On the Power of the Multiple Associative Computing (MASC) Model Related to That of Reconfigurable Bus-Based Models

Mingxian Jin* and Johnnie W. Baker+
(*) Department of Mathematics and Computer Science, Fayetteville State University, Fayetteville, NC 28301
Voice: (910) 672-1294 Fax: (910) 672-1070 Email: *mjin@uncfsu.edu*
(+) Department of Computer Science, Kent State University, Kent, OH 44242
Voice: (330)-672-9980 Fax: (330)-672-7824 Email: *jbaker@cs.kent.edu*

Abstract: *The MASC model is a multi-SIMD model that uses control parallelism to coordinate the interaction of data parallel threads. It supports a generalized associative style of parallel computation. The power of this model has been compared to that of priority CRCW PRAM and enhanced meshes. In this paper, we present the work on simulations between MASC and reconfigurable bus-based models, in particular, different versions of the Reconfigurable Multiple Bus Machine (RMBM). It is shown that MASC and the Basic RMBM (B-RMBM) can simulate each other in constant time if the number of buses on the B-RMBM is Θ(j) where j is the number of MASC instruction streams. Thus, when these two models satisfy the preceding condition, they have the same power. Simulations of other stronger versions of RMBM using MASC are also considered. Since the RMBM model has been shown to be as powerful as a general Reconfigurable Mesh (RM), our simulations can be used to establish a relationship between MASC and RM. As RM has been widely accepted as an extremely powerful model, our work gives a better understanding of the MASC model and provides useful information concerning its power.*
Key Words: *parallel computational model, associative computing, simulation, reconfigurable buses, Multi-SIMD*

1. Introduction

The MASC (for *Multiple Associative Computing*) model is a multi-SIMD model that uses control parallelism to coordinate the interaction of data parallel threads. The MASC model extends the concept of associative computing and provides a complete paradigm to support general parallel computation. Motivated from the STARAN computer built in 1970s by Goodyear Aerospace, the MASC model was proposed by Potter and Baker in 1994 [14] and has been studied at Kent State University as a practical model for years. Since the MASC model is a strictly synchronous model that supports SIMD computation, it is sometimes called a multi-SIMD or MSIMD model (i.e., a SIMD enhanced with multiple instruction streams).

In contrast to a number of other parallel models, the MASC model possesses certain constant time global properties such as constant time broadcasting, constant time global reduction operations, and constant time associative search, which have all been justified in [7]. These properties have made MASC able not only to solve general parallel processing problems effectively [2, 7, 13, 17] but also to solve some problems in special areas such as real-time air traffic control in an extremely efficient manner [9, 10]. Possible techniques for implementing the MASC model have been explored in [5, 6, 19, 21].

The power of a computational model is indicated both by the efficiency of algorithms it supports and by the efficiency with which it can simulate other computational models. In order to evaluate the power of the MASC model, simulation and comparison with other popular parallel models such as PRAM (*Parallel Random Access Machine*) and MMB (*Mesh with Multiple Buses*) have been studied previously. Constant time simulation of PRAM with MASC and constant time simulations between MMB with MASC have both been established [16, 3]. In this paper, we present our work on simulating MASC and reconfigurable bus-based models. We aim to relate the power of the MASC model with that of the RM (*Reconfigurable Mesh*) model, as RM has been widely accepted as an extremely powerful computation model. In order to establish a relationship between MASC and RM, we use a bridge model – RMBM (*Reconfigurable Multiple Bus Machine*) and discuss simulations between RMBM and MASC first. Then we extend the simulations to relate MASC and RM, as it has been shown that RMBM can be as powerful as RM. Our work provides a better understanding of the power of the MASC model.

This paper is organized into seven sections. In particular, Section 2 outlines the MASC, RM, and RMBM models. Sections 3, 4, and 5 present simulations between different versions of RMBM and MASC, respectively. Sections 6 extends the simulation results to relate MASC and RM. Section 7 gives concluding remarks and points out some open problems.

2. Related Parallel Models

In this section, we provide a short summary of the parallel models used in this paper, i.e., the MASC model, the RMBM model, and the RM model.

2.1 The MASC model

The MASC model is a multi-SIMD model that has been enhanced with associative properties. As shown in Figure 1, MASC consists of an array of processing elements, or PEs, and one or more instruction streams, or ISs. A MASC with *n* PEs and *j* ISs is written as MASC(*n*, *j*). Each of ISs is connected to a disjoint set of processing elements (PEs) that can be regrouped during execution of an algorithm.

Each PE, paired with its local memory, is called a *cell*. (In this paper, we use PEs and cells interchangeably when there is no conflict involved.). Instructed by its assigned IS, a PE performs arithmetic and logic operations as a conventional computer. However, a PE neither stores a copy of the program nor participates in decoding this program.

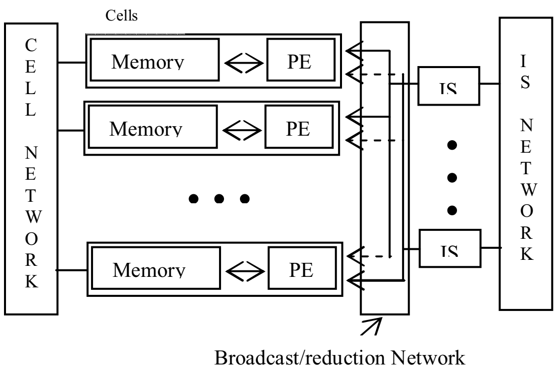

Figure 1 The MASC model

The MASC model may have three networks; namely, a cell network for cell communications, an IS network for IS communications, and a broadcast/reduction network for communication between an IS and its set of cells.

With its broadcast/reduction network, the MASC model supports basic associative operations in constant time (assuming word length is a constant) and broadcast from an IS to its active cells in constant time. These associative operations include the global reduction operations of OR and AND of binary values, the maximum and minimum of integer or real values, and the associative search by a given search pattern. Following an associative search, an IS can select (or "pick one") an arbitrary cell from its active cells that have data items matching the search pattern in constant time. An IS can instruct the selected cell to place a value on the bus and all other cells in its set receive this value in constant time. The feasibility of these assumptions has been justified in [11] and more details can be found there.

The number of ISs is assumed to be considerably smaller than the number of PEs. ISs are coordinated using control parallelism and communicate using the IS network, which may be implemented by a bus, shared memory, or other network. Further information about techniques for coordinating multiple ISs can be found in [5, 6].

A standard associative language that supports the one IS version of MASC or MASC(*n,* 1), also called ASC, has been implemented across a number of platforms. It provides true portability for parallel algorithms [13]. An extension of the ASC compiler is used in [6] to automatically execute the branches of an ASC program simultaneously using multiple ISs.

The MASC model is able to support general algorithm development and analysis. A wide range of different type of ASC algorithms and several large programs has been implemented using the ASC language. These include graph algorithms, computational geometry algorithms, string matching, image processing, database management, compiler optimization, and a real-time air traffic control system. Examples are given in [2, 7, 9, 10, 13, 17].

Simulations of PRAM and enhanced meshes using MASC have been studied in [16, 3]. These simulations allow PRAM algorithms and MMB algorithms to be executed by MASC using the same number of processors. As in [16], let PRAM(n, m) denote a PRAM with n processors and m shared memory. MASC(n, j) without a cell network can simulate priority CRCW PRAM(n, O(j)) in O(1) time with high probability. The "high probability" indicates an average running time. In [3], MASC(n, j) with a 2-D mesh can simulate a $\sqrt{n} \times \sqrt{n}$ MMB in O($\sqrt{n}$ /j) time. When $j = \Omega(\sqrt{n})$, the simulation takes O(1) time. Simulation of MASC(n, j) with a $\sqrt{n} \times \sqrt{n}$ MMB takes $O(jn^{1/6})$ time. It is shown that MASC(n, 1) with a 2-D mesh is more powerful than a $\sqrt{n} \times \sqrt{n}$ mesh with a global bus, and that MASC(n, j) with a 2-D mesh is also more powerful than a $\sqrt{n} \times \sqrt{n}$ MMB when $j = \Omega(\sqrt{n})$.

2.2 Reconfigurable Mesh (RM)

A 2-D RM (*Reconfigurable Mesh*) is a basic mesh model is enhanced by reconfigurable buses. Each processor has four ports, referred to as N, S, E, and W, that can be controlled locally such that disjoint buses can be established dynamically. With the ability of reconfiguring bus connections during algorithm execution, RM can create different communication patterns based on the algorithm needs.

Local connections of a processor can be restricted in different ways to obtain variants of RM. For example, if a processor is allowed to set at most one pair of the ports {EW} or {NS}, the restricted RM is called a BRM (*Basic Reconfigurable Mesh*). If all processors connect their ports as {EW} and {NS}, the restricted RM is essentially a MMB.

The RM model has been widely accepted as an extremely powerful model. A number of constant time algorithms have been discovered for the this model [1,4,11,12] (assuming a constant time bus broadcast), while they require non-constant time on other models [1, 12, 20]. Due to the wide acceptance of RM, we wish to establish a relationship between MASC and RM. This relationship could be a useful tool in evaluating the power of the MASC model. However, dissimilarities of the MASC model and a general RM create some difficulties to hinder a direct simulation of the two models. Instead, we consider a bridge model – the RMBM model – which has been shown to be equally powerful to RM. In this paper, we will simulate the MASC model and the RMBM model, aiming to establishing a relationship between MASC and RM. The RMBM model is introduced next.

2.3 Reconfigurable Multiple Bus Machine (RMBM)

The *Reconfigurable Multiple Bus Machine* (RMBM) is also reconfigurable bus-based model, proposed by Trahan et al. in [15,18]. As in Figure 2 cited from [15, 18]), the RMBM model consists of a set of processors and a set of buses which are used for processor communication. A processor can connect itself to a bus through the use of its local settings. A bus can be split into segments or fused to connect to another bus by a processor. Each processor has a number of packs of switches with each pack dedicated to one bus. There are up to five switches in each pack of switches. Two switches control the connection of the bus to the processor's reading port and writing port, respectively. Two switches are used to segment the bus at the point they are located. One is located to the left of the reading port and the other is between the reading and writing ports. The last switch is used to connect the processor fuse line to the bus. The processor fuse line can connect (not necessarily adjacent) multiple buses together.

A processor can set or reset a local switch in one time step but can only set or reset one switch as a time. Any processor can place a value (i.e., write) on a bus in one step and the other processors connected to the bus receive the value in the same step. A processor may connect its reading and writing ports to different buses. However, at any point of time, a processor can only connect one port to one bus, which means it can only perform either a read or a write in one step.

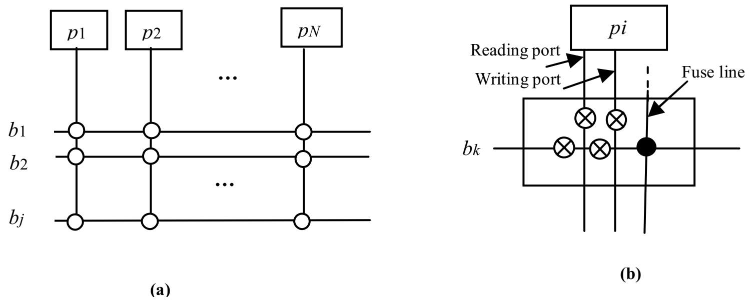

Figure 2. RMBM -- (a) Structure of the Model (b) Illustration of a pack of switches for a processor

Depending on switches available in the processors, there are four versions of RMBM defined in [15,18]:

B-RMBM: a processor only has switches to connect reading and writing ports to a bus.
S-RMBM: In addition to read/write switches, a processor can segment the bus.
F-RMBM: In addition to read/write switches, a processor can fuse buses using its fuse line.
E-RMBM: All five switches are functional for a processor; namely, a processor can connect to a bus for read/write, split the bus into two segments, and fuse the bus to a separate fuse line. This is the strongest version of RMBM.

Concurrent read (CR) on a RMBM bus is assumed in this paper. Although either exclusive write (EW) or concurrent write (CW) on a bus can be assumed, we only consider CW in our simulations. The reason is as follows. When there are multiple processors that wish to write on a bus, there is a predefined rule (e.g., common, priority, or arbitrary) to resolve the collision. One processor will win the competition and is allowed to write on the bus. This takes constant time on a CW bus. MASC can also handle CRCW-type operations. When an IS broadcasts a value to its PEs, all the PEs receive the value in one step. When an IS needs to obtain a value from multiple PEs (i.e., a CW situation), the IS uses an associative operation (e.g., global OR, global MAXIMUM, or PickOne) to determine which PE wins, which also takes constant time.

In [15,18], the relationships among different versions of RMBM and RM have been established. Considering the similarities of RMBM and MASC model, we establish simulations between MASC and RMBM first. Then we can extend the results to relate MASC and RM.

The following three sections present simulations between the MASC model and different versions of the RMBM model. We start with the weakest version – B-RMBM.

3. MASC vs. B-RMBM

Simulating MASC with B-RMBM is fairly straightforward, as there are some similarities between the structures of the two models. Let MASC(n, j) denote a MASC with n PEs and j ISs. Let B-RMBM(n, m) denote a B-RMBM with n processors and m buses. We use B-RMBM($n+j$, j) to simulate MASC(n, j) in O(1) time. Let each of the first j processors on the B-RMBM simulate a unique MASC IS. The read/write ports of each of these processors are always used to connect to unique bus. The other n processors simulate the n MASC PEs. For each of these n

processors, the bus that its reading and writing ports are connected to is determined by the IS that the corresponding MASC PE listens to. It is obvious that each MASC step takes O(1) time on the B-RMBM. We have following theorem.

Theorem 1. MASC(n, j) can be simulated by B-RMBM(n+j, j) in O(1) time.

Next, we consider simulating B-RMBM using MASC. Assume a B-RMBM (n, m) with n processors and m buses and MASC(n, j) with n PE*s* and j ISs. If $j \geq m$, let each MASC PE simulate one of n B-RMBM processors and each MASC IS simulate one of m buses. Depending on how a B-RMBM processor set the reading or writing port to a bus in a particular step, a MASC PE listens to the corresponding simulating MASC IS. The reading and writing ports can be differentiated by setting a read/write flag in the PE. This takes O(1) time.

If $j < m$, let IS_1 simulate B-RMBM buses b_1, b_{j+1}, b_{2j+1}, …; IS_2 simulate B-RMBM buses b_2, b_{j+2}, b_{2j+2}, …; IS_j simulate B-RMBM buses b_j, b_{2j}, b_{3j}, …. Since each IS simulates at most $\lceil m/j \rceil$ buses, all bus read/write operations on the B-RMBM takes O(m/j) time to be simulated sequentially on the MASC. When $m = \Theta(j)$, this is O(1) time. Since CRCW is assumed for bus access, we have Theorem 3, followed by Corollary 3.

Theorem 2. B-RMBM(n, m) can be simulated by MASC(n, j) in O(m/j) time. When $m = \Theta(j)$, this is O(1) time.

Corollary 3. The MASC model has the same power as CRCW B-RMBM when the number of buses on the B-RMBM is $\Theta(j)$ where j is the number of instruction streams on the MASC.

4. MASC vs. S-RMBM

Since S-RMBM is a B-RMBM enhanced by adding processors with the ability of splitting a bus into bus segments, the constant time simulation of MASC using B-RMBM can be obtained from the simulation of MASC using B-RMBM shown in section 3.

Now, we consider simulation of S-RMBM using MASC. Assume a S-RMBM (n, m) with n processors and m bus and a MASC(nm, max(m,n)) with nm PEs and max(n,m) ISs. The nm PEs on the MASC are divided into m groups with n PEs in each group. Initially, only n PEs in the first group are activated and each PE listens to one of n ISs. For any non-bus-access operation, each of these n PEs is used to simulate an S-RMBM processor. Each step takes O(1) time.

For a bus-access operation on an S-RMBM processor, the simulation proceeds as follows.

1) First, all PEs on the MASC are regrouped as follows. Let PE_{kn+i} listen to IS_i while k=0..m-1 (see Figure 3(a)). With this grouping, any data item to be written from PE_i is duplicated to PE_{n+i}, PE_{2n+i}, …, $PE_{(m-1)n+i}$ by an IS_i broadcast. Since an S-RMBM processor is allowed to read from or write to at most one bus at any point of time and there are max(n, m) ISs, this takes O(1) time.
2) Second, all PEs are regrouped with different IS assignments. The first n PEs listen to IS_1; the second n PEs listen to IS_2, …, the last n PEs listen to IS_m (see Figure 3(b)). IS_i is dedicated to simulate an S-RMBM bus b_i. With this grouping, IS_i creates subsets among all its PEs with each subset representing one bus segment on b_i. Assume there are k bus segments on b_i. ISi first only activates those PEs that have segment switches (either at the point of the reading port or the writing port) set and deactivated the rest of PEs. The PEs corresponding the two ends of b_i are exceptions and need to be activated also. IS_i takes two constant time steps to find two smallest index numbers of these active PEs, which determine the two ends of the first bus segment. Then, ISi activates PEs whose indices fall into these two ends and whose reading and/or writing ports are connected to b_i. If there is any read or write operation, it takes O(1) time to complete it. After then, ISi removes all these PEs from its PE set except the PE representing the second end of the bus segment. IS_i iterates the above steps to process

the second segment, the third segment, etc until all bus segments are processed on b_i. Since the maximum number of segments on a bus is $n/2$, the worst case time is O(n).

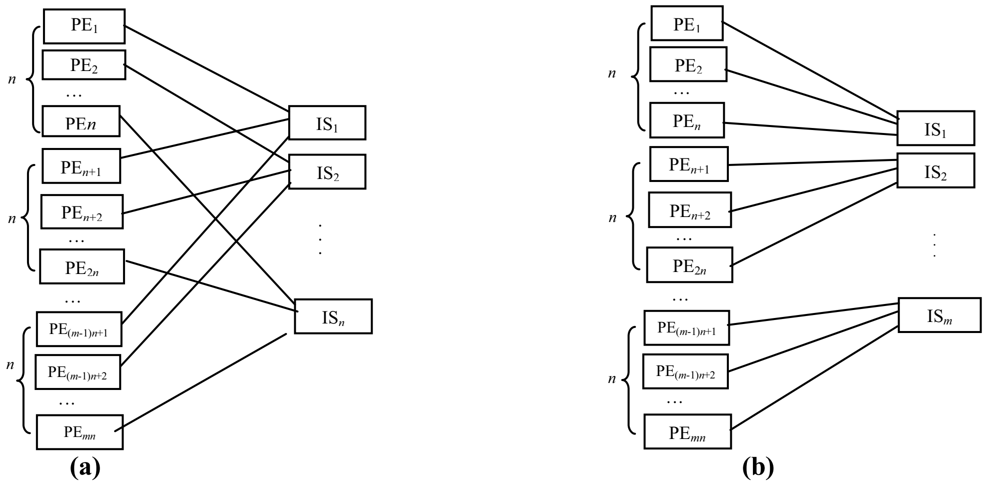

Figure 3. PE grouping on the MASC to simulate a bus-access operation of the S-RMBM (a) First step (b) Second step

3) Last, all PEs are regrouped as in the first step (see Figure 3(a)). The ISs broadcast all values just read back to the first n PEs to update data for future operations on those PEs. Since PEi (i=1..n) that simulates S-RMBM processor i has at most one data value read and there are max(n, m) ISs, this again takes O(1) time.

Summing all the above time, we have the total simulation time O(n) as in Theorem 5.

Theorem 5. A S-RMBM(n, m) can be simulated by MASC(nm, max(m, n)) in O(n) time.

5. MASC vs. F-RMBM and E-RMBM

Since both F-RMBM and E-RMBM are more powerful models than B-RMBM and S-RMBM, obviously it takes O(1) time to simulate MASC (n, j) using either F-RMBM(n+j, j) or E-FMBM(n+j, j) without setting fuse switches nor segment switches.

To simulate F-RMBM(n, m) using MASC(n, m)) (assuming $n > m$), let each of the first n PEs on the MASC simulate an F-RMBM processor and each MASC IS simulate an F-RMBM bus. Each PE on MASC takes O(1) time to simulate a non-bus access operation.

For a bus-access operation, the simulation is less efficient. We present a brief description here due to the space limit. More details will be included in an expanded version of this paper.

On the MASC, an intuitive approach is as follows. Initially, if an F-RMBM processor sets its reading/writing port connected to bus b_i (i = 1..m), the simulating PE is assigned to listen to ISi that is simulating b_i. If an F-RMBM processor fuses buses b_i and b_j, all PEs initially assigned to ISj are switched to ISi in one time step. Switching takes O(1) time.

However, an F-RMBM processor may set its reading/writing port to a bus other than those buses it fuses. So a simulating IS may need to read the fuse information from one of its PEs. This fuse information represents the fuse switch settings of the simulated F-RMBM processor. The IS then passes this information to other simulating ISs to instruct their listening PEs for switching. In order to communicate among ISs, an IS network is needed for the MASC. Assume an array is defined in each PE to store the bus numbers that its simulated F-RMBM processor sets its fuse switches. Since there are at most m items in an array, ISs take O($m*n/m$) or O(n) time to collect the data from all n PEs. Assume the time to route a data item on the IS network is routing(m), which is clearly a function of m. The time to distribute the m data items is then O(nm

routing(*m*)). After the distribution, switching PEs to their destination IS, again, takes O(1) time. The total time is then O(*nm* routing(*m*)).

Theorem 6. F-RMBM(*n*, *m*) can be simulated by MASC(*n*, *m*) with an IS network in O(*nm* routing(*m*)) time while $n > m$ and routing(*m*) is the time to route a data item on the IS network.

To simulate E-RMBM, we may divide the simulation into to two parts. One is simulating fuse lines as in F-RMBM, and one is simulating bus segments as in S-RMBM. We combine the simulation results from Section 4 and this section in the following Theorem 7.

Theorem7. A E-FRMBM(*n*, *m*) can be simulated by MASC(*nm*, max(*n*, *m*)) with an IS network in O(*nm* routing(*m*)) time while routing(*m*) is the time to route a data item on the IS network.

6. Extension on relationships among MASC, PRAM and RM

In [15,18], relative powers of the PRAM, RMBM, and RM models have been well studied. Some results related to our work can be described as follow. Related models can be placed into two groups, i.e.,

***Group* 1**: B-RMBM, S-RMBM, PRAM

***Group* 2**: F-RMBM, E-RMBM, RM

All models in the same group have the same power. Any model in Group 2 is more powerful that any model in Group 1. Note that all cases are in CRCW. For the reason of completeness, we have also included the PRAM model.

Based on these relationships, we have the following observations.

1) Since MASC and B-RMBM have the same power when the number of B-RMBM buses is restricted as in Corollary 3, MASC has the same power as CRCW PRAM (with appropriate restrictions).
2) Since S-RMBM has the same power as B-RMBM, it has the same power as MASC (when the number of buses is appropriately restricted). Although our S-RMBM simulation of MASC in Section 4 takes non-constant time, a constant time simulation is possible.
3) It can be shown that S-RMBM has the same power as BRM (the proof is omitted). Therefore, a constant time BRM simulation using MASC is also possible.
4) Since MASC has the same power as B-RMBM, it is less powerful than RM, E-RMBM and F-RMBM. The main reason is that the limited number of ISs restricts the data movements.

7. Conclusion

In this paper, we have shown simulations between MASC and versions of RMBM – reconfigurable bus-based models that can be as powerful as RM. By taking these simulations as a bridge, we have analyzed the power of the MASC model relative to RM. The power of the MASC model is comparable to that of CRCW PRAM but less than that of RM. For some special cases of RM with the restrictions, a constant time simulation can be obtained (e.g. MMB and possibly BRM).

Some problems still remain open. As seen in the paper, when we increase the number of PEs and the number of ISs, the simulation time can be reduced but it also degrades efficiency. Determining minimal cost simulation models (i.e., ones of minimal size) which achieve simulation time optimality is a problem that needs further work. Although the MASC model is less powerful than RM, finding a lower bound for the simulation time is another problem that we plan to study further.

Acknowledgement

The authors wish to thank the two anonymous reviewers for their valuable comments.

References

1. S. G. Akl, *Parallel Computing: Models and Methods*. Prentice Hall, New York, 1997.
2. M. Atwah, J.Baker, S. Akl, An Associative Implemen-tation of Classical Convex Hull Algorithms, *Proc. of the 8th Int'l Conf. on Parallel and Distributed Computing Systems*, pp. 435-438, 1996.
3. J. Baker, M. Jin, Simulation of Enhanced Meshes with MASC, a MSIMD Model, *Proc. of the 11th IASTED Int'l Conf. on Parallel and Distributed Computing Systems*, pp. 511-516, 1999.
4. Y. Ben-Asher, D. Peleg, R. Ramaswami, A. Schuster, The Power of Reconfiguration, *J. of Parallel and Distributed Computing*, 13, pp. 139-153, 1991.
5. W. Chantamas and J. Baker, A Multiple Associative Model to Support Branches in Data Parallel Applications using the Manager-Worker Paradigm, *Proc. of the 19th IPDPS (Workshop in MPP*), April 2005
6. W. Chantamas, J. Baker, and M. Scherger, Compiler Extension of the ASC Language to Support Multiple Instruction Streams in the MASC Model using the Manager-Worker Paradigm, *Int'l Conf. on Parallel and Distributed Processing Techniques and Applications (PDPTA'06)*, Las Vegas, June 26-29, 2006
7. M. Esenwein, J. Baker, VLCD String Matching for Associative Computing and Multiple Broadcast Mesh, *Proc. of the 9th Int'l Conf. on Parallel and Distributed Computing Systems*, pp. 69-74, 1997.
8. M. Jin, J. Baker, and K. Batcher, Timings of Associative Operations on the MASC model, *Proc. of the 15th IPDPS (Workshop in Massively Parallel Processing*), CD-ROM, April 2001
9. W.C. Meilander, J.W. Baker, M. Jin, Importance of SIMD Computation Reconsidered, *Proc. of the 17th IPDPS(WMPP workshop*), CD-ROM, April 2003
10. W.C. Meilander, M. Jin, J.W. Baker, Tractable Real-Time Control Automation, Proc. of the *14th IASTED Inte'l Conf. on Parallel and Distributed Systems (PDCS 2002)*, pp. 483-488
11. R. Miller, V.K. Prasanna-Kumar, D. Reisis, and Q. Stout, Parallel Computations on Reconfigurable Meshes, *IEEE Trans. Computer*, 42 (1993), pp.678-692
12. S. Olariu, J. Schwing, J. Zhang, On the Power of Two-dimensional Processor Arrays with Reconfigurable Bus Systems, *Parallel Processing Letters* vol. 1, No. 1, pp. 29-34, 1991.
13. J. Potter, *Associative Computing: A Programming Paradigm for Massively Parallel Computers*, Plenum Press, New York, 1992.
14. J. Potter, J. Baker, S. Scott, A. Bansal, C. Leangsuksun, C. Asthagiri, ASC: An Associative-Computing Paradigm, *Computer*, 27(11), 19-25, 1994.
15. J. L. Trahan, R. Vaidyanathan, and R. K. Thiruchelvan, On the Power of Segmenting and Fusing Buses, *Journal of Parallel and Distributed Computing,* vol. 34, no.1, (1996), pp. 82-94.
16. D. Ulm, J. Baker, Simulating PRAM with a MSIMD Model (ASC), *Proc. of the Int'l Conf. on Parallel Processing*, pp.3-10, 1998.
17. D. Ulm, J. Baker, and M. Scherger, Solving a 2-D Knapsack Problem Using a Hybrid Data-Parallel/Control Sytle of Computing, *Proc. of the 18th IPDPS (WMPP Workshop*), April 2004
18. R. Vaidyanathan and J. L. Trahan, *Dynamic Reconfiguration: Architectures and Algorithms*, Kluwer Academic/Plenum Publishers, New York, 2003
19. R. Walker, J. Potter, Y. Wang, and M. Wu, Implementing Associative Processing: Rethinking Earlier Architectural Decisions, *Proc. of the 15th IPDPS (WMPP Workshop*), April 2001
20. Wang, G. Chen, Two-dimensional Processor Array with Reconfigurable Bus System Is At Least As Powerful As CRCW Model, *Information Processing Letters* 36, pp. 31-36, 1990.
21. H. Wang, and R. A. Walker, Implementing a Scalable ASC Processor, *Proc. of the 17th IPDPS (Workshop in Massively Parallel Processing*), CD-ROM, April 2003