维普资讯 http://www.cqvip.com

基于图结构的候选序列生成算法*)

郭 平 刘潭仁

(重庆大学计算机学院 重庆400044)

摘 要 先生成候选序列再判断候选序列是否为频繁序列,最后获得频繁序列是序列数据挖掘中基于候选序列挖掘算法的一般结构,如 Apriori 类算法,GSP 算法,SPADE 算法等。因此,研究候选序列生成算法具有普遍意义。本文首先研究了序列数据集(序列数据库)与图结构间的关系,证明了一个序列是频繁序列的必要条件是该序列对应于一个完全子图。以此为基础提出了基于图结构的候选序列生成算法,文中给出了算法正确性证明。在 T25I10D10K 和 T25I20D100K 数据集上的挖掘实验表明在本文提出的候选序列生成算法上进行挖掘比用 Apriori 算法进行挖掘的效率更高。

关键词 序列模式,数据挖掘,序列模式挖掘

Graph-Based Candidate Frequent Patterns Generating Algorithm

GUO Ping LIU Tan-Ren

(College of Computer Science, Chongqing University, Chongqing, 400044)

Abstract In candidate-sequence-based mining algorithms, the common procedure is first Generating the candidate frequent patterns, then identifying the frequent patterns based on the candidate frequent patterns, last getting the frequent patterns, such as the apriori-like algorithm, GSP algorithm, SPADE algorithm and so on. Thus there is universal meaning to research the candidate frequent patterns generating algorithm. In this article, firstly we investigate the relationship between sequence data set (sequence database) and graph structure and prove that the requirement of a sequence to be a frequent sequence is that there is a corresponding complete subgraph to the sequence in the graph. Then a graph-based candidate frequent patterns generating algorithm is proposed and the correctness of the algorithm is proved in the article. Lastly the algorithm is applied to T25I10D10K and T25I20D100K data set and comparing with the apriori algorithm it higher the efficiency of sequence data mining.

Keywords Sequence pattern, Data mining, Sequence pattern mining

1. 引言

序列挖掘是一类重要的数据挖掘问题,它有着广泛的应用前景并且已提出了许多算法。这些算法基本上可以分为两类。第一类是以高速内存中的操作代替对序列数据库的频繁读写。这类算法将序列数据库中的数据读出并经过压缩后存储在内存特定的数据结构中,该结构保持了数据之间的相关性。挖掘过程中,通过对内存数据结构的操作代替对序列数据库的操作从而提高算法的效率。如 FP-growth 算法[1]即是将序列数据库中的数据读出后存储在树结构中,再通过对频繁树的操作获得频繁序列。又如,SPADE 算法[2]利用内存数组(或链表)存储从序列数据库中读出的数据,再通过对数组的操作获得频繁序列。

第二类是直接读写序列数据库。这类算法首先需要生成频繁序列的候选序列,再通过查询序列数据库以确认候选序列是否为频繁序列,最终获得频繁序列。在确认过程中,需要搜索序列数据库,这是这类算法最大的不足。因此,大量的算法研究是如何使生成的候选序列尽可能少,从而减少确认过程的次数,提高算法的效率。如 Apriori all、Apriori some、Dynamic Some[3]、GSP[4]等算法即是如此。

在理论上,第一类算法不需要生成候选序列。但是实际中这是不可避免的。例如,FP-growth 算法,其生成的频繁树中包含的节点虽然是频繁项目(长度为1的频繁序列),但树中并不是每一条路径上的节点的组合都是频繁序列,因此频繁树本身就是候选序列的集合(称候选集)。再有,SPADE 算法中,也需要生成候选序列,只是它对候选序列是否为频繁序列的判断过程不需要访问序列数据库而是直接访问内存中的数据结构。因此,研究候选序列生成算法具有普遍意义。

本文的研究属于第二类。首先将数据库中的数据读出并形成图结构,其次证明了一个序列是频繁序列的必要条件是它对应于图中的一个完全子图。基于此,给出了基于完全图的候选序列生成算法并比较了该算法与 Apriori 算法中的候选序列生成算法以及它们对挖掘效率的影响。最后,在数据集 T25I10D10k 和 T25I20D100k[1]上进行了候选集生成与挖掘测试。

为叙述方便,这里先给出几个相关的概念。

定义1 可辩识的对象 $a_j(1\leqslant j\leqslant m)$称为项目,它们的集合 $I=\{a_1,a_2,\cdots,a_m\}$称为项目集。$t\in 2^I$ 称为项目集 I 上的序列,$D=\{t|t\in 2^I\}$称为待挖掘的数据集或序列数据库。

定义2 给定常数 ξ 称为最小支持度。对序列 η

$$S(\eta)=\frac{|\{t|\eta\subseteq t\wedge t\in D\}|}{|D|} \tag{1}$$

称为 η 在序列数据库 D 上的支持度。其中:|X|为集合 X 中元素的个数。如果

$$S(\eta)\geqslant\xi \tag{2}$$

称 η 为 D 上的频繁序列模式或频繁序列。

序列挖掘即是要找出序列数据库 D 中的所有频繁序列。

2. 序列数据库对应的矩阵和图

通过下述的算法 CTG 可以由搜索序列数据库 D 生成与它对应的图 G 和矩阵 T。

算法 CTG

*)本文的研究得到国家十五攻关项目(编号:2002BA107B)资助。

维普资讯 http://www.cqvip.com

输入:序列数据库 D,最小支持度 ξ;

输出:图 G,矩阵 T;

步骤:

//生成矩阵 T

第1步　统计 D 中出现的项目并计算每个项目的支持度。将支持度不小于 ξ 的项目作为矩阵 T 的行和列;

第2步　矩阵 T 的元素值初始化为0;

第3步　对 D 中的每一个序列,在 T 中记录各项目对出现的次数。设$\{x_1,\cdots,x_n\}$是 D 中的序列,将 T 中 x_i 所对应的行与 x_j 所对应列交叉处的元素(记为:$T(x_i,x_j)$)的值加1,$1\leqslant i,j\leqslant n$。

第4步　修改 T 的元素的值。设 x,y 是 T 的行列对应的项目,重新给 T(x,y)赋值为:

$$T(x,y)=\begin{cases}0 & \dfrac{T(x,y)}{|D|}<\xi\\ 1 & \text{其它}\end{cases}\qquad(3)$$

//生成图 G

第5步　生成以 T 为邻接矩阵的图 G。图 G 的顶点对应于 D 中的项目。

定义3　由算法 CTG 生成的矩阵 T 与图 G 分别称为序列数据库 D 对应的矩阵和图。并用 T(x,y)记矩阵中与项目 x 对应的行和项目 y 对应列的交叉处的元素。

可以证明,对于给定的最小支持度 ξ,由序列数据库 D 按算法 CTG 生成的矩阵和图是唯一的。

表1　序列数据库 D

I	t_I
1	a,b
2	a,b,c,d
3	b
4	c,d,e,f
5	a,b,c,f
6	a,b,e,f
7	b,f
8	c,d,f
9	a,b,c,d
10	a,b,d

作为例子,当 ξ=0.3,由表1所示序列数据库 D 生成的矩阵和图如图1所示。

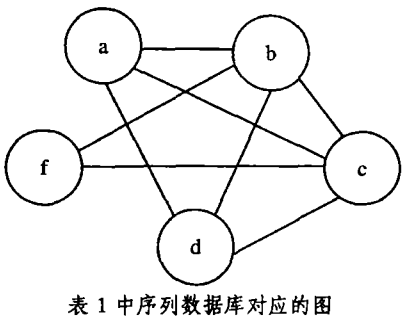

表 1 中序列数据库对应的图

	a	b	c	d	f
a	1	1	1	1	0
b	1	1	1	1	1
c	1	1	1	1	1
d	1	1	1	1	0
f	0	1	1	0	1

表 1 中序列数据库对应的矩阵

图1

3. 频繁序列与完全图

本节基于序列数据库 D 和由它生成的表 T、图 G 来讨论频繁序列与图的关系。对于频繁序列与它的子序列,有结论:

定理1[5]　频繁序列的子序列也是频繁序列。

由算法 CTG 生成的图 G 的边与顶点的下述性质可以直接由算法获得。

性质1　G 的顶点与 D 中长度为1的频繁序列一一对应。

性质2　G 中有边相连的顶点对与 D 中长度为2的频繁序列一一对应。

性质3　D 的频繁序列中的任意两个项目都对应于 G 中有边相连接的顶点。

由定理1和上述性质,可以得到如下结论:

定理2　D 的任意频繁序列都有 G 中的完全子图与它对应。

证明:假设 $S=\{s_1,\cdots,s_n\}$是 D 中的频繁序列。由定理1和性质1,记与 S 中的项目一一对应的顶点集合为 $V=\{v_1,\cdots,v_n\}$,不失一般性,假设 v_i 与 s_i 对应,$1\leqslant i\leqslant n$。下面证明 G 中以 V 为顶点集合的子图 G' 是完全图。

$\forall v_i,v_j\in V$,与它们对应的 S 中的项目为 s_i,s_j,$\{s_i,s_j\}$构成 S 的长度为2的频繁序列,因此,v_i 与 v_j 间有边相连接。即 G' 中任意两个顶点间都有边相连,故 G' 为完全图。定理得证。

值得指出的是,G 中的完全子图并不一定与 D 中的频繁序列相对应。例如,图1中由顶点{a,b,c,d}构成的子图是完全图,但是项目集{a,b,c,d}并不是序列数据库 D(表1)的频繁序列。我们可以得到:

定理3　序列 S 是 D 的频繁序列的必要条件是 G 中以 S 中项目为顶点的子图是完全图。

4. 基于完全图的候选序列生成算法

由定理3,将与 G 中完全子图对应的序列作为 D 上频繁序列的候选序列得到下述算法 SCG。算法中 L_k 记长度为 k 的频繁序列,C_k 记长度为 k 的候选序列,V 记 G 的顶点集合,E 记 G 的边集合。

算法　SCG//生成长度为 k+1的候选序列,$k\geqslant 0$

输入:图 G,候选序列长度 k+1,长度为 k 的频繁集合 L_k

输出:长度为 k+1的候选序列集合 C_{k+1}

步骤:

```
step1:if k=1 then C1={v|v∈V};return C1;
step2:if k=2 then C2={{u,v}|u,v∈V and uv∈E};return C2;
step3:Ck+1=φ;
step4:forall S1,S2∈Lk and |S1∩S2|=k-1 do {
          {u,v}=(S1∪S2)-(S1∩S2);
          if {u,v}∈(L2 then
            Ck+1=Ck+1∪(S1∪S2);//将(S1∪S2)作为长度为
            k+1的候选序列
          }
step5:return Ck+1
```

关于算法的正确性,显然只需证明如下定理:

定理4　C_k 中的任意序列都有 G 中的完全子图与它对应。

证明:使用数学归纳法。

当 k=1,2时,由关于图 G 的性质1,2可知定理的结论正确。

当 k=3时,对$\forall S\in C_3\neq\phi$,根据算法就有 $S_1,S_2\in C_2$,使得:

维普资讯 http://www.cqvip.com

$|S_1 \cap S_2|=1$和$S=(S_1 \cup S_2)$

不妨设$S_1=\{a,b\}$,$S_2=\{b,c\}$,则有$S=\{a,b,c\}$。由于$S_1,S_2 \in C_2$,与a,b对应的顶点V_a,V_b间有边相连接,与b,c对应的顶点V_b,V_c间有边相连接。又由算法可知:与

$\{a,c\}=(S_1 \cup S_2)-(S_1 \cap S_2)$

对应的顶点V_a,V_c间有边相连接。即:以V_a,V_b,V_c为顶点的子图中任意两顶点间都有边相连接,这个子图就是与S对应的完全子图。k=3时,定理的结论成立。

假设k=n-1时定理的结论成立。

对k=n时,$\forall S \in C_n \neq \phi$,根据算法就有$S_1,S_2 \in C_{n-1}$,使得:

$|S_1 \cap S_2|=n-2$和$S=(S_1 \cup S_2)$

不妨设$S_1=\{u\} \cup (S_1 \cap S_2)$,$S_2=\{v\} \cup (S_1 \cap S_2)$,则有$S=\{u,v\} \cup (S_1 \cap S_2)$。

对$\forall$ a,b∈S,分下述四种情况:

(1)当a=u,b=v时,由算法中S的生成过程可知与a,b对应的顶点间有边相连接。

(2)当a=u,b∈$(S_1 \cap S_2)$时,有$\{a,b\} \subset S_1$,由假设,与a,b对应的顶点间有边相连接。

(3)当a=v,b∈$(S_1 \cap S_2)$时,有$\{a,b\} \subset S_2$,由假设,与a,b对应的顶点间有边相连接。

(4)当$\{a,b\} \subset (S_1 \cap S_2)$时,由于$(S_1 \cap S_2) \subset S_1$,因此$\{a,b\} \subset S_1$,由假设,与a,b对应的顶点间有边相连接。

上述四种情况穷举了{a,b}所有可能的取值,所以与S中任意两个项目对应的顶点间都有边相连接,即S与G的一个完全子图对应。

由归纳原理可知,定理得证。

5. 算法分析与实验

在以候选序列为基础的序列挖掘算法中,候选序列的大小和生成算法的复杂度直接影响挖掘算法的效率。为评价SCG算法,我们将Apriori算法中候选序列生成过程替换为SCG算法后仍称为SCG算法,这里仅对SCG算法与Apriori算法进行比较分析。

Apriori算法中由L_k生成L_{k+1}的候选序列C_{k+1}时须进行连接和剪枝两个步骤。

连接步:

$$C'_{k+1}=\{S_1 \cup S_2 \mid S_1,S_2 \in L_k \text{ 且 } |S_1 \cap S_2|=k-1\} \quad (4)$$

剪枝步:

$$APR\_C_{k+1}=\{S \mid S \in C'_{k+1} \text{且 S 的长度为 k 的任意子序列在 } L_k \text{ 中}\} \quad (5)$$

在剪枝步中须对$|C'_{k+1}|$个序列进行处理,每个序列含有k+1个长度为k的子序列,其中用于连接的两个序列肯定在L_k中,因此只需对其余的k-1个序列判定它们是否在L_k中。判定过程如果通过在L_k中查找实现,并假设L_k是排好序的(可以通过改变连接的顺序来保证),可知剪枝步需要的计算次数为:

$$|C'_{k+1}| \times (k-1) \times \log_2(|L_k|) \quad (6)$$

如果SCG的实现中采用类似的方法,即候选序列的生成也被看作连接与剪枝,将这两步合并后有了:

$$SCG\_C_{k+1}=\{S_1 \cup S_2 \mid S_1,S_2 \in L_k \text{ 且 } |S_1 \cap S_2|=k-1 \text{和} (S_1 \cup S_2)-(S_1 \cap S_2) \in L_2\} \quad (7)$$

其计算量为:

$$|C'_{k+1}| \times \log_2(|L_2|) \quad (8)$$

其中C'_{k+1}的含义同于(6)式。

由(6)、(8)式,当$|L_2|<|L_k|^{k-1}$时,SCG产生SCG_C_{k+1}较Apriori生成APR_C_{k+1}具有更小的计算量。

另一方面,比较(5)与(7)式,当k>3时:

$$APR\_C_{k+1} \leqslant SCG\_C_{k+1} \quad (9)$$

即,Apriori算法产生的候选序列数目不会超过SCG算法产生的候选序列数目。这表明SCG在k>3时将比Apriori花费更多的时间去搜索序列数据库以确定频繁序列。然而,这种花费可以被Apriori算法剪枝步的花费所抵消,下面的实验证实了这一点。

作为算法SCG的验证和与Apriori进行挖掘效率比较,我们在T25I10D10k和T25I20D100k数据集进行了算法对比实验。T25I10D10k和T25I20D100k数据集的基本参数如表2。对比实验的结果如表3,表4。

表2 T25I10D10k和T25I20D100k数据集的基本参数

	T25I10D10k	T25I20D100k
项目个数	996	5158
序列数	10000	100000
序列最大长度	51	51
序列最短长度	10	10
序列平均长度	28.7	29.0
项目平均出现次数	288.2	562.3

表3 数据集T25I10D10k的处理结果

序列长度	支持度ξ=0.8%				支持度ξ=1%			
	候选序列数		频繁序列数	挖掘时间比 SCG/Apriori	候选序列数		频繁序列数	挖掘时间比 SCG/Apriori
	SCG	Apriori			SCG	Apriori		
1	948	948	948		920	920	920	
2	2093	897756	2093		1022	845480	1022	
3	2547	2547	1884	0.757	1174	1174	845	0.875
4	2087	2017	1772	0.818	953	929	881	0.884
5	1532	1414	1279	0.782	873	869	845	0.800
6	881	807	777	0.776	678	674	667	0.769
7	435	428	428	0.750	409	403	391	0.792
8	185	185	185	0.700	177	176	175	0.727
9	57	57	57	0.667	56	56	53	1.000
10	11	11	11	1.000	10	7	7	1.000
11	1	1	1	1.000	0	0	0	0
12	0	0	0	0				

维普资讯 http://www.cqvip.com

序列长度	支持度 ξ=2%				支持度 ξ=3%			
	候选序列数		频繁序列数	挖掘时间比 SCG/Apriori	候选序列数		频繁序列数	挖掘时间比 SCG/Apriori
	SCG	Apriori			SCG	Apriori		
1	688	688	688		407	407	407	
2	40	472656	40		0	0	0	0
3	43	43	9	1.000				
4	3	2	0	1.000				
5	0	0	0					

表4 数据集 T25I20D100k 的处理结果

序列长度	支持度 ξ=0.8%				支持度 ξ=1%			
	候选序列数		频繁序列数	挖掘时间比 SCG/Apriori	候选序列数		频繁序列数	挖掘时间比 SCG/Apriori
	SCG	Apriori			SCG	Apriori		
1	1195	1195	1195		822	822	822	
2	1469	1426830	1469		794	674862	794	
3	2214	2214	1934	0.992	1121	1121	994	0.991
4	2386	2333	1879	1.009	1216	1214	1047	0.984
5	1774	1739	1695	0.991	1074	1074	872	0.973
6	1237	1237	1145	0.961	691	676	640	0.997
7	624	622	609	0.968	372	339	89	1.066
8	247	247	247	0.980	21	18	18	1.167
9	69	69	68	0.966	2	2	2	1.000
10	12	12	12	1.000	0	0	0	0
11	1	1	1	1.000				
12	0	0	0	0				

序列长度	支持度 ξ=1.2%				支持度 ξ=1.4%			
	候选序列数		频繁序列数	挖掘时间比 SCG/Apriori	候选序列数		频繁序列数	挖掘时间比 SCG/Apriori
	SCG	Apriori			SCG	Apriori		
1	575	575	575		394	394	394	
2	325	330050	325		170	154842	170	
3	406	406	283	0.994	233	233	178	1.000
4	317	316	301	0.991	253	253	253	1.000
5	275	275	252	0.980	252	252	252	0.990
6	168	168	168	0.984	168	168	138	0.986
7	72	72	72	0.964	53	43	37	1.222
8	18	18	18	1.000	9	9	9	1.000
9	2	2	2	1.000	1	1	1	0
10	0	0	0	0	0	0	0	

序列长度	支持度 ξ=1.6%				支持度 ξ=1.8%			
	候选序列数		频繁序列数	挖掘时间比 SCG/Apriori	候选序列数		频繁序列数	挖掘时间比 SCG/Apriori
	SCG	Apriori			SCG	Apriori		
1	281	281	281		213	213	213	
2	102	78680	102		43	45156	43	
3	177	177	177	1.000	40	40	28	1.000
4	253	253	162	0.991	18	12	0	1.600
5	147	147	99	0.985	0	0	0	0
6	32	30	0	1.000				
7	0	0	0					

序列长度	支持度 ξ=2%				支持度 ξ=3%			
	候选序列数		频繁序列数	挖掘时间比 SCG/Apriori	候选序列数		频繁序列数	挖掘时间比 SCG/Apriori
	SCG	Apriori			SCG	Apriori		
1	148	148	148		34	34	34	
2	12	21756	12		0	0	0	
3	2	2	1	1.000				
4	0	0	0					

考虑到算法实现的差异，表3与表4中给出的是 SCG 与 Apriori 挖掘时间的比值。对序列长度为2的情况没有给出挖掘时间的比，显然这时 SCG 较 Apriori 花费更少的时间。

从表3与表4中可以看出，SCG 生成的候选序列与 Apriori 生成的候选序列的数量差异较小（长度为2的序列除外）。就挖掘时间比来看，绝大多数情况是小于1的，大于1的情况出现在候选序列数和频繁序列数较小的时候，这时两个算法执行的

（下转第141页）

维普资讯 http://www.cqvip.com

$\sin\Omega\cos\theta+\cos\Omega\cdot\sin\varphi\cdot\sin\theta$；$S_{33}=\cos\Omega\cdot\cos\varphi$。

令 $K_1=\sin\Omega$，$K_2=\sin\varphi$，$K_3=\sin\theta$，$K_4=\cos\Omega$，$K_5=\cos\varphi$，$K_6=\cos\theta$

则 M^T 变为：

$$M^T=\begin{pmatrix} K_5K_6 & -K_3K_5 & K_2 \\ K_3K_4+K_1K_2K_6 & K_4K_6-K_1K_2K_3 & -K_1K_5 \\ K_1K_3-K_2K_4K_6 & K_1K_6+K_2K_3K_4 & K_4K_5 \end{pmatrix}$$

以上转换方法即可以将一个协坐标系统转换到另一协坐标系统。因此，此时就可以利用上述矩阵建立起两个协坐标系统间的关系。在两个系统中都知道了至少三个点的前提下，就可以决定此系统中的所有任意点。这个方法可以由以下形式的方程来表示，这些方程是通过替换前述三个变换 X_C，Y_C 和 Z_C 中的参数和向量而得到。

如下所示：

$$X_{A1}=(X_{B1}-X_C)(K_5K_6)+(Y_{B1}-Y_C)(-K_3K_5)+(Z_{B1}-Z_C)(K_2)$$
$$Y_{A1}=(X_{B1}-X_C)(K_3K_4+K_1K_2K_6)+(Y_{B1}-Y_C)(K_4K_6-K_1K_2K_3)+(Z_{B1}-Z_C)(-K_1K_5)$$
$$Z_{A1}=(X_{B1}-X_C)(K_1K_3-K_2K_3K_6)+(Y_{B1}-Y_C)(K_1K_6+K_2K_3K_4)+(Z_{B1}-Z_C)(K_4K_5)$$
$$X_{A2}=(X_{B2}-X_C)(K_5K_6)+(Y_{B2}-Y_C)(-K_3K_5)+(Z_{B2}-Z_C)(K_2)$$
$$Y_{A2}=(X_{B2}X_C)(K_3K_4+K_1K_2K_6)+(Y_{B2}-Y_C)(K_4K_6-K_1K_2K_3)+(Z_{B2}-Z_C)(-K_1K_5)$$
$$Z_{A2}=(X_{B2}-X_C)(K_1K_3-K_2K_3K_6)+(Y_{B2}-Y_C)(K_1K_6+K_2K_3K_4)+(Z_{B2}-Z_C)(K_4K_5)$$
$$X_{A3}=(X_{B3}-X_C)(K_5K_6)+(Y_{B3}-Y_C)(-K_3K_5)+(Z_{B3}-Z_C)(K_2)$$
$$Y_{A3}=(X_{B3}-X_C)(K_3K_4+K_1K_2K_6)+(Y_{B3}-Y_C)(K_4K_6-K_1K_2K_3)+(Z_{B3}-Z_C)(-K_1K_5)$$
$$Z_{A3}=(X_{B3}-X_C)(K_1K_3-K_2K_3K_6)+(Y_{B3}-Y_C)(K_1K_6+K_2K_3K_4)+(Z_{B3}-Z_C)(K_4K_5)$$

最终的方程集合是：

$$X_{A1}=(X_{B1}-X_C)C_1+(Y_{B1}-Y_C)C_2+(Z_{B1}-Z_C)C_3$$
$$Y_{A1}=(X_{B1}-X_C)C_4+(Y_{B1}-Y_C)C_5+(Z_{B1}-Z_C)C_6$$
$$Z_{A1}=(X_{B1}-X_C)C_7+(Y_{B1}-Y_C)C_8+(Z_{B1}-Z_C)C_9$$
$$X_{A2}=(X_{B2}-X_C)C_1+(Y_{B2}-Y_C)C_2+(Z_{B2}-Z_C)C_3$$
$$Y_{A2}=(X_{B2}-X_C)C_4+(Y_{B2}-Y_C)C_5+(Z_{B2}-Z_C)C_6$$
$$Z_{A2}=(X_{B2}-X_C)C_7+(Y_{B2}-Y_C)C_8+(Z_{B2}-Z_C)C_9$$
$$X_{A3}=(X_{B3}-X_C)C_1+(Y_{B3}-Y_C)C_2+(Z_{B3}-Z_C)C_3$$
$$Y_{A3}=(X_{B3}-X_C)C_4+(Y_{B3}-Y_C)C_5+(Z_{B3}-Z_C)C_6$$
$$Z_{A3}=(X_{B3}-X_C)C_7+(Y_{B3}-Y_C)C_8+(Z_{B3}-Z_C)C_9$$

其中：$C_1=K_5K_6$，$C_2=-K_3K_5$，$C_3=K_2$；$C_4=K_3K_4+K_1K_2K_6$；$C_5=K_4K_6-K_1K_2K_6$；$C_6=-K_1K_5$；$C_7=K_1K_3-K_2K_4K_6$；$C_8=K_1K_6+K_2K_3K_4$；$C_9=K_4K_5$。

用以从一个协坐标系统转换到另一个协坐标系统用到的最终参数为：

C_1，C_2，C_3，C_4，C_5，C_6，C_7，C_8，C_9，X_C，Y_C，Z_C

以上方程用到单值分解(singular Value Decomposition，SVD)和回代(back substitution[4])法。单值分解法(SVD)可以有助于对某一给定矩阵的问题进行诊断和提供相应解答。而且，任一 $m\times n$ 矩阵($m>=n$)都可以写作 $m\times n$ 的正交矩阵 U，$n\times n$ 有着非负元素的对角矩阵，以及 $n\times n$ 正交矩阵的转置。

在有回代的单值分解方法中，转换参数首先被设置为0，并且将结果和期望结果相比较。

这个过程不断将新的值集代入进行，直到结果和期望结果相匹配。

2. 一个典型的测量过程中获得的结果

转换之前的协坐标：

3738.567710	2477.67299	133.157941
4446.351355	2734.034259	−400.465783
4099.653905	3085.748699	−1259.239468

转换后的协坐标：

910.993	−2315.65	34.897000
1213.43	−3007.23	179.581
380.843	−1536.02	1131.59

转换参数为：

$C_1=0.9263$，$C_2=-0.935068$，
$C_3=0.21259$，$C_4=-1.80296$，
$C_5=1.552617$，$C_6=-0.349405$，
$C_7=-0.520200328$，$C_8=0.881826$，
$C_9=-0.53737066$
$X_C=700.0296$，$Y_C=320.61397$，$Z_C=-400.0926$

参 考 文 献

1 Analytical Photogrammetry. Second Edition, Sanjib K Ghosh, Pergamon Publishers

2 Photogrammetry. Third Edition, Francis, H Moffitt and Edward. M. Mikhail, Harper and Row Publishers, Newyork

3 Numerical Recipes in C. Cambridge University Press, William H Press, et al. 1993

4 http://www.techsoftpl.com/matrix/doc/matdcmp.htm

5 http://www.library.cornell.edu/nr/nr_index.cg

(上接第139页)

绝对时间差在2秒以内。因此，SCG 比 Apriori 具有更高的挖掘效率。

结论 候选序列的生成是序列挖掘算法中基本而且重要的问题。本文研究了将待挖掘的序列数据库转换为图结构的方法，并以此为基础研究了频繁序列与图的完全子图的关系，证明了一个序列成为频繁序列的必要条件是该序列对应于图中的一个完全子图。利用这个性质，文中给出了一个基于完全图的候选序列的生成算法。除了证明该算法的正确性，文中还将该算法与 Apriori 算法中候选序列的生成过程进行了比较分析。虽然 Apriori 算法生成的候选序列较 SCG 生成的候选序列少，但是 Apriori 在生成候选序列时所进行的剪枝操作较 SCG 多且复杂。在 T25I10D10k 和 T25I20D100k 数据集进上进行的挖掘实验表明 SCG 的挖掘效率比 Apriori 的挖掘效率更高。

本文虽然从数据实验中得到了 SCG 的挖掘效率较高，但是由于挖掘过程与数据集相关性较大，要得到更一般的结论还需要对 Apriori 算法进行更细致的复杂性分析，这是值得进行进一步研究的问题。

参 考 文 献

1 Han J, Pei J, Yin Y. Mining frequent patterns without candidate generation. SIGMOD'00

2 Zaki M. SPADE: An efficient algorithm for mining frequent sequences. Machine Learning, 2001, 40: 31～60

3 Agrawal R, Srikant R. Mining sequential patterns. ICDE'95

4 Srikant R, Agrawal R. Mining quantitative association rules in large relational tables. SIGMOD'96

5 Agrawal R, Srikant R. Fast algorithms for mining association rules. VLDB'94

6 Han J, Kamber M. 数据挖掘——概念与技术. 北京：高等教育出版社，2001