序列关联并行挖掘算法研究*)

李庆华　赵　峰
（国家高性能计算中心　武汉430074）（华中科技大学计算机科学与技术学院　武汉430074）

Parallel Data Ming Algorithm of Sequential Associations

LI Qing-Hua　ZHAO Feng
(National High Performance Computing Center, Wuhan 430074)
(College of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan 430074)

Abstract　Ming sequential associations is becoming increasing essential in many scientific and commercial domains. Developing parallel algorithm becomes quite challenging depending on enormous size of available dataset and possibly large number of mined associations, the nature of input data and the timing constraints imposed on the desired associations. In this paper, we discuss several different parallel algorithms that cater to various situations to speed up the current mining process.

Keywords　Data mining, Sequential associations, Parallel algorithm

如何从海量序列数据集中挖掘出序列关联(Sequential Associations)是当今科学计算和商业数据挖掘领域中一个十分重要的研究课题。人类社会的日益电子化使数据集的数量、种类和规模都在不断增大，数据集的增大导致传统挖掘算法挖掘出的序列关联大规模增多，因而如何从大量候选集中挖掘出有效序列关联面临着新的挑战。序列输入数据固有的特性、期望序列关联相关的时间约束(timing constraints)和海量数据集的结合以及数据仓库多维模型的出现使得传统的串行挖掘算法不能满足快速、高效挖掘序列关联的要求。目前序列关联算法分布并行的研究多数是对传统关联挖掘算法的分布并行化，典型算法有：Count 分布算法[1]，Data 分布算法[2]，Candidate 分布算法[3]等。本文首先介绍一种新型的基于哈希树(Hash-tree)的序列关联挖掘算法，然后就输入数据集和候选集的不同情况提出几种将基于哈希树的序列关联挖掘算法并行化的方式，以便提高挖掘速度。

1　有关定义

为使本文能在概念上自包含，现给出所有可能涉及的定义。

定义1　非空集合 $I=\{i_1,i_2,i_3,\cdots,i_m\}$ 称为项集，其中 i_k 称为项[4]。

定义2　序列是项集的有序表，记为 $a=a_1\rightarrow a_2\rightarrow\cdots\rightarrow a_n$，其中 $a_k\subset I(k=1,\cdots,n)$。含有 k 个项的序列长度为 k，称为 k 序列($k=\sum|a_i|$)[4]。

定义3　序列模式也称序列关联，可表示成如下形式：

when A occurs⇒

B occurs within some certain time

定义4　序列模式相关的限制称为约束。约束可限制：①整个序列；②序列元素；③一个元素到另一个元素的转换。

定义5　整个序列中所允许的事件最晚出现和最早出现的时间差称为最大跨距(Maximum Span)，简记为 ms[5]。

定义6　任一事件集中事件出现的最晚时间和最早时间的差值称为事件集窗口尺寸(Event-set Window Size)，简记为 ws[5]。

定义7　某一事件集中最晚出现事件的时间和该事件集直接前序事件集中最早出现事件的时间的差称为最大间隙(Maximum Gap)，简记为 xg[5]。

定义8　某一事件集中最早出现事件的时间和该事件集直接前序事件集中最晚出现事件的时间的差称为最小间隙(Minimum Gap)，简记为 ng[5]。

例1：有序列((A)(C,B)(D)(F,E,G))，图1表示 ms，ws，xg，ng 四种时间约束。

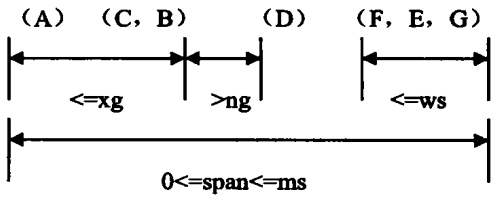

图1　xg、ng、ws、ms 示意图

定义9　判断序列模式是否有效的参数称为模式浓度(Pattern's strength)。

模式浓度通常基于在给定数据下模式发生的频率来计算，模式发生的频率有5种计算方式：COBJ、CWIN、CWIN-MIN、CDIST_O、CDIST。本文中仅用到 CWIN 方式，CWIN 的计算方法是：如果序列在每个跨距窗口出现一次，那么模式发生次数就加1，跨距窗口是一个持续时间和 ms 相等的窗口。若干连续的跨距窗口在它们各自开始和结束时间之间有相同的时间单元差距。它们跨越整个时间持续期间所有的对象，并将所有对象的个数累计。

如果模式出现得足够频繁，我们就称它为有效模式或频繁模式。决定模式是否有效的参数有：①支持度(Support)；②可信度(Confidence)；③重要性(Significance)；④覆盖度(Coverage)。

*)国家高性能计算基金项目(00305)资助。

2 基于 Hash-tree 的串行算法

基于 Hash-tree[6]的序列关联挖掘算法是在 GSP 算法[7]上改进的一种串行挖掘算法，其工作原理和 GSP 算法很相似，算法在序列事件上反复运算，每个反复包含两个阶段：①从长度为(k-1)的频繁序列集拼接和修剪(join-and-prune)而得到长度为 k 的候选集；②在对象时间线(object time-lines)上计算候选集出现的次数，得到长度为 k 的频繁集。

串行算法如下，其中 F_k 表示长度为 k 的频繁集；C_k 表示长度为 k 的候选集，C_k 存储在哈希树中。该算法对 GSP 算法做了改进。

基于 hash-tree 的挖掘算法

```
输入：序列数据
输出：频繁集
形成频繁集 F1，每个 F1仅含一个事件；
 k=2;
while (Fk-1不为空)
{
Ck=join-and-prune(Fk-1)
for 每个对象时间线
{
repeat
 /* hash-tree 遍历 */
  在哈希树上遍历时间线直到叶节点，来寻找当前可能的候选集，
/* 叶节点 */
  在叶节点上计算候选集的出现次数
until
}
通过在 Ck 中保留最大候选集来形成 Fk
}
/* join-and-prune 函数 */
Ck join-and-prune(Fk-1)
{
 if (k=2)
{
  for F1中的每对事件 e1,e2
   形成 c=(e1)(e2);
   将 c 加到 Ck 中;
  for F1中的每对满足 e1<e2的事件 e1,e2
   形成 c=(e1 e2);
   将 c 加到 Ck 中;
}
else
{
 for 每个 Fk-1中的 c1
   在 Fk-1中寻找 c2,c2满足的条件是：删除 c1的第一个事件，删除 c2的
   最后一个事件，可以得到相同的子序列;
for 每个 c2,
    {
    形成 k-序列 c;
    /*
    c 满足的条件是：如果(e)是 c2的最后的事件集，c=(c1)(e)。否则
    如果 e 是 c2的最后的事件，c=(c1 e)。
    */
    if (每个长度为 k-1子序列 c 在 Fk-1中)
       将 c 加到 Ck 中;
   }
}
}
```

3 并行挖掘算法

上节所述基于 hash-tree 的挖掘算法能提高挖掘效率，但不适于哈希树过深或分支过多的情况。目前输入序列数据一般具有海量和高维的特性。当海量数据生成 Hash 树时，对象数量过多，挖掘算法用于计算数据集的时间消耗会很大。高维数据的事件数目大，挖掘生成过多的候选集，对存储空间要求很高，因而并行算法是有效发现序列模式的最佳途径。

3.1 EVE 算法

事件分布(EVE)算法的基本思想是：将输入数据和候选哈希树分别分布和复制到所有处理器上。候选集在各个处理器上线性生成。根据对象数量、时间线的长度、ms 等的不同，EVE 算法又有一些差异。

(1)EVE-S 算法　简单事件分布算法(Simple Event Distribution Algorithm)适用于具有较小时间线和相对较多对象的情况。该算法分布输入数据时，尽量让处理器得到其上每个对象整个的时间线。算法如图2所示。

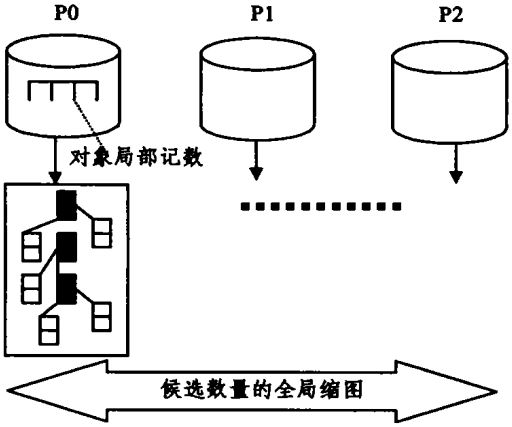

图2　EVS-S 算法

(2)EVE-R 算法　部分数据复制事件分布算法(Event Distribution with Partial Data Replication)适用于相对较少数量的对象(对象数小于处理器个数)，但每个对象有大的时间线，同时 ms 值相对较小的情况。该算法基本思想是：每个对象的时间线穿过不同的处理器；跨距窗口很小，不会跨过两个处理器；序列发生的次数在跨距窗口内计算，当一个处理器没有足够的数据来计算其中的序列次数时，它会向与它相邻的处理器收集数据。算法如图3所示。

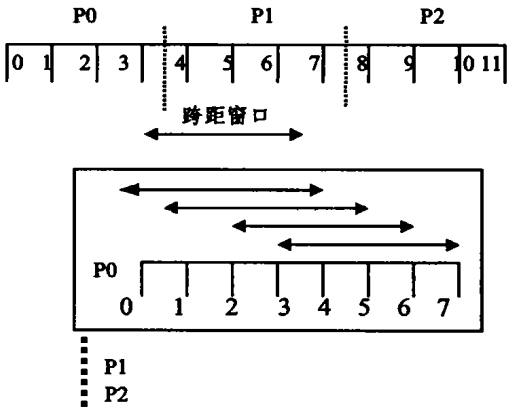

图3　EVE-R 算法

(3)EVE-C 算法　复杂事件分布算法(Complex Event Distribution Algorithm)适用于相对较少数量的对象(对象数小于处理器个数)，但每个对象有大的时间线，同时 ms 值较大，对象时间线要跨过多个处理器，跨距窗口也要跨过2个以上的处理器的情况。该算法基本思想有两种：①将数据复制到处理器，让每个处理器都有完整的跨距窗口。这种想法和 EVE-R 算法类似，但它要求更多的磁盘空间来支持复制所有信息，其优点在于节省了通信开销。②不复制数据。但有两种情况需要考虑：第一，在其他处理器上的跨距窗口可能导致单个处理器上的序列出现次数的计算，也就是重复计算问题；第二，由于单个处理器上没有足够的数据，可能导致一些序列的出现被忽略，也就是遗漏计算问题。因此，EVE-C 算法需要针对不同的情况具体对待。

3.2 EVECAN 算法

在EVE算法中，我们假设候选集复制到所有处理器上，但实际处理问题时，候选集通常会很大，为了降低处理器内存需求和避免遗漏计算序列，提出了事件候选集分布算法(Event and Candidate Distribution)。

EVECAN算法基本思想是：输入数据的划分和EVE算法类似。候选集存在分布的哈希表中，哈希分布本着让所有处理器上候选集数目相等的原则。EVECAN算法中处理器需要处理输入数据和候选集，当ms的值较小时，可将候选集静态置于处理器上，建立本地哈希树，而输入数据采取类似于IDD算法的round-robin方式；当ms值较大时，则反之。

4 实验结果及分析

为了分析和比较串行算法和并行算法的性能，我们对某超市自2000年4月5日至2000年6月19日的销售数据应用上述算法，该销售数据共包括132144条数据，1034种商品。并行机采用曙光3000，3个节点。测试数据如表1所示。

表1 各种算法时间开销比较

测试算法	时间开销/分
串行算法	16.8
EVE-S	15.3
EVE-R	15.5
EVECAN	15.1

在大型数据库上进行序列模式挖掘时，由于海量数据和高维模型的出现，算法的开销往往很大。本文提出的并行算法能加速挖掘过程，提高挖掘效率。进一步的工作包括并行算法复杂性研究。

参 考 文 献

1 Hong J. Incremental Discovery of Rules and Structure by Hierarchical and Parallel Clustering In Knowledge Discovery in Database. In: G. Piatestsky-Shapiro and W. J. Frawly, eds. AAAI/MIT Press, 1991

2 Agrawal R, Imielinski T, Swami A. Mining Association Rules Between Sets of Itenms inLarge Databases. SIGMOD'93, ACM, 1993

3 Agrawal R, Shafer J C. Parallel Mining of Association Rules. IEEE Trans. on knowledge and Data Engineering, 1996, 8(6)

4 周斌，吴泉源. 序列模式挖掘的一种渐进算法[J]. 计算机学报，1999, 22(8): 882～887

5 Joshi M, Karypis G, Kumar V. A Universal Formulation of Seuential Patterns: [Technical Repirt N0. 99-021]. Department of Computer Science, University of Minnesota, 1999

6 Joshi M, Karypis G, Kumar V. Parallel Alogrithms for Mining Sequenctial Associations: Issues and Chanllges: [Technical Repirt N0. 01-021]. Department of Computer Science, University of Minnesota, 2001

7 Ahola J. Ming Sequential Patterns. VTT Information Technology, 2001, 5

(上接第113页)

① 限制智能卡用户的范围

·有些智能卡，任何人都可以读取卡上信息，像记录病人姓名和血型的医疗卡，这种智能卡一般不设密码，只要拿到卡的人都可以读取卡上信息。这时卡体本身就是一种保护。

·对于只许持卡人读取信息的智能卡通常采用一种叫PIN(个人识别码)的密码形式来保护卡上的信息，通过键盘输入读卡器，只有PIN码核对正确了，用户才能对该卡进行操作。典型的应用是手机的SIM卡。

·对于只许第三方读取信息的智能卡便只有发卡人才读取卡上信息，比如银行卡。这时这些智能卡由16～32位数字的密码来保护。Java智能卡可以采取这种方式。

② 限制读取Java智能卡信息的方式(只读、可添加、可修改或可擦写)。存储在Java智能卡上的信息一般被划分为若干个部分：只读信息、只可添加的信息、只可更新的信息和无法读取的信息。这样有些密码信息就可以存储在无法读取的存储区域中。

(2)从卡的结构和支持的加密算法来控制

如上所述只有知道密码的人才有权使用Java智能卡，但如果需要通过无线电或电话线将卡上的信息向异地传送，还必须要有额外的防护手段。

防护手段之一就是加密。Java智能卡有加密和解密的功能，使得在传送存储在卡上的信息的同时，也不用担心会发生泄密。

由于Java智能卡带有微处理器，同时又支持对称密钥算法和公开密钥算法，而且它的尺寸大小极便于携带，因此它本身就是网络数据传递和身份认证极佳的安全模块。目前Java智能卡支持DES算法和RSA算法，可以选择适当的加解密算法。这种防范机制可以确保所用的卡和计算机都真实有效，使得几乎没有可能半路窃取传送的信息。

(3)从是否执行指定的代码段来控制

可根据代码段计算出一个散列函数值，并将其附在代码段后面，然后再加密代码段。这样，Java智能卡可通过验证代码段的散列函数值来决定是否执行该代码段，从而避免执行恶意的代码段，躲开可能的攻击。

结束语 本文在论述当前移动代理的安全性问题及现有的移动代理保护方案的基础上，主要提出了采用Java智能卡(JavaCard)来保护移动代理，构建了采用JavaCard的移动代理系统的安全参考模型，同时对Java智能卡自身的安全问题进行了分析，并给出了较为有效可行的方案。

参 考 文 献

1 Hohl F. A Protocol to Detect Malicious Hosts Attacks by Using Reference States: [Technical Report Nr. 09/99]. Faculty of Informatics, University of Stuttgart, Germany, 1999

2 Vigna G. Cryptographic Traces for Mobile Agents. in Mobile Agents and Security Lecture Notes in Computer Science, Springer-Verlag, June 1998

3 Sander T, Tschudin C F. Towards Mobile Cryptography: [Technical Report 97-049]. International Computer Science Institude, Berkeley, 1997

4 Hohl F. Time Limited Blackbox Security: Protecting Mobile Agents From Malicious Hosts. In: Giovanni Vigna, ed. Mobile Agents and Security, LNCS 1419, Springer, 1998. 44～60

5 Wilhelm U G, Staamann S M, Buttyan L. A Pessimistic Approach to Trust in Mobile Agent Platforms. IEEE Internet Computing, September/October 2000

6 丁建国，柳惠琳，陈涵生，白英彩. 移动代理的一种安全认证机制. 计算机工程，2001, 27(2)

7 朱向华，万燕，孙永强. 移动代理系统的安全机制. 计算机科学，2001, 28(1)

8 李增智，李刚，韩冬，王志文. 智能卡的新发展——JavaCard技术综述. 计算机科学，2001, 28(7)