Design Issues in Multiprocessor, Multicore, and Real-time Scheduling on Linux Machines

Hannah Cebulla, Bridget Wermers, John Hultman, Logan Shy

Multiprocessor Scheduling Design Issues

- Goal of Multiprocessor System
- Granularity
- Types of Multiprocessor Systems
    - Loosely Coupled
    - I/O processor
    - Tightly Coupled

Multicore Scheduling Design Issues

- Definition of Multicore system
- Scheduling
  - Locality
  - Caches
- Cooperative resource sharing
- Resource Contention

Real-time Scheduling Algorithms

- Hard vs Soft Deadlines
- Periodic vs Aperiodic
- Characteristics
    - Determinism
    - Responsiveness
    - User Control
    - Reliability
    - Fail Soft Operations
- Approaches
    - Static table-driven
    - Static priority-driven
    - Dynamic Planning-based
        - Dynamic Best Effort

Linux Kernel Scheduling Design

- Similar to UNIX
- Important scheduling changes
  - 1.2: Round Robin
  - 2.4: Priorities, Queues, and Multicore
  - 2.6: CFS
- CFS implementation
  - Dynamic time quantum
  - Target Latency
  - RB Tree
- Synchronization primitives
  - Atomic
  - Spinlock (r/w)
  - Mutex

Algorithm Comparison

- Stochastic Search Scheduling Method based on the Genetic Algorithm
- Dynamic Critical-Path Scheduling
- Earliest Deadline First (EDF)
- Rate Monotonic (RM)
- Declustering
- Priority Inversion
- Contention Aware Scheduling

Conclusions

- Scheduling is hard (NP-hard!)
- Key differences between multicore and multiprocessor systems
- Schedule design must vary by nature of system (laptop vs space shuttle)
- Many attempts at optimizing under varying circumstances
- Linux CFS and it’s many applications