Distributed Computing with MPI

Sean Lawlor

Computer Networks Group
Department of Electrical and Computer Engineering
McGill University Bellairs Institute, Barbados

March 14, 2012
Outline

- Introduction
  - Motivation
  - MPI-1 and MPI-2 statuses
- Machine Learning with MPI
- Deadlock and Knot detection/removal
- Determining Computation End
- Conclusion
Motivation

- Why would we want to parallelize applications?
- What benefit would pushing to a cluster give?
Motivation

Why would we want to parallelize applications?
What benefit would pushing to a cluster give?
What options exist for this kind of work?
Motivation

- Why would we want to parallelize applications?
- What benefit would pushing to a cluster give?
- What options exist for this kind of work?
- How do we determine computation end?
Motivation

- Why would we want to parallelize applications?
- What benefit would pushing to a cluster give?
- What options exist for this kind of work?
- How do we determine computation end?
Figure: Example of computer cluster [Source: Boise State CS Dept, Beowulf Cluster Lab (http://cs.boisestate.edu/~amit/research/beowulf/)]
MPI is a message passing interface library standard

- Is a specification, not an implementation
- Library, not a language
- Based on the classical message passing programming model
- Barriers
- Multi-Party messages
  - 1:n → broadcast
  - n:1 → reduce (Hadoop)
  - n:n → all-to-all (1-step consensus)
MPI was defined in 1994 \(^1\) by a broadly-based group of parallel computer vendors, computer scientists, and application developers.

Implementations and support grew quickly

Basis for cluster environments with free and open implementations (OpenMPI, MPICH2, ...)

\(^1\)I. Foster, *Designing and Building Parallel Programs*. Addison-Wesley, 1994, online Ch. 8 - Message Passing Interface
Same as MPI-1 with extended functionality
- Extends message passing model
  - Parallel I/O
  - Remote memory operations (not covered here)
  - Dynamic process management
- Adds bindings to include C++ and Fortran-90
- Interaction with Threads
Most parallel systems implement some implementation of MPI-2

Cluster MPIs, such as MPICH2 and LAM, support most of MPI-2 including dynamic process management

We’ll refer mostly to the OpenMPI implementation
Problem Types solved by MPI
- Large Dataset Processing
- Decreased Computation Time (if possible)
- Redundancy for Distributed Systems

Example System Definition
- Deepthought: (Beowulf Cluster)
- 15 Nodes
- Each node has a dual-core Pentium IV processor
- 1 GB of RAM
- 1 (non-included node) as coordinator
What’s the difference?

Can asynchronous send *and* receive be done?
Asynchronous vs. Synchronous

- What’s the difference?
- Can asynchronous send *and* receives be done?
- What’s the best option?
Asynchronous vs. Synchronous

- What’s the difference?
- Can asynchronous send *and* receive be done?
- What’s the best option?
MPI Uses for machine learning applications

- Used to handle massive datasets
- Many programs are parallelizable
MPI Uses for machine learning applications

- Used to handle massive datasets
- Many programs are parallelizable
- Tailored applications for problems
MPI Uses for machine learning applications

- Used to handle massive datasets
- Many programs are parallelizable
- Tailored applications for problems

Example:

MNIST Dataset
15 Node plot of error vs 2 Node calculation
Synchronous
Machine Learning

MPI Uses for machine learning applications
- Used to handle massive datasets
- Many programs are parallelizable
- Tailored applications for problems
- Example:
  - MNIST Dataset
MPI Uses for machine learning applications

- Used to handle massive datasets
- Many programs are parallelizable
- Tailored applications for problems
- Example:
  - MNIST Dataset
  - 15 Node plot of error vs 2 Node calculation
MPI Uses for machine learning applications

- Used to handle massive datasets
- Many programs are parallelizable
- Tailored applications for problems

Example:
- MNIST Dataset
- 15 Node plot of error vs 2 Node calculation
- Synchronous
MPI Uses for machine learning applications

- Used to handle massive datasets
- Many programs are parallelizable
- Tailored applications for problems
- Example:
  - MNIST Dataset
  - 15 Node plot of error vs 2 Node calculation
  - Synchronous
Figure: Results of parallel Logistic Regression (x-axis denotes time and y-axis is error on test set)
Deadlock Detection

- Many works into this (Example\(^2\))
- Determine if directed cyclic graph exists in network

Figure: Example of cyclic graph in network

A deadlock has been detected, now what?

Need to determine which node should give up computation
A deadlock has been detected, now what?
Need to determine which node should give up computation
Problem Specific (Eg. Will you destroy everything or just some things by killing a specific node?)
A deadlock has been detected, now what?
Need to determine which node should give up computation
Problem Specific (Eg. Will you destroy everything or just some things by killing a specific node?)
Knots are a special case of deadlocks\(^3\)

- Cause larger problems
- Killing any single cycle may not kill all cycles
- What should we do here?

Figure: Example of network graph with knot

Figure: Graph with multiple (> 2) distinct cycles, forming multiple knots
Again problem specific

However there are many standard ways depending on the problem
Again problem specific

However there are many standard ways depending on the problem

Variations:
Determining Computation End

- Again problem specific
- However there are many standard ways depending on the problem
- Variations:
  - Timelimit reached
Again problem specific

However there are many standard ways depending on the problem

Variations:
  - Timelimit reached
  - Goal achieved
Again problem specific

However there are many standard ways depending on the problem

Variations:
- Timelimit reached
- Goal achieved
- No more messages to send
Again problem specific

However there are many standard ways depending on the problem

Variations:
- Timelimit reached
- Goal achieved
- No more messages to send

Focus on no more messages situation
Determining Computation End

- Again problem specific
- However there are many standard ways depending on the problem
- Variations:
  - Timelimit reached
  - Goal achieved
  - No more messages to send
- Focus on no more messages situation
- Upon end, global solution must be attained
Again problem specific

However there are many standard ways depending on the problem

Variations:
- Timelimit reached
- Goal achieved
- No more messages to send

Focus on no more messages situation

Upon end, global solution must be attained
Naive Computation End Algorithm

Setup:

- Ring network
- Single marker used at a time\(^4\)
- White determines node is idle
- Black means node is currently responding to a message (i.e. computing)
- Marker colors node white when it leaves that node if idle
- If marker sees a white node, assumed white since last iteration

Example network graph

Figure: Example ring network graph
Algorithm:

- **INIT**: $\forall$ nodes $(n) \in N :$ color black
- $n_i$ sends initial marker $m$ along outgoing edge
- Node $n_j$, where $n_j$ is the child of $n_i$ forwards message with following properties:
  - 0, if $n_j$ is black
  - $m + 1$, if $n_j$ is white
  - Marker paints node white
  - If node receives message, paints itself black
- Termination is achieved when $m = N$
- Collect data at gate, and return
Algorithm by example
Algorithm by example
Algorithm by example
Algorithm by example

[Diagram of a cycle with nodes and arrows, node labeled 0]
Algorithm by example
Algorithm by example
Algorithm by example
Algorithm by example
Algorithm by example
Algorithm by example
Algorithm by example
Algorithm by example
Algorithm by example
Algorithm by example
Algorithm by example

[Diagram of a circular graph with nodes and directed edges]

5
Algorithm by example
Algorithm by example
Algorithm by example
Conclusion

- Definition of MPI 1 and 2
- How MPI can be leveraged for distributed computation
- Determining cyclic graphs and avoiding them
- Determining computation end on the fly
Thank you!
Questions?