Lecture 14: MPI Introduction

CS178: Programming Parallel and Distributed Systems

March 19, 2001
Steven P. Reiss

I. Overview

A. Last time we talked about parallel architecture

1. Basic concepts

a) Lots of processors

b) Control organization :: MIMD

c) Communications :: network

(1) Store & forward communication

d) Memory

e) Input/Output

2. There is no standard parallel machine

a) Lots of different ones

b) Each with different control, communications, ...

B. We also talked some about parallel software

1. Programming models

a) Shared memory, shared objects

(1) These are clean models that approach what we are used to

(2) Implemented via message passing

(3) Require synchronization

(4) Not used because of efficiency reasons

b) Explicit message passing

(1) This is the model that is currently used

(2) This is what we want to explore

II. Message Passing

A. Basic idea

1. Lots of processes that do the work

2. These processes send messages

a) Messages can be sent to a specific processor

b) Messages can be sent to all processors (or a set)

c) Broadcasts can do computations (min, max, sum, ...)

B. But this is different than distributed case

1. Sends can be either synchronous or asynchronous

2. But receives are synchronous

a) Processors have to explicit wait for a message

b) It doesn't just arrive and spawn a thread (RMI)

C. Synchronization is done via messages

1. No shared data

a) No need for mutexes or monitors, etc. to protect it

2. Process synchronization via messages

a) Process B waits for a message, then it can run;

b) Process A waits for messages from all processes it is joining

3. This is also a clean programming model (but different)

4. This constrains our messages

D. Message Guarantees

1. Messages are delivered

a) We want to assume a single system

b) We don't want the overhead of checking all over, etc.

c) Single system -- processes don't fail that often

2. Message are delivered in relative order

a) Message passing diagrams

b) What you don't want to happen

(1) Note that using different routings can cause this

(2) Internet allows this

c) What can happen

E. Message types

1. Buffered versus unbuffered

a) Buffered messages allow continuation as soon as its copied

b) Buffers can exist at each point

2. Synchronous vs asynchronous

a) Synch means that the sender continues only after the receiver reads the message

b) Requires acknowledgements

3. Must the receiver be ready

a) Does there have to be a receive pending for a send to succeed

F. Programming Support

1. Need support for sending & receiving messages

a) Providing appropriate guarantees

b) Supporting point-to-point and broadcast messages

c) Supporting different message types

2. Need support for processes, communications, ...

a) Want to think of all processes as a single system

b) Don't want to worry about communications

3. Need support for network topologies

a) Actual topology of the network

b) Doing embeddings automatically

4. Ideally this should be at the language level

a) CSP -- Hoare's toy language

b) Some languages built this way, but nothing popular or common

5. Currently done a the library level

a) Various standard libraries

b) PVM and MPI are the more common

c) We will study MPI

III. MPI Basics

A. Processes

1. **Want to start up a number of processes simultaneously**
   a) The system should take care of this
2. **MPI assumes all processes are the same**
   a) Same binary
   b) MPI assigns a number to each process
      (1) This is the process rank
      (2) Ranges from 0 (master) to n-1 (children)
3. **Communications is done via Send and Receive calls**
   a) Different send calls handle different message types
   b) Receive is independent of send type

B. Messages

1. **Sent as <address,count,datatype>**
2. **Lots of built-in datatypes**
3. **User datatypes can be defined too**

IV. MPI Sample Program

A. The Problem: approximating PI

1. $\int_0^1 \frac{4}{1+x^2}dx = 4\operatorname{atan}(1) - 4\operatorname{atan}(0) = 4\operatorname{atan}(1) = \pi$
2. **Compute the integral from 0 to 1 of 4/1+x^2**
3. **Do this by dividing the area into intervals and approximating each interval**
4. **Sample drawing (0,4) - (1,2); approx with rectangles**

B. MPI Program

1. **See handout -- go over line by line**