基于密度约束的对比模式挖掘

柴 欣 高一寒 武优西 刘靖宇

(河北工业大学人工智能与数据科学学院 天津 300401) (河北省大数据重点实验室 天津 300401)

摘 要 序列模式挖掘是从序列数据中发现用户感兴趣的模式。对比模式挖掘是其中的一类挖掘方法,其特点是在两类或多类别的序列库中找到特征信息,在实际的生活和生产中应用十分广泛。随着数据规模的不断增加,算法的挖掘效率显得尤为重要,但是当前对比模式挖掘仍存在挖掘速度太慢的问题。为了快速挖掘满足密度约束和间隙约束的对比模式,文中提出了一种近似求解算法 ADMD(Approximately Distinguishing Patterns Mining Based on Density Constraint),该算法在模式的挖掘过程中允许存在小部分的模式丢失,从而换取挖掘速度的大幅提升。该算法采用网树的特殊结构来计算模式的支持数;采用模式拼接的方式来生成候选模式;采用预判式剪枝策略对模式进行剪枝,以避免大量冗余模式的生成。但由于在剪枝过程中可能会剪掉一部分非冗余模式,造成挖掘结果并非完备,因此该算法是一种近似求解算法。在 ADMD 算法的基础上,通过在剪枝策略中设定参数 k 的方式来得到 ADMD-k 算法,该算法可以通过设定 k 的取值来调整剪枝程度,从而在挖掘效率和准确率方面取得平衡。最后在真实的蛋白质数据集上将所提算法与其他算法从挖掘的对比模式数量和挖掘速度方面进行对比实验。实验结果表明,在 $k=1.5$ 的情况下,所提算法仅用不到原来 13%的时间,就可以挖掘到 99%以上的模式,具有近似度高、速度快的特点。

关键词 对比模式,挖掘速度,网树,密度约束,剪枝策略

中图法分类号 TP311 **文献标识码** A **DOI** 10.11896/jsjkx.181202289

Distinguishing Patterns Mining Based on Density Constraint

CHAI Xin GAO Yi-han WU You-xi LIU Jing-yu

(School of Artificial Intelligence, Hebei University of Technology, Tianjin 300401, China)

(Hebei Province Key Laboratory of Big Data Calculation, Tianjin 300401, China)

Abstract Sequential patterns mining is to find interest patterns from sequential data. Distinguishing patterns mining is one of the mining methods, which is characterized by finding feature information in two or more categories of sequence databases. It is widely used in real life and production. With the increasing size of data, the efficiency of algorithm mining is particularly important. However, the mining speed of distinguishing patterns mining is too slow at present. In order to quickly mine the distinguishing patterns that satisfy density constraint and gap constraint, this paper proposed an approximate solution algorithm ADMD (Approximately Distinguishing Patterns Mining Based on Density Constraint). This algorithm allows a small number of patterns to be lost in the process of patterns mining in exchange for a large increase in mining speed. In this algorithm, the support of the pattern is calculated by the special structure of the Net tree, the candidate patterns are generated by patterns growth approach, and the patterns are pruned by the prejudgment pruning strategy to avoid the generation of a large number of redundant patterns. However, some non-redundant patterns may be pruned in the pruning process, resulting in incomplete mining results, so the algorithm is an approximate algorithm. Based on ADMD, the ADMD-k algorithm was proposed by setting the parameter k in the pruning strategy. The algorithm can adjust the pruning degree by setting k, to achieve a balance between mining efficiency and accuracy. Finally, in real protein datasets, the number of mining patterns and mining speed are compared with other algorithms. The experimental results verify that when k is 1.5, the proposed algorithm costs no more than 13% of the time, but can find up more than 99% of patterns. Therefore, the proposed algorithm is very effective with high approximation rate and high speed.

Keywords Distinguishing patterns, Mining speed, Net tree, Density constraint, Pruning strategy

到稿日期:2018-12-11 返修日期:2019-04-15 本文受国家自然科学基金项目(61702157,61571180)资助。

柴 欣(1962—),男,博士,教授,主要研究方向为数据挖掘、智能计算,E-mail:ch212@126.com;**高一寒**(1993—),男,硕士生,主要研究方向为数据挖掘;**武优西**(1974—),男,博士,教授,CCF 高级会员,主要研究方向为数据挖掘、智能计算,E-mail:wuc567@163.com(通信作者);**刘靖宇**(1976—),男,博士,副教授,主要研究方向为数据挖掘、绿色存储。

1　引言

自从序列模式挖掘这一概念被 Agrawal 等[1]提出，其就成为了数据挖掘领域中的一个重要分支，被不断发展和完善[2-5]。根据挖掘过程中模式匹配方式[6-8]和序列集合数量的不同，序列模式挖掘可分为多个类别[9-11]。对比模式挖掘[12-14]就是多种序列模式挖掘中的一个重要类别。

频繁序列模式挖掘仅是在一个序列集合中进行的，并未考虑序列的类别信息。而在实际的生产和生活中，序列数据一般会包含类别属性，如在学生考试的序列中，不同学生可能对应着不同的学校，学校就可以作为他们的类别。对比模式挖掘通过对比两个或多个序列集合，来找出不同数据集中支持度产生很大差异的频繁项集，从而发现不同数据集合中的特征信息。由于对比模式挖掘具有这种特征，因此在实际的应用非常广泛[15-16]。Wei 等[17]提出了一种基于密度约束和间隙约束的对比模式挖掘算法——MPDG 算法。该算法虽然能够挖掘到不同类别的特征信息，但其挖掘速度较为缓慢。对比模式挖掘由于需要在频繁序列模式挖掘的基础上结合类别信息进行挖掘，因此要比频繁序列模式挖掘的难度更大，算法设计更为复杂。目前，对比模式挖掘算法的运行速度缓慢，仍是一个重要问题。

鉴于此，本文提出了 ADMD(Approximately Distinguishing Patterns Mining Based on Density Constraint)算法，该算法采用一种高效的预判式剪枝策略，通过不断生成、更新剪枝条件的判定因子，整体上对算法运行过程中的冗余模式进行高效剪枝，使得挖掘速度大幅度提升。但由于在剪枝过程中可能会剪掉小部分非冗余模式，因此该算法是一种近似求解算法。除了采用这种剪枝策略外，该算法还应用网树结构[18]对数据集中的数据进行匹配挖掘，利用一种高效的模式生成方式来生成候选模式。在此基础上，本文又提出了 ADMD-k 算法，该算法通过参数 k 实现了挖掘效率和挖掘准确率之间的平衡。最后通过以上各方面的改进，该算法的挖掘速度相对于同类挖掘算法有了大幅度的提升。

本文第 2 节给出问题的相关定义；第 3 节详细介绍本文提出的 ADMD 和 ADMD-k 算法；第 4 节展示实验结果，并对其进行分析；最后总结全文。

2　问题定义

二分类序列库是由两个序列集合构成的一个大的序列库。例如，$D=\{S_1,S_2,\cdots,S_M,S_{M+1},\cdots,S_{M+N}\}$ 是一个二分类序列库。前 M 个序列属于正类序列库，即 $D+=\{S_1,S_2,\cdots,S_M\}$，后 N 个序列属于负类序列库，即 $D-=\{S_{M+1},S_{M+2},\cdots,S_{M+N}\}$。其中，$S_i$ 是由若干个元素组成的序列，$len(D)$表示序列库中所有的元素个数。序列 S 由 n 个字符组成，即 $S=s_1s_2\cdots s_n$。模式 $P=p_1p_2\cdots p_m$ 是一个长度为 m 的序列模式。

定义 1　在间隙约束为$[M,N]$时，序列 S 中的子序列 $s_{l1}s_{l2}\cdots s_{lm}$ 中所有相邻的两个字符的下标值都满足条件 $M\leqslant l_{i+1}-l_i-1\leqslant N$，其中的 $l_i(i=1,2,\cdots,m)$为该字符在序列 S 中的位置，则称序列 $s_{l1}s_{l2}\cdots s_{lm}$ 是一个长度为 m 的偏移序列。序列 S 中模式 P 的偏移出现数就是所有长度为$|P|$的偏移序列个数的总和，记为 $ofs(|P|,S)$，根据文献[18]，其计算公式如下：

$$ofs(|P|,S)=\begin{cases}0, & m>l_1\\ (n-(m-1)((M+N)/2+1))W^{(m-1)}, & m\leqslant l_2\\ \text{能够通过公式递归计算得到}, & l_2<m\leqslant l_1\end{cases}\quad(1)$$

其中，模式 P 的长度为 m，序列 S 的长度为 n，间隔约束为$[M,N]$，$W=N-M+1$，$l_1=\lfloor(n+M)/(M+1)\rfloor$，$l_2=\lfloor(n+N)/(N+1)\rfloor$。

定义 2　若序列 S 中的子序列 $s_{l1}s_{l2}\cdots s_{lm}$ 与模式 $P=p_1p_2\cdots p_m$ 相同，则该子序列可以称为一个出现，模式 P 在序列 S 中所有的出现个数称为模式 P 在序列 S 中的支持数，用 $count(P,S)$表示。

定义 3　将模式 P 在序列 S 中的支持数与偏移出现数的比值，定义为模式 P 在序列 S 中的密度，用 $density(P,S)$表示，即 $density(P,S)=count(P,S)/ofs(|P|,S)$。

定义 4　给定二分类序列库 D，密度阈值 MT，间隙约束$[M,N]$，我们用 $sup(D,P,MT)$来表示模式 P 在序列库 D 中的支持度。其计算公式为：

$$sup(D,P,MT)=\sum|\{S\in D|density(S,P)>MT\}|/len(D)$$

如果模式 P 满足以下 3 个条件，则称模式 P 是一个满足挖掘条件的对比模式[13,17]：

1)在正类中频繁，即 $sup(D+,P,MT)\geqslant PCT$；

2)在负类中不频繁，即 $sup(D-,P,MT)\leqslant NCT$；

3)最小性条件，即模式 P 的子模式都不能同时满足条件 1)和条件 2)。

3　挖掘算法

本文提出的 ADMD 算法利用不完整网树结构来计算模式的支持数；通过公式计算偏移出现数；采用模式拼接方式生成候选模式；采用预判式剪枝策略对模式进行剪枝。由于在该算法中采用了预判式剪枝策略，在剪掉大量冗余模式的同时也可能将小部分非冗余模式剪掉，导致挖掘结果并非完备，因此本文所提出的 ADMD 算法是一种近似求解算法。

3.1　计算支持数和偏移出现数

本文所提出的 ADMD 算法是通过不完整网树结构来计算模式的支持数。网树的本质是树结构的一种拓展结构，同样具有树结构中的根结点、叶子结点、层、路径等概念[6,11]。网树结构与一般的树结构不同：一棵网树可以同时拥有多个根结点；相同名称的网树结点可能在网树的不同层中多次出现，用 n_j^i 表示出现在第 j 层上的结点 i；除根结点外的任何一个结点都可能有不止一个双亲结点；从一个网树结点到根结点可能有很多路径。

在网树结构的基础上，若仅保留一棵完整网树的某一层叶子结点，则称这种结构为不完整网树结构[18]。

在网树结构中，从结点 n_j^i 到该网树所有根结点的路径数之和称为该结点的树根路径数，用 $R(n_j^i)$来表示。在网树结构中，根结点的树根路径数均为 1，第 $j(j>1)$层结点的树根路径数为其所有父结点的树根路径数之和。

结合网树的特殊结构，计算长度为 l 的模式在序列中的

支持数，可以将该过程看作在深度为 l 的网树中计算第 l 层叶子结点的树根路径数之和。而在实际计算模式的支持数的过程中，不需要一棵完整的网树结构信息，只需要其上一层叶子结点的信息即可。因此，本文采用不完整网树的结构来计算模式支持数。

在第一遍扫描序列时可以建立长度为 1 的不完整网树；在此基础上，在第 l 遍扫描序列时可以建立长度为 $l(l>1)$ 的不完整网树，然后计算得出不完整网树的树根路径数，即支持数。根据式(1)可以计算各长度模式的偏移出现数。

3.2 候选模式生成方法

传统的候选模式生成方法是在已有的模式后直接添加字符集中的各个元素，但这样会造成大量的冗余，从而拖累挖掘速度。因此，本文采用模式拼接的方式来生成候选模式。下文给出两个定义。

定义 5 给定一个模式 $P=p_1p_2\cdots p_{l-1}p_l$，其中 $l\geqslant 2$，模式 P 的最大前缀 $pre(P)=p_1p_2\cdots p_{l-1}$，模式 P 的最大后缀 $suf(P)=p_2\cdots p_{l-1}p_l$。对于长度等于 1 的模式，最大前缀和最大后缀均是空序列，记作 λ。

定义 6 给定长度为 $l(l>1)$ 的序列 P 和 Q，如果 $pre(Q)=suf(P)$，就可以生成一个长度为 $l+1$ 的候选模式，即 $P\oplus Q=p_1p_2\cdots p_{l-1}p_lq_l$ 或 $P\oplus Q=p_1q_1q_2\cdots q_{l-1}q_l$。

在生成候选模式之前，先采用 HashMap 对队列中的模式进行存储。将模式的最大前缀作为 HashMap 的 key，有着相同最大前缀的所有模式共用一个 key，将这些模式的最后一位字符存储在同一个字符串中。然后将每一个模式的最大后缀与在 HashMap 中搜寻的指定前缀拼接。这样就可以利用已有的长度为 l 的模式来生成长度为 $l+1$ 的候选模式。

3.3 预判式剪枝策略

假定长度为 $l(l\geqslant 2)$ 的模式 P 和 Q，其中 P 是满足条件的对比模式，Q 是不满足条件的候选模式。在 P 的子模式中存在一个正类支持度最小的模式 p_i，由此可得到一个数值 $\alpha=sup(D+,P,MT)/sup(D+,p_i,MT)$；存在一个负类支持度最大的模式 p_j，即可得到一个数值 $\beta=sup(D-,P,MT)/sup(D-,p_j,MT)$。每得到一个新的对比模式时，都会产生新的 α 和 β 值，如果新得到的 α 值大于原有的 α 值，就更新 α 值来保留最大值；如果新得到的 β 值小于原有的 β 值，就更新 β 值来保留最小值。

对模式 Q 进行剪枝判断，将模式 Q 的正负类支持度分别与 α 和 β 值相乘，如果乘积满足对比模式条件，即 $sup(D+,Q,MT)\times\alpha\geqslant PCT$ 且 $sup(D-,Q,MT)\times\beta\leqslant NCT$，则将其加入到队列中等待生成长度为 $l+1$ 的候选模式，否者将不对其生成长度为 $l+1$ 的候选模式，从而起到了剪枝的作用。

利用这一方法可以避免大量冗余模式的生成，但由于在剪枝过程中可能会剪掉小部分非冗余模式，因此会造成挖掘结果并非百分百完整，故该算法(ADMD 算法)是一种近似的对比模式挖掘算法。后文将会分析挖掘结果的完整程度以及挖掘时间的提升幅度。

3.4 ADMD 算法

ADMD 算法的描述如算法 1 所示。

算法 1 ADMD 算法：挖掘所需的对比模式

输入：序列数据库 D，最小最大间隙约束[min，max]，密度约束 MT，正负支持度阈值 PCT 和 NCT

输出：频繁模式集合 freq

```
1. 扫描 D,将其中所有字符加入集合 charset 中
2. for 字符 c:charset
3.       建立模式 P=c
4.       if sup+(P)≥PCT and sup-(P)≤NCT then freq=freq∪P
5.     elseorigin=origin∪P
6. end for
7. 从 charset 中去除所有长度为 1 的频繁模式所对应的字符
8. for 模式 Po:origin
9.       for 字符 c:charset
10.        调用子算法建立模式 P=Po c
11.        if sup+(P)≥PCT and sup-(P)≤NCT then
12.          freq=freq∪P,更新 α,β
13.        else if sup+(P)×α≥PCT and sup-(P)×β≤NCT then
14.          candidate=candidate∪P
15.        end if
16.      end for
17. end for
18. while !candidate.empty do
19.      从 candidate 中取出模式 Ppre=cP′,计算 candidate 中以 P′为
         前缀的模式集 suff
20.      for 模式 Psuff:suff
21.        拼接 Ppre=cP′与 Psuff=P′c′,调用子算法得到模式 P=cP′c′
22.        if sup+(P)≥PCT and sup-(P)≤NCT then
23.          freq=freq∪P,更新 α,β
24.        else if sup+(P)×α≥PCT and sup-(P)×β≤NCT then
25.          candidate=candidate∪P
26.        end if
27.      end for
28. end while
```

子算法的作用是生成新的候选模式，子算法的描述如算法 2 所示。

算法 2 子算法：生成在模式 P_{old} 后接一个字符 c 的新模式 $P=P_{old}c$

输入：序列数据库 D，最小最大间隙约束[min，max]，密度约束 MT，模式 P_{old}，字符 c

输出：模式 P 的不完整网树 $NCNS_P$，正负支持度 $sup_+(P)$，$sup_-(P)$

```
1. for 序列 S:D
2.     currNCN=NCNS_Pold[S.index]
3.     for i=1 to currNCN.length step 1
4.     for gap=currNCN[i]+min+1 to currNCN[i]+max+1 step 1
5.     if S[gap]==c then NCNS_P[S.index][gap]+=currNCN[i]
6.       end for
7.     end for
8.     if 序列 S 为正类序列 then
9.       posSum+=S.length
10.      if sum(NCNS_P[S.index])/offset(S) > MT then posSup+=S.length
11.    else
12.      negSum+=S.length
13.      if sum(NCNS_P[S.index])/offset(S) > MT then negSup+=S.length
14.    end if
15. end for
16. return NCNS_P, sup+(P)=posSup/posSum, sup-(P)=negSup/negSum
```

3.5 ADMD 算法实例

例 1　给定序列库 D,如表 1 所列。表 1 中,字符集 $|\Sigma|=\{A,C,G,T\}$,密度阈值 $MT=0.39$,正类支持度阈值 $PCT=0.5$,负类支持度阈值 $NCT=0.1$,间隙约束为[0,5]。下面用例 1 来说明 ADMD 算法的挖掘过程。

表 1　序列库

Table 1　Sequence dataset

序列号	序列	集合
1	GAACCC	D+
2	AAACCT	D+
3	TAGCACTC	D+
4	GTTAGC	D−
5	ATGGCT	D−

首先扫描序列库 D 中的所有序列,得到长度为 1 的模式分别为 A,C,G,T,分别建立它们的不完整网树并计算它们在这些序列中的支持数。以序列 S_1 为例,如图 1 所示,模式 A,C,G,T 在序列 S_1 中的支持数分别为 2,3,1,0。根据式(1)可以得到长度为 1 的模式在序列 S_1 中的偏移出现数是 6。由此可以得到它们在序列 S_1 中的密度分别为 1/3,1/2,1/6,0,同理可以得到它们在序列 S_2-S_5 中的密度。接下来,可以得到模式 A,C,G,T 在 $D+$ 中的支持度分别为 0.3,0.3,0,0。因此,长度为 1 的模式均不是所挖掘的对比模式,故将它们加入到队列中。

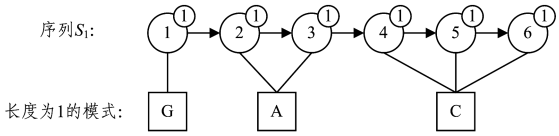

图 1　长度为 1 的模式在序列 S_1 中的出现情况

Fig. 1　Occurrence of 1-length patterns in sequence S_1

根据队列中长度为 1 的模式 A,C,G,T 生成长度为 2 的模式 AA,AC,AG,AT,CA,CC 等。分别建立它们的不完整网树并计算它们在这些序列中的支持数,以序列 S_1 为例,如图 2 所示,模式 AA,AC,AG,AT 在序列 S_1 中的支持数分别为 1,6,0,0。根据式(1)可以得到长度为 2 的模式在序列 S_1 中的偏移出现数是 15,由此可以得到它们在序列 S_1 中的密度分别为 1/15,2/5,0,0,同理可以得到它们在序列 S_2-S_5 中的密度。接下来,可以得到模式 AA,AC,AG,AT 在 $D+$ 中的支持度分别为 0,0.6,0,0,在 $D-$ 中的支持度均为 0。其中模式 AC 满足 $sup(D+,AC,0.39)=(|S_1|+|S_2|)/20=0.6>0.5$ 并且 $sup(D-,AC,0.39)=0<0.1$。因此模式 AC 是所挖掘的对比模式,将其加入到对比模式集中。因为挖掘到了对比模式,所以此时需要对 α 和 β 值进行更新。

将模式 AC 的正类支持度分别比上其子模式 A 和 C 的正类支持度,0.6/0.3=2,0.6/0.3=2,选取比值大的可以得到 α 值,也就是 $\alpha=2$。因为模式 A,C 的负类支持度均为 0,因此 β 值暂时不进行更新。长度为 2 的模式中除了模式 AC 外全部不符合挖掘条件,同时由于此时 β 值还未生成,因此不进行剪枝判断,将其余的模式全部加入到队列中。

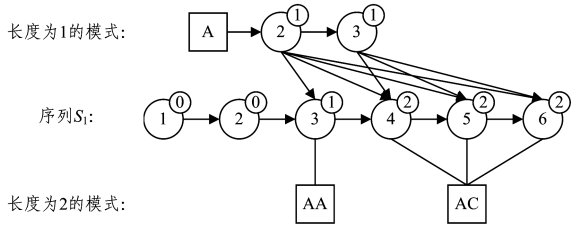

图 2　模式 A 的超模式在序列 S_1 中的出现情况

Fig. 2　Occurrence of hyperpatterns for pattern A in sequence S_1

根据队列中长度为 2 的模式 AA,AG,AT 等生成长度为 3 的模式 AAA,AAG,AAT 等,迭代上述操作直至队列为空时结束。

3.6 ADMD-k 算法

在 ADMD 算法中,α 和 β 值的大小将决定剪枝程度的大小,进而影响挖掘速度和准确率。为此,本文提出了 ADMD-k 算法,该算法将 α 和 β 值分别乘以系数 k 和 $1/k$。显然,ADMD 算法就是 $k=1$ 的情况。

4　实验结果与分析

本文实验是在蛋白质序列集上进行的,实验所采用的蛋白质序列集是从 PFam(Protein Family Database)中获取的,其具体特征如表 2 所列。为了测试本文 ADMD-k 算法的性能,将其与文献[17]的 MPDG 算法进行对比,并且分别取 k 值为 1,1.5,2 和 3,共 4 种参数形式,这些算法均采用 C++ 实现。实验平台为:Intel(R) Core(TM) i5-4210M CPU@2.60GHz,8.0GB 内存,Windows10 操作系统。通过实验测试分析正类支持度阈值的大小对挖掘模式的数量和挖掘时间的影响。

表 2　序列库特征信息

Table 2　Characteristics of sequence dataset

数据集名称	序列类型	$\|D+\|$	$\|D-\|$	备注
Srp	蛋白质序列	5	4	各序列不等长
DUF9495	蛋白质序列	16	5	各序列不等长
Cbi	蛋白质序列	80	76	各序列不等长

具体的实验数据如图 3 和图 4 所示。

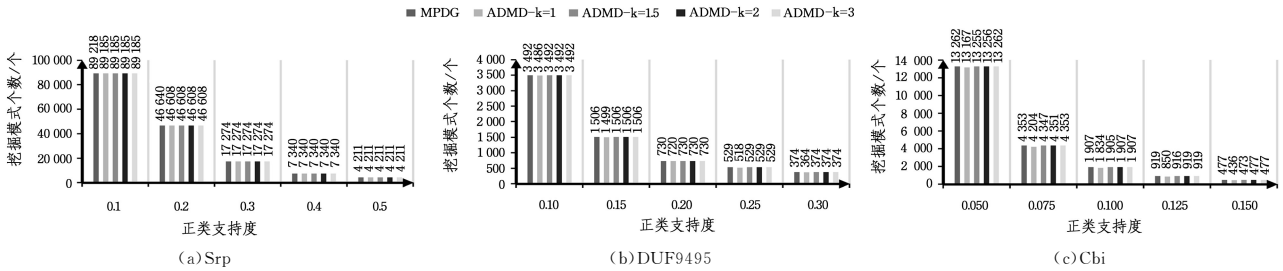

图 3　不同正类支持度阈值对挖掘个数的影响

Fig. 3　Influence of different positive class support thresholds on number of mining

从图 3 可以看出，由于本文算法 ADMD-k 是一个近似算法，随着 k 值的增大，挖掘模式个数逐渐接近 MPDG 算法。例如在图 3(c)中，当阈值为 0.15 时，k 分别为 1，1.5，2 和 3 时，挖掘模式数量分别为 436，473，477 和 477，而该实例中实际模式数量就是 477。通过该实验不难发现，当 $k=1.5$ 时，在这 3 组蛋白质数据集上，ADMD-k 挖掘模式的准确率已达 99% 以上。此外，随着正类支持度阈值的增加，挖掘模式的个数不断下降，其原因是正类库中满足要求的模式数量减少。3 组实验结果均呈现这一特点，例如在图3(a)中，正类支持度从 0.1 增加到 0.5 时，满足条件的模式数量从 89 218 减少到 4 211。

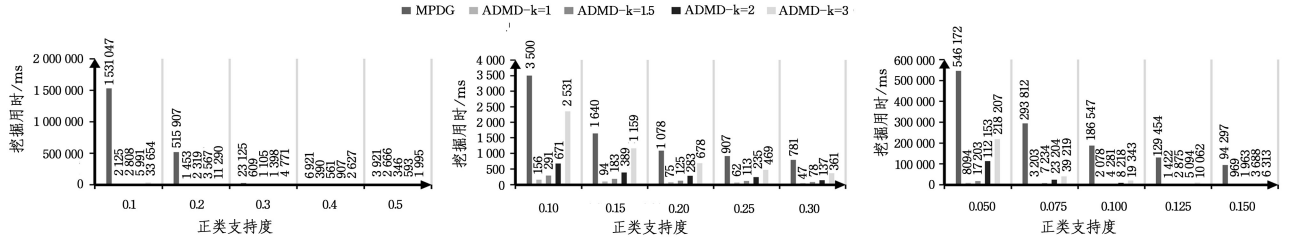

图 4 不同正类支持度阈值对挖掘时间的影响

Fig. 4 Influence of different positive class support thresholds on mining time

从图 4 可以看出，算法 ADMD-k($k=1,1.5,2,3$)的挖掘时间均显著低于 MPDG 算法的挖掘时间，并且伴随 k 值的增大，挖掘时间变长，造成这一现象的原因是 k 值的增大会降低剪枝程度，因而挖掘时间增长，并且由实验数据可知，当 $k=1.5$ 时，在这 3 组数据集上，ADMD-k 消耗的时间都在 MPDG 算法的 13%以下。此外，随着正类支持度的增加，算法的挖掘时间不断缩短，这是因为满足条件的模式数量减少，需要计算的候选模式也会减少。

综上，本文提出的 ADMD-k 算法与 MPDG 算法在挖掘结果近似的情况下，挖掘速度有着大幅提升。其中，当 $k=1.5$ 时，ADMD-k 算法可以用不到 MPDG 算法的 13%的时间来挖掘到 99%以上的模式。

结束语 为了快速地挖掘满足约束的对比模式，本文提出了 ADMD-k 算法，该算法采用不完整的网树结构，使得维护树的结构较为简单，并利用一种高效的模式生成方式，不仅提高了模式生成的效率，而且避免了大量冗余模式的产生，大幅提升了算法的挖掘效率。本文还提出了一种新的剪枝策略，提前对模式进行剪枝，对一些差距需求较大的模式提前进行剪枝，避免其不断产生无用的候选模式，从而大大缩短了算法的挖掘时间。最后，在实际的蛋白质实验中证明了 ADMD-k 算法相较 MPDG 算法在运行速度上有大幅提升。

参 考 文 献

[1] AGRAWAL R,SRIKANT R. Mining sequential patterns[C]// Proceedings of 11th International Conference on Data Engineering. 1995:3-14.

[2] EXARCHOS T P,TSIPOURAS M G,PAPALOUKAS C,et al. A two-stage methodology for sequence classification based on sequential pattern mining and optimization [J]. Data & Knowledge Engineering,2008,66(3):467-487.

[3] DING B,LO D,HAN J,et al. Efficient mining of closed repetitive gapped subsequences from a sequence database[C]// IEEE International Conference on Data Engineering. Shanghai,China:IEEE Computer Society,2009:1024-1035.

[4] WANG H,DING S F. Research and development of sequential pattern mining [J]. Computer Science,2009,36(12):14-17.

[5] MAO G J,HU D J,XIE S Y. Models and algorithms for classifying big data based on distributed data streams [J]. Chinese Journal of Computers,2017,40(1):161-175.

[6] WU Y X,SHEN C,JIANG H,et al. Strict pattern matching under non-overlapping condition [J]. Science China Information Sciences,2017,60(1):012101.

[7] TAN C D,MIN F,WANG M,et al. Discovering patterns with weak-wildcard gaps [J]. IEEE Access,2016,4(1):4922-4932.

[8] MIN F,ZHANG Z H,ZHAI W J,et al. Frequent pattern discovery with tri-partition alphabets [J]. Information Sciences. doi:10.1016/j.ins.2018.04.013.

[9] GONG Y S,XU T T,DONG X J,et al. e-NSPFI:Efficient mining negative sequential pattern from both frequent and infrequent positive sequential patterns [J]. International Journal of Pattern Recognition and Artificial Intelligence,2017,31(2):1-20.

[10] DONG X J,GONG Y S,CAO L B. F-NSP+:A fast negative sequential patterns mining method with self-adaptive data storage [J]. Pattern Recognition,2018(84):13-27.

[11] WU Y X,TONG Y,ZHU X Q,et al. NOSEP:Nonoverlapping sequence pattern mining with map constraints [J]. IEEE Transactions on Cybernetics,2018,48(10):2809-2822.

[12] WANG H F,DUAN L,ZUO J,et al. Efficient mining of distinguishing sequential patterns without a predefined gap constraint [J]. Chinese Journal of Computers,2016,39(10):1979-1991.

[13] YANG H,DUAN L,HU B,et al. Mining top-k distinguishing sequential patterns with gap constraint[J]. Journal of Software,2015,26(11):2994-3009.

[14] WANG X M,DUAN L,DONG G Z,et al. Efficient mining of density-aware distinguishing sequential patterns with gap constraints[C]// International Conference on Database Systems for Advanced Applications. 2014:372-387.

[15] GHOSH S,FENG M,NGUYEN H,et al. Risk prediction for acute hypotensive patients by using gap constrained sequential contrast patterns [C]// AMIA Annual Symposium Proceedings American Medical Informatics Association. 2014:1748-1757.

[16] ZHENG Z G,WEI W,LIU C M,et al. An effective contrast sequential pattern mining approach to taxpayer behavior analysis [J]. World Wide Web-Internet & Web Information Systems,2016,19(4):633-651.

[17] WEI Q S,WU Y X,LIU J Y,et al. Distinguishing sequence patterns mining based on density on density and gap constraints [J]. Computer Science,2018,45(4):252-256.

[18] WU Y X,WANG L L,REN J D,et al. Mining sequential patterns with periodic wildcard gaps [J]. Applied Intelligence,2014,41(1):99-116.