FP-array在计算机犯罪挖掘中的应用

李献礼[1]，陈业纲[2]

(1. 长江师范学院网络信息中心 重庆 涪陵 408000; 2. 长江师范学院计算机科学系 重庆 涪陵 408000)

【摘要】针对现代社会计算机犯罪中电子证据的收集难度很大，且海量的电子证据之间的相关性不易分析的问题，对基于FP-Tree的最大频繁模式 (FP-Max) 挖掘算法的优缺点进行了分析，根据FP-Max算法所存在的缺点并结合实际提出一种通过构建FP矩阵的FP-array的高性能关联规则挖掘算法，并将该算法用于典型的计算机犯罪电子证据的相关性数据挖掘中，可用于成功地挖掘比较常见的五类计算机犯罪数据，挖掘结果可为实际的破案过程提供重要参考。

关 键 词 关联规则; 最大频繁模式; 频繁项集; FP-array; 最大频繁模式

中图分类号 TP311.131 **文献标识码** A doi:10.3969/j.issn.1001-0548.2009.04.027

Computer Crime Data Mining Based on FP-array

LI Xian-li[1] and CHEN Ye-gang[2]

(1. Computer Network Information Center, Yangtze Normal University Fuling Chongqing 408000;

2. Department of Computer Science, Yangtze Normal University Fuling Chongqing 408000)

Abstract Since the electronic evidence is difficult to collect in the computer crime in morden society, a new high-performance algorithm for mining association rules based on the construction of the FP-array is proposed according to the discuss of the maximal frequent patterns (FP-Max) algorithm based on FP-tree. Five kinds of common data of computer-related crime are mined successfully by using the new algorithm. The mining results can provide important reference in actual detection process.

Key words association rules; data mining; frequent item sets; FP-array; FP-Max

随着计算机的出现和普及，各种各样的计算机犯罪手段和犯罪形式层出不穷。我国刑法对计算机犯罪的界定过于狭窄，刑事诉讼法对计算机犯罪过程中的电子证据这种特殊的证据形式认定不准确，致使海量电子证据之间的相关性不易分析，司法人员执法困难，难以遏制、防范和打击计算机犯罪。

将关联规则方法应用于计算机取证分析，能发现经过计算机的各活动之间的相关性。通过分析计算机中留下的使用记录和痕迹，可推断出计算机使用者的习惯、特征，并对出现的异常情况做出判断，分析不同犯罪之间的相似性和不同犯罪人之间的关系。

目前一般采用FP-Tree的FP-Max算法进行计算机犯罪挖掘。该算法的缺点是[1]：(1) 在保持最小支持度阈值不变时可能产生新频繁项目，且原频繁项目还可能增加; (2) 在构造FP-tree两次扫描数据集过程中，需要较多的遍历时间。

本文构造FP-array挖掘最大频繁项集算法，用于挖掘计算机犯罪数据。

1 FP-Max算法

1.1 基本概念

设项的集合为$I=\{I_1,I_2,\cdots,I_n\}$，事务数据库$D=\langle T_1,T_2,\cdots,T_n\rangle$，$T\subseteq I$。若$X$中包含$K$个元素，且 $X\subseteq D$，称X为k-项集。

对于任一X，若其支持数不小于用户给定的阈值m，称X为频繁模式(FP)或频繁项集(FIS)。

若X是一个频繁项集，且X的任意一个超集都是频繁的，那么X是最大频繁模式(MFP)或最大频繁项目集(MFIS)，将所有MFP组成的集合称为最大频繁集(MFS)。

在频繁模式树(FP-tree)中，每个结点都由结点名称、结点计数值、结点链表和父结点指针4个域组成。为便于遍历树，构建由项目名和项目链头2个域组成的头表。项目链头指向FP-tree中与之名称相同的第1个节点[2]。

收稿日期：2008 – 07 – 21; 修回日期：2009 – 03 – 26
基金项目：重庆市教委科学研究项目(KJ091305)
作者简介：李献礼(1960 –)，男，副教授，主要从事数据挖掘方面的研究.

1.2 FP-Tree的FP-Max挖掘算法

FP-Tree的FP-Max挖掘算法的关键是寻找全部的MFP，并将该过程拆分为构造和挖掘FP-Tree。

该算法的优点为[3]：(1) 将最耗时的频繁项集(FI)计算通过对FP-tree进行简单搜索而实现；(2) 对于用户给定的不同预测集，可利用分治策略分离创建FP-tree；(3) 可根据用户提供的置信度和长度一对门限值快速确定FI。

2 在FP-array上挖掘MFI算法

2.1 构造FP-array[4-5]

(1) 建立矩阵。假定矩阵中的每列的值表示的是压缩存储的一条或多条记录，要求矩阵中没有重复设所有事务共涉及N个项目，则列向量的长度为N，若某个项目在该事务中出现，则该位为1，否则为0。在5个项目C_1、C_2、C_3、C_4、C_5中，当前事务为C_1和C_5，则当前事务对应的列向量为10001。如果矩阵中存在相同的列，该列向量权值加1，将求出的列向量作为一列添加到关联矩阵中，该列的权值为1。每一列的权值表示事务在数据库中出现的次数。

(2) 计算各项目的支持数。根据设定的最小支持度阈值，去掉所有非频繁项目的行，根据支持数的降序排序，得到调整好的FP-array。

设项目顺序为C_1、C_2、C_3、C_4、C_5，事务数据库D如表1所示。

表1　事务数据库D

T_D	T_{10}	T_{20}	T_{30}	T_{40}	T_{50}	T_{60}	T_{70}	T_{80}	T_{90}
项集	$C_1C_2C_5$	C_2C_4	C_2C_3	$C_1C_2C_4$	C_1C_3	C_2C_3	C_1C_2	$C_1C_2C_3C_5$	$C_1C_2C_3$

FP-array如下：

$$\begin{array}{cc|cccccccc} & \text{支持数} & 1 & 1 & 2 & 1 & 1 & 1 & 1 & 1 \\ & \text{权} & & & & & & & & \\ \hline C_2 & 8 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 \\ C_1 & 6 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \\ C_3 & 5 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 1 \\ C_5 & 2 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ C_4 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \end{array}$$

2.2 FP-array挖掘最大频繁项集算法

先将支持度从大到小排序，对排好序的项进行处理：(1) 找出当前正在处理的k项的投影矩阵$\boldsymbol{B}$，去掉项k所在的行中所有元素为0的列；(2) 删除项k行以下且支持度计数值小于最小支持度的行；(3) 合并相同的列，其权值相加。

设最小支持度阈值为m，对矩阵$\boldsymbol{B}$中每个列向量$\boldsymbol{P}_i$，若MFS中的$\boldsymbol{P}_i$不是某元素的子集需分别3种情况进行讨论[6-7]：

(1) 若$\boldsymbol{P}_i$的权值$\text{count}_i \geqslant m$，则将$\boldsymbol{P}_i$添加到MFS中。(2) 若$\boldsymbol{P}_i$的权值$\text{count}_i < m$，且其他的列向量$\boldsymbol{P}_j$中包含$\boldsymbol{P}_i$，则将$\boldsymbol{P}_j$的权值加到$\text{count}_i$中。(3) 如果既不是$\text{count}_i \geqslant m$，也不是$\text{count}_i < m$的情况，则先分别计算$\boldsymbol{P}_j$和$\boldsymbol{P}_i$包含的项，然后求$\boldsymbol{P}_j$和$\boldsymbol{P}_i$的交集$\boldsymbol{P}$，其权值等于$\boldsymbol{P}_j$和$\boldsymbol{P}_i$权值的和；如果交集$\boldsymbol{P}$的权值大于或等于$m$，但$\boldsymbol{P}$不是MFS中某元素的子集，则将$\boldsymbol{P}$加入到MFS中。具体算法如下[8-10]：

输入事务数据库D的FP-array、最小支持度阈值m、频繁项目个数M；

输出D的最大频繁集MFS；

MFS=Φ(Φ为空)；
for (k=M; k>=2; k - -)
//支持度按从大到小顺序依次处理每个项k
{
求出项k的投影矩阵$\boldsymbol{B}$；
for (i=1; i<=h; i + +)
//h为$\boldsymbol{B}$的列数
if ($\boldsymbol{P}_i$不是MFS中某元素的子集)
//处理每个列向量$\boldsymbol{P}_i$
if ($\boldsymbol{P}_i$. count>=m)
//$\boldsymbol{P}_i$. count为$\boldsymbol{P}_i$列的权值
{MFS=MFS∪$\boldsymbol{P}_i$；
if ($\boldsymbol{P}_i$全为1)返回;}
else
//判断包含$\boldsymbol{P}_i$的列的权值和与最小支持数的关系//，同时把$\boldsymbol{P}_i$放MFS中；
{count=$\boldsymbol{P}_i$. count;
for (j=1; j<=h; j + +)
if (i!=j) and ($\boldsymbol{P}_i$∧$\boldsymbol{P}_j$= =$\boldsymbol{P}_i$)
count+=$\boldsymbol{P}_j$. count；
if (count>=m)
{MFS=MFS∪$\boldsymbol{P}_i$; if ($\boldsymbol{P}_i$全为1)返回;}
else
//如果交集$\boldsymbol{P}$的权值大于或等于m，但$\boldsymbol{P}$不是MFS中//某元素的子集，则将$\boldsymbol{P}$加入到MFS中；
{for(j=1; j<=h; j + +)
if (i!=j){$\boldsymbol{P}$=$\boldsymbol{P}_i$∧$\boldsymbol{P}_j$;
计算P的支持数Pcount;

if (Pcount＞=*m*) and (***P***不是MFS中某元素的子集)

MFS=MFS∪***P***;}}}}

若$\boldsymbol{P}_i$全为1，表明不可能再产生新的最大频繁项目集了，挖掘过程终止。

3 挖掘计算机犯罪的关联

3.1 数据整理

2007年某市部分区县共查出计算机犯罪数据940条，表2是公安局提供的部分计算机犯罪的字段和数据。

表2 计算机犯罪数据

姓名	案件类型	案件数	起始日期	终止日期
甲	破坏计算机	7	2007/01/1	2007/12/1
甲	篡改计算机数据	8	2007/01/1	2007/12/1
乙	非法入侵计算机	4	2007/01/1	2007/12/1
乙	盗用计算机	3	2007/01/1	2007/12/1
丙	盗窃计算机数据	7	2007/01/1	2007/12/1
丙	破坏计算机数据	3	2007/01/1	2007/12/1
丁	散布有害信息	4	2007/01/1	2007/12/1
丁	网上赌博	5	2007/01/1	2007/12/1
丁	计算机软件侵权	4	2007/01/1	2007/12/1

很显然，上述数据都是原始数据，直接处理不方便。在数据挖掘进行之前，需要对这些数据进行预处理。经过分析后知，发生计算机犯罪，犯罪类型往往不只一种，所以，首先需要找到犯罪种类大于一定阈值的计算机刑事案件。假定阈值为2，对数据表中的940条记录进行初步处理后，得到有意义的记录770条，其中包含重点计算机刑事案件303条。通过比较分析发现，计算机犯罪类型通常包含9种[11]，如表3所示。

表3 计算机犯罪类型

犯罪编码	犯罪名称	挖掘编码
C_1	破坏计算机犯罪	1
C_2	非法入侵计算机系统犯罪	2
C_3	盗窃计算机数据犯罪	3
C_4	盗用计算机犯罪	4
C_5	篡改计算机数据犯罪	5
C_6	破坏计算机数据犯罪	6
C_7	散布有害信息犯罪	7
C_8	网上赌博	8
C_9	计算机软件侵权犯罪	9

使用关联规则算法挖掘各种计算机犯罪之间的关系，经过Java程序处理后，计算机犯罪数据挖掘结果如表4所示。

表4 某市部分计算机犯罪案件挖掘训练集

案件名称	案件类型编码	数据挖掘编码训练集	T_D
3、1事件	$C_1\,C_3\,C_7$	{1,3,7}	T_1
4、12事件	$C_1\,C_8\,C_9$	{1,8,9}	T_2
⋮	⋮	⋮	⋮

3.2 用FP-array算法进行关联规则挖掘

对以上的数据进行处理后得到训练数据103条，最小支持度定为20%，运行基于FP-array算法的程序，对数据进行一遍扫描后，得到频繁1-项集，然后按照1-项集支持度对训练集从高到低排序，最后剪枝(即去掉非频繁1-项集)[12]，处理过程步骤如下：

(1) 定义训练集的文件名为crimalItem.txt

假设记录总数、项集的宽度和最小支持记录分别为103、9和64。通过排序、剪枝后的少量候选项集如下：

{5 9 7 4}

{5 9 2 8}

{5 9 2 7}

{9 7 4}

剪枝以后，将只剩余〈5〉、〈9〉、〈2〉、〈7〉、〈8〉、〈7〉、〈4〉这7项1-项集(重新排序后的数据)。

(2) 求解FP-array

矩阵中的每列代表的是剪枝后经过压缩的存储的一条或者多条记录，假设如下：

$$\begin{array}{c} \\ 2 \\ 4 \\ 5 \\ 7 \\ 8 \\ 9 \\ \\ \end{array}\begin{array}{c} \begin{array}{cccccc} 2 & 3 & 45 & 2 & 3 & 41 \end{array} \\ \begin{pmatrix} 0 & 3 & 45 & 2 & 3 & 41 \\ 3 & 0 & 32 & 17 & 17 & 4 \\ 45 & 32 & 0 & 5 & 5 & 24 \\ 2 & 24 & 3 & 7 & 7 & 26 \\ 3 & 17 & 5 & 0 & 0 & 4 \\ 41 & 4 & 24 & 4 & 4 & 0 \end{pmatrix} \\ \begin{array}{cccccc} 2 & 4 & 5 & 7 & 8 & 9 \end{array} \end{array}$$

(3) 求解频繁项集

假设求解的频繁项集数量为13，即：

{5}=81

{9}=67

{9 5}=52

{2}=50

{2 5}=45

{2 9}=41

{2 9 5}=38

{7}=30

{7 5}=27

{7 9}=25

{8}=20

{8 5}=12

{4}=4

如果将最小支持度定义为20%，则频繁项集将达到16个，其中频繁为3的项集数量为6个。

3.3 挖掘结果

某市2007年有5种类型的计算机刑事犯罪。通过基于FP-array算法的关联规则挖掘，成功挖掘计算机犯罪数据，如表5所示。

表5 FP-array算法挖掘计算机犯罪数据果

犯罪编码	犯罪名称	挖掘编码
C_2	非法入侵计算机系统犯罪	1
C_5	篡改计算机数据犯罪	3
C_4	盗用计算机犯罪	4
C_8	网上赌博	8
C_9	计算机软件侵权犯罪	9

3.4 实验结果分析

对FP-Max算法与FP-array算法进行对比实验测试。实验环境为双核Intel 3.0 GHz、1 GB内存的Windows Vista操作系统，测试数据库为crimal数据库，采用Java程序实现。crimal数据库包含10 124条记录，记录了计算机犯罪的19个属性。

不同最小支持度下，FP-array算法与FP-Max算法的性能比较如图1所示。

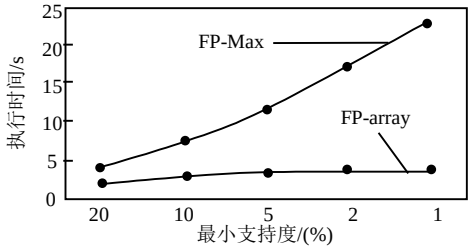

图1　FP-Max 和FP-Array算法性能比较

从图1可以看出，当数据集很大并且稀疏时，本文算法的平均性能比FP-Max算法高。此外，在不同最小支持度下，此算法的执行时间变化幅度很小，在实验中最小支持度为1%时的执行时间只比20%时多0.2 s，说明本算法具有较好的稳定性和可扩展性[13-14]。但是当一个数据集很稠密时，计算关联的FP-array的计算量很大，因此通常不用此方法。

4 结 束 语

本文提出的FP-array算法不需要经常向事务数据库中增加新事务，所以，在保持最小支持度阈值不变的条件下，原频繁项目或新产生的频繁项目的计数值保持不变。而且FP-array的构造过程，只需扫描一次数据集[15]，从大量数据试验发现，构建FP矩阵时不会递归地产生大量条件模式矩阵，具有较高的时间效率，并节省了存储空间。

参 考 文 献

[1] 李忠哗, 吴聪聪, 戴维迪, 等. 基于频繁模式矩阵的最大频繁项目集挖掘算法[J]. 计算机应用与软件, 2007, 24(7): 45-53.
LI Zhong-hua, WU Cong-cong , DAI Wei-di, et al. An FP-tree based incremental updating algorithm for maximal frequent patterns mining[J]. Computer Applications and Software, 2007, 24(7): 45-53.

[2] 李忠哗, 任春龙, 何丕廉. 一种基于FP-树的最大频繁模式增量更新挖掘算法[J]. 计算机应用与软件, 2007, 24(5): 47-49.
LI Zhong-hua , REN Chun-long, HE Pi-lian. An FP-tree based incremental updating algorithm for maximal frequent patterns mining[J]. Computer Applications and Software, 2007, 24(5): 47-49.

[3] 金　光, 刘士荣, 李荣茜, 等. 数据挖掘技术在犯罪行为分析中的应用[J]. 宁波大学学报(理工版), 2002, 9(3): 56-58.
JIN Guang, LIU Shi-rong, LI Rong-qian, et al. Research on crime-analysis using data-mining technology[J]. Journal of Ningbo University(Natural Science & Engineering Edition), 2002, 9(3): 56-58.

[4] AGRAWAL R, IMIELINSKI T, SWAMI A. Mining association rules betweensets of items in large databases[C]//IProc of the ACM SIGMOD Conf on Management of Data(SIGMOD '93). New York: ACM Press, 1993: 207-216.

[5] HAN J, PEI J, YIN Y. Mining frequent patternswithout candidate genera2tion[C]//Proc of the 2000 ACM SIGMOD Int'l Conf on Management of Data(SIGMOD 2000). New York: ACM Press, 2000: 1-12.

[6] LU S F, LU Z D. Fastminingmaximum frequent itemsets[J]. Journal of Software, 2001, 12(2): 293-297.

[7] 吉根林, 杨　明, 宋余庆, 等. 最大频繁项目集的快速更新[J]. 计算机学报, 2005, 28(1): 128-135.
JI Gen-ling, YANG Ming, SONG Yu-qing, et al. Fast updating maximum frequent itemsets[J]. Chinese Journal of Computers, 2005, 28(1): 128-135.

[8] 胡学钢, 刘　卫, 王德兴. 基于剪枝概念格的项集知识表示与挖掘[J]. 计算机工程与应用, 2007, 43(22): 176-178.
HU Xue-gang, LIU Wei, WANG De-xing. Representation and mining of frequent itemsets based on the pruned concept lattice[J]. Computer Engineering and Application, 2007, 43(22): 176-178.

[9] GRAHNE G, ZHU J F. Hight performance mining of maximal frequent itemsets[C]//Proceeding of the 6th SIAM International Workshop on Hight Performance Data mining(HPDM 2003). San Francisco, CA: [s.n.], 2003: 135-143.

编　辑　熊思亮