Irregular Parallel Algorithms

Department of Computer Science | University of North Carolina at Chapel Hill | February 2004

The Challenge

Modern science and engineering disciplines make extensive use of computer simulations. As these simulations increase in size and detail, the computational costs of naive algorithms often grow faster than can be accommodated even with the largest parallel computing resources available today. Improving the efficiency of algorithms for these problems requires the use of sophisticated modeling techniques that vary resolution as needed, coupled with sparse and adaptive solution methods that vary computational effort in time and space.

The distribution of work and data in such algorithms cannot be characterized a priori because these quantities are input-dependent and evolve with the computation itself. Algorithms with these properties are said to be irregular, and pose problems for high-performance parallel implementations, where equal distribution of work over processors and locality of reference are required within each processor.

To obtain high-performance for irregular parallel algorithms, we are trying to integrate several levels of the algorithm development and implementation process. First, we are concerned with how irregular parallel algorithms can best be expressed in modern programming languages. Next, given a form in which irregular parallelism is specified, how can we translate this to an efficient parallel execution strategy. And finally, how can we maximize the performance of our approach on different target parallel architectures.

Highlights

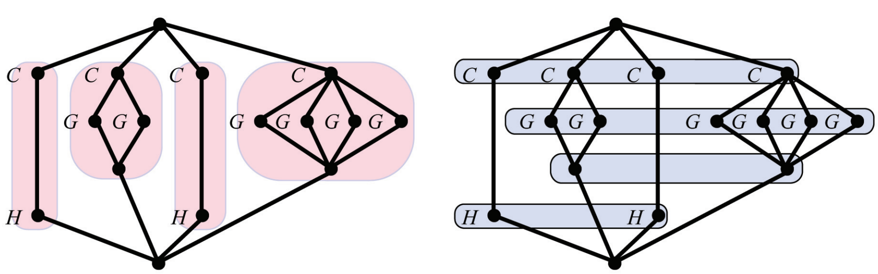
- **Christopher J. Riley. High-Performance Java Codes for Computational Fluid Dynamics. M.S. Thesis 2001, UNC–Chapel Hill.**
- **Wolf Pfannenstiel. Piecewise Execution of Nested Parallel Programs. Ph.D. Dissertation 2001, Technische Universität Berlin.**
- **Gabrielle Keller. Transformation-Based Implementation of Nested Data Parallelism for Distributed Memory Machines. Ph.D. Dissertation 2000, Technische Universität Berlin.**
- **James W. Riely. Applications of Abstraction for Concurrent Programs. Ph.D. Dissertation 1999, UNC–Chapel Hill.**

The Approach

How are Irregular Algorithms Expressed? We generalize data parallelism to nested data parallelism, because this yields a succinct and expressive mechanism for the expression of irregular parallel computations. The central concept of data parallelism is the application of an operation to all elements of a collection. In nested data parallelism, the elements of a collection may themselves be collections, and the operation may itself be data parallel. This enables the simple expression of parallel operations on irregular domains such as graphs and trees. We are investigating the expression of nested data parallelism in performance-oriented programming languages such as Fortran (Fortran 95/HPF) and object-oriented languages such as Java.

How Can Irregular Algorithms be Executed in Parallel? We are pursuing two approaches: (1)

```
FORALL (i = 1,4)
   WHERE C(i) DO
      FORALL (j = 1,i) DO
         G(i,j)
      END FORALL
   ELSEWHERE
      H(i)
   END WHERE
END FORALL
```



Nested data-parallel constructs in Fortran-95, and two implementation strategies expressed on the program dependence graph. Left: Multithreading of outer construct and serialization of the inner construct, leading to potential load imbalance and loss of concurrency, but high locality. Right: Flattening of nested data-parallel constructs into vector operations with full load balance, but potential loss of locality.

multithreading with run-time thread scheduling for load balance, and (2) flattening nested parallelism through compilation techniques that create vector operations preserving the available parallelism and total work to within a constant factor. Load-balanced parallel execution of the individual vector operations requires minimal run-time support. With both approaches, we are concerned with increasing the efficiency and decreasing the memory requirements of the resulting programs.

What is the Best Performance and How Can it be Obtained? Parallel computer architectures that hide memory latency for bulk references generated by vector operations or multithreading are a particularly good match to irregular computations. For commodity architectures requiring higher degrees of reference locality, we are investigating compilation and execution strategies that exploit partial serialization of the computation in combination with multithreading and flattening.

Parallel Programming with Explicit Locality. In conjunction with researchers at the University of Maryland at College Park and Ohio State University, we are evaluating the UPC (Unified Parallel C) programming language, a thread-parallel extension of the ANSI C language in which variables are allocated to shared or private thread memory, with shared values explicitly distributed across memories so that each thread can operate efficiently on local portions of shared values.

Principal Investigator
Jan F. Prins, professor

Graduate Research Assistants
Jeffrey Feasel
Jun (Luke) Huan

Past Project Members
Brian Blount, Wolf Pfannenstiel, Christopher J. Riley, James Riely, Martin Simons

Research Sponsors
Lucite Foundation
National Science Foundation (International Programs Grant #INT-9726317)

Selected Publications

Riley, C. J., S. Chatterjee, and R. Biswas. "High-Performance Java Codes for Computational Fluid Dynamics," *Proc. Joint ACM Java Grande—ISCOPE 2001 Conference*, June 2001.

Riely, J. W., and J. Prins. "Flattening is an Improvement," *ACM Static Analysis Symposium*, 2000.

Nyland, L., J. Prins, A. Goldberg, and P. Mills. "A Design Methodology for Data-Parallel Applications," *IEEE Transactions on Software Engineering*, 26(4), IEEE, 2000.

Prins, J., S. Chatterjee, and M. Simons. "Irregular Computations in Fortran—Expression and Implementation Strategies," *Scientific Programming*, 7(3), 1999.

Blount, B. L., and S. Chatterjee. "An Evaluation of Java for Numerical Computing," *Scientific Programming*, 7(2), 1999.

Blount, B. L., S. Chatterjee, and M. Philippsen. "Irregular Parallel Algorithms in Java," *Proc. Irregular '99: Sixth International Workshop on Solving Irregularly Structured Problems in Parallel*, 1999.

Key Words
Adaptive algorithms; irregular computation; nested data parallelism; flattening; multithreading; Java; Fortran

For More Information

Jan Prins, professor
Department of Computer Science
University of North Carolina at Chapel Hill
CB#3175, Sitterson Hall
Chapel Hill, NC 27599-3175
Phone: (919) 962-1913
Fax: (919) 962-1799
E-mail: prins@cs.unc.edu
Web: www.cs.unc.edu/Research/IPA