Evaluation to Database Optimization

Scott Daniels and Bennet Vance
daniels@cse.ogi.edu

The theory and practice of *partial evaluation* has attracted increasing interest in recent years, and the techniques it offers have been applied to a variety of problems in compilation and the design of algorithms. To our knowledge, however, little work has been done to investigate its possible applications to database systems. In this paper we describe some very preliminary work we have done with a view to remedying this gap in partial-evaluation research. Our specific focus of attention is *optimization* of database implementations.

We discuss both actual partial-evaluation experiments we have already carried out and more speculative ideas that we have thought about but not implemented. The techniques we consider have potential application both to relational and object-oriented database systems. However, most of our initial work is based completely on relational databases.

The full paper may be obtained as an OGI-CSE Technical Report, number 93-019.

Towards Automatic Collection of the U.S. Census

Daniel Burnett
burnett@cse.ogi.edu

The goal of this project is to demonstrate the feasibility of using a computerized spoken language system (SLS) to automatically collect information for the U.S. Census over the telephone. This entails developing a prototype SLS for both English and Spanish languages. In this talk, we will present an overview of the SLS prototype development procedure, including dialog design, data collection, and SLS design. We will also report on progress to date and outline future work.

Improving Parallel Heuristic Search in an Expert System

Tito Autrey
tito@cse.ogi.edu

This talk will analyze parallelism opportunities in heuristic search. It will use the SYNCHEM expert system for its examples. SYNCHEM is a large expert system in the domain of organic chemistry. This particular problem domain was chosen because it is extremely complex and successful solution instances can have high value. This is in contrast to simple domains such as Block World. SYNCHEM solves the organic synthesis problem which is, given a target molecule, to generate a set of starting materials and a sequence of reactions to apply to produce the target molecule. The current system takes 10 CPU days on the available platform, a MicroVAX-II, to find successful solutions to its most difficult problems. It is hoped that a vastly faster and larger system could find a synthesis for a valuable and difficult compound such as Taxol.

I will also describe the SYNCHEM architecture, its current capabilities and future research directions. There are two primary areas of research, knowledge acquisition to expand the set of rules available to SYNCHEM, and parallelizing heuristic search to make effective use of the expected environment for computational chemists which is a network of multiprocessor workstations. This talk will not cover the knowledge acquisition project in more than cursory detail.

Modularity *and* Performance – the Promise of Incremental Specialization

Bill Trost
trost@cse.ogi.edu

The Synthesis project demonstrated how careful use of fine-grained modules, combined with partial evaluation techniques, can dramatically improve operating system throughput and response time. However, Synthesis itself was a highly experimental system, hand-crafted from the bottom up and designed for a particular microprocessor. The optimizations applied were also custom-designed, and did not take advantage of existing results in partial evaluation.

Our work will extend the ideas of Synthesis by developing and applying systematic methods to optimize part of an existing operating system, the filesystem of HP-UX. Our intent is to demonstrate that the techniques developed by Synthesis can be implemented portably, and in existing systems.

In my talk, I will describe the approach we envision to transform an existing body of code into a fine-grained structure, and, if time permits, show how the fine-grained structure permits a variety of unusual and powerful optimizations.