Message passing concept

Joachim Hein
LUNARC & Centre of Mathematical Sciences
Lund University

1

Overview

- Introduction
- Distributed data parallel programming
- Message passing between the processes

2

Why parallel programming

- Improve computational speed
  - Get faster time to solution
  - Larger, hopefully more realistic problem
- Divide problem into subtasks
  - Assign subtasks to different CPUs → parallel computing
- Parallel computing is presently the **only** game in town to get more performance
  - CPUs do not get faster (stuck around 3 GHz for decades)

3

Parallelism at various levels

- Inside CPU (multiple functional units, vector FPU, ...)
- Multicore processors
- Stream processors (GPGPU, FPGA, ...)
- Multiple processors in a node (SMP, CC-NUMA, ...)
- Multiple nodes (largest systems available)

4

Message Passing

- Most applications on multi-node systems use one or more of
  - C, C++, Fortran, Python
  - Message Passing Interface (MPI) for communication
- Applications using MPI also very suitable for
  - Multicore systems
  - Multiprocessor systems (SMP, CC-NUMA)
  - Often outperform threaded applications (Posix, OpenMP)
- MPI typically harder to develop
  - Might need to develop from scratch

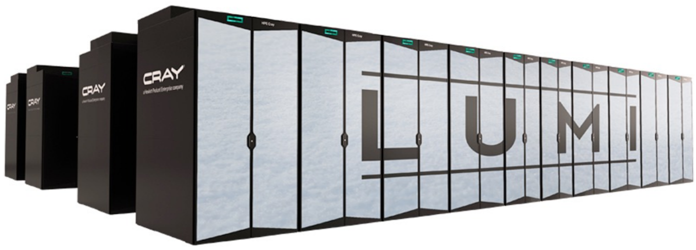
5

Images of typical hardware

6

LUMI (Large Unified Modern Infrastructure)
Peak: >550 PFlop/s = 550,600,000,000,000,000 Flop/s

- 10 European countries (incl. Sweden)
- Located in Kajaani/Fi
- GPU partition
  - 4 AMD MI250x GPU/node
  - 1 host CPU/node
  - Q1 2023
- CPU partition
  - 1536 nodes
  - 2 CPUs/node (AMD EPYC “Milan”)
  - 128 cores/node



7

Dardel
Peak expected ≈ 13.5 Pflop/s

- Next generation SNIC machine
- Located at KTH/Stockholm
- CPU partition
  - 554 nodes
  - 2 CPUs/node (AMD EPYC “Rome”)
  - 128 cores/node
- GPU partition
  - 54 nodes
  - 4 AMD Instinct™ next gen. GPU/node
  - 1 AMD EPYC™ next gen. CPU/node
  - Under installation

8

Tetralith @ NSC

- 1892 compute nodes
  - 2 Xeon Gold 6130 Proc
  - 32 cores per nodes
  - Over 60000 cores total
- Intel Omni-Path network
  - 100 Gbps
  - Fat-tree topology

Backside of the cabinets to show the cables

9

Kebnekaise @ HPC2n

- 602 nodes
  - Heterogenous system
- System includes
  - 20 Nvidia V100
  - 160 Nvidia K80
  - 36 Intel KNL
  - 20 large Mem-nodes (3TB)

10

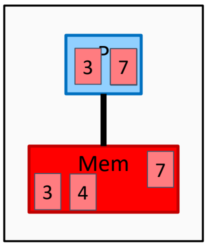
This course

- Introduce the concept: message passing programming
- Give you a good foundation to the vast functionality of MPI
- Teach you to write you own programs
- We do not cover, e.g.:
  - Single-sided communication
  - New features in MPI 3.x
  - MPI-IO

13

Serial programming

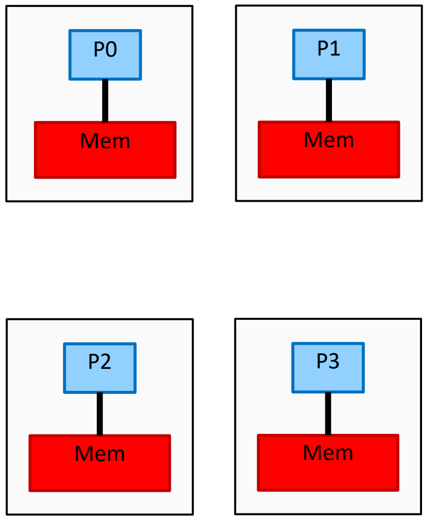
- We want to do a long calculation
- Pencil & Paper to slow ☹
- Take a computer
  - Write a program in C, Fortran, ...
  - Run the program
- What happens when the program runs?
  - Processor reads data from memory
  - Processes these data
  - Writes result back to memory
  - Write final result to disk
- What can we do if that is still too slow ???



14

Parallel programming

- Take more than one computer
- Partition the problem into sub-problems
- Write a program for each sub-problem
- Place one program on each computer
- Should be faster now!
- But often the sub-problems are **not** independent !!!



15

Example

- Calculate large sum:

$$\sum_{n=1}^{40000} a_n$$

- Split into four sums, place each on a computer:

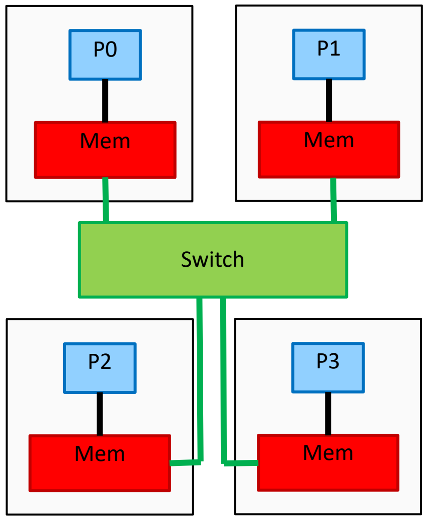
$$\sum_{n=1}^{10000} a_n \quad \sum_{n=10001}^{20000} a_n \quad \sum_{n=20001}^{30000} a_n \quad \sum_{n=30001}^{40000} a_n$$

- In the end: four partial results on different computers ?!?

16

Message passing

- Build an interconnect network
- The programs now pass (data-)messages between them
  - Send data into the network
  - Receive data from the network
- In the previous example:
  - Each processor sends its partial result to P0
  - P0 receives from P1, P2, P3
  - Adds these to get final result

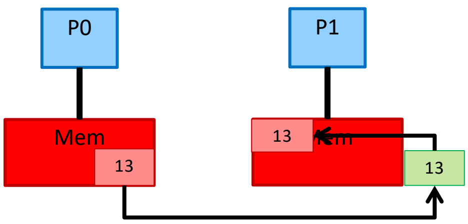


17

Double sided point-to-point communication

- Most basic form of communication in message passing:

Program on P0:	Program on P1:
`a = 13` `send(a,1)`	`recv(b,0)` `c = c + b`



18

Summary

- Concepts of distributed memory parallel computing
  - Several programs running simultaneously
  - Each has its own memory
  - Communication via message passing

19