Avoiding Scheduler Subversion using Scheduler-Cooperative Locks

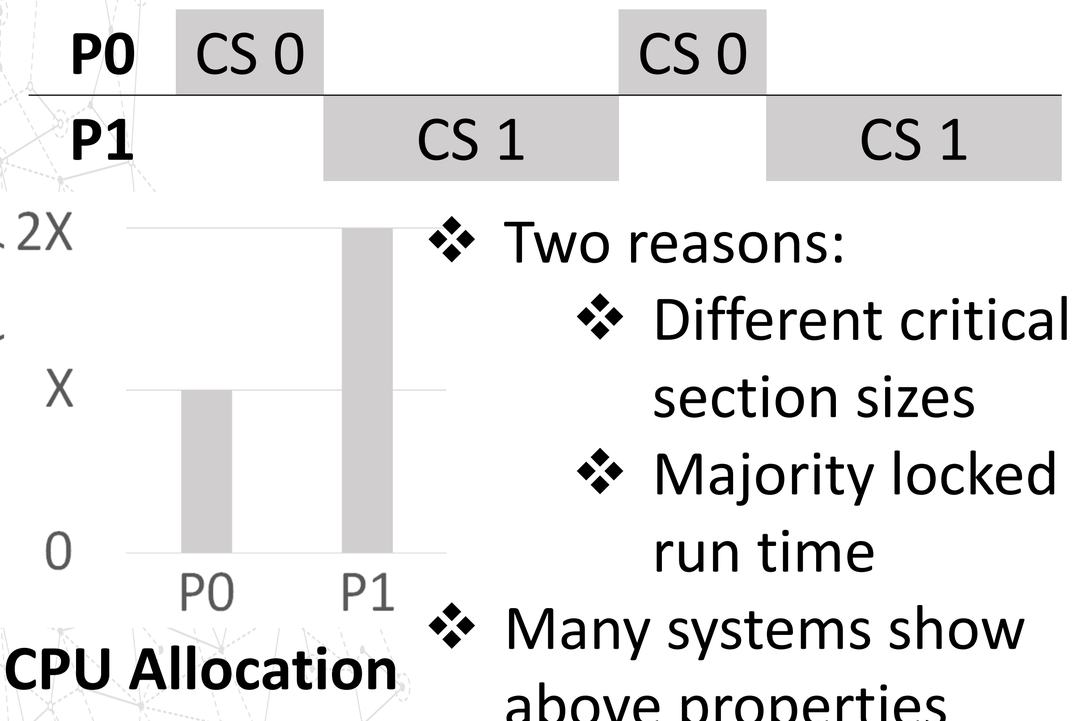
Yuvraj Patel, Leon Yang, Leo Arulraj,
Andrea C. Arpaci-Dusseau, Remzi H. Arpaci-Dusseau, Michael Swift
Computer Sciences Department, University of Wisconsin-Madison

The Problem

Scheduler Subversion

Locks determine which process is scheduled

- Example: 2 processes P0 & P1 accessing a ticket lock, default priority, P1 holds lock for twice as long as P0



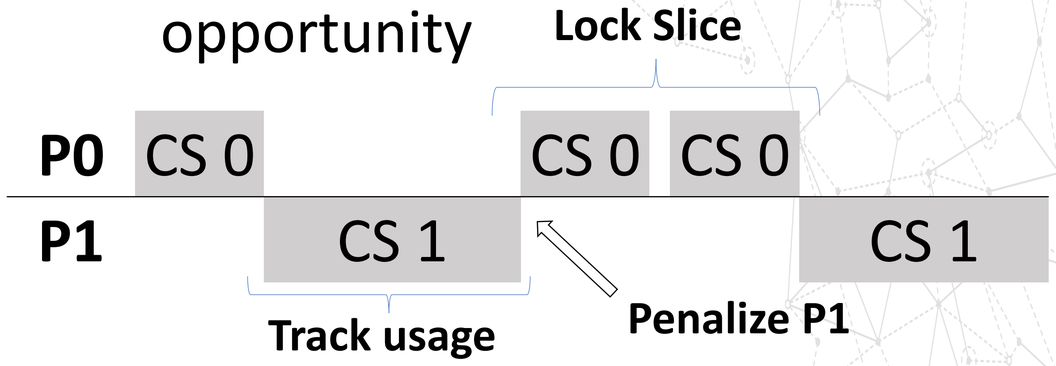
- Two reasons:
  - Different critical section sizes
  - Majority locked run time
- Many systems show above properties

The Solution

Scheduler-Cooperative Locks

Align lock usage with CPU scheduling goals

- Important design components
  - Track lock usage of all users
  - Penalize dominant users
  - Lock slice – dedicated window of opportunity

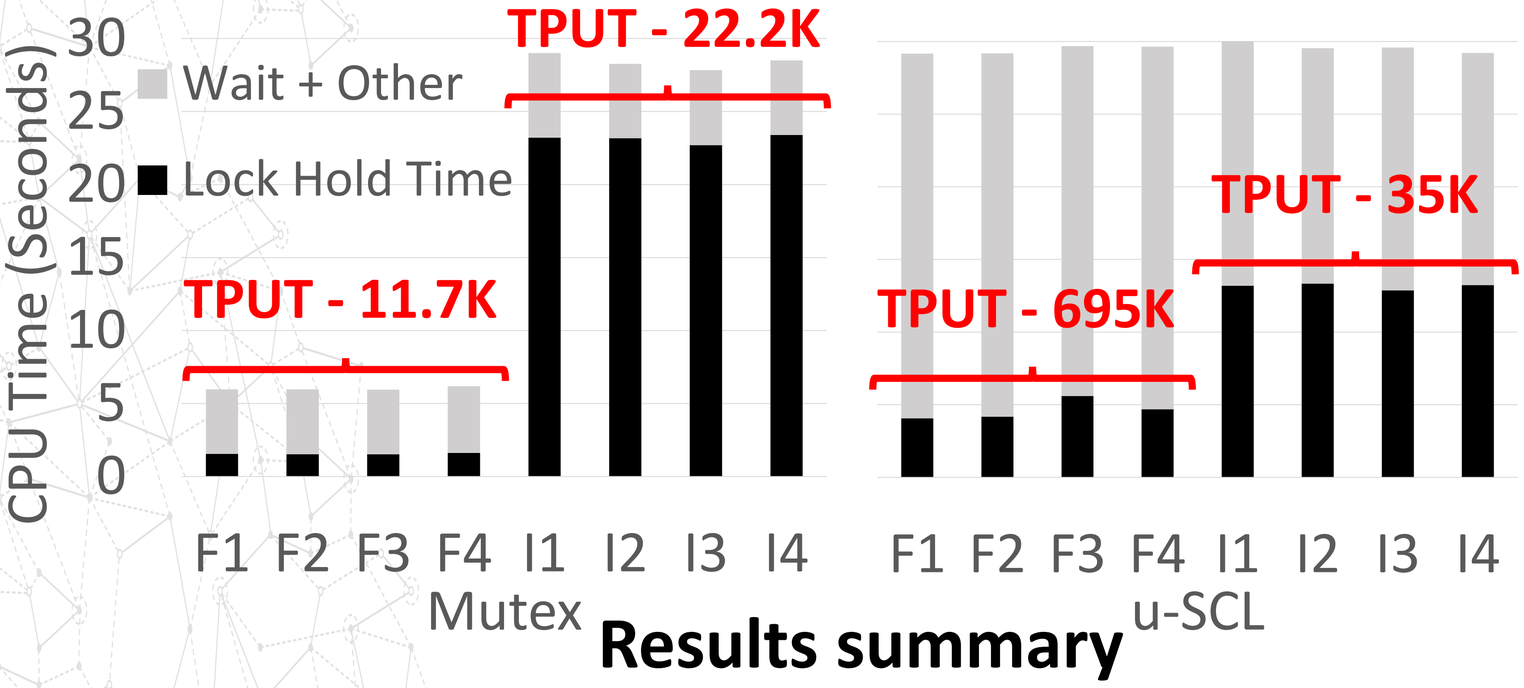


- Implement 3 different types of SCL
  - 2 user space – u-SCL, RW-SCL
  - 1 kernel – k-SCL

Evaluation

Example result – UpScaleDB + Linux CFS

4 CPU, 4 threads – insert, 4 threads – find, default priority



Results summary

- Allocate CPU proportionally in extreme cases
- Efficient and scale well at large scale
- Handles interactive and batching threads
- Demonstrate real-world utility

Conclusion

- Locks usage determines CPU allocation subverting scheduling goals
- Introduce Scheduler-Cooperative Locks that aligns with CPU scheduling goals
- Evaluation shows the performance capabilities and versatility of SCLs
- SCL can support any type of schedulable entity - thread/process/container
- Source code - http://tiny.cc/o3ocnz