Building Blocks: Software

Operating Systems, Processes, Threads

Partners

Funding

Horizon 2020
European Union Funding
for Research & Innovation

Reusing this material

Outline

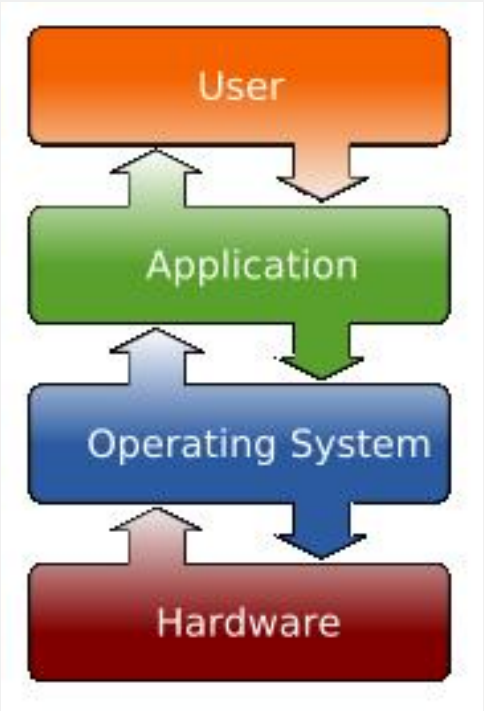
- Operating Systems
  - What does an OS do?
  - Operating Systems for HPC
  - The HPC OS Environment
- Processes
- Threads
  - Threads on accelerators

Operating Systems

What do they do? Which ones are used for HPC?

Operating System (OS)

- Underlying software that steers a computer's operation
- Provides a platform for applications to run
- Orchestrates access to the hardware by applications:
  - Manages use of system resources (processor, memory, disk, network connections, etc.)
  - Provides user interface
  - Abstracts hardware details from user and programmer



Operating System (OS)

- Allows multiple programs to run concurrently on hardware
  - Including internal OS (“system”) processes and user programs
- Some operating systems allow multiple users to access the system and run applications at the same time
- Running applications are controlled through the concepts of *processes* and *threads.*
  - an application / program is a single process…
  - …which may have multiple threads

Operating Systems for HPC

- HPC systems used to always used Unix
  - vendors (DEC, SUN, Cray, IBM, SGI, …) all wrote their own version
- Now dominated by Linux (of various flavours)
  - Most HPC vendors modify a commercial Linux distro (RedHat or SUSe) and tailor to their own system.
  - Many commodity clusters run a free Linux distro (CentOS is particularly popular).
  - Every single HPC machine in the November 2017 Top500 uses Linux
- Only IBM Power systems offer vendor Unix (AIX) (or Linux)
- Windows really not used for HPC

HPC compute node OS

- On the largest supercomputers the compute nodes often run an optimised OS to improve performance
  - Interactive (front-end) nodes usually run a full OS

- How is the OS optimised?
  - Remove features that are not needed (e.g. USB support)
  - Reduce OS processes running in the background ("OS noise") that could degrade performance or cause it to vary unpredictably
  - Restrict scheduling flexibility and increase interrupt period
  - Bind processes and threads to specific cores
  - Remove support for virtual memory (paging)
  - …

HPC OS Environment

- Usually access is via remote login
  - Secure Shell (SSH)
  - File transfer over SSH (SCP/SFTP) or network mount point
  - Graphics display becomes an issue
- “X windows systems” often used
  - Allows windows to be displayed remotely
  - Graphics over SSH
  - Has performance issues
- Always multi-user
  - Users distinguished by username, authorised by password
  - Users also assigned a group id

HPC OS Environment

- Each user provided a “home” directory (enter on login)
  - e.g. `/home/username`
  - Can create / destroy anything within that directory
  - Barring any quotas/limits on disk space used
- OS protects system from user, and users from each other
  - File permissions (read, write, execute depending on file ownership)
  - Can often access other user / shared directories within the same group
- Users have limited permissions to change things
  - Only system administrators are “super users” with additional privileges
  - “sudo apt-get install my_preferred_software” will not work
  - “sudo anything” is usually against security policy (and will fail)

HPC OS Environment

- Software provided varies from system to system
- Most HPC services make centrally-installed software available to users through environment modules
  - Typically for scientific applications that are in high demand
  - Try “module available”, “module list”, module --help
- Linux package managers (apt-get etc.) won’t work
- You may have to build software yourself (or ask for the service to install it for you)

The Command Line

- Interaction almost always through Linux command line
  - e.g. which two files or folders are taking up the most space?

```
user@hpcsystem> du -sm * | sort -n | tail -2
```

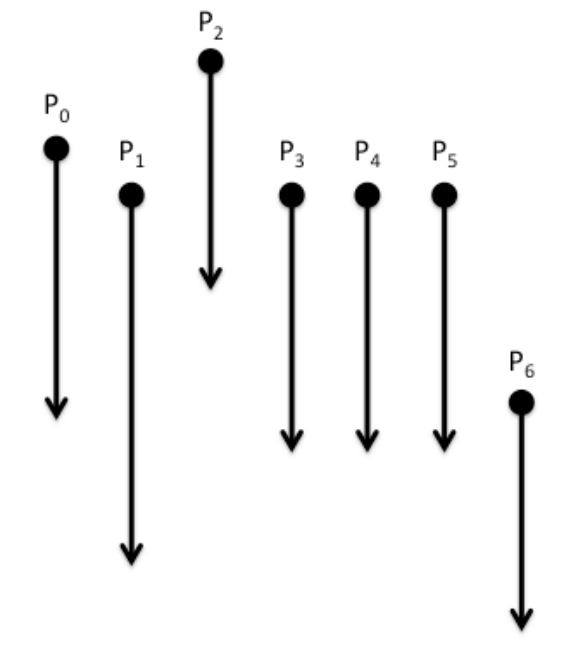
  - often a reasonably large barrier to new people adopting HPC.
- For any serious use of HPC you will have to learn to use the command line
  - often also useful for using command line on your own laptop/PC
- Basic Linux commands should always work
  - Depends a bit on shell (bash, tcsh, ksh, zsh)
  - Bash is default on most modern HPC systems
- Should also learn basic operation of in-terminal text editor
  - vi/vim is generally available
  - emacs is another popular choice

Processes

To each their own….

Processes

- Each application is a separate *process* in the OS
  - a process has its own memory space which is not accessible by other running process.
  - processes are ring-fenced from each other: if web browser crashes, it can't scribble over document stored in the memory your word processor
- Each process is scheduled to run by the OS



OS and multicore processors

- "*Multicore parallelism – manually specified by the user*"
  - what's the use of a multicore laptop if I run non-parallel code?
- OS's have *always* scheduled multiple processes (even on single-core processors)
  - regularly check which process is running
  - give another process a chance to run for a while
  - rapid process switching gives *illusion* applications run concurrently even on a single core
- With a multicore processor
  - multiple processes can *really* run at the same time

Process Scheduling

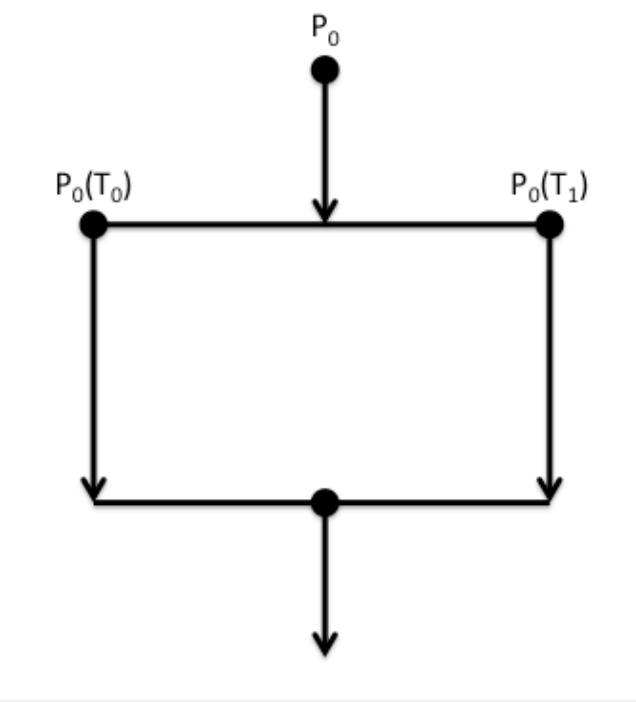
- The OS has responsibility for interrupting a process and granting the core to another process
  - Which process is selected is determined by the *scheduling policy*
  - Interrupt happens at regular intervals (every 0.01 seconds is typical)
  - Process selected should have processing work to do
- On a quad core processor, OS schedules 4 processes at once
- Some hardware supports multiple processes per core
  - Known as *Symmetric Multi-threading* (SMT)
  - Usually appears to the OS as an additional core to use for scheduling
- Process scheduling can be a hindrance to performance
  - in HPC, typically want a single user process per core

Threads

Sharing memory

Threads

- For many applications each process has a single *thread*…
  - … but a single process can contain multiple threads
  - each thread is like a child process contained *within* parent process



Threads (cont.)

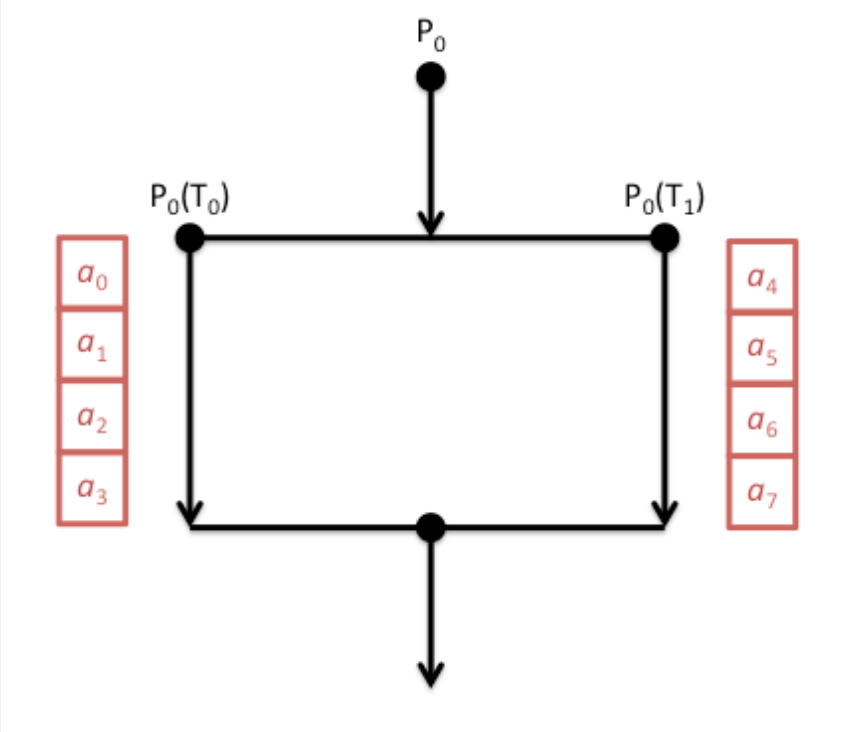
- All threads in a process have access to the same memory
  - the memory of the parent process
- Threads are a useful programming model pre-dating multicore
  - e.g. a computer game (a process) creates asynchronous threads
    - one thread controls the spaceship
    - another controls the missile
    - another deals with keyboard input
    - …
  - but all threads update the same game memory, e.g. the screen
- OS scheduling policy is aware of threads
  - ensures all of the game operations progress
  - switching between threads usually quicker than between processes

Threads and multicore processors

- With multiple cores
  - multiple threads can operate at the same time on the same data to speed up applications
- Cannot scale beyond the number of cores managed by the operating system
  - to share memory, threads must belong to the same parent process
- In HPC terms cannot scale beyond a single *node*
  - using multiple nodes requires multiple processes
  - this requires inter-process communication – see later

Shared-memory concepts

- Process has an array of size eight
  - each thread operates on half the data; potential for 2x speedup



Threads and Accelerators

- Accelerators (GPU / GPGPU cards etc.) provide a lot of computational power for low energy consumption
- Often need a huge number of threads for efficient usage
  - Encouraged to create many more threads than cores (oversubscription)
  - Accelerator hardware supports fast switching of execution of threads
    - switch off a thread when it is waiting for data to arrive from memory
    - switch on a thread that is ready to do computation
    - tries to hide memory latency (delay)
  - GPGPUs can have 1000's of cores, so it can be difficult to implement oversubscription when programming a scientific software application
- Threading is becoming more and more important on modern HPC machines for both performance and power efficiency