diligentyang
码龄5年

288	484	1017	175	153万+
原创	粉丝	获赞	评论	访问

1万+	529	2万+	1295	
积分	收藏	周排名	总排名	等级

TA的主页 私信 关注

目录

死锁的方法

17:09:49 5462 收藏 2 版权

个或几个。

由于资源本身固有特性的限制，此方法不可行。

配方法，即要求进程在运行之前一次性申请它所需要的全部资源，在它的资源未满足前，不把它投入运

需要的所有资源分配给它，在整个运行期间便不会再提出资源要求，从而摒弃了请求条件。在该进程的
因而也摒弃了保持条件，这样可以保证系统不会发生死锁。

有饥饿现象，同时必须**预知进程所需要的全部资源。**

的资源后，便开始运行。

须释放已分配给自己的且已经用完的全部资源，然后才能再请求新的所需资源。

务，提高设备利用率，还可减少进程饥饿的几率。

规定一个进程在申请新的资源不能立即得到满足而变为等待状态之前，必须释放已占有的全部资源，若

寸，可以通过操作系统抢占这一资源（适用于状态易于保存和恢复的资源）。

夏杂，需付出很大的代价。会增加系统开销，降低系统吞吐量，且进程前段工作可能失效。

充中所有资源都按类型赋予一个编号，要求每个进程均严格按照资源序号递增的次序来请求资源，从而保
不路。

于排队。如输入机 =1，打印机 =2，磁带机 =3，磁盘机 =4。

资源，它接下来请求的资源只能是那些排在 i 之后的资源。

以请求 Rj，而进程 B 获得 Rj 再请求 Ri不可能发生，因为资源 Ri 排在 Rj 前面。

呈占据了较高序号的资源，此后它继续申请的资源必然是空闲的，因而进程可以一直向前推进。

资源不便（原序号已排定），用户不能自由申请资源，加重了进程负担。使用资源顺序与申请顺序不同

止死锁的必要条件（死锁预防）；在资源的动态分配过程中，采用某种策略防止系统进入不安全状态，从而

分每一个系统能够满足的资源申请进行动态检查，并根据检查结果决定是否分配资源，若分配后系统可能
以分配。

按某种进程顺序(p1,p2,…，pn) 来为每个进程 Pi 分配其资源，**直到满足每个进程对资源的最大需求**，使
此时的系统状态为安全状态，称序列(p1,p2,…，pn) 为安全序列。若某一时刻系统中不存在这样一个安全
安全状态。

动态申请资源，系统在进行资源分配之前，先计算资源分配的安全性，若此次分配不会导致系统进入不安
否则进程等待。

下会死锁。

尤有可能死锁。

入不安全状态。

P2 和 P3，共有 12 台打印机，三个进程对打印机的需求和占有情况如下表所示：

可用
3

，P1，P3），所以系统是安全的。

的转换

入不安全状态。

可用
2

有可用的打印机，但却不能分配给它，必须让 P3 一直等待到 P1 和 P2 完成，释放出资源后再将足够

s Algorithm)

锁算法（Dijkstra 于1965 年提出）

动态分配资源后不进入不安全状态，以避免可能产生的死锁。

一个距最大请求最近的客户。如果有，这笔贷款被认为是能够收回的。继续检查下一个客户，如果所有投
全的，最初的请求可以批准。

进程必须预先提出自己的最大资源请求数量，这一数量不能超过系统资源的总量，系统资源的总量是一

，…，Pn），m 类资源（R1，R2，…，Rm），银行家算法中使用的数据结构如下：

，表示 Rj类资源有 k 个可用

Pi最大请求 k 个 Rj类资源

程 Pi已经分配到 k 个Rj类资源

还需要 k 个 Rj 类资源

n [i,j]

分配算法)

k 个，设 Request[i,j]=k。当进程Pi 发出资源请求后，系统按如下步骤进行检查：

转 (2)；否则出错，因为进程申请资源量超过它声明的最大量。

]，转 (3)；否则表示资源不够，需等待。

作如下修改：
uest[i,j]

Request[i,j]
i,j]

则正式进行分配，否则恢复原状态让进程 Pi 等待。

法算法)

口下数据结构：

资源。开始时，Work=Available。

的资源分配给进程，使之运行完成。开始时，finish[i]=false；当有足够资源分配给进程 Pi 时，令

sh [i] = false
程 Pi

如果找到，转 (3)，否则转 (4)
可顺利执行完，并释放分配给

ocation[i,j] ；Finish[i] := true ，

= true ，则表示系统处于安全状
态。

到 P4，3 类资源及数量分别为 A(10 个) ，B（5 个），C（7 个），T0 时刻的资源分配情况如下：

ocation	Need	Available
B C	A B C	A B C
1 0	7 4 3	3 3 2
0 0	1 2 2	
0 2	6 0 0	
1 1	0 1 1	
0 2	4 3 1	

Available
A B C
3 3 2

Allocation A B C	Work+ Allocation A B C	Finish
0 1 0		false
2 0 0		false
3 0 2		false
2 1 1		false
0 0 2		false

---

2)
(3,3,2)

2)
,3,2)
应的向量Available、Need、Allocation

able、Need、Allocation

tion C	Need A B C	Available A B C
0	7 4 3	3 3 2
		(2 3 0)
0	1 2 2	
2)	(0 2 0)	
2	6 0 0	
1	0 1 1	
2	4 3 1	

初始：Work = Available = [2 3 0]

Allocation A B C	Work+ Allocation A B C	Finish
3 0 2	5 3 2	true
2 1 1		false
0 0 2		false
3 0 2		false
0 1 0		false

)，系统按银行家算法进行检查：

)
,3,0)

)，系统按银行家算法进行检查：

)
,3,0)

应的向量

tion C	Need A B C	Available A B C
0	7 4 3	3 3 2 (2 3 0)
0 2) 2 1 2	1 2 2 (0 2 0) 6 0 0 0 1 1 4 3 1	

ion C	Need A B C	Available A B C
0	7 4 3	3 3 2 (2 3 0)
0] 0 2) 2 1 2	[7 2 3] 1 2 2 (0 2 0) 6 0 0 0 1 1 4 3 1	[2 1 0]

任何进程需要，故系统进入不安全状
系统不分配资源。

配资源数量	
B	C
1	2
0	2
0	5
0	4
1	4

(2, 3, 3)

，请给出安全序列。

已分配资源数量			尚需资源Need		
A	B	C	A	B	C
2	1	2	3	4	7
4	0	2	1	3	4
4	0	5	0	0	6
2	0	4	2	0	1
3	1	4	1	1	0

态？若是，请给出安全序列。

7, 4, 11) (9, 5, 13) (13, 5, 15)
P1 P2 P3
) (2, 1, 2) (4, 0, 2)

安全序列：p4, p5, p1, p2, p3

（0，3，4），是否能实施资源分配?

ilable （2, 3, 3）所以不能满足进程 P2 的请求。

），是否可以资源分配?

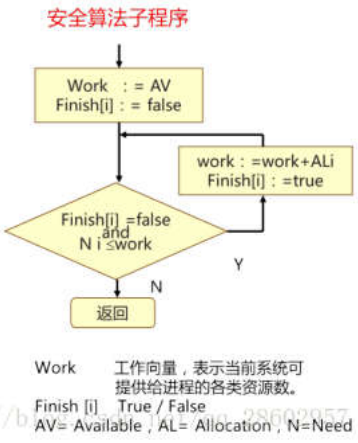
2, 0, 1)
ble (2, 3, 3)
Allocation4 =(4, 0, 4)
3, 3)

已分配资源数量			尚需资源Need		
A	B	C	A	B	C
2	1	2	3	4	7
4	0	2	1	3	4
4	0	5	0	0	6
4	0	4	0	0	1
3	1	4	1	1	0

Available = (0, 3, 3)

4, 11) P1 (9, 5, 13) P2 (13, 5, 15) P3
(2, 1, 2) (4, 0, 2)

p4, p5, p1, p2, p3 ，可以分配



, 进程重启。

资源请求。

就是说, 它们执行的顺序没有任何同步要求的限制。

分配的资源数目必须是固定的。

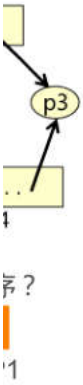
预防方法，也未采用死锁避免方法，而是直接为进程分配资源，则系统中便有可能发生死锁。

并加以消除，为此需提供死锁检测和解除死锁的手段。

中保存资源的请求和分配信息，利用某种算法对这些信息加以检查，以判断是否存在死锁。

是描述进程和资源间的申请和分配关系的一种有向图。

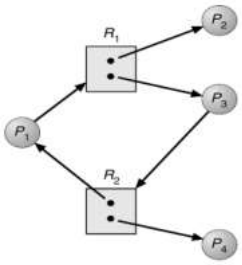
源节点 R，方框中的小黑点数表示资源数。



?

1

分配给了p1和p2，然后p1请求r1，而r1中就1个资源，分配给了p2，所以此时p1阻塞，p2请求r3，而r3
此时p2也阻塞，r4中的一个资源分配给了p3，所以，待p3运行完毕，释放r3，p2阻塞解除，运行完毕后
同样的分析思路，此处不再详解。



有环路无死锁

存在环路，则系统中不存在死锁；反之，如果资源分配图中存在环路，则系统中可能存在死锁，也可能不

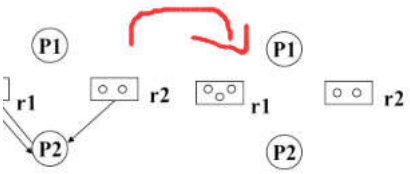
进程结点 Pi，若无则算法结束；
，使 Pi 成为一个孤立结点；

图中所有的边，使所有进程都成为孤立结点，则称该图是可完全简化的；反之，称该图是不可完全简化

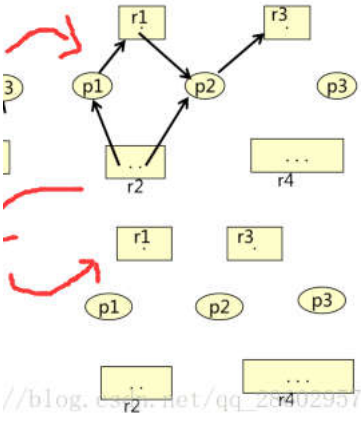
与之相连。

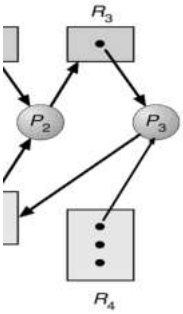
满足其要求。

件是资源分配图不可完全简化。



分配图可完全简化





，三个进程结点均为阻塞结点。
化简顺序无关，最终结果相同。

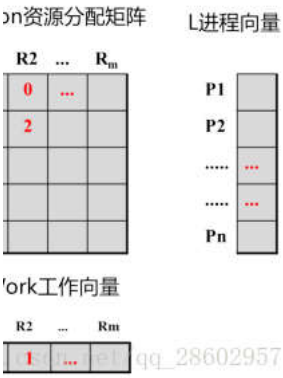
分配图化简和死锁定理来检测死锁。

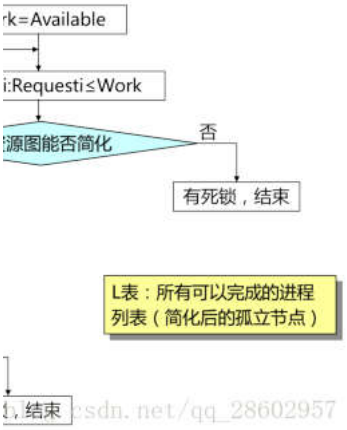
的措施。资源分配时不检查系统是否会进入不安全状态，被请求的资源都被授予给进程。需要周期性检测

相当于死锁避免）。

死锁。

行家算法的数据结构：





类型：*A*（7个），*B*（2个），

st	*Available*
	A B C
	0 0 0

对所有i，*Finish*[*i*] = true.

n	*Request*	*Available*
	A B C	*A B C*
	0 0 0	0 0 0
	2 0 2	
	0 0 1	
	1 0 0	
	0 0 2	

还所有的资源，但是资源不够

P_4

应将陷入死锁的进程从死锁状态中解脱出来，常用的解除死锁方法有两种:

它进程剥夺足够数量的资源给死锁进程，以解除死锁状态。

系统中撤消一个/一部分死锁进程，并剥夺这些进程的资源供其它死锁进程使用。

，直至有足够的资源可用，使死锁状态消除为止。

个进程直到取消死锁循环为止。

liangchuan的博客 6779

)方法可归结为以下四种： 1）预防死锁。这是一种较为简单和直观的事先预防的方法。该方法是通过设置某些限制条...

高权重 评论

详细，可以将目录放在最上面，方便来访者寻找自己想看的内容 2年前

上理策略 wenlijunliujuan的专栏 4万+

资源的类型3.1 可重用资源和消耗性资源3.1.1 可重用资源（永久性资源）3.1.2 消耗性资源（临时性资源）3.2 可抢占...

2016的博客 - CSDN博客 11-29

死锁状态中解脱出来,常采用的方法有: 剥夺资源:从其它进程剥夺足够数量的资源给死锁进程,以解除死锁状态; 撤消进...

824的博客-CSDN博客_解除死锁的两... 6-27

决的方法及死锁的检测与解除 产生死锁的原因和必要条件:产生死锁的原因:1.竞争资源。当系统中供多个进程共享的资...

常用方法) _BigMao的博客 54

死锁跟阻塞的sql；如果需要杀死进程可以直接解注下面的杀死进程sqlGO/****** Object: StoredProcedure [dbo].[s...

解决方法【转】 weixin_34209851的博客 223

9-24 16:48:58阅读767评论1 字号：大中小订阅 一、要点提示（1） 掌握死锁的概念和产生死锁的根本原因。（2） ...

数据库_Beyond_2016的博客-CSDN博客 4-8

避免死锁、检测死锁、解除死锁解决四多的常用策略如下:鸵鸟策略、预防数据库

用方法)__BigMao的博客-CSDN博客 6-3

阻塞sql的方法(两种常用方法) _bigmao 2020-06-03 10:41:39 4收藏最后发布:2020-06-03 10:41:39首发:2020-06-0...

杨森源的博客 3721

等待对方所拥有的资源，且这些并发进程在得到对方的资源之前不会释放自己所拥有的资源。从而造成大家都想得到...

weixin_30325487的博客 30

造成的一种僵局，若无外力作用，这些进程都将永远不能再 向前推进。安全状态与不安全状态：安全状态指系统能按...

- CSDN博客 11-21

中:1.查询是否锁表show OPEN TABLES where In_use > 0;2.查询进程(如果您有SUPER权限,您可以看到所有线程。...

博客 - CSDN博客 11-18

个条件同时具备:互斥条件、不可抢占条件、占有且申请条件、循环等待条件。 (3) 记住解决死锁的一般方法,掌握死锁...

Yaqin_lee的博客 738

方法同样可以类比处理火灾，从而将处理死锁分为预防、避免、检测与恢复四个方面。死锁预防 预防的目的就是...

产生及解决死锁的方法 小小柴的博客 1516

共享系统资源时，系统必须提供同步机制和进程通信机制，然而，对这种机制使用不当的话，可能会出现进程永远被...

的博客 - CSDN博客 11-29

会遇见思索的问题,这里介绍几种常见的避免死锁的方法:1、避免一个线程同时获取多个锁2、避免一个线程同时占用多...

gchuan的博客-CSDN博客 3-26

一种措施。当检测到系统中已经发生死锁时,将进程从死锁状态中解脱出来。常用的方法是撤销或挂起一些进程,以便...

杨森源的博客 1万+

调度信息（就绪时间、开始截止时间和完成截止时间、处理时间、资源要求、优先级）系统处理能力强。在实时系统...

点赞1 评论1 分享 收藏2 手机看 ... 关注

程序员的首选! Python大本营的博客 5万+

置顶公众号” Python大本营 IT人的职业提升平台最近有很多程序员在CSDN博客发帖讨论：用Windows还是Linux？...

博客 - CSDN博客 12-1

牛不能确定出现了死锁,每种方法总是有缺陷的。有时为了执行某个任务。某个线程花了很长的时间去执行任务,如果在...

:死锁的基本方法_死锁_孤..._CSDN博客 1-6

死锁状态中解脱出来,常采用的方法有: 剥夺资源:从其它进程剥夺足够数量的资源给死锁进程,以解除死锁状态; 撤消进...

半暖的博客 6万+

发进程因争夺系统资源而产生相互等待的现象。 原理：当一组进程中的每个进程都在等待某个事件发生，而只有这...

Cookie1997的小屋 2213

进行了解：死锁的概念及资源分配图我们都知道，操作系统可能会遇到死锁这种情况，那么处理死锁的方法有哪些呢...

jianbai_的博客 480

锁：在并发环境下，各进程因竞争资源而造成的一.种互相等待对方手里的资源，导致各进程都阻塞，都无法向前推进...

horst Hu的专栏 1511

i个以上的进程（线程）在执行过程中，因争夺资源而造成的一种互相等待的现象，若无外力作用，它们都将无法推进...

伊甸园的博客 63

feizai1208917009的博客 530

e/details/80643753

冲吧，不要停! 3万+

se/201303/193062.htmlhttp://www.itnose.net/detail/6399027.html解除正在死锁的状态有两种方法：第一种：1...

一路前行 7598

进程中因争夺资源而造成的一种僵局。具体来讲在多进程环境中，当一个进程请求资源时，如果该资源不能立即获得...