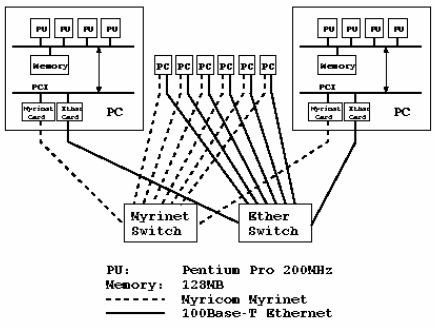
Cluster of Multiprocessor Systems

A Pentium Pro PC-Based SMP Cluster

Presenters:
Matias Dons. Dollerup
Mikkel Stensgaard

Introduction

- Each node is a SMP containing 4 Pentium Pro Processors.
- 8 nodes connected
- Comminication
  - Ethernet using MPICH
  - Myrinet using NICAM
- Solaris OS
- Single user, single job



Agenda

- Inter node communication
- NICAM – remote memory implementation
- Programming SMP clusters
- Case studies
- Conclusion

Internode Communication (1)

- Memory bandwidth
  - ~100-400 MB/s
- Ethernet bandwidth
  - ~ 4 MB/s

This is a mayor bottle neck!

- Myrinet bandwith
  - ~ 100 MB/s

Internode Communication (2)

- Message passing
  - Handling of incomming messages
  - Mutual exclusion on buffers
  - Message copying limits bandwidth
  - Main CPU are involved
  - Explicit coordination between nodes.

Internode Communication (3)

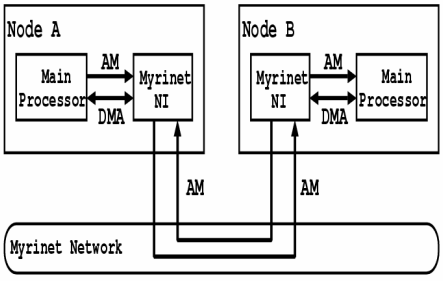
- Remote memory
  - Low overhead
  - High bandwidth
- NICAM
  - Network Interface Communication using Active Messages.

NICAM – Overview

- Utilizes the CPU on the Myrinet NI.
  - Operations performed directly on NI
  - No overhead from main CPU besides startup
  - Is build upon Active Messages
- Requires synchronization primitives
  - Because of asynchronous access from nodes
- Synchronization performed between NI's
  - No involvement from main CPU

NICAM – Active messages

- Uses AM both internal and internode to simplify design.
- Maps local address to physical directly on NI
- Uses DMA to access memory



NICAM – Synchronization

- Uses flags to indicate completeness
- Exploits cache coherence mechanism.
  - Waiting CPU's does not generate traffic

NICAM

- NICAM uses the NI exclusively.

Only possible when a single job is running

NICAM – Primitives

- Copy
  - Copy from one node to another node
  - Possible to start multible parallel copy operations.

  nicam_bcopy(src_node, src_addr, dst_node, dst_addr,size)
  nicam_sync()

  nicam_bcopy_notify(.....,flag_addr)

- Barriers
  - Multible barrier possible at same time.
  - Implemented using a multi-stage barrier.

  nicam_barrier(flag_addr)

Software development for COMPaS

Programming SMP Clusters (1)

Shared memory architecture

- Thread synchronization and mutual exclusion is needed
- Communication overhead is low
- Performance is limited by system bus bandwidth
- Possibility for full cache utilization

Programming SMP Clusters (2)

Distributed memory architecture

- Implicit synchronization when exchanging messages
- Communication overhead is high
- Performance limited by network bandwidth

Suggested programming model

- All message passing programming
  - Not ideal for SMP systems
- All shared memory programming
  - Inter-node communication costs are not controllable
- Hybrid memory programming
  - Avoids passing messages in intra-node communication
  - Explicitly specify inter-node communication

Hybrid memory program structure

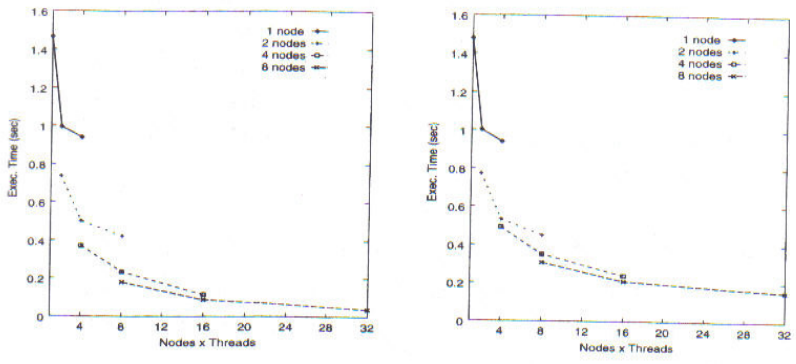
1. Partition and distribute data to each node
2. Each node partitions it's own data and distribute it to its threads
3. Threads synchronize when finished
4. Nodes synchronize and exchanges data
5. Step 3-4 is repeated until the task is complete

Case studies: Laplace equation (1)

- Iterative Laplace equation solver
  - Solves the Laplace equation on a 640x640 domain
- Distribution: matrix rows are equaly divided among nodes
- Sharing: each nodes subdivides rows to local threads

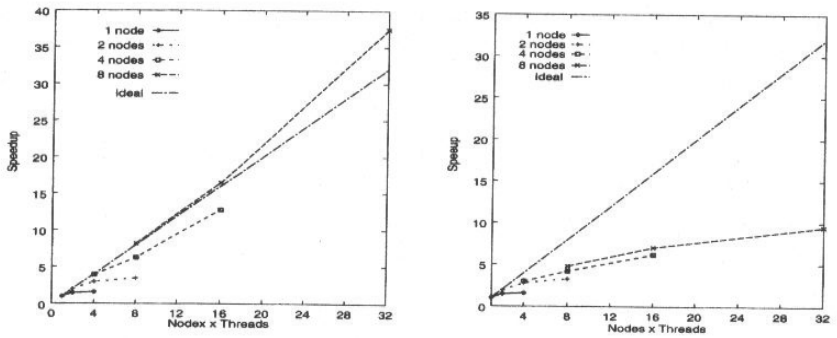
Case studies: Laplace equation (2)

- Performance: Myrinet vs. Ethernet



Case studies: Laplace equation (3)

- Performance: Myrinet vs. Ethernet

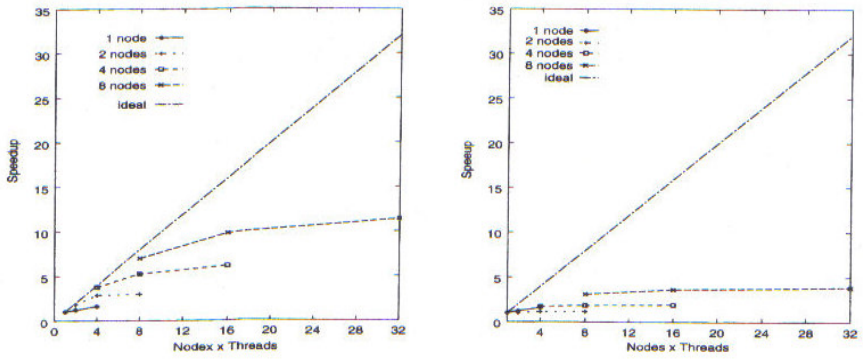


Case studies: Radix sort (1)

- The radix sort algorithm
  - Idealy sorts elements in linear time
- Distribution: elements are equaly divided among nodes in the cluster.
- Sharing: each nodes subdivides elements equaly among local threads

Case studies: Radix sort (2)

- Performance: Myrinet vs. Ethernet



Overall hybrid programming guidelines

- The system bus bandwidth limits each threads data throughput
- Aim for cache-aware applications

Conclusion

- Optimal architecture utilization via the NICAM inferface
- The one-job architecture limits useability
- Test setup is outdated, thus we can expect much more performance using state of the art computers and networks
- The hybrid programming model requires specific applications

The End

- Questions?