Simulation Between Enhanced Meshes and the Multiple Associative Computing (MASC) Model

Johnnie W. Baker and Mingxian Jin
Department of Math and Computer Science
Kent State University, Kent, OH 44242
Voice: (330) 672-4004 Fax: (330) 672-7824
{jbaker, mjin}@mcs.kent.edu

ABSTRACT

MASC (for Multiple Associative Computing) is a practical, highly scalable joint control parallel, data parallel model that naturally supports massive parallelism and a wide range of applications. In this paper, we propose efficient algorithms for the MASC model with a 2-D mesh to simulate enhanced meshes, e.g., meshes with multiple broadcasting (MMB), and basic reconfigurable meshes (BRM). The results not only show the power of the MASC model in terms of the enhanced mesh models but also provide an automatic conversion of numerous algorithms designed for enhanced meshes to the MASC model.

Keywords

Parallel models of computation, associative computing, simulation, enhanced meshes, MSIMD, MMB, MASC.

1. INTRODUCTION

The MASC (for *Multiple Associative Computing*) model of parallel computation, developed at Kent State University, is a generalized version of an associative style of computing that has been in use since the early 1970's. It is a hybrid SIMD/MIMD model with an array of processing elements (PEs) and one or more instruction streams (ISs), each of which issues commands to a unique set in a dynamic partition of all PEs. A MASC machine with n PEs and j ISs is written as MASC(n, j), where j is normally expected to be small in comparison to n.

Detailed features of the MASC model can be found in [3]. A brief description follows. Each PE (or *cell*) has a local memory and is capable of performing the usual functions of a sequential processor other than issuing instructions. An IS is logically a processor which has a bus connecting it to each cell and can send an instruction to all cells in constant time. Each cell listens to only one IS and can switch to another IS based on local data tests. Cells can be active, inactive, or idle. An active cell executes the program steps from its current IS while an inactive cell only listens. An IS can instruct an inactive cell to become active again. An idle cell is currently inactive and contains no essential program data and may be reassigned to an IS as an active cell (Figure 1).

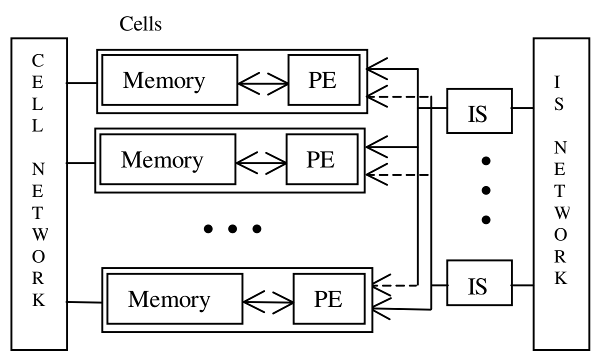

Figure 1. The MASC model

If the word length is assumed to be a constant, then the MASC model supports the global reduction operations of OR and AND of binary values and of maximum and minimum of integer or real values for each IS and its active PEs in constant time. The MASC model also supports a constant time associative search, which allows data in the local memories of the processors to be located by content rather than by address. The cells whose data value match the search pattern are called *responders*. Each IS can select (or "pick one") arbitrary responder from the set of active cells in constant time. This IS can also instruct the selected cell to broadcast a data item on the bus and all other cells listening to this IS receive this value in constant time.

This model also includes three real or virtual networks; namely, a PE network used for communications among PEs, an IS broadcast/reduction network used for communication between an IS and a set of cells, and an IS network used for IS communications.

A wide range of types of algorithms and several large programs have been developed for the MASC model and many of these have appeared in the literature. Moreover, results of simulating PRAM with MASC and self-simulation of MASC have been established recently[4].

The power of a computational model is indicated both by the efficiency of algorithms it can support and by the efficiency it can simulate other computational models. Thus, simulations between MASC and the well-known enhanced mesh models provide a better understanding the power of the MASC model and a way to easily convert the numerous algorithms designed for enhanced meshes to MASC.

Enhanced meshes are basic mesh models augmented with fixed or reconfigurable buses. In this paper, we consider *the mesh with a single global bus, the mesh with multiple broadcasting* (MBB) model using row and column buses, and *the reconfigurable mesh* (RM) model using buses that are dynamically created while a problem is being solved. Assume each processor in RM has four ports, referred to as N,S,E, and W. We consider only the *basic RM* (BRM) in which every processor may set at most one connection involving one of the reconfigurable pairs {N,S} or {E,W}.

2. SIMULATIONS

Our work provides efficient simulation algorithms between enhanced meshes and MASC. Some results based on these simulations are also included. In our simulation algorithms, it is assumed that MASC(n, j) has a 2-D mesh network for the PEs with size $\sqrt{n} \times \sqrt{n}$ with row–major ordering and that $j = \sqrt{n}$. The enhanced meshes considered are assumed to have the same size and ordering. We also assume that each PE in MASC has the exactly same computation power and local communication power (between neighbors) as a processor of an enhanced mesh processor. The processors in MASC are mapped to those in the same position in the enhanced meshes. So only the simulation of broadcasting the enhanced meshes needs to be dealt with. Since $\sqrt{n}$ ISs are assumed, we assign the ith IS in MASC to both the ith row and the ith column. Simulating an enhanced mesh with a global bus with MASC(n, 1) in constant time is trivial.

A constant time simulation of MMB with MASC is given first. In order to simulate the MMB row broadcast, the algorithm proceeds as follows. First, all PEs switch to their assigned row IS. Each PE row in an MMB should have at most one PE that needs to broadcast. The IS for each row checks to see if there is a PE that wishes to broadcast a value and, if true, instructs this PE to place its broadcast value on the MASC bus. The broadcast operation is completed in O(1). The simulation of broadcast operations along column buses is done analogously. The total running time is O(1). Notice we need no extra memory here. Further, by identifying a problem that can be solved faster using this MASC model than is possible using the MMB model, we observe that MASC (n, j) with a 2-D mesh is strictly more powerful than a $\sqrt{n} \times \sqrt{n}$ MMB when $j = \Omega(\sqrt{n}\,)$. Hence, any algorithm for a $\sqrt{n} \times \sqrt{n}$ MMB can be executed on MASC(n, j) with $j = \Omega(\sqrt{n}\,)$ and a 2-D mesh with a running time at least as fast as the MMB time. Making obvious algorithm adjustments, we also obtain the following two extended results. First, for any constant c, MASC(n,1) with a $n \times c$ mesh connection is more powerful than a $n \times c$ MMB. Any algorithm on a $n \times c$ MMB can be executed on a MASC(n,1) with a $n \times c$ mesh connection with a running time of the same complexity or smaller. Second, MASC(mn, j) with a mesh connection is more powerful than a $m \times n$ MMB, when j is $\Omega(\max\{m, n\})$. Any algorithm on a $m \times n$ MMB can be executed on MASC(mn, $\Omega(\max\{m, n\})$) with a $m \times n$ mesh connection with a running time of the same complexity or smaller. The following theorem summarizes the primary results of the above simulation.

Theorem 1. MASC(n, j) with a 2-D mesh is strictly more powerful than a $\sqrt{n} \times \sqrt{n}$ MMB when $j = \Omega(\sqrt{n}\,)$. Any algorithm for a $\sqrt{n} \times \sqrt{n}$ MMB can be executed on MASC(n, j) with $j = \Omega(\sqrt{n}\,)$ and a 2-D mesh with a running time at least as fast as the MMB time.

Next, we simulate BRM with MASC. The simulation algorithm includes two parts. The first part is preprocessing when any switch is reconfigured. The ISs in parallel assign leader PEs for each of their subbuses in a right-to-left order by checking the connection status of each BRM processor. Each PE sets a variable to store the connection status of its corresponding BRM processor. Once the leader of a subbus has been found, the IS sends the column number of the leader to all PEs that share the same subbus. This takes $O(\sqrt{n}\,)$ in the worst case. The second part is to simulate a broadcast. This is performed by setting a flag for those PEs that wish to broadcast a value, and then sending the value to all PEs that share the same leader number, i.e., the same subbus. A broadcast operation over a subbus can be simulated in O(1). However, if there are several subbuses in one row and several broadcast requests simultaneously, the row IS must handle these requests sequentially. This results in the worst case time of $O(\sqrt{n}\,)$. The simulation of a broadcast along column buses is done analogously. Since each PE needs only constant extra local variables to store data in the simulation, the total extra memory used is $O(n)$ for all PEs, which is an insignificant cost. For more detail about this algorithm, see [1]. This simulation yields the following theorem.

Theorem 2. MASC(n, j) with a 2-D mesh connection where $j = \Omega(\sqrt{n}\,)$ can simulate a $\sqrt{n} \times \sqrt{n}$ BRM in $O(\sqrt{n}\,)$ time and $O(n)$ extra memory.

The reverse simulation of these models is also given. To simulate MASC with MMB, a local computation, a memory access, and a 2-D mesh data movement instruction by one PE is executed exactly same in both of the models. Each of the $\sqrt{n}$ MMB processors in the first column is also assigned to simulate an IS and to broadcast an instruction stream. Each of these processors also stores a copy of the program. The MMB simulates the execution of the instruction of the ISs sequentially. An instruction stream is first broadcast to the processors in the leftmost column bus and then these processors broadcast it along the rows to all processors. Each MMB processor checks two predefined variables to decide whether or not to execute the current instruction. For the MASC reduction operations (i.e., OR, AND, maximum, and minimum), more work is required. When a MMB-processor receives a reduction operation from an IS, it does nothing at this step provided it currently is assigned to this IS, and is active. Otherwise, it determines the null value for this reduction operation and prepares to use this value in the reduction operation. The null values are as follows: 0 for OR, 1 for AND, MININT for maximum, etc. Next, the optimal algorithm provided by [2], which takes $O(n^{1/6})$, is used to compute the reduction and get the value in the processor located in the first row and the first column, i.e., P(1,1). The final step is to send this result from P(1,1) to the corresponding IS in MMB using the first column bus. As $\sqrt{n}$ ISs are assumed for MASC, the worst time for the simulation is $O(\sqrt{n} \times n^{1/6})$ or $O(n^{2/3})$. In the algorithm, there are constant extra variables for each MMB-processor used. Additionally, the MMB processors in the first column require extra memory to store a copy of the program to be executed, which is constant length. So the total extra memory used is $O(n)$.

Since the BRM is more powerful than the MMB, it can also execute the above simulation of MASC with a 2-D mesh. This gives the following theorem.

Theorem 3. MASC(n, $\sqrt{n}$) with a 2-D mesh connection can be simulated by a $\sqrt{n} \times \sqrt{n}$ MMB or BRM with $O(n^{2/3})$ time and $O(n)$ extra memory.

3. CONCLUSION

The comparison of MASC with the well-established enhanced meshes models provide an effective means to understand the power of MASC. Also, the constant time simulation of MMB by MASC enables MASC to execute each of the numerous algorithms designed for the MMB with the same running time.

For additional related results and information, see [1].

REFERENCES

[1] J.Baker, M. Jin, Simulation of Enhanced Meshes with MASC, a MSIMD Model, Proceedings of the 11th International Conference on Parallel And Distributed Computing Systems, Boston, MA, Nov., 1999, to appear.

[2] D. Bhagavathi, S. Olariu, W. Shen, L. Wilson, A Unifying Look at Semigroup Computations on Meshes with Multiple Broadcasting, Parallel Processing Letters, Vol. 4, 1994, pp. 73-82.

[3] J. Potter, J. Baker, S. Scott, A. Bansal, C. Leangsuksun, C. Asthagiri, ASC: An Associative-Computing Paradigm, Computer, 27(11), 1994, pp. 19-25.

[4] D. Ulm, J. Baker, Simulating PRAM with a MSIMD Model (ASC), Proceedings of the International Conference on Parallel Processing, 1998, pp. 3-10.