1 of 1

DOI/ER/25043--71

A MULTI-LEVEL DATA-FLOW ARCHITECTURE FOR SIGNAL AND DATA PROCESSING APPLICATIONS

DE-FG03-87ER25043

Final Report

Technical Point of Contact

Jean-Luc Gaudiot, Ph.D.

Associate Professor

Department of Electrical Engineering - Systems

University of Southern California

Los Angeles, CA 90089-2563

213-740-4484

213-740-4449 (FAX)

gaudiot@usc.edu (E-mail)

September 2, 1993

DISCLAIMER

This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency thereof, nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

MASTER

DISTRIBUTION OF THIS DOCUMENT IS UNLIMITED

A grant was awarded to us by the Department of Energy, Office of Energy Research, in May 1987 to support the design and performance analysis of a large grain data-driven multiprocessor system for numerical applications. The basic idea of the work is to apply the data-driven principles of execution at a more appropriate level than conventional “atomic” instructions. For this purpose, a level such as that of vector operations was under study.

This document represents the final report concerning the results of the research supported by this grant.

1 Results

The goals of the project entailed an analysis of Partial Differential Equation solvers on data-driven environments, a preliminary design of our multi-level architecture, an in-depth study of some of the mechanisms of execution (*e.g.*, matching), and a design of the software environment. As enumerated in the original proposal, our work has yielded results in three different domains:

1. Specifications of the application programs
2. Design of the general concepts of the architecture and simulation
3. Implementation of a translating environment

We will now turn to each of the above items and examine specific research results.

1.1 Specifications of the application programs

We have undertaken the simulation of large scale numerical applications in order to evaluate the final performance of our Multi-level Tagged-token Data-flow (MTD) system. These applications have included basic iterative linear system solvers, such as Jacobi method and chaotic relaxation, and iterative multigrid Elliptic PDE solvers.

Two steps have been followed to implement these applications on the system. First, we have described each algorithm in SISAL and proposed proper language constructs for different types of algorithms. Second, we have translated the SISAL program into data-flow graphs and implemented the graphs on our MTD system.

Algorithm parallelization and macro-actor generating schemes have been studied to decide the trade-off between parallelism and speedup.

We have analyzed several Partial Differential Equations solvers in the context of data-driven execution. Multi-grid problems have proven to be particularly well suited to partitioning at a higher level and execution on a vector basis [17]. “Conventional” Jacobi problem solvers have been studied in micro-data-flow environments in both *non-chaotic* and *chaotic* implementations [15]. A new structure handling approach (*token no-labeling* and *matching store with locks*) was introduced for the data-flow execution of the latter. It was shown that, although chaotic relaxation would consistently produce lower execution times, both

methods of scheduling would present the same limitations in terms of speed-up. Indeed, for an 8-point grid, chaotic relaxation would proceed twice as fast as synchronous relaxation. However, speed-up in both cases would be limited to 3 which is much below the potentially available parallelism of 8. This phenomenon has been traced to the overhead required by the implementation of the model of synchronization (namely the U-interpreter). It was verified that the granularity of the synchronization model (micro-data-flow) introduced much overhead (on the order of 80%). Furthermore, this overhead (operations which do not belong to the algorithm *per se*) is constituted of mostly *sequential* instructions which severely limit performance improvement (Amdahl's law). This confirmed that a more appropriate level of data-flow execution is needed.

We have pursued our work on Partial Differential Equation Solvers reported in [15] and are reaching interesting observations in the domain of macro-actor execution. The work has obtained the following results:

1. From simulation observations, our data-driven architecture shows promises on those basic iterative methods, which are used as parts of many advanced PDE solvers.
2. A parallel multigrid algorithm, which is suitable for data-driven architectures, has been developed [16].
3. New matching methods for Chaotic execution have been developed and presented in [15].
4. The effect of Macro-actors (lump the sequential actors) has been shown by simulation to compare favorably with micro-actor execution. Very importantly, it has been observed that macro-actor lumping would provide not only an improvement in execution time but also a corresponding improvement in speed-up. This is being traced to the overhead introduced by the low level of the micro execution model. In addition, it can be seen that this overhead is, in many cases, largely sequential.

In another set of experiments, we are working on solving discrete relaxation labeling algorithm problems. Studies involving three different levels of resolution have been undertaken and preliminary results demonstrate the validity of our approach. More simulation runs will be undertaken.

1.2 Design of the general concepts of the architecture and simulation

A deterministic simulator [25] which incorporates the principles of macro-actor execution has been designed and has now been in use for a few weeks in our research group. At the lowest level, execution is assumed to be sequential. Its upper level is a tagged token data-flow machine with "fat" data tokens while the lower level execution can be programmed by micro instructions.

In addition, it should be noted that a particular effort has been pursued in the direction of studying the models of matching. Along these lines, we have developed the *partial matching* with *pre-emptive execution* described in [13]. In this model, although execution still proceeds in macro-data-flow fashion, matching can be performed as soon as an *executability subset* of the operands has arrived. When execution proceeds and encounters an *operand fault*, the actor is pre-empted until enough data is received on another partial execution subset. This model of execution combines the advantages of micro and macro-data-flow execution: on the one hand, it reduces overhead as allowed in macro-data-flow. On the other hand, however, it retains tolerance to latency as would a micro-data-flow system.

Scheduling in micro-data-flow execution has been studied (reported in [5, 3]), and we have demonstrated methods of program partitioning which will improve the speed of iterative algorithms. These results will be directly applicable to our macro-data-flow architecture in that they show how lumping several iterations into a single macro operation improves performance.

On another front, we have begun looking at practical implementations of our model. Our underlying philosophy is to be able to do “rapid prototyping” of our architecture by utilizing off-the-shelf components. Therefore, each of the blocks we envision should remain simple at the hardware level, but its function would be designed by microprogramming. Graph operations (token matching, formatting etc) and computational operations (conventional instructions) would be executed in an asynchronous manner. Among the issues under study, the problem of virtual-memory matching is given priority. (*i.e.*, how to fit on a necessarily limited machine a large number of tokens without introducing bottlenecks or deadlocks).

1.3 Implementation of a translating environment

Several subtasks are being undertaken under this heading:

1. Automatic translator from SISAL is in progress. A simple (one-to-one) translator is expected to be ready at the end of the semester while a more complex version (many-to-one) should become available at some point during the summer.
2. Assembler: it has become increasingly important to generate medium sized graphs rapidly and with a minimum amount of debugging. Hand drawing is a notoriously poor method. Therefore, we are designing a simple assembly language and the corresponding assembler.
3. Graphic display and statistics generation: with the advent of our simulator and the complex test programs, we find that the amount of information is difficult to interpret. In addition, we also realized that simplistic measures (execution time, speed-up) do not tell the whole story, particularly with regard to what happens *during* execution. We are therefore investigating other measures (involving histogramming).
4. I/O operations [2]: we have proposed introduced new constructs in SISAL and have analyzed the decomposition of the program into “computation” graph and “I/O” graph.
5. Some modifications to SISAL: This came as a spin-off of our visit to the Royal Melbourne Institute of Technology in Australia. We realized that many forms of updates would be difficult to specify in SISAL. In [1], we have demonstrated a new reduction operator, and have proposed implementation schemes for shared memory systems, as well as message passing machines.

References

- [1] D. Abramson, W. Heath, and J-L. Gaudiot. Parallel manipulation of arrays in SISAL. Technical report, RMIT Technical Report, January 1989.
- [2] P. Evripidou and J-L. Gaudiot. Input / output operations in a data-driven environment. Technical Report CRI 88-45, USC Dept. Electrical Engineering-Systems, Computer Engineering, July 1988.
- [3] P. Evripidou and J-L. Gaudiot. Numerical algorithms in a data-driven environment. Technical Report CRI 88-19, USC Dept. Electrical Engineering-Systems, Computer Engineering, April 1988.
- [4] P. Evripidou and J-L. Gaudiot. A decoupled data-driven architecture with vectors and macro actors. In *Proceedings of the VAPP-CONPAR '90 Joint Conference on Vector and Parallel Processing*, Zürich, Switzerland, September 1990.
- [5] P. Evripidou and J-L. Gaudiot. A decoupled graph/computation data-driven architecture with variable resolution actors. In *Proceedings of the 1990 International Conference on Parallel Processing*, St Charles, Illinois, August 1990.
- [6] P. Evripidou and J-L. Gaudiot. Distributed Input/Output processing in data-driven multiprocessors. In *Proceedings of the Second IEEE Symposium on Parallel and Distributed Processing*, Dallas, Texas, December 1990.
- [7] P. Evripidou and J-L. Gaudiot. Input/Output operations for hybrid data-flow/control-flow systems. In *Proceedings of the Fifth International Parallel Processing Symposium*, Anaheim, California, April 1991.
- [8] M. E. Exum and J-L. Gaudiot. Network design and allocation considerations in the Hughes Data-Flow Machine. *Parallel Computing*, 1990.
- [9] J-L. Gaudiot. Data-driven multicomputers in digital signal processing. *Proceedings of the IEEE*, pages 1220–1234, September 1987.
- [10] J-L. Gaudiot. Parallel computing: one opportunity, four challenges. In *Proceedings of the 1989 IEEE Conference on Data Engineering*, February 1989.
- [11] J-L. Gaudiot and C. M. Lin. Performance of asynchronous algorithms in multi-level data-driven systems. In *International Federation for Information Processing*, pages 527–557, December 1989. Lyon France.
- [12] J-L. Gaudiot and C.M. Lin. Chaotic linear system solvers in a variable-grain data-driven multiprocessor systems. In *Proceedings of the Conference on Parallel Architectures and Languages Europe*, Eindhoven, The Netherlands, June 1991.

- [13] J-L. Gaudiot and W. Najjar. Macro-actor execution on multilevel data-driven architectures. In *Proceedings of the IFIP Working Group 10. 3 Working Conference on Parallel Processing*, Pisa, Italy, April 1988.
- [14] J-L. Gaudiot and Y.H. Wei. Token relabeling in a tagged token Data-Flow architecture. *IEEE Transactions on Computers*, September 1989.
- [15] C. M. Lin, M. H. Hosseiniyar, and J-L. Gaudiot. Solving partial differential equations in a data-driven multiprocessor environment. In *Proceedings of the 15th Annual International Symposium on Computer Architecture*, May-June 1988.
- [16] C. M. Lin, W. Proskurowski, and J-L. Gaudiot. A parallel multigrid method for data-driven multiprocessor systems. In *Proceedings of the Fourth Cooper Mountain Conference on Multigrid Methods*, April 1989.
- [17] W. Najjar and J-L. Gaudiot. A hierarchical data-driven model for multigrid problem solvers. In *Proceedings of the International Symposium on High Performance Computer Systems*, Paris, France, December 1987.
- [18] W. Najjar and J-L. Gaudiot. Scalability analysis in gracefully degradable large systems. *IEEE Transactions on Reliability*, June 1991.
- [19] A. Sohn and J-L. Gaudiot. Data-driven production systems: A macro actor/token implementation. In *Proceedings of the International Symposium on Intelligent Control*, Philadelphia, Pennsylvania, September 1990.
- [20] A. Sohn and J-L. Gaudiot. A macro actor/token implementation of production systems on a data-flow multiprocessor. In *Proceedings of the Twelfth International Joint Conference on Artificial Intelligence*, Sydney, Australia, August 1991.
- [21] A. Sohn and J-L. Gaudiot. Performance evaluation of the multiple root approach to the rete pattern matcher for production systems. In *Proceedings of the 1992 Fifth Generation Computer Systems Conference*, Tokyo, Japan, June 1992.
- [22] Y. H. Wei and J-L. Gaudiot. Demand-driven interpretation of FP programs on a data-flow multiprocessor. *IEEE Transactions on Computers*, August 1988.
- [23] Y. H. Wei and J-L. Gaudiot. Lazy evaluation of FP programs: a data-flow approach. In *Proceedings of the International Conference on Fifth Generation Computer Systems 1988*, November 1988.
- [24] Y. H. Wei and J-L. Gaudiot. Compiling programs to direct access data-flow graphs. In *Proceedings of the 1990 International Conference on Parallel Processing*, 1990.
- [25] N. Yoo and J-L. Gaudiot. The USC macro-data-flow simulator. Technical Report Technical Report Number: CENG-89-27, USC, October 1989.

reprints & preprints
removed; conf papers
cycled separately.

**DATE
FILMED**

10 / 18 / 93

END