DOI:10.3969/j.issn.1673-4785.201409021
网络出版地址:http://www.cnki.net/kcms/detail/23.1538.tp.20150514.1432.001.html

一种基于银行家算法的网络爬虫资源配置策略

王庆红,李广凯,周育忠,韦嵘晖
(南方电网科学研究院有限责任公司 技术情报所,广东 广州 510080)

摘 要:死锁是多用户操作系统正常运行的一个重要问题,系统资源不足会导致爬虫算法进入不安全状态,进而引发死锁等问题。引入被广泛用于操作系统的银行家算法,调度多个网络爬虫进程并发运行,并且为每个进程合理分配系统资源,当进程无法获取系统资源时,则等待其他进程分配完成后释放系统资源,从而完成资源分配,有效降低死锁率。采用 C++编程,设计并实现基于银行家算法的网络爬虫配置策略。通过 2 h 21 min 35 s 工程测试,urllib2 算法死锁率为 30%,新算法死锁率仅为 2%,测试证明该策略能够有效降低死锁率,能高效完成多个任务进程的资源分配。

关键词:操作系统;资源配置;死锁;系统安全;银行家算法;网络爬虫
中图分类号:TP361; TM75 **文献标志码**:A **文章编号**:1673-4785(2015)03-0494-05

中文引用格式:王庆红,李广凯,周育忠,等. 一种基于银行家算法的网络爬虫资源配置策略[J]. 智能系统学报, 2015, 10(3): 494-498.
英文引用格式:WANG Qinghong, LI Guangkai, ZHOU Yuzhong, et al. A web crawler resource allocation strategy based on the Banker's algorithm[J]. CAAI Transactions on Intelligent Systems, 2015, 10(3): 494-498.

A web crawler resource allocation strategy based on the Banker's algorithm

WANG Qinghong, LI Guangkai, ZHOU Yuzhong, WEI Ronghui
(Technology Information Department, Electric Power Research Institute of China Southern Power Grid, Guangzhou 510080, China)

Abstract: Deadlock is a major issue for the normal operation of a multi-user operating system. Insufficient system resource will make the crawler algorithm go into the unsafe state, which will further cause problems such as deadlock. The introduction of the Banker's algorithm, which is widely used in the operating system can schedule multiple web crawler processes running concurrently and allocate system resources rationally for each process. When the process is unable to get the system resources, the other processes need to release resources to complete the allocation of resources, thereby reducing the rate of deadlock effectively. In this paper, a web crawler resource allocation strategy based on Banker's algorithm is designed and implemented using C++ programming. After approximately 2.5 hours of engineering testing the results showed that, the deadlock rate of urllib2 algorithm is 30% and the improved algorithm is only 2%. It is proven that the improved algorithm can reduce deadlock rate effectively and complete resource allocation for multi-process with high efficiency.

Keywords: operating system; resource allocation; deadlock; system safety; Banker's algorithm; web crawler

网络爬虫资源分配不足会导致发生死锁,发生死锁的爬虫进程无法继续运行,必须通过释放爬虫资源重新抓取网页信息,因此造成爬虫算法效率低下。G. Ricart 等[1]研究并实现了计算机网络优化算法,通过理论分析证明了理想条件下可以避免死锁。Wang[2]通过实验验证了银行家算法在判断系统安全性中的有效性。本文研究的多用户操作系统属于共享资源系统,Hao 等[3-4]研究了共享资源系统的鲁棒死锁控制方法。Petri 网可用于描述多用户操作系统模型,Ma 等[5]将银行家算法用于解决 Petri

收稿日期:2014-09-12. 网络出版日期:2015-05-14.
通信作者:王庆红. E-mail: wangqh@csg.cn.

网中的死锁问题,有效降低了死锁率,表明银行家算法可应用于解决多用户操作系统的死锁问题。多处理器片上系统适用于多用户操作系统,Xiao 等[6]研究了多处理器片上系统上的死锁问题,提出了一种硬件死锁检测算法。多用户操作系统的死锁问题本质上是多进程访问公共资源导致的死锁问题,Lou 等[7]研究了多进程访问公共资源导致的死锁问题,提出了一种避免死锁的算法,实验结果表明该算法有效地解决了分布式事务管理系统中的死锁问题。S. A. Reveliotis 等[8]研究的多车辆系统资源配置算法解决死锁问题。Lang 等[9]基于银行家算法提出了二次时间算法,更好地实现了操作系统资源优化配置。本文研究的网络爬虫配置策略基于 Windows 操作系统进行开发,Windows 操作系统是基于事件驱动的程序模型,Tang 等[10]提出了一种在事件处理系统中有效避免死锁的方法。Duda 等[11]将 Cache 模块增加到爬虫系统中,并采用了热点检测机制,避免从服务器端重复地获取相同内容,采用 MFC 编程设计并实现了基于银行家算法的网络爬虫资源配置应用软件,增加 Cache 模块的设计并引入热点检测机制,提高了系统的运行效率。Peng 等[12]将 Ajax 页面状态的抓取转换为 DRPP,在完全状态转换图拓扑结构已知的情况下,核心状态转换图增加少量取自它的边后成为平衡、强连通图,该欧拉回路即为最优搜索路径,从而提高了效率。杨梅等[13]针对银行家算法中的安全分配算法进行了研究,该算法缩小了安全检查的范围,提高了系统效率。陈昊成[14]针对预防死锁问题和调度策略问题提出了解决方案,应用于资源管理分配系统。章韵等[15]经过仿真实验提出了一种资源分配算法,实验结果证明该算法可以有效避免死锁。

以上研究成果表明,银行家算法是一种避免死锁的有效算法。本文提出了基于银行家算法的网络爬虫资源分配策略,以降低多任务处理系统的死锁率,提高网络爬虫资源的利用率,避免死锁的发生。

1　基本概念

1.1　多用户操作系统

根据在同一时间最多允许多少个用户同时操作计算机,操作系统可分为单用户操作系统和多用户操作系统。同一时间内允许多个用户同时使用计算机,称为多用户操作系统;反之,同一时间内最多允许一个用户操作计算机,则称为单用户操作系统。常见的多用户操作系统有 Windows Server 2003 和 Windows Server 2008 等。系统资源的有限性及进程并发性是导致发生死锁的原因,当系统无法满足进程的资源请求时,资源申请失败导致发生死锁。

1.2　死锁

在多用户操作系统,进程是并发执行的,但是存在一种危险—死锁。若无内部鲁棒容错或外部干预,系统将长期处于封锁状态。竞争资源及进程推进顺序非法是导致死锁的主要原因。在当前资源限制下,寻找一组资源分配的执行顺序,从而避免产生死锁,是本文研究的主要内容。将银行家算法为避免死锁获取安全进程序列的递归思想,借鉴到网络爬虫算法中。将请求集长度作为一个变量,边界条件为请求集长度及节点状态数组,从而能够确定请求集节点的合法性。该算法的时间复杂度比 urllib2 算法较小,并且能够获得最优请求集长度。

1.3　安全状态

只要保持系统时刻在安全状态,便可避免死锁。假设系统中并发存在 n 个进程,分别为 P_1,P_2, …, P_n,如果按照某种序列顺序进行分配资源,使得 n 个进程都能顺利完成资源分配,则该序列称为安全序列,此时系统处于安全状态。通过一个实例说明,例如有 4 个进程共享 20 个同类资源,进程 P_1共需 15 个资源,进程 P_2共需 5 个资源,进程 P_3共需 8 个资源,进程 P_4共需 10 个资源。假设在 T_n时刻,P_1、P_2、P_3、P_4分别已经获得 8、4、4、2 个资源,尚有 2 个空闲资源未分配。经过分析,T_n时刻是安全的,因为存在一个安全序列<P_1,P_2,P_3,P_4>,只要系统按此进程序列分配资源,每个进程都可以完成资源分配;反之,系统处于不安全状态。如果满足系统安全性条件,则可以分配;否则暂不分配,以免系统进入不安全状态。

综上所述,本文借用安全性检查算法,将剩余未分配的空闲资源 K 作为递归的边界条件,不断地寻找合法的节点进入请求集,判断下一个节点进入的合法性可以确定当前节点进入的合法性,进而确定出一组合理的安全序列防止死锁问题的发生。这种思想在处理多用户操作系统资源配置问题中是一个创新点和突破点,值得继续优化和深入研究。

2　银行家算法

银行家算法可以有效避免死锁,该算法最初是用于银行贷款[9]。如果存在某个状态序列,能满足所有客户的贷款需求,则该状态是安全的;反之,则可能导致发生死锁。若安全检查结果表明系统处于危险状态,则该请求不合法;反之,请求合法。

2.1 算法数据结构

算法中存在 4 类数据类型。1)第 1 类数据类型称为可利用资源向量,该向量随资源分配、回收动态变化,用含有 m 个元素的一维数组 $\boldsymbol{A}_{\mathrm{v}}$。2)第 2 类数据类型称为最大需求矩阵,用 $\boldsymbol{M}$ 矩阵表示。3)第 3 类数据类型称为分配矩阵,表示每一类资源已分配给每个进程的资源数量,用 $\boldsymbol{A}_1$矩阵表示。4)第 4 类数据类型称为需求矩阵,用 $\boldsymbol{N}$ 矩阵表示。

2.2 银行家算法流程

算法包括系统分配资源流程及系统安全性检查流程。

1)系统分配资源步骤:

a)当满足 $\boldsymbol{R}_i \leqslant \boldsymbol{N}_{ij}$ 条件时,执行下一步操作,否则程序返回 false;

b)初始化假设 $\boldsymbol{F}_i$ = false,满足一定条件则置为 true;

c)假设系统试分配及安全性检查通过,则可以进行分配,分配算法为

$$\boldsymbol{A}_{\mathrm{v}} := \boldsymbol{A}_{\mathrm{v}} - \boldsymbol{R}_i$$
$$\boldsymbol{A}_1{}^i := \boldsymbol{A}_1{}^i + \boldsymbol{R}_i$$
$$\boldsymbol{N}_i := \boldsymbol{N}_i - \boldsymbol{R}_i$$

2)系统安全性算法步骤:

a)设置 2 个向量:工作向量与完成向量。$\boldsymbol{W}$ 向量即工作向量,表示系统能够提供给各个进程的资源数量;$\boldsymbol{F}$ 向量即完成向量,表示系统能够顺利地为每个进程完成分配。初始化假设 $\boldsymbol{F}_i$ = false,满足一定条件则置为 true。

b)当某个进程满足条件 $\boldsymbol{F}_i$ = false 以及 $\boldsymbol{N}_{ij} < \boldsymbol{W}_j$,则跳转到步骤 c),否则跳转到步骤 d)。

c)当进程 P_i 获得资源后,可顺利执行,直至完成,并释放该进程分配得到的资源,$\boldsymbol{W}_j = \boldsymbol{W}_j + \boldsymbol{A}_1{}^{ij}$;$\boldsymbol{F}_i$ = true,继续执行步骤 b)。

d)如果所有进程的 $\boldsymbol{F}_i$ = true 都完成,则系统处于安全状态,否则系统将处于一个不够安全的状态,此时不能完成资源分配工作。

本文提出的基于银行家算法的网络爬虫资源分配算法一个重要的特点是将银行家算法中核心的递归思想引入到请求集生成过程中。通过测试每个纳入请求集数组是否合法来决定是否允许当前节点进入。合法指的是当前节点进入之后可以保障下一个节点进入请求集。这种递归思想可称为全局递归思想或完全递归,消耗时间和系统资源较多。局部递归算法指的是:对经过相应初始化处理的请求集求差集,如果请求集不能覆盖系统所有节点,则只对未被表示的节点递归,最后纳入请求集。在初始化请求集的过程中,通过初始化请求集向量 $\boldsymbol{N}$ 的位置节点,可以减少初始化节点的差集重复率从而降低请求集长度。

2.3 算法设计与实现

采用 Visual C++ 6.0 设计 Windows 界面程序。界面分为三部分:参数输入区域、资源分配区域和结果显示区域。用户在参数输入区域输入可利用资源向量 $\boldsymbol{A}_{\mathrm{v}}$ 等信息,单击动态添加按键添加进程资源,单击安全检查,可以检查当前所有进程对当前资源的申请是否安全,是否会使系统因资源不足或其他原因进入不安全状态从而不能完成分配。如果进程申请资源安全,软件则提示安全,否则提示不安全。

软件可以支持最多 10 类资源进行分配,如图 1 所示。可利用资源向量为[5, 5, 5, 5, 5, 5, 5, 5, 5, 5],输入进程名称 process0,输入 process0 最大需求矩阵为[1, 1, 2, 1, 1, 1, 1, 1, 1, 1], process0 分配矩阵为[0, 0, 0, 0, 0, 0, 0, 0, 0, 0],需求矩阵为[1, 1, 2, 1, 1, 1, 1, 1, 1, 1],完成输入之后单击“动态添加”按键添加进程成功。单击“安全检查”按键,即提示系统状态安全,可以执行分配操作。

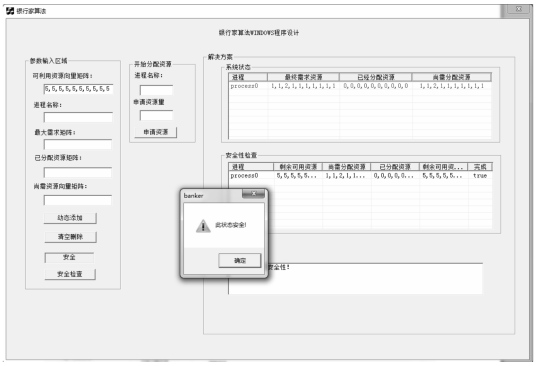

图 1 软件安全性检查

Fig. 1 Software security check

开始执行分配,在进程名称文本框输入“process0”以及申请资源量。假设输入的申请资源量为[1, 1, 1, 1, 1, 1, 1, 1, 1, 1],单击“申请资源”,软件弹出“资源分配后安全,可以分配”的消息提示框,表明输入的资源量可以满足申请要求并执行申请请求。单击确认,进程 process0 已经分配资源为[1, 1, 1, 1, 1, 1, 1, 1, 1, 1],尚需分配资源为[0, 0, 1, 0, 0, 0, 0, 0, 0, 0],表明 process0 申请资源成功。此时系统可用资源数向量由原来的[5, 5, 5, 5, 5, 5, 5, 5, 5, 5]变为[4, 4, 4, 4, 4, 4, 4, 4, 4, 4],系统为进程 process0 分配资源成功,如图 2 所示。

图 2　进程分配资源成功

Fig. 2　Allocation successfulness for processes

2.4　实验结果

本文算法和 urllib2 爬虫算法对死锁问题进行测试实验对比。urllib2 是 Python 中处理 HTTP 协议的算法库,可用于监测网络爬虫资源的死锁[16]。设置超时时间为 2 m,当爬虫资源发生死锁后超时,记录该任务的状态为 Locked,完成其他任务后重启该任务;未发生死锁从而完成任务请求,则记录改任务的状态为 Completed,并记录等待时间。urllib2 爬虫算法抓取日志如表 1 所示,本文算法抓取日志如表 2 所示。

表 1　Urllib2 爬虫算法抓取日志

Table 1　Urllib2 crawler algorithm crawl log

事　件	任务开始时间	等待时间	死锁边界时间
Locked:Request1	19:47:08	0:02:00	0:02:00
Locked:Request2	19:49:52	0:02:00	0:02:00
Locked:Request3	19:52:13	0:02:00	0:02:00
Completed:Request4	19:54:38	0:01:21	0:02:00
Completed:Request5	19:55:59	0:02:00	0:02:00
Completed:Request6	19:58:42	0:01:06	0:02:00
Locked:Request7	19:59:48	0:02:00	0:02:00
Completed:Request8	20:02:01	0:00:41	0:02:00
Completed:Request9	20:02:42	0:00:27	0:02:00
Completed:Request10	20:03:09	0:01:12	0:02:00
Completed:Request11	20:04:20	0:01:58	0:02:00
Locked:Request12	20:06:18	0:02:00	0:02:00
Completed:Request13	20:09:00	0:00:36	0:02:00
Completed:Request14	20:09:36	0:00:36	0:02:00
Locked:Request15	20:10:12	0:02:00	0:02:00
Locked:Request16	20:12:26	0:02:00	0:02:00
Locked:Request17	20:15:17	0:02:00	0:02:00
⋮	⋮	⋮	⋮
Completed:Request100	22:08:43	0:00:36	0:02:00

表 2　本文算法抓取日志

Table 2　Banker´s algorithm crawl log

事　件	任务开始时间	等待时间	死锁边界时间
Completed:Request1	19:47:08	0:00:41	0:02:00
Completed:Request2	19:49:52	0:01:01	0:02:00
Completed:Request3	19:52:13	0:01:05	0:02:00
⋮	⋮	⋮	⋮
Locked:Request36	20:42:29	0:02:00	0:02:00
⋮	⋮	⋮	⋮
Locked:Request48	20:58:38	0:02:00	0:02:00
⋮	⋮	⋮	⋮
Completed:Request100	22:08:43	0:00:36	0:02:00

分别采用 urllib2 爬虫算法和本文算法进行 100 次请求操作,如果在超时时间之内,爬虫资源完成请求操作,则日志中记录该次请求操作为 Completed,反之在所设置的超时时间之内未能请求成功,则日志中记录该次请求操作为 Locked,表示发生死锁。本文算法和 urllib2 爬虫算法死锁边界时间如图 3 所示,纵坐标表示任务等待时间,横坐标表示测试实验的时间范围为 19:47:08—22:08:43,等待时间等于 2 min 表明任务请求发生死锁,小于 2 min 表明任务请求完成。

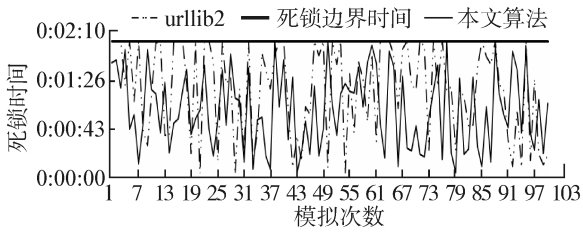

图 3　死锁边界时间

Fig. 3　Deadlock boundary time

Urllib2 爬虫算法等待时间合计为 2 h 9 min 7 s,死锁次数为 30 次,死锁率为 30%;银行家算法等待时间合计为 1 h 38 min 28 s,死锁次数为 2 次,死锁率为 2%,算法对比实验结果如表 3 所示。实验结果表明,本文算法明显降低了死锁率,提高了系统运行效率。传统资源配置算法死锁率高、效率低下。

表 3　实验结果对比

Table 3　Comparison of the experimental results

算　法	等待时间合计	死锁次数	死锁率/%
urllib2 算法	2:09:07	30	30
银行家算法	1:38:28	20	2

3 结束语

本文基于银行家算法,在多用户操作系统中,提出一种新的网络爬虫资源配置策略。这种基于银行家算法的网络爬虫资源配置策略不但提高了网络爬虫资源的利用率,而且能够有效避免死锁的发生。下一步的研究工作可以在目前已有的策略上有针对性、方向性地进一步分析和研究降低死锁率的方法。

参考文献:

[1] RICART G, AGRAWALA A K. An optimal algorithm for mutual exclusion in computer networks[J]. Communications of the ACM, 1981, 24(1): 9-17.

[2] WANG Hong. The study of banker´s algorithm base on experiment[J]. Applied Mechanics and Materials, 2013, 442: 303-308.

[3] HAO Yue, HU Hesuan. Robust deadlock control using shared-resources for production systems with unreliable workstations[C]//2013 IEEE International Conference on Automation Science and Engineering. Madison, USA, 2013: 1095-1100.

[4] YUE Hao, HU Hesuan. A polynomial deadlock avoidance policy for a class of assembly processes based on petri nets [C]//2013 IEEE International Conference on Automation Science and Engineering. Madison, USA, 2013: 1151-1156.

[5] MA Xiaohui, YAN Junya. An improved parallel banker´s algorithm based on petri net[C]//2011 International Conference on Electronic and Mechanical Engineering and Information Technology. Harbin, China, 2011: 1538-1541.

[6] XIAO Xiang, LEE J J. A true 0(1) parallel deadlock detection algorithm for single-unit resource systems and its hardware implementation[J]. IEEE Transaction on Parallel and Distributed System, 2010, 21(1): 4-19.

[7] LOU Lin, TANG Feilong, YOU I, et al. An effective deadlock prevention mechanism for distributed transaction management[C]//2011 Fifth International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing. Seoul, Korea, 2011: 120-127.

[8] REVELIOTIS S A, ROSZKOWSKA E. Conflict resolution in free-ranging multivehicle systems: a resource allocation paradigm[J]. IEEE Transactions on Robotics, 2011, 27(2): 283-296.

[9] LANG S D. An extended banker´s algorithm for deadlock avoidance[J]. IEEE Transactions on Software Engineering, 1999, 25(3): 428-432.

[10] TANG Feilong, YOU I, YU Shui, et al. An efficient deadlock prevention approach for service oriented transaction processing[J]. Computers & Mathematics with Applications, 2012, 63(2): 458-468.

[11] DUDA C, PREY G, KOSSMANN D, et al. Ajax crawl: making Ajax applications searchable[C]//IEEE 25th International Conference on Data Engineering. Shanghai, China, 2009: 78-89.

[12] PENG Zhaomeng, HE Nengqiang, JIANG Chunxiao, et al. Graph-based Ajax crawl: mining data from rich Internet applications[C]//2012 International Conference on Computer Science and Electronic Engineering. Hangzhou, China, 2012: 590-594.

[13] 杨梅, 滕少华. 基于死锁避免的资源安全分配算法[J]. 计算机工程与设计, 2011, 32(1): 40-43.
YANG Mei, TENG Shaohua. Improved safe resource allocation algorithm based on deadlock avoidance[J]. Computer Engineering and Design, 2011, 32(1): 40-43.

[14] 陈昊成. 基于网格计算的资源管理与分配系统的设计与实现[D]. 哈尔滨: 哈尔滨工业大学, 2010.
CHEN Haocheng. Design and implementation of resource management and allocation system based on grid computing [D]. Harbin, China: Harbin Institute of Technology, 2010.

[15] 章韵, 汤楠. 服务计算中避免死锁和活锁的资源分配算法[J]. 微电子学与计算机, 2010, 27(12): 69-73, 77.
ZHANG Yun, TANG Nan. Deadlock and livelock free resource allocation algorithm in service oriented computing [J]. Microelectronics & Computer, 2010, 27(12): 69-73, 77.

[16] SHETTY K S, BHAT S, SINGH S. Symbolic verification of web crawler functionality and its properties[C]//2012 International Conference on Computer Communication and Informatics. Coimbatore, India, 2012: 1-6.

作者简介:

王庆红,男,1976 年生,高级设计师,技术情报所所长,主要研究方向为电力系统运行、规划与设计以及企业情报系统建设与管理。

李广凯,男,1975 年生,副教授,博士,主要研究方向为电力系统企业技术情报咨询及直流输电技术。

周育忠,男,1974 年生,高级工程师,主要研究方向为行业情报系统建设、运维管理和资源整合。