Operating System

lina liu Apr. 13

1. how ls cmd process executed?
2. what is process?
3. what is virtual memory?
4. what the difference between memory cache and virtual memory?
5. what is CPU protect mode?
6. difference between BIOS and UEFI under CPU protect mode?

3 pieces operating system

Virtualization

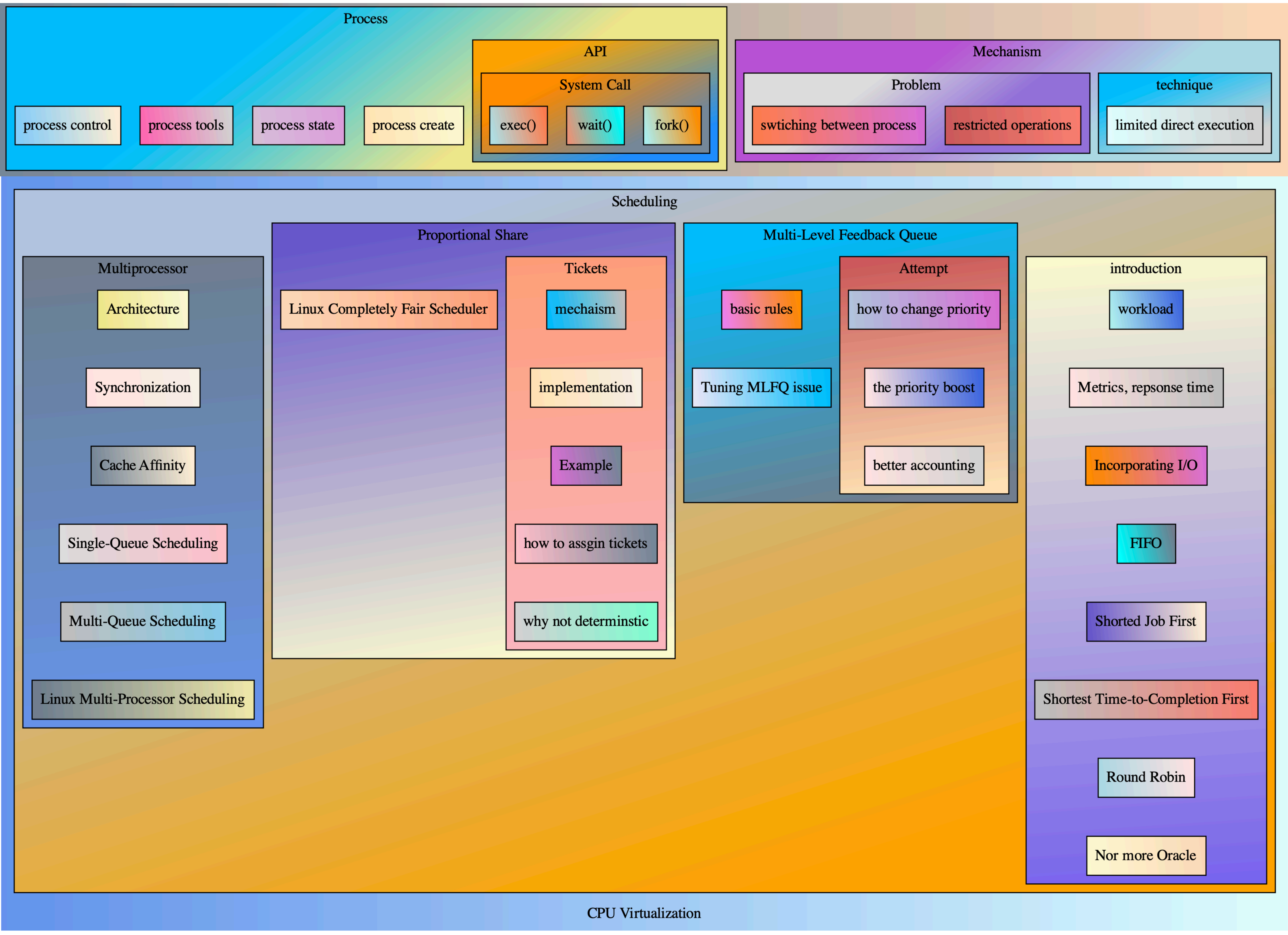
- The Abstraction: The process
- Interlude: Process API
- Mechanism: Limited Direct Execution
- Scheduling: Introduction
- Scheduling: The multi-level feedback queue
- Scheduling: Introduction
- Mechanism: Limited Direct Execution
- Scheduling: Proportional Share
- Multiprocessor Scheduling
- CPU Virtualization
- Memory Virtualization
- The abstraction: Address Space
- Interlude: Memory API
- Mechanism: Address Translation
- Segmentaion
- Free-Space Management
- Paging: Introduction
- Paging: Faster Translation(TLBs)/Smaller tables
- Beyond Physical Memory: Mechanisms/Policies
- Complete Virtual Memory Systems
- Memory Virtualization Summary

Concurrency

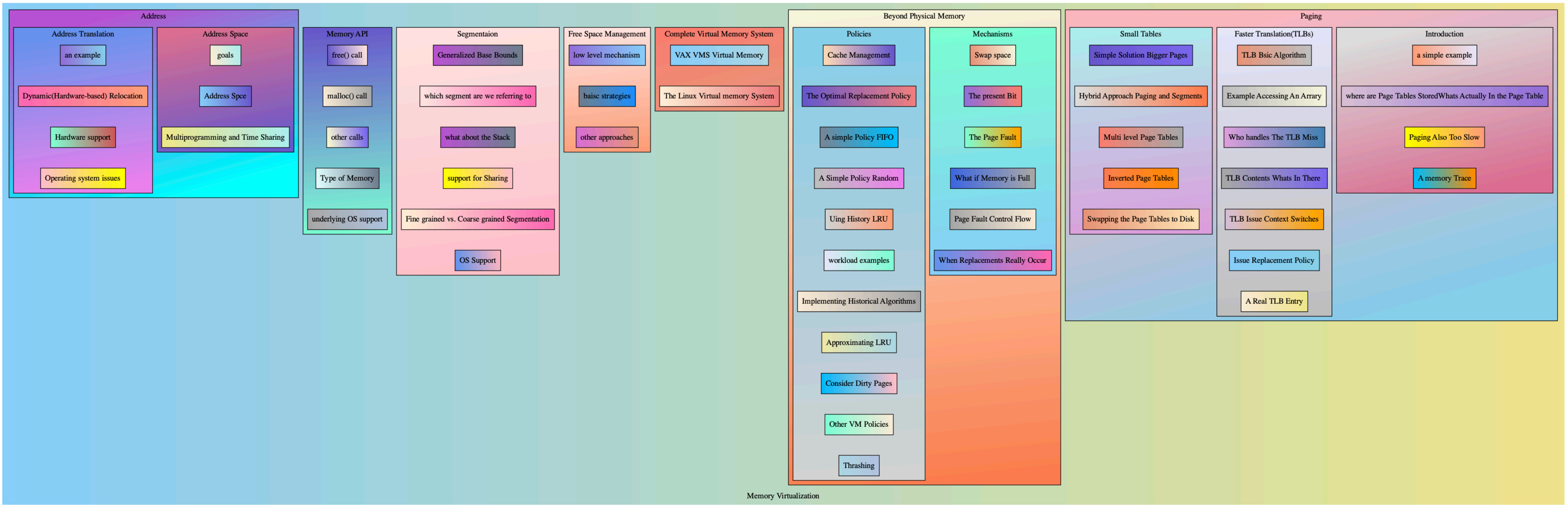
- Concurrency: An Introduction
- Interlude: Thread API
- Locks
- Lock-based concurrent data structure
- Condition Variables
- Semaphores
- Common Concurrency Problems
- Event-based concurrency(Advanced)
- Summary of Concurrency

Persistence

- A Dialogue on Persistence
- I/O Devices
- Hard Disk Drives
- Redundant Arrays of Inexpensive Disks(RAIDs)
- Interlude: Files and Directories
- File System Implementation
- Locality and the fast file system
- Crash Consistency: FSCK and Journaling
- Log-structured file systems
- Flash-based SSDs
- Data Integrity and Protection
- Summary Dialogue on Persistence
- A Dialogue on Distribution
- Distributed Systems
- Sun’s Network File System(NFS)
- The Andrew File System(AFS)
- Summary Dialogue on Distribution


Process
process control
process tools
process state
process create
API
System Call
exec()
wait()
fork()
Mechanism
Problem
swtiching between process
restricted operations
technique
limited direct execution
Scheduling
Multiprocessor
Architecture
Synchronization
Cache Affinity
Single-Queue Scheduling
Multi-Queue Scheduling
Linux Multi-Processor Scheduling
Proportional Share
Linux Completely Fair Scheduler
Tickets
mechaism
implementation
Example
how to assgin tickets
why not determinstic
Multi-Level Feedback Queue
basic rules
Tuning MLFQ issue
Attempt
how to change priority
the priority boost
better accounting
introduction
workload
Metrics, repsonse time
Incorporating I/O
FIFO
Shorted Job First
Shortest Time-to-Completion First
Round Robin
Nor more Oracle
CPU Virtualization

Memory Virtualization



Concurrency

Introduction

1. Why use Threads?
2. How to create a thread?
3. Why it get worse, because of shared data?
4. The heart of the problem: uncontrolled scheduling
5. One more problem: waiting for another
6. The wish for atomicity
7. why in OS class?

Interlude: Thread API

1. Thread Creation
2. Thread Completion
3. Locks
4. Condition Variables
5. Compiling and Running

Locks

1. The basic idea
2. Pthread Locks
3. Building A Lock
4. Evaluating Locks
5. Controlling Interrupts
6. A Failed Attempt: Just Using Loads Stores
7. Building working spin locks with Test-and-set
8. Evaluating Spin Locks
9. Compare and swap
10. Load-Linked and Store-Conditional
11. Fetch-And-Add
12. Too much spinning, what now?
13. Using Queues: Sleeping instead of Spinning
14. Different OS, different support
15. Two-Phase Locks

Lock-based Concurrent Data Structures

1. Concurrent Counters
2. Concurrent Linked Lists
3. Concurrent Queues
4. Concurrent Hash Table

Condition Variables

1. what is condition variable?
2. The routines of condition variable
3. The Producer/Consummer(Bounded Buffer) Problem4. Convering Conditions

Semaphores

1. What is semaphore?
2. Binary Semaphores(Locks)
3. Semaphores for ordering
4. The Producer/Consumer(Bound Buffer) Problem
5. Reader-Writer Locks
6. The Dining Philosophers
7. How to implement Semaphores?

Common Concurrency Problems

1. What types of bugs exist?
2. Non-Deadlock bugs
3. Deadlock Bugs

Event-based Concurrency(Advanced)

1. The basic idea: an event loop
2. An Important API: select()/poll()
3. using select()
4. Why Simpler? No locks needed
5. A problem: Blocking System Calls
6. A Solution: Asynchronous I/O
7. Another Problem: State management
8. What is still difficult with events

Concurrency