

银行家算法在高校排课系统中的应用

彭志勇, 赖晓风

(西华师范大学 计算机学院, 四川 南充 637002)

【摘要】银行家算法是操作系统中用来避免死锁的一种典型的算法。本文主要论述了银行家算法的思想, 利用该算法在高校排课系统中针对选修课程教室安排中的应用, 并设计了一种排课的方案, 从而使每个教室都能得到充分合理的安排, 突出了银行家算法相对其他算法在高校排课系统中的优势。

【关键词】银行家算法; 死锁; 数据结构; 安全性算法; 遗传算法

【中图分类号】TP311.52 **【文献标识码】**A **【文章编号】**1673-1891(2011)02-0057-03

引言

银行家算法是一种最具有代表性避免死锁的算法, 由于该算法可以用于银行现金贷款而得名。一个银行家把他的固定资金贷给若干顾客, 只要不出现一个顾客借走所有资金后还不够, 银行家的资金就是安全的。它有很多的应用, 只要是涉及多个独立个体对某种资源的动态申请和回收就可以应用此算法。在计算机科学中一般用此算法检测进程的推进顺序是否是安全队列, 如果不是则会因为对资源的争夺而造成死锁。本文主要是介绍银行家算法在高校排课系统中的应用, 用该算法来得到一个可用方案, 以避免高校教学管理人员在安排可用教室资源时出现死锁的情况。

1 银行家算法中的数据结构

银行家算法适用于每种资源类型有多个实例的资源分配系统, 该算法如此命名是因为这一算法可用于银行系统, 以确保银行决不会分配其现金以致使它不能满足其所有客户的需要^[1]。当新进程进入系统时, 它必须说明其可能需要的每种资源类型的实例的最大数量。这一数量不可能超过系统资源的总量。当用户申请一组资源时, 系统必须确定这些资源的分配是否仍会使系统处于安全状态。如果会, 就可分配资源; 否则, 进程必须等待直到某个其他进程释放足够资源为止。为了实现银行家算法, 系统要设置几个数据结构。这些数据结构对资源分配系统的状态进行了编码。设 n 为系统进程的个数, m 为资源类型的种类。需要如下数据结构:

1.1 可利用资源向量 Available

可利用资源向量也称为空闲向量, 是一个含有 m 个元素的数组。长度为 m 的向量表示每种资源的现有实例的数量, 其初始值是系统中所配置的该类全部可用资源的数目, 其数值随该类资源的分配和回收而动态地改变^[2]。如果 $Available[j]=K$, 那么资源

类型 R_j 现有 K 个实例。

1.2 最大需求矩阵 Max

最大需求矩阵是一个 $n \times m$ 的矩阵, 它定义了系统中每一进程已占有的每一类资源数的最大需求。如果 $Max[i, j]=K$, 那么第 i 个进程最多可申请 K 个资源类型 R_j 的实例^[3]。

1.3 分配矩阵 Allocation

分配矩阵也叫做占有矩阵, 是一个 $n \times m$ 的矩阵, 它定义了系统中每个进程现在所分配的各种资源类型的实例数量。如果 $Allocation[i, j]=K$, 那么第 i 个进程现在当已分配了 K 个资源类型 R_j 的实例^[3]。

1.4 需求矩阵 Need

需求矩阵也叫做申请矩阵, 是一个 $n \times m$ 的矩阵, 表示每一个进程还需要的剩余资源。如果 $Need[i, j]=K$, 那么第 i 个进程还可能申请 K 个资源类型 R_j 的实例^[3]。

显然, 前三个矩阵之间存在如下关系: $Need[i, j]=Max[i, j]-Allocation[i, j]$, 这些数据结构的大小和值会随着时间而改变。

2 银行家算法的实现

2.1 进程申请资源的情况

设 $Request_i$ 是进程 P_i 的请求向量, 如果 $Request_i[j]=K$, 表示进程 P_i 需要 R_j 类资源的个数为 K 。 $Request_i$ 与 $Need$ 的关系可能为以下 3 种情况:

(1) $Request_i > Need[i]$ 。这种情况表示该进程的资源需求已超过它所宣布的最大值, 因此认为出错。

(2) $Request_i = Need[i]$ 。这种情况表示该进程现在对它所需的全部资源一次申请完成。

(3) $Request_i < Need[i]$ 。这种情况表示该进程现在对它所需资源再进行部分的申请, 剩余的资源以后可再次申请。

2.2 银行家算法的描述

当进程 P_i 发出资源请求后, 系统按下述步骤进

行检查;

(1)如果 $Request_i \leq Need_i$,便转向步骤(2);否则显示出错,因为它所需的资源数已超过它事先要求的最大值。

(2)如果 $Request_i \leq Available_i$,便转向步骤(3);否则,表示尚无足够资源, P_i 须等待。

(3)假设系统将资源分配给 P_i ,则需修改如下数据结构的值:

$$Available_i := Available_i - Request_i;$$

$$Allocation_i := Allocation_i + Request_i;$$

$$Need_i := Need_i - Request_i;$$

(4)系统利用安全性算法,检查此次资源分配后,系统是否处于安全状态。如果是安全的,则将资源真正地分配给进程 P_i ,否则,将本进程的试探分配作废,恢复原来的资源分配状态,进程 P_i 等待。

2.3 安全性算法

工作向量 $Work$ 表示在算法执行过程中,系统可提供给进程继续运行所需的各类资源数目,它含有 m 个元素,在执行安全算法开始时,初始值 $Work = Available$ 。

完成向量 $Finish$ 表示系统能否运行完成。它有 n 个分量,分别表示各进程是否可执行完成。若 $Finish[i] = true$,表示第 i 个进程能够获得足够的资源,运行完成;若 $Finish[i] = false$,表示该进程不能获得所需全部资源,不能运行完成。

设初值 $Finish[i] = false, i = 0, 1, 2, \dots, n-1$ 。当有足够资源分配给第 i 个进程时,再令 $Finish[i] = true$ 。

安全算法的步骤:

(1)进行安全性检查。从进程集合中找到一个能满足下述条件的进程 i :

$$Finish[i] = false \text{ 并且 } Request_i \leq Work;$$

若找到这样的进程,执行步骤(3);若找不到这样的进程,则转步骤(4)。

$$(2) Work := Work + Allocation[i];$$

$$Finish[i] := true;$$

因为进程 P_i 若能执行完成,则能够释放它所占的资源。

(3)若所有进程的 $Finish[i] = true$ 都满足,则表示系统处于安全状态,正式将资源分配给进程;否则,系统处于不安全状态,系统不能进行这次试探性分配,恢复原来的资源分配状态,让进程 P_i 等待。

如果已判定系统处于安全状态,则通过运算过程同时可以找到一个安全序列。如果系统处于不安全状态,则所有 $Finish[i] = false$ 的进程都参与了死锁。

3 银行家算法在高校排课系统中的应用

排课系统是高校不可缺少的一部分,它的内容对于学校的决策者和管理者来说都至关重要,随着高校的扩招和学生课程的增加,而学校的资源又是有限的,用传统的人工管理方式排课就显得比较复杂,所以高校教务处如何使当前所有选课的学生在有限的时间内修完课程,并归还所有申请的资源,这一点非常重要。

在此,引入两个向量 $Resource$ (资源总量)和 $Available$ (剩余资源量)以及两个矩阵 Max (每个进程的最大需求量)和 $Allocation$ (已为每个进程分配的数量)。它们共同构成了任一时刻系统对资源的分配状态。

向量模型为:

R_1	R_2	R_3	R_4
-------	-------	-------	-------

假设教室资源总量 $Resource$ 为 R_1 类教室 5 个(最多能坐 100 人的教室), R_2 类教室 3 个(最多能坐 80 人的教室), R_3 类教室 2 个(最多能坐 50 人的教室), R_4 类教室 1 个(最多能坐 20 人的教室),即向量 $Resource(5, 3, 2, 1)$ 。所选修的课程分别为进程 P_1, P_2, P_3 , 学生选课的人数分别为: 380 人、570 人和 120 人,提出的请求 $Request$ 就是他们的最大需求 Max 。

银行家算法应用于高校排课系统中,主要是让系统能够判断现有的教室资源是否能满足学生上课的需求,找到一个合理的方案,既安全序列,而不至于陷入死锁。根据对实际情况的调查和研究,很显然从上面给出的数据可以看出,如果课程安排不合理,系统很可能会发生死锁,这样的安排就不具备可行性,因此需要设计一个合理的方案。

笔者的设计方案思想如下:

(1)先安排选其中一门课的学生上课,等这批学生上完课后就会将资源释放,这样选其他课程的学生就可以分配到足够多的资源,这样就能正常完成任务。因为选第一门课程的人数为 380 人,而 R_1 类教室有 5 个,且最多能坐 100 人, R_2 类教室有 3 个,最多能坐 80 人,所以可以给 P_1 这个进程分配 R_1 类资源 3 个,这样还剩下 80 人没有分配资源,再给进程 P_1 分配 R_2 类资源 1 个,这样就刚好分配完,资源也得到充分的利用, R_3 类资源和 R_4 类资源在此不需要分配给进程 P_1 (如表 2 所示)。

(2)给进程 P_1 分配完资源后, R_1 类资源还剩下 2 个, R_2 类资源也剩下 2 个,即向量 $Available(2, 2, 2, 1)$,而选第二门课程的人数为 570 人,就算把剩下的所有的资源全部分配给进程 P_2 也是不够的,所以只能先给进程 P_2 分配一部分的资源,等进程 P_1 执行完并将资源释放以后,再分配给进程 P_2 ,这样就可以正常完成任务了。暂且先给进程 P_2 分配 R_1 类资源 2

个, 分配 R_2 类资源1个, 不分配 R_3 类资源和 R_4 类资源(如表2所示), 这样进程 P_2 还剩290个人没有分配, 等进程 P_1 释放资源以后, 再将 R_1 类资源分配3个进程 P_2 即可。

(3)此时 R_1 类资源已经分配完了, R_2 类资源还剩下1个, R_3 类资源还剩下2个, R_4 类资源还剩下1个, 因为选第三门课的人数为120人, 先给进程 P_3 分配1个 R_2 类资源, 进程 P_3 还剩下40人没有分配资源, 再给进程分配 R_3 类资源1个(如表2所示), 这样进程 P_3 就能够正常完成任务了。

由以上的方案设计思想可以得出矩阵Max如表1所示。

表1 矩阵Max(每个进程的最大需求量)

进程	资源			
	R_1	R_2	R_3	R_4
P_1	3	1	0	0
P_2	5	1	0	0
P_3	0	1	1	0

其中, Max(2, 1)元素值为5, 表示进程 P_2 对资源 R_1 的最大需求为5个, 矩阵Allocation如表2所示。

表2 矩阵Allocation(已为每个进程分配的数量)

进程	资源			
	R_1	R_2	R_3	R_4
P_1	3	1	0	0
P_2	2	1	0	0
P_3	0	1	1	0

因为选择第二门课程的还没有分配到足够的资源, 所以, 还要设置另外一个矩阵: 各个进程尚需资源量(Need), 可以看出 $Need=Max-A$, 矩阵Need如表3所示。

表3 矩阵Need(尚需资源量)

进程	资源			
	R_1	R_2	R_3	R_4
P_1	0	0	0	0
P_2	3	0	0	0
P_3	0	0	0	0

从上述表格中的数据可以看出, 系统中的资源和所开设课程的教室正好分配完, 资源得到了充分的利用, 这就是银行家算法的一个应用。

4 总结

以上是对银行家算法的介绍以及在高校排课系统中的应用, 并设计一种合理的排课方案, 从而避免了死锁的情况发生, 充分而又合理的利用了学校的资源。银行家算法是一种动态策略避免死锁的算法, 银行家算法的优点是能有效的合理的安排系统已有的资源, 与其它的算法相比较, 限制的条件少, 资源的利用程度提高, 所以在高校排课系统中, 教室的安排得到了很充分的利用, 并且该算法能保证所有的客户在有限的时间内得到满足。而传统的遗传算法在适度函数选择不当的情况下有可能收敛于局部最优, 而不能全局最优, 对于动态数据, 用遗传算法求最优解比较困难。

注释及参考文献:

- [1]肖竞华, 陈建勋. 计算机操作系统原理: Linux实例分析[M]. 西安: 西安电子科技大学出版社, 2008: 228-333.
- [2]徐德民. 操作系统原理Linux篇[M]. 北京: 国防工业出版社, 2004: 66-68.
- [3]王红. 操作系统原理及应用(Linux)[M]. 北京: 中国水利水电出版社, 2005: 93-96.
- [4]Abraham Silberschatz. Operating System Concepts[M]. Sixth Edition, 2003: 188-191.
- [5]HuMin. Research on Combine of Load2based and Queue2based AQM Algorithm[J]. Journal of Wuhan University of Technology, 2005, 27(12): 89-92.
- [6]Andrew S Tanenbaum. Modern Operating Systems[M]. Second Edition. New Jersey: Prentice Hall, 2002.

Banker's Arithmetic in the University System of Arrangement

PENG Zhi-yong, LAI Xiao-feng

(School of Computer Science, China West Normal University, Nanchong, Sichuan 637002)

Abstract: Banker's algorithm is a typical algorithm, used to avoid deadlock in the operating system. This paper mainly discusses the idea of banker's algorithm, and its application to arrange the classroom for elective courses in the University Course Scheduling System, And designed a course arrangement of the program, thus each classroom can be arranged sufficiently and reasonably. In order to highlight the advantages of banker's algorithm compared with other algorithms in the University Course Scheduling System.

Key words: Banker's arithmetic; Deadlock; Data structure; Security arithmetic; Genetic algorithm