Computer Systems B

COMS20012

Introduction to Operating Systems and Security

bristol.ac.uk

University of BRISTOL

Scheduling across multiple cores

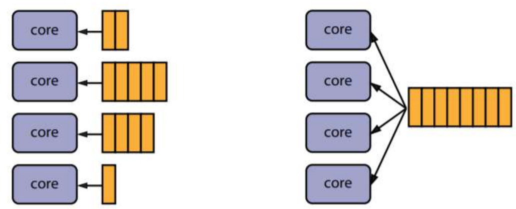
bristol.ac.uk

Two possible approaches

- Per-core queues
- Shared queue

Which scales better?

Which offers better performance?



Scalability

- Contention and scalability
  - Access to the shared queue is a critical section, **mutual exclusion** is needed
  - As the number of cores grow, **contention** to access the resource become a problem
- Per-core design scale to many core

bristol.ac.uk

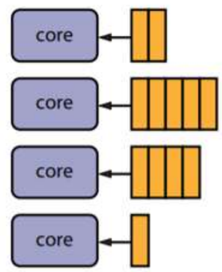
Performance

- As a thread run, the data it accesses is loaded in the **CPU cache(s)**
- If thread changes CPU, the data needs to be loaded again
- **Affinity** between a thread and one core grows because the cached data
- **Per-core approach benefit** from this affinity by keeping threads on the same core
- This is not the case for shared queue design

bristol.ac.uk

Load balancing

- In per-core design, queues may have different length
- This create **load imbalance** across cores
  - Cores may be idle why others are busy
  - Threads on lightly loaded core have more CPU time than threads on heavy loaded ones
- Non-issue in shared queue design
- Per-core designs need a mechanism to balance the load across all cores
  - Migrate threads from lightly loaded core to heavy loaded ones



bristol.ac.uk

University of
BRISTOL

- Pause the video
- Open *kern/thread/thread.c*
- Resume the video

bristol.ac.uk

kern/thread/thread.c

- **thread_consider_migration** defined line 881
- Called from hardclock function (*kern/thread/clock.c* line 93)
  - Watch Week 7 Video 2 again if confused
- Line 889-901
  - Calculate average load per core (**one_share**)
  - Load on current core (**my_count**)
- Line 902-904
  - If bellow average load nothing to do
- Line 906-913
  - Build a list of thread to migrate (**victims**)
  - Remove them from the thread running on the current core **(&curcpu->c_runqueue**)

bristol.ac.uk

kern/thread/thread.c

- Line 915-966
  - Find a core bellow average load (line 921)
  - Migrate victim threads to that core
- Line 973-982
  - Internal management
  - Put back thread that could not be assigned
  - Clean up data structure

bristol.ac.uk

University of BRISTOL

Thank you

bristol.ac.uk