第 37 卷 第 14 期
Vol.37 No.14

计 算 机 工 程
Computer Engineering

2011 年 7 月
July 2011

• 软件技术与数据库 •　　文章编号：1000—3428(2011)14—0062—03　　文献标识码：A　　中图分类号：TP311

最大亚频繁模式挖掘算法研究

张海清[1]，刘胤田[1,2]

(1. 成都信息工程学院智能信息处理实验室，成都 610225；
2. 四川大学数学学院，成都 610065)

摘　要：为解决传统最大频繁模式在项集频繁度与项集长度规模之间的制约关系，提出最大亚频繁模式概念及其挖掘算法 MSFP-mining，包括最大亚频繁模式概念并分析其要素特点，基于 AFP-tree、CMP-tree、SFP-tree、SFP-growth 的候选 MSFP 挖掘方法，基于 MSFP-tree 的最大亚频繁模式超集检测和剪枝策略及对 MSFP-mining 挖掘性能的实验验证。实验结果表明，该算法利用差别频繁度实现核心项集、附加频繁项集、补充频繁项集的阶段性求取和组合，在保证项集频繁度基础上实现最大亚频繁模式挖掘，扩展频繁模式规模。

关键词：模式挖掘；最大亚频繁模式；数据集；超集检测；MSFP-tree 结构

Research on Mining Algorithm of Maximal Sub-Frequent Pattern

ZHANG Hai-qing[1], LIU Yin-tian[1,2]

(1. Intelligent Information Processing Lab, Chengdu University of Information Technology, Chengdu 610225, China;
2. College of Mathematics, Sichuan University, Chengdu 610065, China)

【Abstract】To solve the problem of traditional maximal frequent pattern mining that it can not find frequent pattern remaining more items than traditional maximal frequent pattern with the same support threshold, this paper proposes the conception of Maximal Sub-Frequent Pattern(MSFP) and relative mining algorithm MSFP-mining. The main contributions include: the conception of MSFP and analysis of MSFP character, the MSFP-mining algorithms of MSFP, such as AFP-tree, CMP-tree, SFP-tree, SFP-growth, and MSFP-tree, the superset check method of candidate MSFP and the pruning strategy of MSFP-tree, the efficiency of MSFP-tree based mining algorithms by extensive experiments. Experimental result shows that MSFP can effectively expand the scale of maximal frequent pattern.

【Key words】pattern mining; Maximal Sub-Frequent Pattern(MSFP); data set; superset check; MSFP-tree structure

DOI: 10.3969/j.issn.1000-3428.2011.14.019

1　概述

随着数据挖掘的广泛应用，挖掘最大频繁项集已成为多种数据挖掘中的关键问题。根据搜索空间树的遍历策略，可以把经典的最大频繁项集挖掘算法分为宽度优先算法 MaxMiner[1]和深度优先算法 Mafia[2]、FpMax[3]、GenMax[4]。文献[5]方法有效提高了频繁项集的挖掘效率。本文提出最大亚频繁模式 MSFP(Maximal Sub-Frequent Pattern)概念及其挖掘算法。

2　最大亚频繁模式挖掘

亚频繁模式(Supplemental Frequent Pattern, SFP)[6] 相对于传统频繁模式，实现了频繁项集长度的扩展。将项集中的项区分为核心项集 B、附加频繁项集 C 和补充频繁项集(S-X)，其中，S 为 B 的所有补充频繁项的集合；$X \subseteq S$。根据文献[6]，若满足以下 4 个约束条件，称 $B \cup C \cup (S\text{-}X)$为基于核心项集 B 的一个亚频繁模式。

(1) $Suport(B) \geqslant \min\_Sup + \min\_Sup\_Share$；

(2) $\min\_Sup\_Share < Support(B \cup X_i) < \min\_Sup$，其中，$X_i \in S$;

(3) $Suport(B \cup C) > \min\_Sup$；

(4) $\begin{matrix} Suport(B \cup C \cup X) < \min\_Sup, \\ Suport(B \cup C) - Suport(B \cup C \cup X) \geqslant \min\_Sup \end{matrix}$

其中，$X \subseteq S$。

文献[1]提出基于 SFP-tree 实现亚频繁模式挖掘。根据其工作可以得出数以百计的基于核心项集 B 的 SFPs，因为 SFPs 由不同的补充亚频繁项(S-X)组成，而这些补充亚频繁项(S-X)之间又彼此包含，所以 SFPs 中有较大一部分是无效的。剔除 SFPs 中的冗余亚频繁模式，通过超集检测挖掘最大 SFPs 的问题随之转化为最大亚频繁模式的挖掘。

在约束条件(1)~(4)的基础上，增加约束条件，若满足约束条件(5)~(8)，则定义($B \cup C \cup (S\text{-}X)$)为最大亚频繁模式，挖掘 MSFP 即为本文提出的最大亚频繁模式挖掘。

(5) $Support(B \cup C \cup X) < \min\_Sup$；

(6) $Support(B \cup C \cup \neg X) \geqslant \min\_Sup$；

(7) $\forall a \in (S - X), Support(B \cup C \cup \neg X \cup \neg a) < \min\_Sup$；

(8) $\forall k \in (T - B - C - S)$,
$Support(B \cup k) < \min\_Sup \ \| \ (Support(B \cup k) \geqslant \min\_sup \ \&\& \ Support(B \cup C \cup k \cup \neg x) < \min\_sup)$

2.1　相关概念以及性质

本文的数据依据 P2P 资源分配需求，表 1 给出了样例事务数据集，且 $\min\_Sup = 3$，$\min\_SupShare = 2$，核心项集长度阈值为 2。

定义 MSFP-tree 的结构包含：

(1)头表(*HeadTable*)按照 AFP-tree 的升序排序方式。头表包含项名和节点链 2 个字段。其中，节点链的作用是指向同

基金项目：国家自然科学基金资助项目(60773169, 60702075)

作者简介：张海清(1986－)，女，硕士研究生，主研方向：数据挖掘；刘胤田(通讯作者)，副教授、博士

收稿日期：2010-12-07　**E-mail：**wonderful0303860@163.com

一项名的前驱。

(2)核心项(*coreItems*)包含 2 个字段：当前的核心项名(*coreItem-name*)，核心节点链(core-link)。树中的分枝代表了基于当前核心项的 MSFI。

表 1　样例数据集

TID	项	频繁项	完整频繁项
1	*b,h,i,o,p,a*	*p,a,h,o,b*	*p,a,h,o,b* | ¬*m*, ¬*c*, ¬*k*, ¬*l*, ¬*e*, ¬*n*
2	*c,d,g,i,m,p*	*p,c,m*	*p,c,m* | ¬*b*, ¬*k*, ¬*a*, ¬*o*, ¬*l*, ¬*e*, ¬*n*, ¬*h*
3	*a,b,l,m,o,g*	*l,a,o,m,b*	*l, a, o, m, b* | ¬*e*, ¬*n*, ¬*h*, ¬*p*, ¬*c*, ¬*k*
4	*b,h,m,o,c*	*h, o,c,m,b*	*h,o,c,m,b* | ¬*e*, ¬*n*, ¬*p*, ¬*k*, ¬*l*, ¬*a*
5	*b,c,k,h,p,s*	*k,p,h,c,b*	*k, p,h,c,b* | ¬*e*, ¬*n*, ¬*l*, ¬*a*, ¬*m*, ¬*o*
6	*a, c, e, l, m, n, p*	*l,n,e,p,a,c,m*	*l, n, e, p, a, c, m* | ¬*b*, ¬*k*, ¬*o*, ¬*h*
7	*a, b, c,m,d*	*a,c,m,b*	*a, c, m,b* | ¬*o*, ¬*k*, ¬*l*, ¬*n*, ¬*e*, ¬*p*, ¬*h*
8	*b, c, h, j, m*	*h,c,e,m,b*	*h,c,e, m, b* | ¬*o*, ¬*k*, ¬*l*, ¬*n*, ¬*p*, ¬*a*
9	*a,b,k,m,o,s*	*k,a,o,m,b*	*k,a,o,m,b* | ¬*e*, ¬*n*, ¬*p*, ¬*h*, ¬*c*, ¬*l*
10	*a,c,e,l,n,o,p*	*l,n,e,p,a,o,c*	*l, n, e, p, a, o, c* | ¬*b*, ¬*m*, ¬*k*, ¬*n*
11	*b,m,c,o,h,q*	*h,o,c,m,b*	*h,o,c,m,b* | ¬*e*, ¬*n*, ¬*p*, ¬*k*, ¬*l*, ¬*a*
12	*b,a,m,d,f,h*	*h,a,k,m,b*	*h,a,k,m,b* | ¬*e*, ¬*n*, ¬*p*, ¬*l*, ¬*c*, ¬*o*
13	*b,e,m,j,n,o*	*n,e,o,m,b*	*n,e,o,m,b* | ¬*p*, ¬*k*, ¬*l*, ¬*c*, ¬*h*, ¬*a*
14	*p,o,n,k,e,a*	*n,e,p,o,a*	*n,e, p,o,a* | ¬*k*, ¬*l*, ¬*c*, ¬*h*, ¬*b*, ¬*m*
15	*k,o,h,e,c,p*	*k,e,p,h,o,c*	*k,e, p,h,o,c* | ¬*b*, ¬*m*, ¬*a*, ¬*l*, ¬*h*

定理 基于同一核心项集的候选 MSFP 是连续出现的。

证明：基于同一核心项集的候选 MSFP 有 2 种情况：只有单一分枝，具有非负项组成的多条 MSFP 分枝。

(1)具有单一分枝时，当前核心项集的候选 MSFP 只有一个，显然是连续的。

(2)当由非负项组成的多条 MSFP 分枝，也就是多个 MSFP 时，考查其生成过程。CMP-tree、SFP-tree 的头表均来自 AFP-tree(升序排列)，由此在 SFP-tree 中核心项集是按照 AFP-tree 头表顺序出现的；基于基项集的 CMP-tree、SFP-tree 的分枝迭代过程均是按照次序依次产生的，同时在 MSFP-tree 生成中也是按照此次序产生的，因此，在 MSFP-tree 构建过程中基于同一核心项集的候选 MSFP 必定是连续出现的。

定理得证。

2.2　MSFP-Mining 算法

为提高超集检测效率，需要在挖掘算法的各阶段进行超集检测，即在生成各类树之前进行超集检测，而不是仅在算法第 10 步进行超集检测，包括 CMP-tree(Conditional Mix Pattern-tree)、SFP-tree(Supplemental Frequent Patterns-tree)、CFP-tree(Conditional Frequently Pattern-tree)，只有基项集及其条件模式基的合集在 MSFP-tree 中不存在超集时，才生成相应的树，因此，当到达算法第 9 步时，所生成的亚频繁模式必定不存在超集，从而可以跳过第 10 步和第 11 步直接通过第 12 步插入 MSFP-tree。

算法 MSFP-Mining

输入 数据集 *T*，频繁支持度阈值 min_*Sup*，亚频繁共享支持度阈值 min_*Sup_share*，核心项集长度阈值 max*Flag*

输出 MSFP-trees

1. AFP-tree←buildtree_A(T)
2. for each item X in headerTable of AFP-tree
3. CMP-tree←genConditionalMixPatterntree(X)
4. while baseItemSize of CMP-tree < maxFlag-1
5. 递归调用 genConditionalMixPatterntree 生成 CMP-tree
6. if baseItemSize == maxFlag–1
7. 调用 genConditionalPatterntree_withnon 生成 SFP-tree
8. 对 SFP-tree，递归调用 SFP-growth
9. if SFP-tree 为单路径 or 条件模式全为补充频繁项
10. 生成候选 MSFP 并调用 superCheck 进行超集检测
11. if superCheck 结果为真
12. 调用 insertMSFP 将 MSFP 插入 MSFP-tree
13. Return

2.3　MSFP-Tree 的生成样例以及优化策略

根据 MSFP-Ming 算法，基于表 1 中的样例数据集，生成基于核心项集{*b*,*m*}的最大亚频繁模式项集为：{{*c*,*o*,*h*,*m*,*b*}，{*h*,*c*,*k*,*m*,*b*}，{*o*,*k*,*m*,*b*}，{*o*,*a*,*c*,*m*,*b*}，{*c*,*a*,*m*,*b*}，{*a*,*h*,*c*,*m*,*b*}}；基于核心项集{*m*,*c*}生成的最大亚频繁模式项集为：{{*o*,*a*}，{*a*,*h*}}；依次完成以{*b*,*m*}和{*m*,*c*}为核心项集的 MSFP-Mining 操作后的 MSFP-Tree 如图 1 所示，其中，(*m*,*c*)指向为当前核心项的节点链。

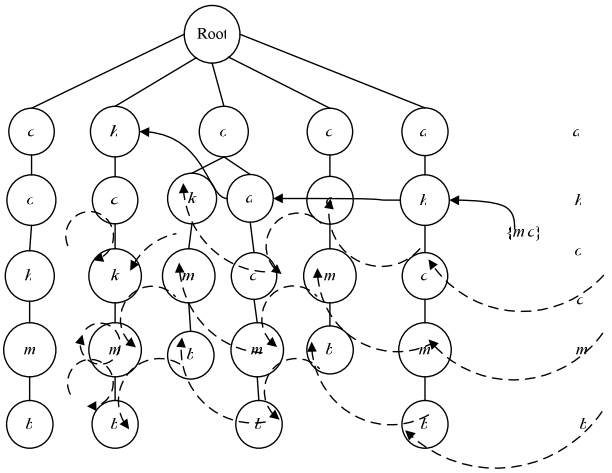


图 1　基于样例数据集生成的 MSFP-Tree

构建基于事务数据集的 AFP-Tree，AFP-Tree 中并不包含负项集，项头表按照升序排列，项集的插入过程与 FP-Tree 的构造过程相同。与 FP-Tree 不同的是，AFP-Tree 项头表中按照升序排列，构造 AFP-Tree 的过程中，将新节点放在链表的头部，若头表中没有此节点则增加该节点，若头表中包含该项，就由项头表通过节点链找到要插入的位置，这样插入算法的时间复杂度为 *O*(1)。

CMP-Tree 项头表根据 AFP-Tree 的头表的排序方式依然采用升序排序，CMP-Tree 不紧包含频繁项同时也包括亚频繁项。基于 CMP-Tree 的频繁条件模式树生成 SFP-Tree。在生成 AFP-Tree、CMP-Tree、SFP-Tree、MSFP-Tree 的过程时，分别对频繁项进行了超集检测和减支的操作。根据 SFP-Tree 的生成规则以及 MSFP-Tree 的项头表的排序方式，核心项集必然出现在 MSFP-Tree 的最低端，因此，对核心项集的链表中的核心节点是否连续出现进行了标记。联合 MSFP-Tree 构造特点的分析，核心节点链的使用，使得超集检测更快更有效。

3　实验与分析

本文实验主要针对 MSFP-Mining 算法实现最大(亚)频繁模式挖掘的时间性能以及最大亚频繁模式的模式规模扩展性能进行分析。实验使用 FIMI Repository(http://fimi.cs.helsinki.fi/)上的稀疏事务数据集(BMPOS)和密集数据集 Mushroom。其中，BMPOS 是比较稀疏的数据集，Mushroom 是典型的非常密集的数据集。数据集主要特征包括数据集名称、事务项大小、事务大小、最大事务长度、平均事务长度，具体数据

见表 2。算法实验平台采用 AMD AthlonII X4 2.6 GHz 处理器，4 GB 内存，500 GB SATA 硬盘，操作系统为 Linux RedHat 5，所有算法采用 C++语言实现并在 GCC 环境下编译。

表 2 事务数据集描述

数据集	项	事务	Max|*T*|	Avg|*T*|
Bms-pos	1 296	65 535	164	7
Mushroom	119	8 124	23	23

实验结果图例格式采用最小支持度_最小共享支持度_核心项集长度阈值_挖掘任务(*minSup_minSupShare_coreItem Num_missionMining*)描述，如 20%_15%_8_MSFP 表示以事务集的 20%为最小支持度、以事务集的 15%为最小共享支持度、挖掘核心项集长度为 8 的所有最大亚频繁项集，0.5%_MFP 表示采用 MSFP-Mining 算法对数据集进行以事务集的 0.5%为最小支持度阈值的传统最大频繁模式挖掘，20%_MFP_fpMax 表示采用 fpMax 算法对数据集进行以事务集的 20%为最小支持度阈值的传统最大频繁模式挖掘。

3.1 时间性能

时间性能实验结果表明，MSFP-Mining 算法在实现传统最大频繁模式挖掘方面性能优于基于 MFI-tree 的 fpMax 挖掘算法，见图 2、图 3。在 MSFP 挖掘方面，MSFP-Mining 算法挖掘稀疏数据集的时间明显长于密集数据集，原因在于挖掘过程中核心项集的补充亚频繁项数量的大幅增加，但正是补充亚频繁项的存在实现了最大亚频繁模式的规模扩展，包括模式的长度和模式的数量。

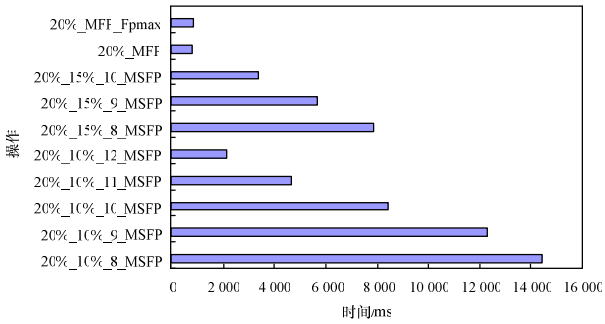


图 2 密集型数据集执行时间

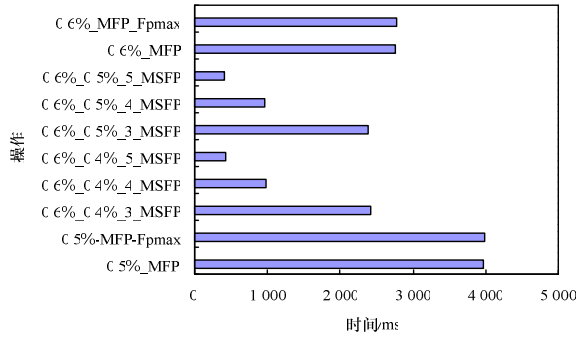
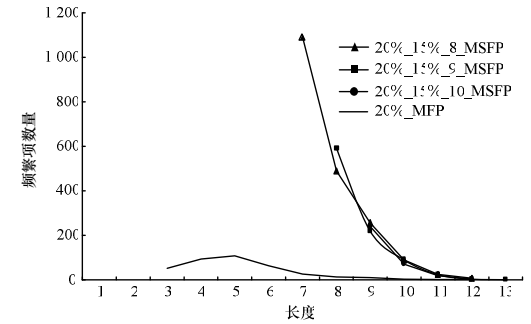


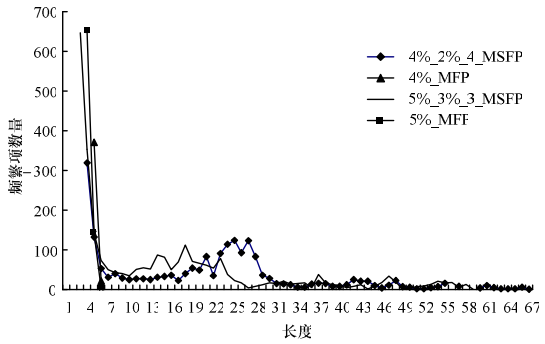
图 3 稀疏型数据集执行时间

3.2 模式规模扩展性能

模式规模扩展实验结果表明，针对密集数据集 mushroom，MSFP-Mining 方法大幅增加了最大频繁模式的数量，但增加的最大亚频繁模式主要分布在核心项集长度阈值旁边，即 MSFP 中包含的补充亚频繁项数量不多，主要原因在于数据集的足够密集要求最小支持阈值保持较高的水平(超过 20%)，针对数据集 BMS-POS 的挖掘增加了较多不同长度的最大亚频繁模式，原因在于 BMS-POS 数据集比较稀疏且有较多的长事务存在，见图 4。



(a)数据集 Mushroom



(b)数据集 BMS-POS

图 4 MSFP-Mining 模式规模扩展效果比较

结果表明，在模式规模扩展方面，对于稀疏数据集，只要能保证事务的相对长度，则规模扩展能力较强(包括数量规模和长度规模)；对于密集型数据集，在保证足够大的最小支持度阈值前提下，仍能在数量规模方面进行有效扩展。

4 结束语

最大亚频繁模式有效地扩展频繁项集的规模，其概念和算法在理论上能用于模糊分类/聚类的研究，在应用上能实现包括 P2P 资源均衡分配和社会化网络用户关系发现，今后工作将对 MSFP-Mining 算法的封闭性及挖掘结果特点进行分析研究，对挖掘过程的有限干预和对挖掘结果进行优化、剪枝和层次提升。

参考文献

[1] Bayardo R. Efficiently Mining Long Patterns from Databases[C]// Proc. of ACM SIGMOD Int'l Conf. on Management of Data. New York, USA: ACM Press, 1998: 85-93.

[2] Burdick D, Calimlim M, Gehrke J. Mafia: A Maximal Frequent Itemset Algorithm for Transactional Databases[C]//Proc. of the 17th Int'l Conf. on Data Engineering. Heidelberg, Germany: IEEE Computer Society, 2001: 443-452.

[3] Grahne G, Zhu Jianfei. High Performance Mining of Maximal Frequent Itemsets[C]//Proc. of the 6th SIAM Int'l Workshop on High Performance Data Mining. Bethesda, Maryland, USA: IEEE Press, 2003: 135-143.

[4] Gouda K, Zaki M J. Efficiently Mining Maximal Frequent Itemsets[C]//Proc. of the 1st IEEE Int'l Conf. on Data Mining. San Jose, USA: IEEE Press, 2001: 163-170.

[5] 张忠平, 李 岩, 杨 静. 基于矩阵的频繁项集挖掘算法[J]. 计算机工程, 2009, 35(1): 84-86.

[6] Liu Yintian, Liu Yingming, Zeng Tao, et al. Mining Supplemental Frequent Patterns[C]//Proc. of the 4th International Conference on Advaced Data Mining and Applications. Chengdu, China: [s. n.], 2008.

编辑 张正兴