

操作系统原理

Operating System Principle

田丽华

§7-3 死锁预防

Methods for Handling Deadlocks

处理死锁的方法

➤ 忽略、预防、避免、检测、解除

- Ignore the problem and pretend that deadlocks never occur in the system; used by most operating systems, including UNIX. (忽略这个问题, 假装系统中从未出现过死锁。这个方法被大部分的操作系统采用, 包括UNIX) 鸵鸟策略
- Ensure that the system will never enter a deadlock state.
(确保系统永远不会进入死锁状态) 预防死锁, 避免死锁
- Allow the system to enter a deadlock state and then recover.
(允许系统进入死锁状态, 然后恢复系统) 死锁检测

鸵鸟策略

- 像鸵鸟一样对死锁视而不见
- 处理死锁的代价很大，而且常常给用户带来许多不便的限制。
- 大多数用户宁可在极偶然的情况下发生死锁，也不愿限制每个用户只能创建一个进程、只能打开一个文件等等。
- 于是不得不在方便性和正确性之间作出折衷。

Deadlock Prevention

(死锁的预防)

Restrain the ways request can be made.
(抑制死锁发生的必要条件)

Mutual Exclusion – not required for sharable resources; must hold for nonsharable resources.

(互斥: 共享资源不是必须的, 必须保持非共享资源)

Deadlock Prevention

(死锁的预防)

Hold and Wait – must guarantee that whenever a process requests a resource, it does not hold any other resources. (占有并等待：必须保证进程申请资源的时候没有占有其他资源)

01

Require process to request and be allocated all its resources before it begins execution, (要求进程在执行前一次申请全部的资源)

02




or allow process to request resources only when the process has none.没有资源时，可以申请资源。在申请更多其它资源之前，需要释放已有资源

03

Low resource utilization; starvation possible. (利用率低，可能出现饥饿)

死锁的预防 (续)

No Preemption – (非抢占)

-  If a process that is holding some resources requests another resource that cannot be immediately allocated to it, then all resources currently being held are released. (如果一个进程的申请没有实现, 它要释放所有占有的资源)
-  Preempted resources are added to the list of resources for which the process is waiting. (抢占的资源放入进程等待资源列表中)
-  Process will be restarted only when it can regain its old resources, as well as the new ones that it is requesting. (只有进程能够重新得到旧的资源和新申请的资源时, 才可以重新开始)

死锁的预防 (续)

Circular Wait – impose a total ordering of all resource types, and require that each process requests resources in an increasing order of enumeration.
(循环等待: 将所有的资源类型放入资源列表中, 并且要求进程按照资源表申请资源)

- 🚀 所有进程对资源的请求必须严格按资源序号递增的次序提出。
- 🚀 总有一个进程占据了较高序号的资源, 它继续请求的资源必然是空闲的, 可以一直向前推进。
- 🚀 在资源分配图中不可能出现环路, 因而摒弃了“环路等待”条件
- 🚀 这种策略可以提高资源利用率, 但在进程使用各类资源的顺序与系统规定的顺序不同时会造成资源浪费的情况。

死锁的预防 (续)

- 上述预防死锁的方法通过限制资源请求来打破死锁的四个必要条件之一，从而预防死锁的发生。
- 其可能的副作用：
 - 降低设备利用率和吞吐量
 - 可能有进程饥饿