**操作系统产生死锁的原因和处理策略**

[strongerHuang](javascript:void(0);) 昨天

**关注、星标公众号**，不错过精彩内容

**产生死锁的原因**

当进程需要以独占的方式访问资源时，可能会发生死锁（Deadlock）。死锁是指两个或以上进程因竞争临界资源而造成的一种僵局，即一个进程等待一个已经被占用且永不释放的资源。若无外力作用，这些进程都无法向前推进。

**产生死锁的根本原因**是系统能够提供的资源个数比要求该资源的进程数要少。

**产生死锁的基本原因**可以分为两类：资源竞争和进程推进顺序不合理。

在资源竞争场景下，系统所拥有的资源是有限的，不能满足每个进程的需要。

例子：

A有纸，B有笔

A：你不给我笔，我就写不了作业

B：你不给我纸，我就写不了作业

彼此僵持不下……

多个程序同时运行时，进程推进顺序不合理。

例子：

A要前进2步，到桌子前，再后退2步。

但如果执行顺序不合理：A先后退，就永远到不了桌子前，触发不了后续动作，就会死锁。

**产生死锁的必要条件**

**产生死锁的四个必要条件：**

* 互斥条件涉及的资源是非共享的，即一次只能有一个进程使用。如果有另一个进程申请该资源，那么申请进程必须等待，直到该资源被释放。
* 不剥夺条件（非抢占）进程所获得的资源在未使用完毕之前，不能被其他进程强行夺走，即只能由获得该资源的进程自行释放。
* 占有并等待（部分分配）进程每次申请它所需要的一部分资源。在等待一新资源的同时，进程继续占用已分配到的资源。
* 环路条件（循环等待）存在一种进程收尾相接的循环链，链中每个进程都在等待下一个进程所持有的资源，造成这组进程处于永远等待状态。

**注意：**这四个条件是死锁的必要条件，只要系统发生死锁，这些条件必然成立。反之，上述条件只要有一个不满足，就不会发生死锁。所以要避免发生死锁，只需要破坏其必要条件。  
**死锁的处理策略**

**对于死锁一般有三种处理策略：预防死锁、避免死锁、死锁的检测及解除**

* **预防死锁**

通过设置一些限制条件，破坏死锁的四个必要条件中的一个或几个，让死锁无法发生。例如，将资源分层，得到上一层资源后才能够申请下一层资源，这样就破坏了环路等待条件。用户申请资源时，要求一次性申请所需要的全部资源，这就破坏了占有并等待条件。当一个已经占有某些不可剥夺资源的进程，请求新的资源而得不到满足时，它必须释放已经占有的所有资源，待以后需要时再重新申请，这就破坏了不剥夺条件。这些预防死锁的方法破坏了系统的并行性和并发性，通常会降低系统的效率。

* **避免死锁**

该方法同样属于事先预防，但它并不事先采取各种限制措施去破坏产生死锁的四个必要条件，而是在动态分配资源的过程中，用一些算法来防止系统进入不安全状态，避免死锁的发生。

具体策略如下：

1. 如果进程请求的资源会导致死锁，系统就拒绝启动该进程；

2. 如果对一个资源的分配会导致下一步的死锁，系统就拒绝本次分配；

显然要避免死锁，系统必须事先知道所拥有的资源数量及其属性。

一个著名的避免死锁的算法是银行家算法。

银行家算法是DijkstraE W于1968年提出的。之所以称为银行家算法，是因为该算法可用于银行系统。

所谓银行家算法，是指分配资源之前先确定资源分配是否会造成系统死锁。如果会死锁，则不分配，只有确认不会死锁后才进行分配。

银行家算法，需要按如下原则判断是否分配资源：

* 新进程进入系统时，它必须说明对各类资源的最大需求量，这一数量不能超过系统的资源总数。只有满足这一条件系统才接纳该进程。

* 当进程申请一组资源时，该算法需要检查进程对各类资源的最大需求量，如果系统现存的各类资源的数量可以满足此时的资源最大需求量时，就分配资源；否则进程必须等待，直到其他进程释放足够的资源为止。
* 进程需要在一定时间内无条件地归还它所申请的全部资源。
* **死锁的检测及解除**

死锁预防和避免都是对资源分配进行适当限制，属于事前措施，并不利于系统资源的充分共享。而死锁检测不会试图阻止死锁，即在死锁发生前不会做任何操作，只是通过设置的检测机制，检测当前是否发生死锁。若发生死锁，则采取一些措施来解除死锁。

判断死锁的法则主要基于第四条死锁的必要条件：

* 资源分配路径中没有环路，则系统不会出现死锁
* 资源分配路径中存在环路，则系统可能出现死锁
* 如果环路中的每个资料类中都只有一个资源，则系统存在死锁
* 如果环路中的每个资源类的资源个数不止一个，则环路的存在是产生死锁的必要条件但不是充分条件

解除死锁的方法包括资源剥夺法、进程撤销法、进程回退法、系统重启法等：

* 资源剥夺法

剥夺陷入死锁的进程所占用的资源，但并不撤销此进程，再将这些资源分配给需要的进程，直至死锁解除。

* 进程撤销法
* 一次性撤销陷入死锁的所有进程，回收所有占用的资源，等死锁解除后，再重新运行进程。
* 逐个撤销陷入死锁的进程，依次回收其资源并重新分配，直至死锁解除。可以优先撤销优先级低、预计剩余执行时间最长、CPU消耗时间少的进程。
* 进程回退法

让所有的进程回退到系统保存的检查点，这种方法要求系统建立并保存检查点、建立回退机制。

系统重启法结束所有进程并重启系统。这种方法很简单，但损失很大，先前的工作可能都浪费了。