Final

Introduction And System Structures

Operating system is a set of software for managing computer resources.

- Exception
  - Interrupt
    - I/O
    - Clock
  - Exception
    - Fault
    - Trap (syscall)

I/O:

- (Programmed I/O)
- Interrupt I/O: Synchronous or asynchronous.
- DMA
- (Channel)
- Batch system: Multiprogramming or not.
- Time-sharing (multitasking) system.

Response time: Process (user) count n, time slice q, Response time $s = n \times q$.

System call: ⋯

Microkernel v.s. monolithic kernel: ⋯

Caveats:

- **Linux** is a multi-user time-sharing system.
- The purpose of system calls is to allow user-level processes to **request services** of the operating system
- The main disadvantage of the batch system is lack of **interaction** instead of concurrency (can use multiprogramming).
- 为了在通用操作系统管理下的计算机上运行一个程序，需要经历几个步骤。但是，不一定需要**向操作系统预定运行时间**，而非用控制台监控程序执行过程。
- **Issuing a trap instruction** or **switching from user mode to kernel mode** need not to be privileged.

Processes And Threads

Process state: Running, ready, waiting. No transition directly from waiting to running.

fork(): ⋯

Kernel / user threads: Many-to-one, Many user thread to single kernel thread.

Caveats:

- **Heap memory** and *global variables* are shared across threads, while register values and stack memory are not.
- A message-passing system is a kind of *inter-process* communication.
- A running process may be switched to release CPU, when one of following events occurs EXCEPT **the process calls a subroutine**.
- When a process is waken up, it means that **its state is changed into ready state**.

CPU Scheduling

Non-preemptive: Not switched away until finish or blocked.

Preemptive: Can be switched away while running.

- Throughput: Number of processes completed per time unit.
- Turnaround time: Time between arrival and finish. (waiting and execution)
- Waiting time: Time waiting in ready queue.
- Response time: Time between a request is submitted and the first response is produced.
- FCFS (First Come, First Served)
- SJF/SPF (Shortest Job / Process First): Non-preemptive. Minimal average waiting time (optimal).
- SRTF (Shortest Remaining Time First): Preemptive SJF.
- HRRN (Highest Response Ratio Next): R = (waiting time + requested execution time) / requested execution time.
- Priority: Lower integer, higher priority. Solve starvation with priority aging.
- RR (Round Robin)
- Multilevel Feedback: Foreground and background.

Gantt chart: Pn in cells, time below cell divider.

Caveats:

- In Round Robin scheduler on threads, if a user thread finished before using up its time slice, the next user thread in the same

process will run within the remaining slice (by user thread implementation); if a kernel thread finished before using up its time slice, a new slice starts.

- 降低进程优先级的合理时机是**进程的时间片用完**，而非进程从就绪态转为运行态。

Process Synchronization

Critical section:

- Mutual exclusion
- Progress (空闲让进)
- Bounded waiting

Semaphore:

- P() / wait() / down(): wait until S> 0 and consume 1.
- V / signal() / up(): ++S.
- **S > 0: S number of resources available.**
- **S = 0: No resource available.**
- **S < 0: |S| number of process waiting.**

Mutex: binary semaphore, (normally) initialized to 1.

Sequence of wait()s matters while signal()s does not, synchronization wait should be before mutex wait. Implementing synchronization: Use a semaphore initialized to 0 for every pair of synchronization. (the sequence of concurrent start up does not matter.)

Bounded buffer (producer / consumer) problem:

- full = 0, empty = n, mutex = 1.
- Producer: wait empty, wait mutex, signal mutex, signal full.

- Consumer: wait full, wait mutex, signal mutex, signal empty.

Caveats:

- 有两个进程 P1、P2，它们分别执行下面的程序体，其中 total 是两个进程都能访问的共享变量，初值为 0（可理解为共享存储段中的存储单元），count 是每个进程的私有变量。假设这两个进程并发执行，并可自由交叉（interleave），则这两个进程都执行完后，变量 total 可能得到的最小取值是 **3**。 `P1:{ int count; for (count = 1; count <= 50; ++count) total = total + 1; } P2: { int count; for (count = 1; count <= 50; ++count) { total = total + 2; } }`
- Suppose 9 producers and 6 consumers share a buffer with size of 8. In order to use the buffer properly, the semaphore mutex of critical section of the buffer is initialized to **1**.
- Suppose 5 processes share mutual exclusive sections. If 3 processes are permitted to enter the mutual exclusive sections at the same time, then the semaphore of mutual exclusion sections should be initialized to **3**.
- **Special machine instruction** as a critical section problem solution results in busy-waiting.

Deadlocks

Necessary conditions for deadlock:

- Mutual exclusion.
- Hold and wait.
- No preemption.
- Circular wait.

Resource allocation graph: ⋯

Graph contains a cycle: Deadlock for one instance per resource type, possible of deadlock for multiple instances per resource type.

Methods for handling:

- Prevention: No circular wait by requiring each process request resources in a total ordering.
- Avoidance: Safe state is the state with a safe sequence. Unsafe state means possibility of deadlock. For single instance per type, use resource allocation graph to avoid cycle; for multiple instances per type, use Banker’s algorithm.
- Detection: For single instance per type, use wait-for graph where nodes are processes and search for cycle; for multiple instance per type, use detection algorithm.
- Recovery: ⋯
- Ignore.

Main Memory And Virtual Memory

Multiple-partition allocation:

- Fixed partitioning.
- Dynamic partitioning.

Dynamic storage allocation algorithm:

- First-fit.
- Best-fit: Smallest fitting hole.
- Worst-fit: Largest hole.
- Next-fit: Starting search from last position.

Frame for physical memory, page for logical memory, with the same size.

Logical address: Page number, page offset.

TLB look up time ε, memory access time t, hit ratio α, Effective Access Time EAT = α(ε + t) + (1 - α)(ε + t + t).

二级页表：页表大小最大为页大小，故页表最少条目为页大小除以页表项大小，由此可计算页目录表最少条目数。

Segmentation: ⋯

- Internal fragmentation:
  - Fixed-size partition.
  - Paging.
  - Segmentation with paging.
- External fragmentation:
  - Variable-size partition.
  - Segmentation.

When page fault is possible, memory access time t, average page fault service time s, page fault rate p, EAT = p t + (1 - p) s.

Page replacement algorithms:

- FIFO (First In, First Out): Belady’s anomaly.
- Optimal: Will not used for longest time.
- LRU (Least Recently Used)
- Second chance (clock): Clockwise with a reference bit.
- Counting based:
  - LFU (Least Frequently Used)
  - MFU (Most Frequently Used)

Frame allocation:

- Fixed allocation:
  - Equal allocation.
  - Proportional allocation.
- Priority allocation.

Frame replacement:

- Global replacement
- Local replacement

Strategy:

- Fixed allocation, local replacement.
- Priority allocation, global replacement.
- Priority allocation, local replacement.

Trashing: Busy swapping.

Working set: the set of pages in the most recent Δ references.

Caveats:

- **Multiple-partition** memory allocation scheme may produce external fragmentation.
- Implementing LRU precisely in an OS is expensive, so practical implementations often use an approximation called **NRU**.
- **Segmentation memory management** may have no internal fragmentation.- The fundamental basis for virtual memory management is **locality**.
- To fetch a data from main memory in a demand paging system requires **2** accesses to the physical memory.
- With demand paging, **hash tables** have worst system performance.

- 总体上说，请求分页 (demand-paging) 是个很好的虚拟内存管理策略。但是，有些程序设计技术并不适合于这种环境。例如，**二分法搜索**。
- 考虑页面置换算法，系统有 m 个页帧供调度，初始时全空；引用串长度为 p，包含了 n 个不同的页面，无论用什么缺页算法，缺页次数不会少于 **n**。
- 首次适应算法的空闲区是**按地址递增顺序连在一起**。