On Using the UML to Describe the MASC Model of Parallel Computation

M. Scherger, J. Potter, and J. Baker
Department of Mathematics and Computer Science
Kent State University
Kent, Ohio 44242

Abstract - *A Unified Modeling Language (UML) description of the MASC model of parallel computation is presented. This UML description identifies MASC objects and specifies various object and inter-object relationships, dependencies, and behaviors. This was achieved by describing various views of the MASC model by using many of the UML structural and behavioral diagrams. The use of using UML to describe MASC has been highly effective for further parallel modeling techniques, comparisons to other parallel models, MASC parallel system software research, and MASC algorithm development.*

Keywords: parallel models, object oriented, parallel architectures.

1 Introduction

The Unified Modeling Language (UML) developed by Booch et al. [1] is becoming the standard modeling tool for specifying, visualizing, constructing, and documenting software intensive systems. The UML is a modeling language capable of providing several different views of systems for various target software and hardware disciplines. This is achieved by presenting different diagrams detailing various structural, object, behavioral, activity, and usage interactions of the components in the model; each intended to detail a particular aspect of component interaction.

The MASC (for Multiple Instruction Stream Associative Computing) model for parallel computation supports a generalized version of an associative style of computing that has been in use since the introduction of the associative SIMD computers (STARAN, MPP) in the early 1970s and 1980s. The MASC model includes the well-known data parallel-programming paradigm and extends this paradigm to a complete computational model. The associative features of the model allow data in the local memories of the processors (PE's) to be located by content rather than by address. A complete description of MASC (a renaming of the original ASC model to emphasize multiple instruction streams) can be found in [4] and [5].

The original description in [5], however, provides a conceptual view of the principal components and basic component interactions of the MASC model. This conceptual description of MASC is primarily used to introduce the model and not for research and development. Using the UML to describe the MASC transforms this conceptual description into one that is not only object oriented, but provides several views for further MASC research in algorithm development and predictability, system software design and implementation, and hardware simulation and modeling. This object-oriented description of MASC details the principal objects and inter-object behaviors by using the UML structural and behavior diagrams.

The remainder of this paper will first give a conceptual description of the MASC model of parallel computation and also present the

MASC predictability parameters used in algorithm predictability analysis. Next, the basic objects of the MASC model are presented and organized into classes for basic structural modeling. The structural UML descriptions are presented including a class hierarchy of the various objects in MASC. Once the structural elements are defined, the object responsibilities are outlined and the basic use-case diagrams of MASC are presented. Other behavioral UML diagrams are presented that describe the sequence diagrams for the MASC predictability parameters.

2 The MASC Model

The following is a conceptual description of the MASC model of parallel computation. As shown in figure 1, the MASC model consists of an array of processor-memory pairs called cells and an array of instruction streams.

A MASC machine with *n* cells and *j* instruction streams is denoted as *MASC(n, j)*. It is expected that the number of instruction stream processors be much less than the number of cells. The model also includes three virtual networks:

1. A cell network used for cell-to-cell communication. This network is used for the parallel movement of data between cells. This network could be a linear array, mesh, hypercube, or a dynamic interconnection network.
2. A broadcast/reduction network used for communication between an instruction stream and a set of cells. This network is also capable of performing common reduction operations.
3. An instruction stream network used for inter-instruction stream communication.

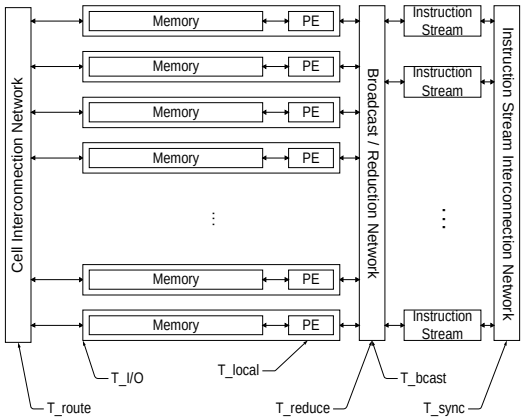

Figure 1: Conceptual view of MASC.

Cells can receive their next set of instructions to execute from the instruction stream broadcast network. Cells can send and receive messages to each other using some communication pattern via the cell network. Each instruction stream processor is also connected to two interconnection networks. An instruction stream processor broadcasts instructions to the cells using the instruction stream broadcast network. The instruction streams also may need to communicate and may do so using the instruction stream network. Any of these networks may be virtual and be simulated by whatever network is present.

MASC provides one or more instruction streams. Each is assigned to a unique dynamic partition of cells. This allows a task that is being executed in a data parallel fashion to be partitioned into two or more tasks using of control parallelism. The multiple IS's supported by the MASC model allows for greater efficiency, flexibility, and reconfigurability than is possible with only one instruction stream. While SIMD architectures can execute data parallel programs very efficiently and normally can obtain near linear speedup, data parallel programs in many applications are not completely data parallel and contain several non-trivial regions where significant

branching occurs. In these regions, only a subset of traditional SIMD processors can be active at the same time. With MASC, control parallelism can be used to execute these different branches simultaneously. Other MASC properties include:

- The cells of the MASC model consist of a processing element (PE) and local memory. The accumulated memory of the MASC model consists of an array of cells. There is no shared memory between cells.

- Each instruction stream is a processor with a bus or broadcast/reduction network to all cells. Each cell listens to only one instruction stream and initially, all cells listen to the same instruction stream. The cells can switch to another instruction stream in response to commands from the current instruction stream.

- An active cell executes the commands it receives from its instruction stream, while an inactive cell listens to but does not execute the command from its instruction stream. Each instruction stream has the ability to unconditionally activate all cells listening to it.

- Cells without further work are called idle cells and are assigned to a specified instruction stream, which among other tasks manages the idle cells.

- The average time for a cell to send a message through the cell network to another cell is characterized by the parameter t_{route}. Each cell also can read or write a word to an I/O channel. The maximum time for a cell to execute a command is given by the parameter t_{local}. The time to perform a broadcast of either data or instructions is given by the predictability parameter t_{bcast}. The time to perform a reduction operation is given by the predictability parameter t_{reduce}. The time for a cell to perform this I/O transfer is characterized by the parameter $t_{i/o}$. The time to perform instruction stream synchronization is characterized by the parameter t_{synch}.

- An instruction stream can instruct its active cells to perform an associative search in time $t_{bcast} + t_{local} + t_{reduce}$. Successful cells are called *responders*, while unsuccessful cells are called *non-responders*.

- The instruction stream can activate either the set of responders or the set of non-responders. It can also restore the previous set of active cells in $t_{bcast} + t_{local}$ time.

- Each instruction stream has the ability to select an arbitrary responder from the set of active cells in $t_{bcast} + t_{local}$ time.

- An active instruction stream can compute the *OR, AND, GLB,* or *LUB* of a set of values in all active cells in t_{reduce} time.

- An idle cell can be dynamically allocated to an instruction stream in $t_{synch} + t_{bcast}$ time.

3 MASC Structural Diagrams

The structural diagrams of the UML provide the model with a foundation of classes, objects, aggregations, and inheritance. The basic UML aggregation diagram for a MASC cell is illustrated in figure 2.

Beginning with the memory in a cell, the basic class for storage element is the Field class. The Field class is an abstract base class in which other concrete Field types

are derived (integer fields, string fields, real fields, Boolean fields, etc). Each cell has a collection of Fields used to store parallel data for program execution.

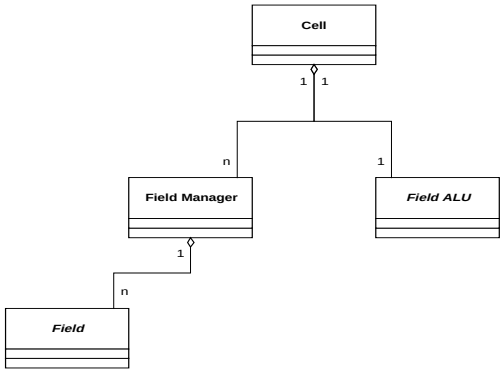

Figure 2: MASC cell aggregation diagram

To manage the Fields, the Field Manager class which maintains the collection of Fields. This purpose of this collection is to provide the memory addressing capability for the Cell and Instruction Stream classes. The Field Manager shown in figure 2 is identified with a cell; however, the functionality of the Field Manager could be associated with an instruction stream if the cell does not have any memory addressing capabilities.

Each cell also has a Field ALU class capable of performing basic arithmetic and logical operations on Fields. The functionality of the Field ALU class is not specified to allow for different types of processing elements to be "plugged-in" the MASC model. The cardinality of to cell to a Field ALU is 1:1. The Field ALU can be implemented as a singleton pattern to reflect that when a parallel model of computation (or implementation of a model) is supporting virtual parallelism, there are fewer physical processing elements than data to be processed. As illustrated in figure 3, allowing a Virtual Cell Manager class to maintain a collection of Field Managers supports virtual parallelism.

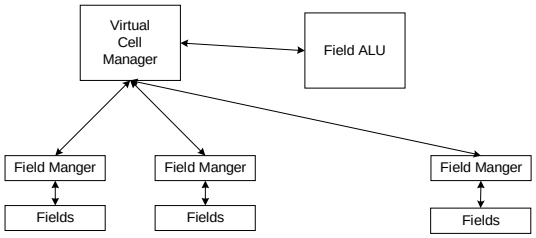

Figure 3: MASC Virtual cell organization.

A MASC instruction stream has the same basic components and functionality of a MASC cell; i.e. it has the capability of performing computations on local (scalar) Fields and communicates with other instruction streams. Since instruction streams must also be able to broadcast instructions to its partition of cells, and perform reductions from a partition of cells. Therefore, it is natural for the properties and functionalities of the Instruction Stream class to be derived from a Cell class. Using the UML inheritance diagram, an Instruction Stream class can be derived from a cell and this is illustrated in figure 4.

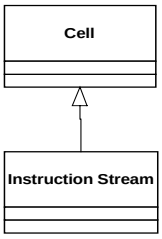

Figure 4: Instruction stream / cell inheritance.

An Interconnection Network class is a class used for communication between cells and/or instruction streams. The UML representation of the interconnection network used in MASC is an abstract class. This will allow different interconnection network class

implementations to be plug-in compatible with the existing MASC architecture. Thus, it will be possible to design and specify different types of networks used in MASC for different deployments of the model. For example, the cell network could be implemented using a grid mesh network, the broadcast reduction network could be implemented using a bus based network, while the instruction stream network could be implemented as a type of intelligent shared memory with basic network functionality.

Now that the basic structural components are defined, the basic UML aggregation diagram of MASC is illustrated in figure 5.

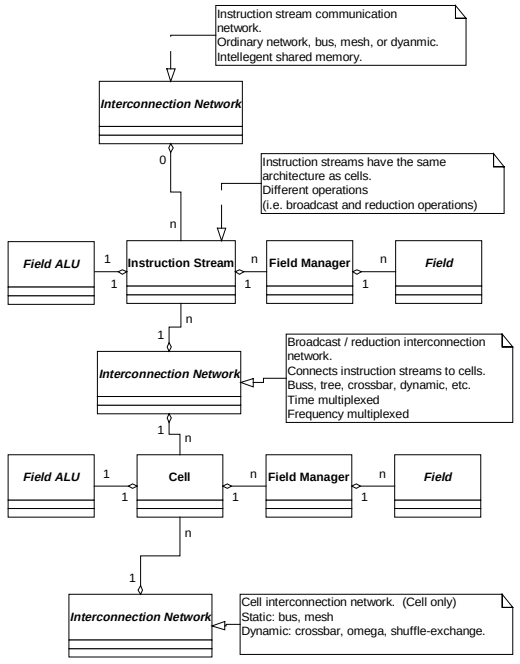

Figure 5: MASC aggregation diagram

The top Interconnection Network class is used for an object to allow for Instruction Stream communication and synchronization. The aggregation of an Instruction Stream is identical to that of a Cell, each allowing for Fields, Field Managers, and Field ALU classes. The middle Interconnection Network object is used for instruction stream broadcast and cell reductions. Finally the lower Interconnection Network class is used for inter-cell communication.

4 MASC Behavioral Diagrams

The MASC behavioral diagrams define class responsibilities by using the UML use cases. Behavioral diagrams also can define the object state transitions by using the UML state diagrams and event constraints by using the UML sequence diagrams.

Two such MASC actors are the Cell class and the Instruction Stream class. The responsibilities and behaviors of the Cell class and Instruction Stream class are described in figure 6. Notice that even though an Instruction Stream is derived from a Cell, the behavior and responsibilities are very different.

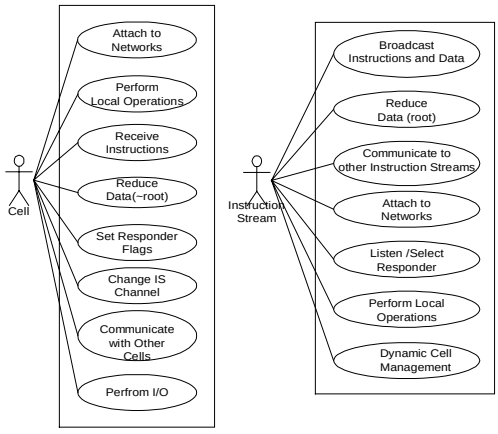

Figure 6: Example MASC actors and use case diagrams.

The MASC predictability parameters can also be defined using the UML sequence diagrams; two are presented in this paper. The T_{route} is a measurement of cell-to-cell communication. Since the communication

network and protocol are not specified however, the sequence diagram in figure 7 illustrates that the time T_{route} is bounded by the completion of all sends from one cell to another cell (sends to multiple cells are acknowledged by the * symbol).

Another UML sequence diagram is for the MASC predictability parameter T_{local} that measures the maximum time a local cell operation requires. This is illustrated in figure 8. The time T_{local} is bounded by the time from when an instruction stream broadcasts the instruction to the cell to the time the instruction sequence is complete. Since not all instructions are arithmetic, the ALU request and complete sequence is optional.

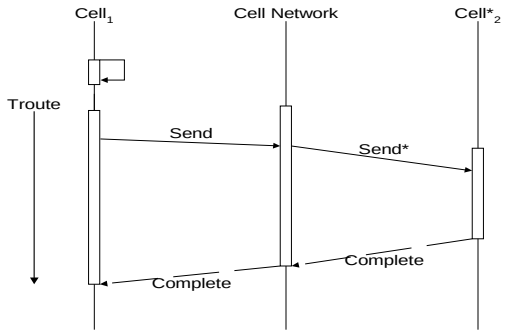

Figure 7: MASC predictability T_{route} using the UML sequence diagram.

This sequence diagram also considers that an instruction stream may broadcast a series of instructions to a cell to execute instead of broadcasting each instruction individually.

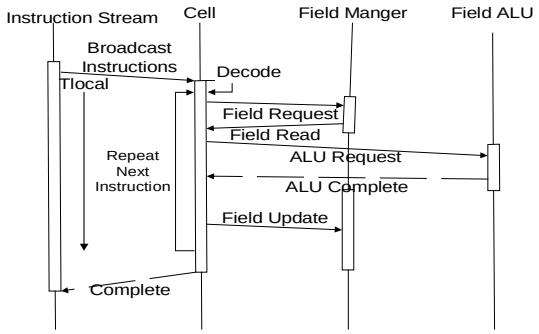

Figure 8: MASC predicability T_{local} using the UML sequence diagram.

A feature of parallel associative computing is to perform an associative search and process all the responding cells. This is a typical operation performed often using a basic parallel selection programming structure along with program iteration. Using the UML sequence diagram, the basic associative search-process-retrieve cycle is illustrated in figure 9. Note that "Cell*" is used to illustrate that all cells of an Instruction Stream are used.

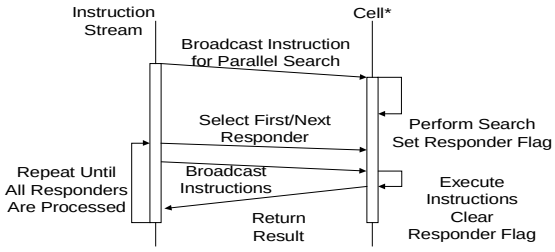

Figure 9: MASC description of the associative search-process-retrieve cycle using UML sequence diagram.

The instruction stream would first instruct all cells to perform a parallel search by broadcasting the datum and fields to search. Once complete, the first responder cell is identified, selected, and processed. This is repeated iteratively for the remaining responding cells.

5 Conclusions

The Unified Modeling Language has provided a structured and organized communication vehicle to describe and develop an object-oriented description of the MASC model of parallel computation. The MASC model is a model of parallel computation that supports an associative style of parallel computation that allows memory to be addressed by content rather than by address. By using the various UML structural and behavioral diagrams and views, the same model can be used for various disciplines of parallel computing research such as parallel architecture, parallel algorithm development, and implementations of parallel runtime environments. The structural UML diagrams of MASC provided the foundation classes and class aggregations. The behavioral UML diagrams of MASC provided the responsibilities and requirements of the classes in the model. The UML was instrumental in identifying duplicate responsibilities and classes among the various components of MASC; such as clarifying that an instruction stream and a cell have the same basic functionality, which is not illustrated in the MASC conceptual diagram.

The future of the object-oriented MASC model is currently being used as a development reference for implementing the MASC model using a cluster of workstations. However implementing MASC on other parallel computing hardware can use the same UML description as a reference. The UML MASC description can also be used as a common reference for comparing MASC with other models of parallel computation.

6 References

[1] Grady Booch, James Rumbaugh, and Ivar Jacobson, The Unified Modeling Language User Guide, Addison Wesley, Reading, MA, 1999.

[2] Bruce Powel Douglass, Real-Time UML: Developing Efficient Objects for Embedded Systems, Addison Wesley, Reading, MA, 1998.

[3] Todd Heywood and Claudia Leopold, "Models of Parallelism", Abstract Machine Models for Highly Parallel Computers, John R. Davy and Peter M. Dew, Eds., Oxford Science Publications, Oxford, England, 1995, pp. 1-16.

[4] Potter, Jerry L., Associative Computing: A Programming Paradigm for Massively Parallel Computers, Plennum Press, New York, NY, 1992.

[5] Potter, Jerry, Johnnie Baker, Stephen Scott, Arvind Bansal, Chokchai Leangsuksun, and Chandra Asthagiri, “ASC: An Associative Computing Paradigm”, IEEE Computer, Nov., 1994, pp. 19-25.