Synchronization and Packet Ordering in Packet Switching Interconnection Networks

Bryan Barney Reagan
University of Illinois At Chicago

4/26/00
Proposed Research Goals

• Develop techniques to implement shared common memory consistency models on top of a Multi-Computer using packet switching, Integrated Network Barriers, and independent wave fronts.

*Definition*. **Multi-Computer** – a parallel architecture in which explicit messages are sent between processors (Zargham, 1996).
Principle Background Areas

- Interconnection Networks & Packet Switching
- An A Priori Packet Switching Network
- Integrated Network Barriers
- Wave Fronts
- Shared Memory And Consistency Models
I. Interconnection Networks & Packet Switching
Basic Concepts

A parallel computer consists of \( p \) processors and \( m \) memory modules, connected by an interconnection network.
Packet Switching Networks

*Definition.* **Packet** – the smallest routable unit of information (data and/or control). A packet must contain an address field, and may contain other information (Culler, Singh, and Gupta, 1999).

*Definition.* **Packet Switching Interconnection Network** - interconnection network, which connects processors and memory modules, and which routes **packets** between these entities.
A Packet Switching Network Consists of the Following Parts

• **Sources (drains) of packets**, typically devices such as processors (processors and/or memory modules) which can send (receive) packets.
• **Switches** route packets from one link to another.
• **Links** are the connections between switches, sources, and drains of packets.

Some Typical Topologies used in Packet Switching include:

- Hypercube
- N-Dimensional Mesh
- Iliac
- Torus
- Omega

(Zargham, 1996).
A switch is composed of the following elements

• **Ports**, which are the endpoints of links, and contain:
  – One or more associated queues to hold packets
  – A Link level protocol, which determines how packets are transferred along the long the links.

• **Arbitration circuitry**, (Arbiter) which moves packets from input queues to output queues within the switches ports, according to a routing algorithm.
II. An A Priori Packet Switching Network
A Packet Switching Network with a 2-Dimensional Grid Topology
This is the layout of a very simple switch for a two dimensional mesh. Links are represented by arrows which show the flow of packets. By convention, the port to the processor is drawn on a diagonally trimmed corner.
Routing Algorithm

Definition. **Routing Algorithm** – an algorithm which selectively moves packets within the switch from input queues, into output queues (Tanenbaum, 1999).

The routing algorithm may also be viewed as selecting the path thought the interconnection network from the sender to receiver node (Tanenbaum, 1999).

In all implemented parallel processors, the routing algorithm is deadlock free.
Queue Dependency

*Definition.* **Queue Dependency** — A Queue A is said to depend upon queue B if packets can move from queue B to queue A (Pifare, Gravano, Fleperin and Sanz, 1994).

Queue Dependence depends upon the routing algorithm being used.
Queue Dependency Graph

**Definition.** **Queue Dependency Graph (QDG)** – a directed graph, \( G = (V, E) \). Each vertex \( v \in V \) corresponds to a queue in a switching system. Each edge, \( E \in (u, v) \mid u, v \in V \), indicates that a packet may route from the queue corresponding to \( u \) to the queue corresponding to \( v \). (Pifare, Gravano, Fleperin and Sanz, 1994).

**Lemma.** An acyclic Queue Dependency Graph corresponds to a deadlock free routing scheme (Pifare, Gravano, Fleperin and Sanz, 1994).

A Queue Dependency graph is more accurate than a Dally’s channel dependency graph (Dally, 1987).
Dimension Routing

Definition. Dimension Routing – a routing scheme for n-dimensional meshes, in which packets are routed minimally to the correct coordinate in each dimension in turn, in a fixed order (Tanenbaum, 1999).

Dimension routing results in an acyclic queue dependency graph (Tanenbaum, 1999).

(In our 2-D meshes, we route first in the Y dimension, and then on X.)
Virtual Cut Through Switching

*Definition.* Virtual Cut Through Switching - a packet switching technique in only the head of a packet may block (Tannenbaum, 1999).
III. Integrated Network Barriers
Barrier & Barrier Synchronization

Definition. **Barrier Synchronization**—a point in the instruction sequences of a set of simultaneously executing processors, which all processors must reach before any may proceed to execution of their subsequent instructions (Jordan, 1978).

Definition. **Barrier** A Barrier implies that all pre-barrier operations complete before barrier synchronization.

Barriers are often provided in software and hardware for multi-processors and multi-computers.
Integrated Network Barriers

*Definition.* **Integrated Network Barrier** – a barrier implementation in which the completion of pending memory operations is integrated with barrier synchronization (Solworth and Stamatopoulos, 1993), (Birk, Gibbons, Sanz, and Soroker, 1989).

Integrated Network Barriers have been used as a congestion control device for packet switching networks (Solworth and Stamatopoulos, 1993).
Integrated Network Barriers Operation

Assume an acyclic queue dependency graph.

- Processors inject a barrier marker into their network input queues.
- Barrier markers propagate to successor queues (in the queue dependency graph) when all predecessors of that queue have barrier markers at their heads.

V. Wave Fronts
Barrier Markers

Definition. **Barrier Marker** – a message which indicates the start of a new barrier (Birk, Gibbons, Sanz, and Soroker, 1989).

Barrier markers are carried in packets with or without other data. A barrier message incorporated into a data packet is said to be “**piggybacked**”.
Essential Properties of Pipelined Barriers

A Pipelined barrier must have the following properties to insure correctness:

- **Flushing Property** – A barrier message traverses a link only after all pre-barrier packets.
- **Restraining Property** – A barrier message traverses a link before all post-barrier packets.
- **Clear Route Property** – The path of a pre-barrier packet is always clear of post-barrier packets.

(Birk, Gibbons, Sanz, and Soroker, 1989).
Sample INB on a 4x4 Grid
T = 0

T=0: Processors insert barrier markers

- Unswept
- Barrier Marker
- Swept
T=0: Processors insert barrier markers
T= 2

T=0: Processors insert barrier markers

Unswept
Barrier Marker
Swept

4/26/00
T=3

T=0: Processors insert barrier markers

 better marker

 Unswept

 Barrier Marker

 Swept

 4/26/00
T= 4

T=0: Processors insert barrier markers

Unswept
Barrier Marker
Swept
T = 5

T=0: Processors insert barrier markers

T=5 Barrier messages begin moving in the X direction.
T= 6

T=0: Processors insert barrier markers

T=5 Barrier messages begin moving in the X direction.
T=7

T=0: Processors insert barrier markers

T=5 Barrier messages begin moving in the X direction.
T=0: Processors insert barrier markers

T=5 Barrier messages begin moving in the X direction.
T= 9

T=0: Processors insert barrier markers

T=5 Barrier messages begin moving in the X direction.

T = 9 Center 4 nodes have completed the barrier

Unswept Barrier Marker Swept
T=0: Processors insert barrier markers

T=5 Barrier messages begin moving in the X direction.

T = 9 Center 4 nodes have completed the barrier
T=0: Processors insert barrier markers.

T=5 Barrier messages begin moving in the X direction.

T = 9 Center 4 nodes have completed the barrier.

T=11 8 nodes complete the barrier.

Unswept
Barrier Marker
Swept
T = 12

T=0: Processors insert barrier markers

T=5 Barrier messages begin moving in the X direction.

T = 9 Center 4 nodes have completed the barrier

T=11 8 nodes complete the barrier

- Unswept
- Barrier Marker
- Swept
T= 13

T=0: Processors insert barrier markers

T=5 Barrier messages begin moving in the X direction.

T = 9 Center 4 nodes have completed the barrier

T=11 8 nodes complete the barrier

T = 13 The barrier has completed on the last four nodes.

Unswept

Barrier Marker

Swept
Problem

Problem: Barrier markers are delayed at points inside the network for rendezvous.

Desirable: To have wave-fronts delay only at destinations (if necessary) (McShane and Solworth, unpublished.)
Wave-front

*Definition.* **Barrier wave-front** - a separator, which partitions the queue dependency graph into three sub-graphs:

- **pre-front**, consisting of unswept queues
- **the front**, containing all of the barrier markers
- **post-front**, consisting of all queues which have not been swept by barrier markers.

*Definition.* **Barrier Wave** – a wave whose front consists of barrier messages.
Complete Wave

Definition. **Route covered by a wave** - A route is covered by a wave if all queues traversed by the route are also traversed by the wave.

Definition. **Complete set of waves** - A set of waves is complete if they cover all routes.

We shall use four waves, each starting at a corner of the mesh, and named based on the direction in which they route, (positive or negative) in each dimension in routing order.

- **Y-Positive-X-Positive** (covers routes in Y+X+)
- **Y-Positive-X-Negative** (covers routes in Y+X-)
- **Y-Negative-X-Positive** (covers routes in Y-X+)
- **Y-Negative-X-Negative** (covers routes in Y-X-)
Y-Positive X-Positive Wave On a 4 x 4 Mesh

- Each wave moves one diagonal per cycle (on an empty mesh.)
- Extends pipelining to the network across all processor nodes.
- Note this wave also covers routes in Y+ and X+ direction.
Disjoint Waves

Definition Disjoint Waves. A pair of waves which do not share any queues.

For a two-dimensional mesh, the pairs of disjoint waves are:

- Y-Positive-X-Positive wave and Y-Negative X-Negative wave.
- Y-Positive-X-Negative wave and Y-Negative X-Positive wave.
Multi-Ply Networks

*Definition.* Multi-Ply Networks— a network which is composed of more than one, logically identical networks, whose only shared queues are sources and drains.

*Lemma:* We can execute disjoint waves on the same ply.

On a 2-D Mesh, two ply are required for four waves:

- **Ply 1**: Y-Positive-X-Positive and Y-Negative-X-Negative
- **Ply 0**: Y-Positive-X-Negative and Y-Negative-X-Positive
Single Virtual Channel of a Wave
Channel INB
T=0

• T=0 All processors insert barrier markers
T=1

- T=0 All processors insert barrier markers
- T=1 Corner nodes inject barrier markers into wave channels
• T=0 All processors insert barrier markers

• T=1 Corner nodes inject barrier markers into wave channels
T=0 All processors insert barrier markers

T=1 Corner nodes inject barrier markers into wave channels
- T=0 All processors insert barrier markers
- T=1 Corner nodes inject barrier markers into wave channels

4/26/00
T=0 All processors insert barrier markers

T=1 Corner nodes inject barrier markers into wave channels
T=6

- T=0 All processors insert barrier markers
- T=1 Corner nodes inject barrier markers into wave channels
T=7

• T=0 All processors insert barrier markers
• T=1 Corner nodes inject barrier markers into wave channels
• T=7 Four nodes complete barrier
• T=0 All processors insert barrier markers
• T=1 Corner nodes inject barrier markers into wave channels
• T=7 Four nodes complete barrier
T=9

• T=0 All processors insert barrier markers
• T=1 Corner nodes inject barrier markers into wave channels
• T=7 Four nodes complete barrier
• T=9 Six Nodes completed this cycle

4/26/00
T=10

- T=0 All processors insert barrier markers
- T=1 Corner nodes inject barrier markers into wave channels
- T=7 Four nodes complete barrier
- T=9 Six Nodes completed this cycle
$T=11$

- $T=0$ All processors insert barrier markers
- $T=1$ Corner nodes inject barrier markers into wave channels
- $T=7$ Four nodes complete barrier
- $T=9$ Six Nodes completed this cycle
- $t=11$ Four Nodes completed this cycle

4/26/00

---

<table>
<thead>
<tr>
<th>Unswept</th>
<th>Y-X-</th>
<th>Y+X+</th>
<th>Both</th>
</tr>
</thead>
<tbody>
<tr>
<td>Barrier Marker</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Swept</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

58
T=12

- T=0 All processors insert barrier markers
- T=1 Corner nodes inject barrier markers into wave channels
- T=7 Four nodes complete barrier
- T=9 Six Nodes completed this cycle
- t=11 Four Nodes completed this cycle

4/26/00
T=13

• T=0 All processors insert barrier markers
• T=1 Corner nodes inject barrier markers into wave channels
• T=7 Four nodes complete barrier
• T=9 Six Nodes completed this cycle
• t=11 Four Nodes completed this cycle
• t=13 Last two nodes complete the barrier.
VI. Shared Memory & Consistency Models
Shared Memory

Each processor node shall be assumed to have local, physical memory.

A globally addressable memory can be implemented on top of separate local memories, using packets to request operations upon remote locations.
Memory Operation

Definition. Memory Operation – a procedure that performs the following steps:

- Perform zero or more Read() accesses
- Compute a Turing computable function based on the values read.
- Perform zero or more Write() accesses based on the values read, and the values computed from them.
Example Memory Operations

• Load a value from shared memory:
  
y ← Mem[&B]

• Store a value into shared memory
  
Mem[&C] ← Z

• Incrementing a counter stored at location &A. Let X denote a temporary, local variable, possibly a register.

  X ← Mem[&A]  // Read (&A)
  X ← X + 1
  Mem[&A] ← X  // Write(&A,X)
Memory Hazards

Definition Memory Hazards – potential memory value inconsistencies which arise when memory accesses from multiple concurrently executing processes are interleaved. These conflicts can be classified into three groups (Hennessy and Patterson, 1990).

• Write after Write Hazards (ex. \( X \leftarrow 1; X \leftarrow 2; \))
• Read after Write Hazards (ex. \( X \leftarrow 3; Y \leftarrow X; \))
• Write after Read Hazards (ex. \( Y \leftarrow X; X \leftarrow 4; \))
Memory Consistency Models

Definition. Memory Consistency Model – a formal specification of how the effects of a program’s memory operations appear to occur to the programmer (Adve and Hill, 1993).

Each Memory Consistency Model makes certain guarantees about which types of hazards are prevented, or how they are handled (Tanenbaum, 1999).
Four Common Memory Consistency Models

- Sequential Consistency (The strongest)
- Release Consistency
- Processor Consistency
- Weak Consistency

The weaker models are a tradeoff of semantics versus performance.
Sequential Consistency

Definition. **Sequential Consistency** – a multi-processor system is sequentially consistent if

- the result of any execution is the same as if the operations of the processors were executed in some sequential order,

- and the operations of each individual processor appear in this sequence in the order specified by its program. (Lamport, 1979).

Sequential consistency has been studied in great detail, and appears to be the de-facto standard at this time.
Sequential Consistency Example

<table>
<thead>
<tr>
<th>Processor 0</th>
<th>Processor 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\text{Op}_0^1$</td>
<td>$\text{Op}_1^1$</td>
</tr>
<tr>
<td>$\text{Op}_0^2$</td>
<td>$\text{Op}_1^2$</td>
</tr>
</tbody>
</table>

The effects of these operations are the same as any interleaving with:

- $\text{Op}_0^1$ before $\text{Op}_0^2$
- $\text{Op}_1^1$ before $\text{Op}_1^2$
Proposed Current and Future Work
Simple Technique for Sequential Consistency

• Each processor can issue one memory request per barrier instance. Hence on barrier completion, all requests have arrived at the location where they will be serviced.

• At the memory, each operation request can be serviced in order of arrival. In the case of a read request, the value read will be sent after the barrier epoch.

**Problem** - the use of one message per Integrated Network Barrier limits network bisection utilization to around 13% on a 16x16 network.
Phased Sequential Consistency Implementation

• A more sophisticated technique would use sequence numbers on the requests within epochs. Each memory request is marked with its originating processors number, and an internal sequence number for its processor within the barrier.

• Requests are held in a linked lists sorted by sequence number, until a barrier marker arrives.

• When barrier marker arrives, requests are performed in sequence order. (In the event of a tie for the same location, the processor number may be used to create an absolute order, possibly alternating the order to ensure fairness.)

• Again, responses to read requests are sent as the requests are processed, in the post-barrier epoch.

Note: For n = 1, 2, 4, 8, 16, and 32; n messages per barrier gives a bisection utilization of 13%, 25%, 44%, 73%, 91% and 95% respectively.
Other Issues to be examined

Other issues to be examined include:

1. Programmatic impacts (control & data dependencies)
2. Other consistency models (related semantics, increased concurrency)
3. Caching issues (INB’s to support higher performance cache accesses)
4. Use of less than complete wave sets

This work is primarily experimental.
The author has created a simulator for a 2 dimensional mesh interconnect network called Inetsim. Currently, it uses rifle and turn queues, Y-first dimension routing, virtual cut through switching, and Integrated Network Barriers.

The simulator has parameters for network size, message and barrier frequencies, number of plies and queue size.

The source code is around 170 Kilobytes of C++ code.
Proposed Current and Future Work

• Design a switch structure which supports wave-front routing using two virtual channels and a Wave-Front based Integrated Network Barriers.
• Create Protocols to ensure a Sequential Consistency Memory Model using the features of the above described network.
• Simulate these new techniques versus existing ones, and measure relative performance.