操作系统

Operating system

孔维强

大连理工大学

内容纲要

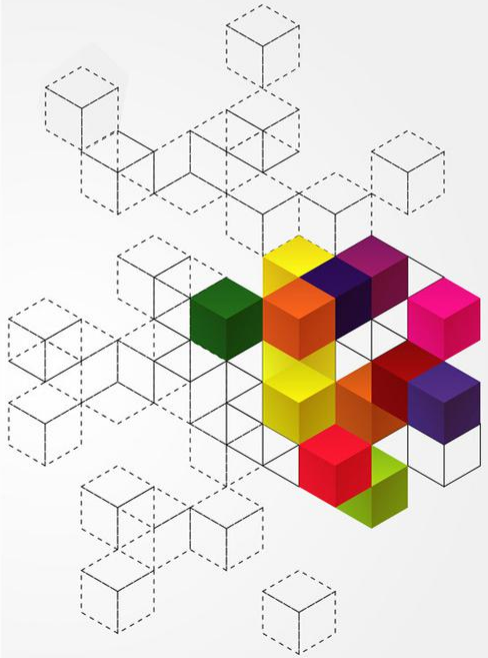
7.4 死锁处理方法

一、 死锁处理方法概述

二、 死锁预防

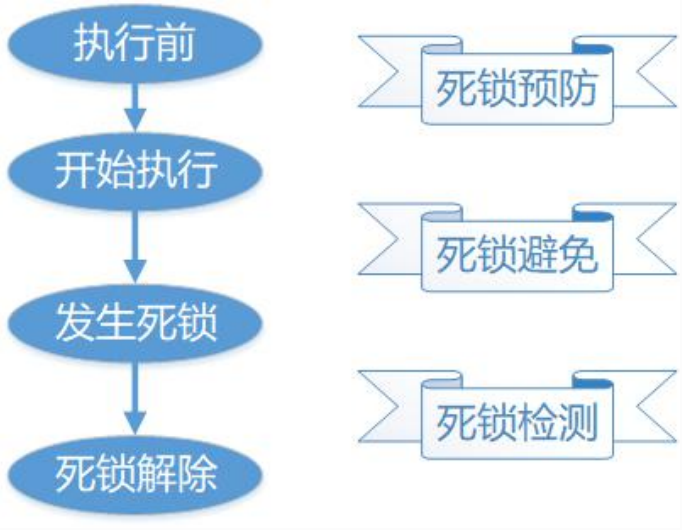
三、 死锁避免

四、 死锁检测

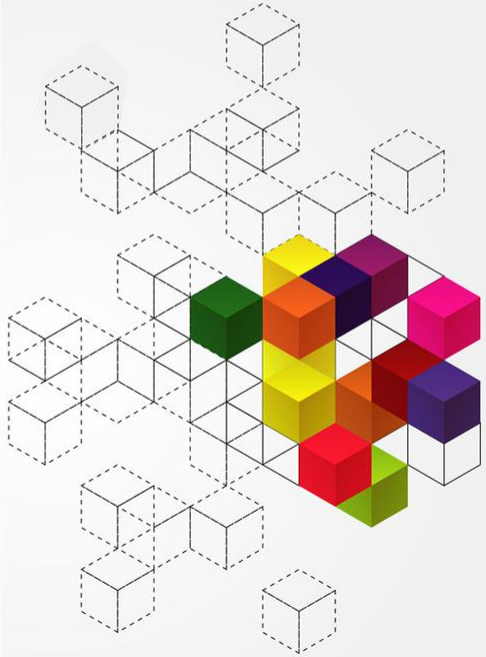


一、死锁处理方法概述

- **死锁处理方法一览图（按处理时机）**



- 死锁预防
- 死锁避免
- 死锁检测（与处理）



二、死锁预防

- **死锁预防思路：实现制定资源使用规则，保证程序按照规则使用资源就必然不会发生死锁**

策略1： Each process try to acquire all the resources it needs before the process run

进程： 运行前申请所需全部资源

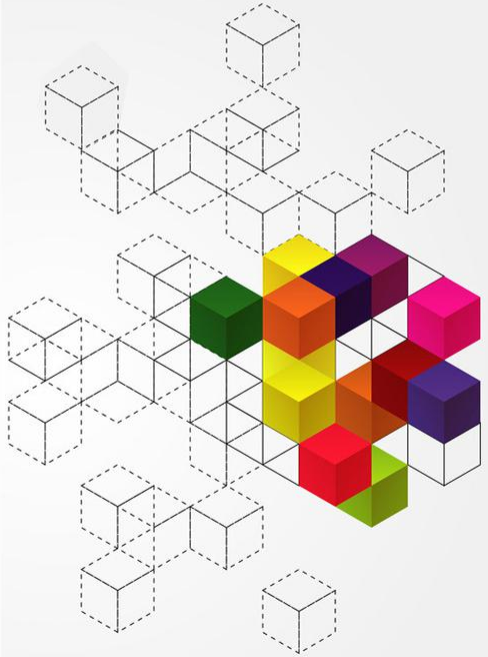
系统： 对于每个进程的资源申请，如果能够满足，则一次性全部分配；否则，进程等待

破坏“hold-and-wait”条件

缺点：

资源利用效率低

由于系统很难同时满足多个进程的一次性资源需求，可能出现大量进程等待现象



二、死锁预防

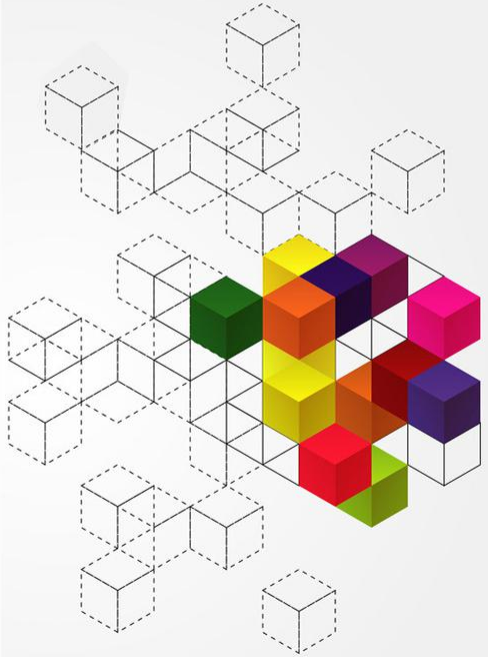
- **死锁预防思路：实现制定资源使用规则，保证程序按照规则使用资源就必然不会发生死锁**

策略2： **对资源进行编号，约定进程按照编号大小顺序对资源进行分配**

资源集：R={r1,r2,...,rn}

函数：F：R→N（为资源定一个级别）

进程pi可以申请资源rj中的实例⇔∀rl,pi占有rl, F(rl)<F(rj)



二、死锁预防

- **死锁预防思路：实现制定资源使用规则，保证程序按照规则使用资源就必然不会发生死锁**

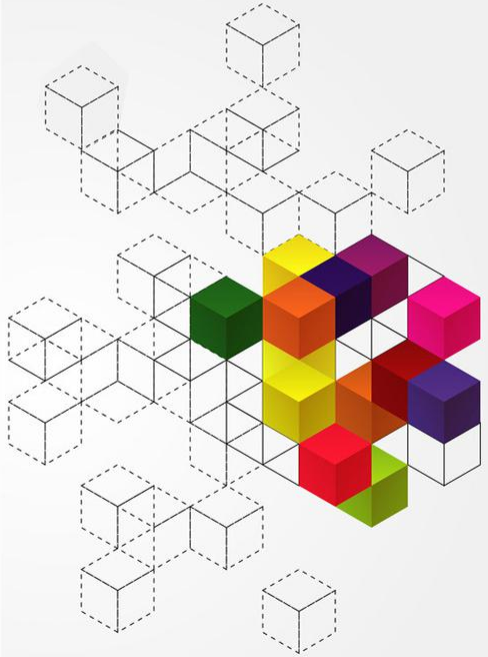
策略2： **对资源进行编号，约定进程按照编号大小顺序对资源进行分配**

例如：R={scanner,tape,printer}

F(scanner)=1; F(tape)=2; F(printer)=3;

要求：进程代码中申请资源必须按照先申请scanner，再申请tape，最后申请printer的原则进行

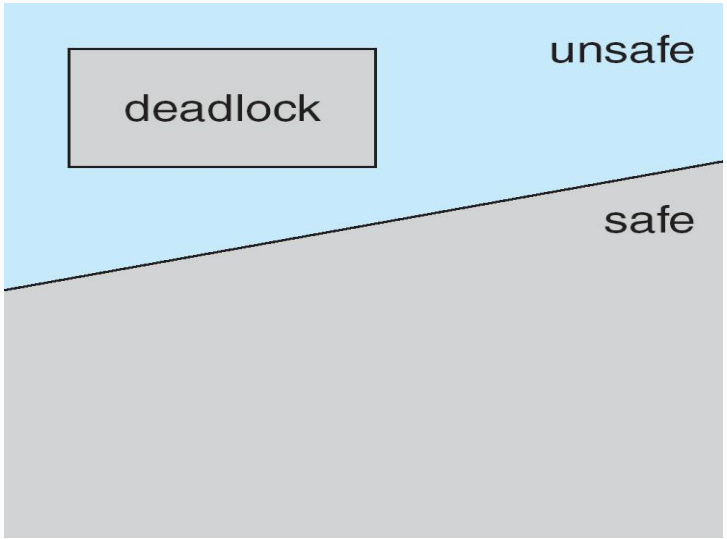
破坏“循环等待”条件：成功预防死锁

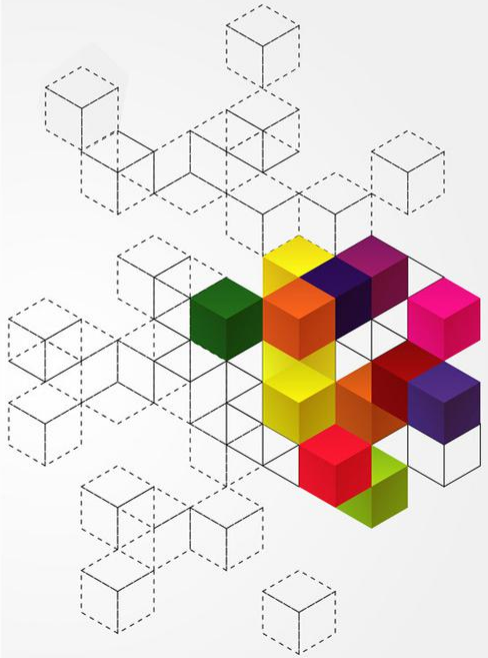


三、死锁避免

2.死锁避免（Deadlock Avoidance）

- 保证死锁永远不会进入有死锁风险的状态



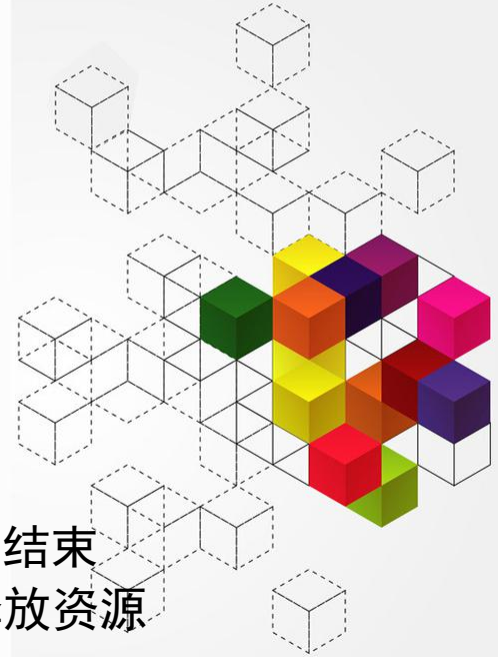


三、死锁避免

2.死锁避免（Deadlock Avoidance）

- 保证死锁永远不会进入有死锁风险的状态

- 当进程请求资源时，系统决定立即分配资源是否保证系统处于安全状态。
- 安全状态：当存在序列 <$P_1, P_2, \ldots, P_n$> 使得 P_i 请求资源可由当前可用资源以及所有进程 P_j $(j < i)$持有资源的总和满足。（如不存在这样的序列，则不安全）
- 亦即
  - 如果 P_i r所需资源无法立刻满足，则 P_i 可等待所有 P_j 结束
  - 当 P_j 结束后，P_i 可以取得所需资源、执行、结束、释放资源
  - 当 P_i 结束, P_{i+1} 可取得其所需资源，以此类推。



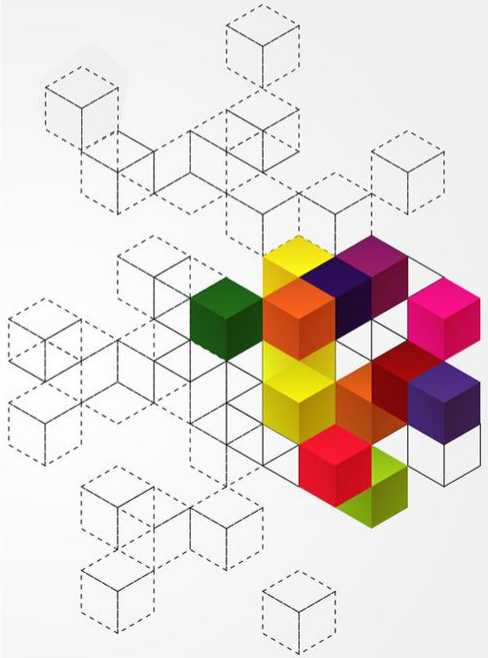
四、死锁检测

3.死锁检测(Deadlock Detection)

- 不采取任何预防或避免死锁的措施，在死锁发生时再决定如何处理

- 解决方案：
- （1）不处理，忽略，鸵鸟机制
- （2）检测，并解除

鸵鸟机制被多数操作系统采用，
原因是死锁检测代价很高



本讲小结

- 死锁处理方法概述
- 死锁预防
- 死锁避免
- 死锁检测