PrefixSpan算法与CloSpan算法的分析与研究

李锡娟,刘 锋,闫娜娜,徐 杰,耿 波
(安徽大学 计算机信息学院,安徽 合肥 230039)

摘 要:数据挖掘领域的一个活跃分支就是序列模式的发现,即在序列数据库中找出所有的频繁子序列。介绍序列模式挖掘的基本概念,然后对序列模式中的经典算法PrefixSpan算法和基于PrefixSpan框架的闭合序列模式CloSpan算法进行了描述,并对它们的执行过程及其特点进行了分析与比较,总结了各自的优缺点,指出PrefixSpan算法适用于短序列方面挖掘,而CloSpan算法在长序列或者阈值较低时胜过PrefixSpan算法且CloSpan算法挖掘大型的数据库有更好的性能,得出的结果对序列模式挖掘的设计有重要的参考价值。

关键词:序列模式挖掘;PrefixSpan算法;CloSpan算法
中图分类号:TP301.6 **文献标识码**:A **文章编号**:1673-629X(2008)01-0070-04

Research and Analysis Between Algorithm of PrefixSpan and CloSpan

LI Xi-juan, LIU Feng, YAN Na-na, XU Jie, GENG Bo
(School of Computer and Information, Anhui University of Technology, Hefei 230039, China)

Abstract: An active research in data mining area is the discovery of sequential patterns, which finds all frequent sub - sequences in a sequence database. Firstly introduces the basic concept of sequential pattern mining, then describes PrefixSpan algorithm and CloSpan which is based on PrefixSpan framework algorithm. The execution process and features of the sequencial mining classic algorithms were finally compared and analysed each other. It shows that PrefixSpan algorithm adapts to mine short sequences, but CloSpan outperforms PrefixSpan when the minimum support is low and sequence is long, furthermore CloSpan has better performance when minning longer frequent sequneces in a large data set. The result gained can be of important value as reference to the design of sequence mining.

Key words: sequential pattern; PrefixSpan algorithm; CloSpan algorithm

0 引 言

序列模式挖掘[1]自1995年被提出以来,已成为当前数据挖掘领域的一个热门分支。它具有广泛的应用价值,例如DNA序列分析、WEB访问记录分析、网络入侵检测、疾病诊断分析以及顾客消费行为模式分析、软件调试和系统I/O优化等等。目前,已经提出了许多序列模式挖掘算法,包括基于Apriori特性的AprioriAll算法[1]、GSP算法[2]、PSP[3]和SPADE[4]和基于模式扩展的FreeSpan[5],PrefixSpan[6]算法等。

主要介绍了PrefixSpan算法和基于PrefixSpan框架的闭合序列模式CloSpan算法,并分析了它们之间的关系,经分析和研究表明,当序列长度很长和阈制值和低的时候CloSpan[7]算法能明显胜过PrefixSpan算法。

收稿日期:2007-04-02
作者简介:李锡娟(1983-),女,安徽安庆人,硕士研究生,研究方向为数据挖掘;刘 锋,教授,硕士生导师,研究方向为并行分布计算、计算机网络。

1 基本概念

定义1 设$I=\{i_1,i_2,\cdots,i_m\}$是一个项目的集合。I的一个子集叫做一个项集。一个序列$s=<t_1,t_2,\cdots,t_m>(t_i\in I,1\leqslant j\leqslant m)$是$m$个项集的有序集合。不失一般性,假设所有在项目集中的项目已经按一定的顺序排序(例如按字母顺序)。序列大小$|s|$是指序列中项目集的数目。序列长度$l(s)=\sum_{i=1}^{n}|t|$是指序列中所有项目的个数。

定义2 序列a的支持度是指在序列数据库D中包含a的序列的个数,$support(\alpha)=||\{s\mid s\in D \text{ and } \alpha\subseteq s\}||$。给定一个最小支持度min_sup,频繁序列模式的集合(frequent sequence,简称FS),就是包含所有的支持度不小于min_sup的序列。

定义3 给定两个序列$a=<a_1,a_2,\cdots,a_m>$和$b=<b_1,b_2,\cdots,b_n>$,如果存在一组整数$i_1<i_2<\cdots<i_n$使得$a_1\subseteq b_{i1},a_2\subseteq b_{i2},\cdots,a_m\subseteq b_{in}$,则称序列$a$被序列$b$包含,即$b$为父序列,$a$为子序列,也可以表示为$a<=b$(如果$a$不等于$b$,那就表示为$a<b$),如

果 b 包含 a 并且它们的支持度相同,称 b 吸收 a。

2 基于投影的序列模式挖掘算法 PrefixSpan

2.1 PrefixSpan 相关概念及术语

定义 4　假设项目集中的项目按某种次序(如字典)排列。给定两个序列模式 a 和 b,$a = < e_1, e_2, \cdots, e_n >$,$b = < e_1', e_2', \cdots, e_m' >$,$m \leqslant n$。$b$ 称为 a 的前缀(Prefix),当且仅当同时满足如下三个条件:

(1) $e_i' = e_i \quad i \leqslant (m-1)$。

(2) $e_m' \subseteq e_m$。

(3) $(e_m - e_m')$ 中的各项目均排在 e_m' 中各项目的后面。

例如:$s = < (a), (a,b,c), (a,c), (d), (c,f) >$ 的前缀有:$< (a) >$,$< (a), (a) >$,$< (a), (a,b) >$,$< (a), (a,b,c) >$,$\cdots$,$< (a), (a,b,c), (a,c), (d), (c) >$,而 $< (a), (b) >$,$< (a), (b,c) >$ 则不是 s 的前缀。

定义 5　给定序列 $\alpha, \beta(\beta \subseteq \alpha)$,$\alpha'$ 称为 α 对应序列 β 的投影(Projection),当且仅当同时满足如下两个条件:

① β 是 α' 的前缀;

② α' 是 α 的满足条件 ① 的最大子序列。

定义 6　设 $\alpha' = < e_1, e_2, \cdots, e_n >$ 为 α 对应于序列 $\beta = < e_1, e_2, \cdots, e_{m-1}, e_m' >$ 投影 $(n \geqslant m)$,称序列 $< e_m - e_m', e_{m+1}, \cdots, e_n >$ 为序列模式 α 对应于序列 β 的后缀(Postfix)。

例如:设序列模式 $s = < (a), (a,b,c), (a,c), (d), (c,f) >$,则序列模式 s 对应于序列 $< (a), (a) >$ 的后缀为:$< (\_,b,c), (a,c), (d), (c,f) >$;$s$ 对应于序列 $< (b), (c) >$ 的后缀为 $< (d), (c,f) >$;s 对应于序列 $< (b), (d) >$ 的后缀为 $< (c,f) >$。

定义 7　设 α 为序列数据库 SD 的一个序列模式,则 α - 投影数据库为 SD 中序列数据对应于 α 的后缀所组成的集合,记为 $SD|_\alpha$。

2.2 PrefixSpan 算法的设计与实现

基于投影的序列模式挖掘算法 PrefixSpan

输入:序列数据库 SD;最小支持度 minsup。

输出:序列数据库 SD 中的所有频繁序列模式。

方法:CALL PrefixSpan(<>,0,SD)

Procedure PrefixSpan($a, l, SD|_a$)

/* a:一个序列模式;l:序列模式 a 的长度;$SD|_a$:如果 a 不等于 <>,$SD|_a$ 为 a - 投影序列数据库,否则,$SD|_a = SD$ */

Begin

(1) 扫描 $SD|_a$ 一次,寻找满足如下条件的频繁项目 b:

将 b 加入到 a 的最后一个元素中后所形成的序列模式为频繁序列模式;将 $< (b) >$ 作为 a 的最后一个元素后所形成的序列模式为频繁序列模式。

(2) 对任意满足条件(1) 的项目 b,将 b 加入到 a 中后所形成频繁序列模式为 a',输出 a';

(3) 对任何 a',构造 a' - 投影序列数据库 $SD|_{a'}$;

(4) 对所有的 $SD|_a$;

(5)CALL PrefixSpan($a', l+1, SD|_{a'}$)

End

下面以一个例子(见表 1) 来说明 PrefixSpan 算法的思想。

表 1　对应的序列数据库

顾客号	顾客序列
1	$< ch >$
2	$< (ab)(ch)(dfg)h >$
3	$< (ceg) >$
4	$< (c)(dgh)(h) >$
5	$< h >$

例 1:表中的序列数据库,最小支持数为 2,PrefixSpan 算法过程如下:

步骤 1:对交易数据库扫描一次得到全部频繁项目,它们同时也是频繁 1 - 序列。它们分别是 $< (c) >$:4,$< d >$:2 $>< c >$:3,$< h >$:4。

步骤 2:基于以上发现的四个频繁项目,序列模式的完整的集合可被分为四个具有不同前缀的序列模式的子集:(1) $< (c) >$ 为前缀的;$\cdots$(4) 以 $< h >$ 为前缀的 >。

步骤 3:发现序列模式的子集。通过构造相应的投影数据库并在其中递归地挖掘可得到序列模式的子集,具体过程如下:

首先发现以 $< (c) >$ 为前缀的序列模式,将包含项 c 的数据序列收集起来,且在这些序列中,只考虑其中以第一次出现的 c 为前缀的子序列。S 中包含 $< c >$ 的投影构成了 $< c >$ - 投影数据库,它包含 4 个后缀序列。通过对该投影库扫描一次,可发现所有以 $< c >$ 为前缀的 2 - 序列模式。它们是 $< cd >$:2,$< cg >$:2,$< ch >$:3。递归地,所有以 $< c >$ 为前缀的序列模式可分为 3 个子集:

① 以 $< (cd) >$ 为前缀的;

② 以 $< (cg)$ 为前缀的 >;

③ 以 $< ch >$ 为前缀的。

通过构造各自的投影库并在其中递归挖掘可得到各个子集。相似地，通过构造 $<d>-$、$<g>-$、$<h>-$ 投影库来构造相应的序列模式的子集。投影数据库和序列模式的完整集合见表 2。

表 2 投影数据库和序列模式

前缀	投影数据库	序列模式
$<c>$	$<h>$, $<(\_h)(dg)h>$ $<(\_g)>$, $<(dgh)h>$	$<c>$, $<cd>$, $<cg>$, $<ch>$, $<c(dg)>$, $<cdh>$, $<c(dg)>$, $<cgh>$
$<d>$	$<(\_g)h,>$ $<(\_gh)h>$	$<d>$, $<(dg)>$, $<dh>$, $<(dg)h>$
$<g>$	$<(h)>$, $<(\_h)h>$	$<g>$, $<gh>$
$<h>$	$<(dg)h>$, $<h>$	$<h>$, $<hh>$

从算法的描述中可见，基于投影的序列模式挖掘算法具有以下三个特点：

①无需产生候选频繁序列模式，这是与类 Apriori 算法的根本区别；

②相对于原始数据库而言，投影数据库的规模不断减小；

③该算法的工作量主要集中在投影数据库的构造上。

3 闭合的序列模式挖掘

上述 PrefixSpan 挖掘算法在挖掘频繁短序列方面已经取得了较好的效果，但是在挖掘序列较长或支持度阈值较小时，频繁序列数量会呈指数级增加，算法性能急剧下降，导致存储空间过度膨胀。近期研究提出了两种方案解决上述问题：第一，只找出最大序列模式集合[8]，而不是挖出全集，但该方法会丢失原有全集的信息；第二，挖掘更加紧凑的闭合序列模式集合[7,9]，该方法不但可以表达与全集相同的信息，而且节省了存贮空间，提高了算法效率。X. Yan, J. Han 提出了首个挖掘闭合序列模式的算法 CloSpan[7]（Closed Sequential pattern mining，闭合序列模式挖掘），遵循候选闭合序列的生成－测试的模式。首先把生成的闭合序列候选集存储在一个 Hash 索引的结果树结构中，在 PrefixSpan 的基础上利用公共前缀、回溯子模式和回溯超模式剪枝的方法来降低搜索空间，以提高算法效率。

3.1 CloSpan 算法的相关概念及术语

定义 8 给定一个序列 $s=<t_1,t_2,\cdots,t_m>$ 和一项 α，$s\diamondsuit_s\alpha=<t_1,t_2,\cdots,t_m,\{\alpha\}>$ if $\forall k\in t_m, k<\alpha$ 叫做序列扩展，记做 S 扩展，即是通过向原序列 s 的末尾加一个新的项集 α 来生成的新序列。

定义 9 给定一个序列 $s=<t_1,t_2,\cdots,t_m>$ 和一项 α，$s\diamondsuit_i\alpha=<t_1,t_2,\cdots,t_m\cup\{\alpha\}>$ if $\forall k\in t_m$, $k<\alpha$ 叫做项集扩展，记做 I 扩展，即是通过向原序列 s 的最后一个项集中添加一个比该项集中任何一项都大的新项 α 以组成新的项集，从而生成的新序列。

例如：$<(a)(b)>$ 是 $<(b)>$ 的序列扩展；$<(ab)>$ 是 $<(b)>$ 的项集扩展。

定义 10 若 α 是序列数据库 D 中的一个频繁序列，且不存在这样的序列 β：(1)β 是 α 的父序列；(2) β 和 α 的支持度相同，则称 α 是一个闭合序列模式。

闭合序列模式发现问题可以描述为：给定顾客数据库 SDB 和用户指定的最小支持度，找到这样的闭合序列集（CS），满足：$CS=\{\alpha\mid\alpha\in FS$ 且 $\neg\exists\beta\in FS$，当 $\alpha\subseteq\beta$ 并且 $\text{supper}(\alpha)=\text{supper}(\beta)\}$，由于 CS 不包含一个与父序列有相同支持度的序列，所以 $CS\subseteq FS$。表 3 是一个顾客序列数据库，SDB 只含有三个顾客序列，$|\text{SDB}|=3$，如果 min_sup = 2，对应的 FS 集合有 16 个序列，如下：

$FS(\text{SDB})=\{a:3,b:2,d:2,e:3,f:2,(af):2,<(a)(b)>:2,<(a)(e)>:2,<(a)(d)>:2,<(e)(a):3,<(e)(b)>:2,<(f)(d)>:2,<(f)(e)>:2,<(af)(d)>:2,<(af)(e)>:2,<(e)(a)(b)>:2\}$，

而 $CS=\{<(af)(d):2>,<(af)(e)>:2,<(e)(a)>:3,<(e)(a)(b)>:2\}$，与 FS 有着相同的信息，但是有更少的模式。但闭合序列模式比频繁序列模式更紧凑。换句话说，挖掘闭合序列可以在保持信息完备性的前提下更加有效地解决序列冗余问题。

表 3 示例数据库表

Sequence ID	Sequence
0	$<(af)(d)(e)(a)>$
1	$<(e)(a)(b)>$
2	$<(e)(abf)(bde)>$

定义 11 设 $I(D)$ 表示 D 中项目的总数，定义为：$I(D)=\sum_{i=1}^{n}l(s_i)$，称 $I(D)$ 是数据的大小。

定义 12 定义投影数据库闭合集合（LS）：$LS=\{s\mid\text{support}(s)\geqslant\text{min\_sup}$ 且不存在 s'，s.t $s\subseteq s'$ 且 $L(D_s)=L(D_{s'})$，则有 $CS\subseteq LS\subseteq FS$。

算法中，取代直接挖掘 CS 是首先产生完全的 LS 集合，然后把非闭合的 LS 序列派出来生成 CS。

定理 1：（投影数据库的等价关系）给定两个序列 s,s'，则 $D_s=D_{s'}\Leftrightarrow I(D_s)=I(D_{s'})$

定理 2：（利用等价关系进行早期终止）给定两个序列，$s\subseteq s'$ 并且 $I(D_s)=I(D_{s'})$，那么 $\forall\gamma$，$\text{support}(s\diamondsuit\gamma)=\text{support}(s'\diamondsuit\gamma)$。

PrefixSpan 算法终止的条件是：当 s 投影数据库中的序列数量小于 min_sup 时，就不需要扩展 s，而 CloSpan 算法是尽可能地早地结束递归过程，也就是算

法在某些树枝上尽早地提前结束。根据定理 2,有效的剪枝方法被发现了。那就是子模式回溯和超模式回溯。

推论 1　子模式回溯:如果一个序列 $s < s'$,$s \supset s'$,则只需条件 $I(D_s) = I(D_{s'})$ 成立,即可停止搜索在前缀搜索树中的所有后代。

推论 2　父模式回溯:如果一个序列 $s < s'$,$s \subset s'$,则只需条件 $I(D_s) = I(D_{s'})$ 成立,则将 s 在前缀搜索树中的所有的子树移植给 s',而不必搜索 s' 任何后代。

PrefixSpan 算法并没有用到这两个剪枝技术,因而搜索空间比较大,而 CloSpan 算法运用这两个剪枝方法在早期终止条件下能够合并子树,用前缀序列格代替了前缀序列树。前缀序列格不仅是一个搜索空间,也是存储 LS 集的内部数据结构。挖掘 CloSpan 结果的第一步是一个前缀序列格。

3.2 CloSpan 算法的设计与实现

3.2.1 算法的总体框架

算法 ClosedMining (D,min_ sup,L)

输入:数据库 D_S 和 min_ sup。

输出:闭合序列集 L。

(1) 删除非频繁项集和空的序列,在数据库 D_S 中对序列中的每个项集进行排序。

(2) 对长度为 1 的频繁项集调用 CloSpan 算法。

(3) 删除结果集中的非闭合序列。

3.2.2 CloSpan 算法的递归调用过程

CloSpan 与 PrefixSpan 相似,可是,在搜索剪枝方面进行了很大的改进。也就是,在探测一个发现序列之前,CloSpan 首先检查是否存在一个 $s \subseteq s'$ 或者是 $s \subseteq s'$ 且 $I(D_s) = I(D_{s'})$。如果这个条件满足了,根据定理 1,就没必要扩展了,因为所有可能的后代已经被发现了。

CloSpan(s,D_s,min_ sup,L) 算法描述如下:

输入:序列 s,投影数据库 D_S,min_ sup

输出:前缀搜索序列格 L

(1) 检查是否发现了 $s \subseteq s'$ 或者是 $s \subseteq s'$ 且 $I(D_s) = I(D_{s'})$,如果发现了则终止,如果没有执行则转到(2)。

(2) 将 s 加入当前的结果集中,扫描 D_s,在 D_s 中寻找频繁项集,如果没有找到就返回,如果有执行(3)。

(3) 对 s 用每个频繁项集进行 S 和 I 扩展,并以 s 的扩展序列和投影数据库为参数调用 CloSpan 算法,回到第(1)步继续往下面执行。

可以对算法的第一步进行优化,算法的第一步包含了两个问题:①包含关系;②投影数据库的大小。包含检测涉及到一个很大的测试空间。如果首先检查包含关系,例如,找出当前序列所有的子序列和父序列,代价是非常大的。因此,设计了一个方法,那就是在投影数据库的大小上使用哈希索引。这样的话,只有投影数据库大小和当前序列的投影数据库大小相等的序列才被检查到。使用这种方法能够明显地提高性能,使得这种检查的时间与总的运行时间相比可以忽略不计。

CloSpan 算法在 PrefixSpan 算法上进行了改进:加入了一些修剪方法,定义了一个类似于投影数据库的新概念,用一个基于哈希的算法优化搜索空间。在文献[7]中表明,CloSpan 不仅生成了全部的闭合子序列集合,而且当序列长度很长或阈值很低时运行效率比 PrefixSpan 高很多。但是由于 CloSpan 需要保存所有的候选闭合序列,当频繁闭合模式很多时,它会消耗大量的内存并导致模式闭合检验时的巨大的搜索空间。因此,它在频繁闭合序列很大时空间性能并不好。

4 总　结

全面分析了 PrefixSpan 和 CloSpan 的两种算法,并对它们之间的关系和各自的优缺点进行了分析。所有的实验均表明,CloSpan 算法挖掘大型的数据库有更好的性能,它不仅有计算最后的结果的效率高,而且在不改变 PrefixSpan 算法的框架下,优于以前的一些算法。实际上当序列很短时或者支持度不是很低的时候 PrefixSpan 算法能胜过 CloSpan 算法,这个在 BIDE[9]算法中讨论过了。事实上,每一种算法都有其相应的应用背景,只有结合实际的背景,一个高效的算法才能体现它真正的内涵。

参考文献:

[1] Agrawal A, Srikant R. Mining Sequential Patterns[C]//In: Proc. of the 11th Int. Conf. on Data Engineering. Taipei: [s. n.], 1995.

[2] Srikant A, Agrawal R. Mining Sequential Patterns: Generalizations and Performance Improvements[C]//In: Proc. of the Fifth Int. Conf. on Extending Database Technology (EDBT). Avignon, France: [s. n.], 1996.

[3] Masseglia F, Cathala F, Poncelet P. The psp approach for mining sequential patterns[C]//In proc. 1998 european symp. principle of data mining and knowledge discovery (PKDD' 98). Nantes, France: [s. n.], 1998.

[4] Zaki M J. SPADE: An Efficient Algorithm for Mining Frequent Sequences[J]. In: Proc. of Machine Learning Journal, special issue on Unsupervised Learning (Doug Fisher, ed.), 2001, 42(1/2): 31 - 60.

(下转第 76 页)

均值代替，如果滑动窗口规定了在取均值过程中窗口各个像素点所占的权重，也就是各个像素点的系数，这时候就称为加权均值滤波。它一般采用 3″3 领域加权均值算法，权值为[1/16,1/8,1/16;1/8,1/4,1/s;1/16,1/s,1/16]。均值平滑有时处理图像的效果不是很好，它虽然去除了一定的噪声，但同时使图像中的边缘变得模糊，这主要和选取的窗口大小有关。这里使用一种既能保持边缘清晰又能消除噪声的自适应平滑滤波方法：

在图像中取 5 * 5 的区域，包含点 (i,j) 的五边形和六边形各 4 个；3 * 3 的区域一个，计算这 9 个区域的标准差和灰度的平均值，取标准差最小区域的灰度平均值作为点 (i,j) 的灰度。

该方法的基本思想是采用一个局部的加权模板与原始的图像信号进行迭代卷积，在每一次迭代时，各个像元点的加权系数是改变的，它是该像元点的梯度函数。同时滤波器的加权系数还依赖于一个参数，这个参数控制了迭代过程中所要保留下来的突变点的幅度，反映了图像灰度值连续性的程度。经过多次的迭代后，滤波器的输出图像变为若干均匀强度区域所组成，且这些区域之间存在很好的边缘。自适应平滑有两个明显的作用：一是锐化了区域边缘；一是使区域内部得到平滑。

8)从扩展区域中提取出原区域的恢复图像。从处理后的效果来看，提高了图像的信噪比，一定程度上改善了图像的可识别性。

3 试验结果及分析

原图如图 2 所示，图像的复原结果如图 3 所示。

图像的复原效果比较好，但是在图像的棱边附近存在振铃波纹，可采用汉明窗等对图像边界部分做处理，从而减弱图像棱边附近的振铃波纹。另外，对复原频谱中的零点采用插值方法进行逼近，可以减少复原图像的振铃波纹，改善复原效果。

实验结果表明该方法能够以较少的运算时间代价获取较好的复原效果，适用于信噪比高的离焦模糊图像的快速复原。

图 2 离焦模糊图

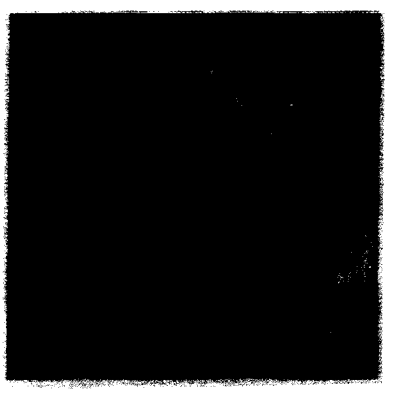

图 3 离焦恢复图

参考文献：

[1] Gonzalez R C. 数字图像处理[M]. 北京：电子工业出版社，2003.

[2] 吴 锡，刘子骥. 非线性图像复原算法的研究[J]. 国外电子测量技术，2006(7)：43－45.

[3] 张 航，罗大庸. 一种改进的全变差盲图像复原方法[J]. 电子学报，2005(7)：34－36.

[4] 郑楚君，李 榕，常鸿森. 离焦模糊数字图像的 Wiener 滤波频域复原[J]. 激光杂志，2004(5)：26－28.

[5] 胡延军，苗春卫，张 兼. 含噪图像的处理方法及评价[J]. 宇航计测技术，2005(2)：59－63.

(上接第 73 页)

[5] Han J, Pei J, Mortazavi－Asl B, et al. FreeSpan: Frequent Pattern－Projected Sequential Pattern Mining[C]//In: Proc. 2000 Int. Conf. Knowledge Discovery and Data Mining (KDD'00). Boston, MA: [s. n.], 2000: 355－359.

[6] Pei J, Han J, Mortazavi－Asl B, et al. PrefixSpan: Mining sequential patterns efficiently by prefix－projected pattern growth[C]//In: Proc. 2001 Int. Conf. Data Engineering (ICDE'01). Heidelberg, Germany: [s. n.], 2001: 215－224.

[7] Yan X, Han J, Afshar R. CloSpan: Mining closed sequential patterns in large datasets[C]//In Proc. 2003 SIAM Int. Conf. Data Mining (SDM'03). San Fransisco, CA: [s. n.], 2003: 166-177.

[8] Afshar R. Mining frequent max, and closed sequential patterns [D]. Canada: School of Computing Science, Simon Fraser University, 2002.

[9] Wang J, Han J. BIDE: Efficient mining of frequent closed sequences[C]//In: Proc. 2004 int. conf. data engineering (ICDE'04). Boston, MA: [s. n.], 2004: 79－90.