Building Blocks

Operating Systems, Processes, Threads

Dr Mark Bull, EPCC
markb@epcc.ed.ac.uk

Outline

- What does an Operating System (OS) do?
    - OS types in HPC
    - The Command Line
- Processes
- Threads
    - Threads on accelerators
- OS performance optimisation
    - Why is the OS bad for performance?
    - Approaches to improving OS performance

Operating Systems

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

Operating System (OS)

- The OS is responsible for orchestrating access to the hardware by applications.
  - Which cores is an application running on?
  - How is the memory allocated and de-allocated?
  - How is the file-system accessed?
  - Who has authority to access which resources?
  - How do we deal with oversubscription (e.g. more applications running than cores available).
- Running applications are controlled through the concepts of *processes* and *threads.*

OS's for HPC

- HPC sector is 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 (Scientific Linux is particularly popular).
- Only IBM Power systems still use UNIX (AIX)
  - 16 HPC systems in the June 2013 Top500 list use UNIX
- Windows HPC is used on a small number of HPC systems
  - 3 HPC systems in the June 2013 Top500 list use Windows

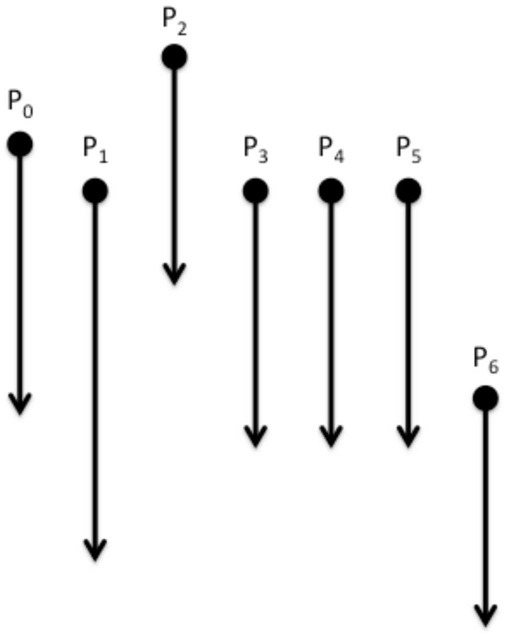
The Command Line

- HPC sector is dominated by Linux
- Interaction is almost always through the Linux command line.
  - 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.
  - Knowledge is often useful for using the command line on your own laptop/PC
  - You should also learn the basic operation of an in-terminal text editing program – “vi” is probably the simplest to learn and is available everywhere.

Processes

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.
  - Each process is scheduled to run by the OS – it can be tied to a particular core or can be migrated between cores



Process Scheduling

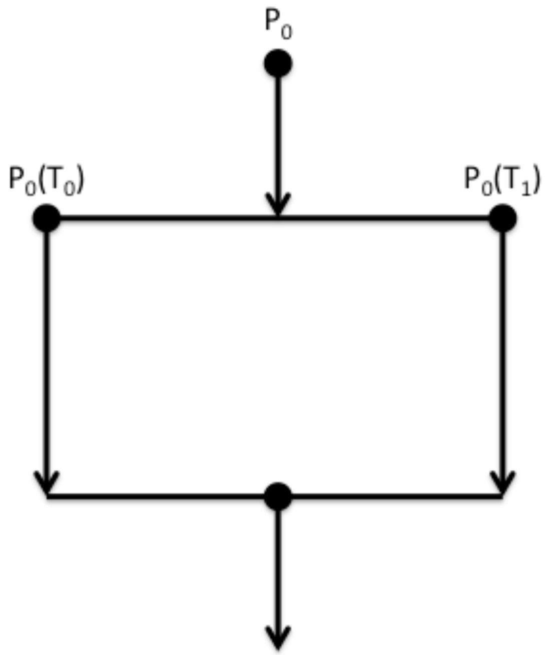
- The OS has responsibility for interrupting a process and granting the core to another process
  - Which process is granted access is determined by the *scheduling policy*
  - Interrupt happens at regular intervals (every 0.01seconds is typical)
  - Process selected should have processing work to do
- Hardware can support scheduling of multiple processes
  - 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

Threads

Sharing memory

Threads

- For many applications each process has a single *thread*…
- …but with the advent of multicore processors it is becoming more common for a process to contain multiple threads



Threads (cont.)

- All the threads in a process have access to the same memory
  - Can operate in parallel on the same data to speed up applications
  - Can have threads operating asynchronously (often used in GUIs)
- OS scheduling policy is aware of threads
  - Usually scheduled as one thread per core but not a requirement
  - Switching between threads is usually a bit quicker than switching between processes

Threads and Accelerators

- The Accelerator programming model generally requires a huge number of threads to provide efficient usage
  - Oversubscription of the accelerator by threads is encouraged
  - Hardware supports fast switching of execution of threads
  - As GPGPUs can have 1000's of computing elements, oversubscription can be difficult!
- Threading is becoming more and more important on modern HPC machines

OS Optimisation

How do vendors get performance?

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
  - Often means that you are *cross-compiling*
- How is the OS optimised?
  - Remove features that are not needed (e.g. USB support)
  - Restrict scheduling flexibility and increase interrupt period
  - Remove support for virtual memory (paging)

Summary