一种高效的关联发现算法:令牌群挖掘算法

樊建聪[1] 阮久宏[2] 梁永全[1] 曾庆田[1] 唐雷雨[3]
(山东科技大学信息科学与工程学院 青岛 266510)[1]
(山东交通学院科研处 济南 250023)[2] (山东科技大学理学院 青岛 266510)[3]

摘 要 令牌是一种具有生命周期的结构体,它从创建、运行到消亡形成一个完整的生命周期。令牌的创建是生成一个六元组,为要完成的任务设置各项参数。令牌的运行即执行各项命令,完成相关任务并返回执行结果,最后取消并回收令牌,以达到更好的效率。应用这种结构设计了一种新的关联规则发现算法,该算法通过创建并发送令牌完成对数据集的一次扫描,在扫描过程中对数据对象进行标记,然后完成数据的收集和规则模式的生成。实验结果表明,该算法具有线性的时间和空间复杂性,在增量挖掘方面具有良好的性能。

关键词 关联规则发现,令牌,令牌群,数据挖掘
中图法分类号 TP181 **文献标识码** A

Efficient Association Rule Discovery Algorithm: ToARD Algorithm

FAN Jian-cong[1] RUAN Jiu-hong[2] LIANG Yong-quan[1] ZENG Qing-tian[1] TANG Lei-yu[3]
(College of Information Science and Engineering, Shandong University of Science and Technology, Qingdao 266510, China)[1]
(Scientific Research Department, Shandong Jiaotong University, Jinan 250023, China)[2]
(College of Science, Shandong University of Science and Technology, Qingdao 266510, China)[3]

Abstract Token is a kind of structure. A token has a life cycle. The life cycle includes creating token, token executing and canceling token. Creating token is to generate a six-tuple and to set parameters. Token executing is to implement commands, fulfill relevant tasks and return results. A new association rule discovery algorithm was proposed according to the characteristics of tokens. The algorithm creates tokens, and then the tokens scan the data set once. In the process of scanning tokens mark the data objects with different marks. Then the data objects can be collected according to the different marks and the rule patterns can be generated. Experimental results show that the algorithm has better performances in incremental mining, time and space complexities.

Keywords Association rule discovery, Token, Token group, Data mining

1 引言

随着实际应用中数据量的急剧膨胀,对大规模、分散式的异构数据集的处理越来越紧迫,对这些数据的处理和利用也是解决诸如机器学习、决策支持、生物信息等领域问题的基础。对于超大规模数据集的处理方法,通常采用集中或分治策略。所谓集中处理就是将数据集看作一个整体,处理这个整体的方法也是采用单一算法的简单策略,例如通过指针的移动来扫描数据集,同时根据事先确定的条件进行处理。这样的策略,优点是方法简洁,得到的处理结果几乎不涉及不一致等问题,对于小型结构化数据集非常有效,但对于大规模、高维、异构、多目标结构的数据集来说效率非常低,于是就出现了分布式、优化处理的策略——分治策略来处理这类数据。分治策略常用的方法是 Multi-Agent 系统[1-3]、PSO(微粒群算法)[4]等。

Agent 具有自治性、目的性等特性,Multi-Agent 系统又能实现多个 Agent 之间的协作、通信等功能,因此具有一定的分布式处理优势。但是由于 Multi-Agent 系统较为复杂,需要解决规划问题、学习问题、协作问题等一系列问题,因此实现比较困难,复杂度较高。

PSO 也是一种多个体、分治处理问题的方法,把每个个体视为在搜索空间中的一个没有重量、体积,也没有结构的微粒,并在搜索空间中根据个体和群体的飞行经验动态调整参数以寻找最优。目前 PSO 在分类、聚类中有一些算法的研究[5-10],但由于 PSO 主要是用于优化处理等方面,对海量数据的加工和处理就不是很优越。

考虑到 Multi-Agent 系统与 PSO 在处理超大规模数据集的局限,本文提出一种新的策略——令牌群方法,该方法也可以看作是介于 Multi-Agent 系统与 PSO 方法之间的一种策略,用以弥补在数据处理中 Agent 过于复杂、微粒结构过于简

到稿日期:2008-07-01 返修日期:2008-10-06 本文受国家自然科学基金(No. 60603090),山东省自然科学基金(No. Y2007G07),山东科技大学春蕾计划资助。

樊建聪 博士研究生,讲师,研究方向为数据挖掘等,E-mail:fanjiancong@sdust.edu.cn;阮久宏 博士,副教授,研究方向为智能交通等;梁永全 博士,教授,博士生导师,研究方向为人工智能等;曾庆田 博士,副教授,研究方向为知识工程、Petri 网等。

单的不足。3 种策略之间的关系如图 1 所示。

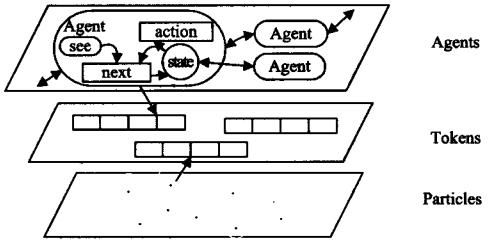

图 1　Agent,Tokens,Particles 关系图

2　相关工作

关联规则挖掘[11]用于发现海量数据中项集之间有趣的关联或其它联系。随着大量数据的不断产生并存储,从海量数据集中挖掘关联规则越来越重要并引起人们的兴趣。

关联规则挖掘实质上是寻找给定数据集中项之间的有趣联系,即形如 $A \rightarrow B$ 的模式。衡量此模式是否有趣,主要有两个最常用的量,即支持度与置信度。

只有同时满足预先设定的最小支持度和最小置信度阈值的规则才被认为是有趣的。

从海量数据集中发现有趣的关联规则需要有效的算法。最基本也是最有影响的关联规则挖掘算法是 Apriori 算法。Apriori 算法[12,13]是一种逐层搜索的迭代方法,通过 k 项集搜索 $k+1$ 项集,算法执行过程中需要多遍扫描数据库,多次连接和剪枝,效率较低。为了进一步提高 Apriori 算法的有效性,又提出了将散列技术、划分方法和选样等技术与 Apriori 算法相结合的算法,但都无法避免多次扫描数据集,甚至降低挖掘结果的精确度。

为了克服多次扫描数据集的不足,又提出了频繁模式树(FP-树)方法[14],该方法通过一种树结构记录项之间的关系及各项出现的次数,能大大降低搜索开销。而当数据库很大时,构造基于内存的 FP-树也是不现实的。

本文提出令牌的定义并设计令牌群的结构,利用令牌的特点及关联规则挖掘的特征,设计出一种基于令牌群的关联规则发现算法。

3　令牌与令牌群

3.1　基本概念

定义 1(令牌)　一种结构,由一个六元组 T 进行描述,

$T=\langle TID, TS, DSA, TC, CS, Description\rangle$

其中,TID 表示令牌编码,根据挖掘任务的类型或特征设置的编码;TS 表示标记,一个令牌可以对多个数据做一种标记且仅做一种标记;DSA 表示数据集的属性子集,确定完成某一特定任务需要的属性;TC 表示令牌做标记的条件,也就是只有数据对象符合 TC 设定的条件时,令牌才能给这个数据对象做标记;CS 表示命令集,设置相关挖掘命令或参数;$Description$ 表示描述集,存储关于某一特定任务的一些描述信息。

如果数据集的数据量较大,可以设置多个令牌做同一种标记,即多个令牌完成同一项任务。如果在一次数据扫描过程中涉及到这种多令牌现象,则可以称之为令牌群。为引入令牌群的定义,先定义令牌冲突的概念。

定义 2(令牌冲突)　设 D 为数据对象,有令牌 t_1 和 t_2,t_1 和 t_2 的条件分别是 TC_1 和 TC_2,记为 $t_1.TC_1$,$t_2.TC_2$。若存在 $TC'=t_1.TC_1 \cap t_2.TC_2$,$TC' \neq \Phi$,且该数据 D 符合 t_1 和 t_2 预定的参数和条件,则称令牌 t_1 和 t_2 在 D 上发生冲突。

令牌在执行任务过程中遇到冲突需要解决,本文使用如下的冲突解决方法:

设令牌 t_1 和 t_2 在数据对象 $D(attri_1,\cdots,attri_n)$ 上发生冲突,其中 $attri_1,\cdots,attri_n$ 是 D 的 n 个属性。令 α 表示 D 的属性满足令牌 t_1 所设定所有条件的个数,β 表示 D 的属性满足令牌 t_2 所设定所有条件的个数,

if $\alpha>\beta$ then 选择 t_1 作标记;

else if $\alpha<\beta$ then 选择 t_2 作标记;

else 按照先来先作标记的原则。

定义 3(令牌群)　令 $G=\langle G_1,G_2,\cdots,G_n\rangle$,$G_1,G_2,\cdots,G_n$ 是 n 个集合,其中 $G_i=\{t_{i1},\cdots,t_{ik}\}$ 表示完成同一任务的一组令牌。每一组令牌完成相同的任务,每一个组的令牌数量根据问题规模而变化。称 G_i 为子令牌群或令牌组,G 称为令牌群。其中 k 的确定较为关键,如果 k 较大,容易与其它组令牌发生冲突;如果 k 较小,对完成任务的效率有影响。本文根据分析结果,建议 n 和 k 按如下标准取值:

(1) 取任务最相关属性 Arelevance;

(2) 计算 Arelevance 的不同属性值数 diff(Arelevance)=n,或不同属性值区间数 diff_interval(Arelevance)=n(可以把整个属性作为一个区间,此时 $n=1$);

(3) 取任务次最相关属性 A'relevance;

(4) 计算 A'relevance 的不同属性值数 diff(A'relevance)=k,或不同属性值区间数 diff_interval(A'relevance)=k(可以把整个属性作为一个区间,此时 $k=1$);

(5) 令牌组数为 n,组中令牌个数为 k。

3.2　令牌的生命周期

令牌具有生命周期。确定任务之后首先要创建令牌,即产生令牌并设置参数;然后发送令牌,让令牌执行相关任务;任务执行完毕后,删除各项参数,回收各项资源。基本流程如图 2 所示。

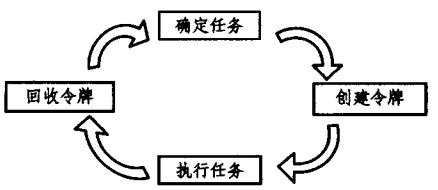

图 2　令牌的生命周期

(1) 创建令牌:创建令牌结构,结构图如下:

TID	TS	DSA	TC	CS	Description

如果令牌属于某个令牌组,则 TS,DSA,TC 的设置可以在令牌组级进行更抽象的设定。创建令牌的另一个重要任务是根据具体任务设置参数值。

(2) 执行任务:对关联规则发现任务来说,令牌需要执行的任务包括扫描数据库、为不同数据对象做标记、令牌之间的冲突消解等。

(3) 回收令牌:删除令牌结构体及其占用的内存空间,以便创建新令牌群,执行其它任务。

3.3　令牌群的结构

令牌群结构可用以下形式化描述：

$G=\{\langle G_i,k\rangle \mid i=1,2,\cdots,n;\ k=1,2,\cdots,m\}$

G_i 表示令牌组，k 表示令牌组 G_i 中的令牌个数，同一个令牌组中的 k 个令牌完成相似的任务。

例 设有一个令牌群 G，G 具有 2 个子令牌组 G_1 和 G_2，G_1 中有 3 个令牌，G_2 中有两个令牌。该令牌群结构如表 1 所列。

表 1 令牌群结构示例表

G	TID	TS	DSA	TC	CS	Des
G_1	11	Θ_{11}	$\{A_{i1},\cdots A_{k1}\}$	$Cond_{11}$	$ComSet_{11}$	
	12	Θ_{12}	$\{A_{i2},\cdots,A_{k2}\}$	$Cond_{12}$	$ComSet_{12}$	
	13	Θ_{13}	$\{A_{i3},\cdots,A_{k3}\}$	$Cond_{13}$	$ComSet_{13}$	
G_2	21	Θ_{21}	$\{A_{i4},\cdots,A_{k4}\}$	$Cond_{21}$	$ComSet_{21}$	
	22	Θ_{22}	$\{A_{i5},\cdots,A_{k5}\}$	$Cond_{22}$	$ComSet_{22}$	

4 ToARD 算法设计

4.1 ToARD 算法一般过程

根据令牌定义、生命周期及其结构，本文设计了基于令牌群的关联规则发现算法——ToARD 算法。该算法的设计思想主要依据令牌特点及令牌群结构，从根据任务创建令牌和令牌群，到发送令牌执行关联规则发现任务。在执行任务期间，很关键的一点就是根据任务条件给数据对象做标记，并根据模式生成规则。该算法主要有几个关键过程，如表 2 所列。

表 2 ToARD 算法主要过程

过程名	描述
createTokens (t_i)	创建一个令牌 t_i
send(t_i)	发送令牌 t_i
executing (t_i)	令牌 t_i 执行任务
collection (Data (Θ_i))	数据对象收集
rules_gen (S)	规则生成

ToARD 算法的一般描述如图 3 所示。

```
输入：令牌相关参数、数据库 D
输出：关联规则
(1) 确定具体任务，根据任务确定令牌组数 n 值以及每个令牌组中的令牌数 k 值；
(2) for (int i =1; i++; i<=k) createTokens (ti);
(3) for (each tokeni) { send( ti); executing (ti); }
(4) for (根据数据对象的不同标记 Θi) {
(5)   Si← collection ( Data (Θi) ) ; S ← Si∪ Si+1 ;}
(6) rules_gen (S) ;
```

图 3 ToARD 算法一般描述

创建令牌的一般描述如图 4 所示。在此过程中，很关键的一步是确定属性子集，可以调用已有的属性选择算法来确定选择哪些任务相关属性。另一个关键任务是确定最小支持度和最小置信度参数。

```
function CreateTokens(ti){
    TID←idi;
    TSi←Θk;
    DSAi←(Attrib0, Attrib1, ···, Attribn)
    CSi←CommandSeti();
    Descriptioni←描述信息;
}
function CommandSeti(){
    double Min_ Support{
    }
    double Min_ Confidence{
    }
    Θ←{标记类型集};
    DMQL←{DMQL 命令集};
}
```

图 4 令牌创建过程一般描述

图 5 所示的是令牌的执行过程，令牌完成扫描数据集，为数据做标记等。

```
function Executing(ti){
for(each datai){
    scan(datai);
    if (datai的属性子集都符合约定条件)
        markToken(Θj) ; //Θj∈Θ, 是一种标记
    if (Θj 与 Θi 在 datak 存在冲突) Conflict_Resolution();
}
```

图 5 令牌执行任务过程一般描述

4.2 算法时间复杂度分析

为分析 ToARD 算法的时间复杂度，先引入两个引理。

引理 1 创建 p 个令牌的时间复杂度是 $O(p)$。

证明：一个令牌是一个六元组，可以看作一个线形表。创建 n 个长度为 a（a 是常数）的线形表的时间是 na，时间复杂度是 $O(n)$，所以创建 p 个令牌的时间复杂度是 $O(p)$。

引理 2 设拥有 p 个令牌的令牌群，对于具有 n 个数据的数据集，Executing()过程的时间复杂度为 $O(n+pm)$，其中 m 是每个令牌处理冲突的平均次数。

证明：当 $p=1$ 时，即只有 1 个令牌的令牌群，不存在冲突，则显然事件复杂度是 $O(n)$；当 $p\geqslant 2$ 时，每个令牌平均处理 $[n/p]$ 个数据，设第 i 个令牌处理冲突的次数为 m_i，则 $\sum_i m_i=pm$，$O(\sum p\ [n/p]+pm)=O(n+pm)$。

定理 ToARD 算法具有线性的时间复杂度。

证明：由引理 1 和引理 2 可直接推得该定理。

5 实验分析

根据个人信息数据集，从中找出收入超过 50k 的人的最常见特征。数据集有 48842 条记录和 14 个属性。根据属性与收入之间相关性，取两个与收入（Inc）最相关的属性年龄（Age）和受教育程度（Edu）来分析年龄段、受教育年限与收入之间的内在关系。

根据此任务，设计 6 个子令牌组的令牌群（如表 3 所列），每个令牌分组中有 4 个令牌，如表 4 所列。设最小支持度 $min\_support=0.02$，最小置信度 $min\_confidence=0.5$，且 $\{min\_support, min\_confidence\}\in TC$。

表 3 子令牌组表

ID	TS	DSA	TC
G_1	Θ_1	Age Edu Inc	Age∈(0,10]
G_2	Θ_2		Age∈(10,30]
G_3	Θ_3		Age∈(30,50]
G_4	Θ_4		Age∈(50,70]
G_5	Θ_5		Age∈(70,90]
G_6	Θ_6		Age∈(90,150)

表 4 第 k 个令牌组中 4 个令牌的设置表

k	TID_k	TS_k	TC_k	CS
G_k	K01	Θ_{k01}	Edu∈[1,4]	Age()∧Edu()→Inc()
	K02	Θ_{k02}	Edu∈[5,8]	Age()∧Edu()→Inc()
	K03	Θ_{k03}	Edu∈[9,12]	Age()∧Edu()→Inc()
	K04	Θ_{k04}	Edu∈[13,16]	Age()∧Edu()→Inc()

ToARD 算法执行后，根据收集的各项属性值计算关于

Age,Edu 和 Inc 的支持度值如表 5 所列。

表 5　支持度计算表

Θ	Age	Edu	support_value	Θ	Age	Edu	support_value
21	(10,30]	[1,4]	6.142E-05	41	(50,70]	[1,4]	0.0007
22	(10,30]	[5,8]	0.00046	42	(50,70]	[5,8]	0.00199
23	(10,30]	[9,12]	0.0094	43	(50,70]	[9,12]	0.0284
24	(10,30]	[13,16]	0.0089	44	(50,70]	[13,16]	0.0253
31	(30,50]	[1,4]	0.00027	51	(70,90]	[1,4]	9.2134
32	(30,50]	[5,8]	0.0025	52	(70,90]	[5,8]	9.2134
33	(30,50]	[9,12]	0.0674	53	(70,90]	[9,12]	0.00107
34	(30,50]	[13,16]	0.0719				

由于 Θ_{33}，Θ_{34}，Θ_{43}，Θ_{44} 大于 *min_support*，所以只计算这 4 条规则的置信度，计算结果如表 6 所列。

表 6　置信度表

Θ	confidence_value
33	0.248923146678758
34	0.579806978470676
43	0.291286568103177
44	0.619654913728432

因为 Θ_{34}，Θ_{44} 大于 *min_confidence*，所以可生成的规则是：

Age(30,50] ∧ Edu[9,12]→Inc>50k

Age(50,70) ∧ Edu[13,16]→Inc>50k

对本例设相同的 *min_support*，*min_confidence* 值，在同一台计算机上执行 Apriori 算法，计算所得结果的支持度和置信度表分别如表 7 和表 8 所列。

表 7　Apriori 算法的支持度值

ID	Age	Edu	Support_value
1	(30,50]	[9,12]	0.0501827339455176
2	(30,50]	[13,16]	0.129664322348822
3	(50,70]	[9,12]	0.0462516507478272

表 8　Apriori 算法的置信度值

Θ	confidence_value
1	0.367852149663231
2	0.573661771672193
3	0.622177954847278

比较 ToARD 算法和 Apriori 算法可以看出，用 ToARD 算法的分析结果更准确简洁。

两种算法时间上的性能比较结果如图 6 所示。

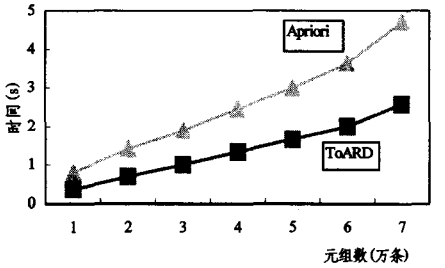

图 6　两种算法的时间性能表较

结束语　数据挖掘是一个应用性很强的研究领域，如何快速、准确、有效地从海量、异构数据中发现有趣的知识是其根本任务。目前也有很多数据挖掘理论研究，也有很多成果，但目前主要还是针对一个或几个特定的应用领域来研究其理论依据。本文主要针对关联规则研究领域，给出了一种新颖的令牌的概念及令牌群的结构，指出创建令牌和令牌群的基本方法，以及令牌挖掘过程中的令牌生命周期的设置等。根据这些特性，本文设计了令牌群关联规则发现算法——ToARD 算法。实验分析表明，该算法具有较好的性能表现和较高的执行效率。

基于令牌群的方法是一种分布式的数据处理方法，除了进行分类、聚类、关联规则发现等数据挖掘应用以外，还应该有形式化的方法和数学基础对其进行描述和研究。下一步的工作除了进一步完善其在特定领域的应用，要从其理论研究方面着手，研究该方法的可扩展性、鲁棒性以及形式化描述方法等。

参 考 文 献

[1] Wooldridge M. An Introduction to Multi-Agent Systems[M]. John Wiley and Sons,2002:1-35

[2] Luo Jiewen, Wang Maoguang, Hu Jun, et al. Distributed data mining on agent grid: issues, platform and development toolkit [J]. Future Generation Computer Systems,2007,23:61-68

[3] Luo Ping,Lu Kevin,Huang Rui,et al. A Heterogeneous Computing System for Data Mining Workflows in Multi-agent Environments[J]. Expert Systems: the Journal of Knowledge Engineering,2007,23(5):258-272

[4] Sousa T F,Silva A P,Silva A F. Swarm Optimization as a New Tool for Data Mining[C]//Proc. of the 17th International Parallel & Distributed Processing Symposium-Workshop on Nature Inspired Distributed Computing (NIDISC) 2003. IPDPS. Nice, France,April 2003

[5] Shi X H,Wan L M,Lee H P,et al. An Improved Genetic Algorithm with Variable Population-size and A PSO-GA Based Hybrid Evolutionary Algorithm[C]//Proceedings of the Second International Conference on Machine Learning and Cybernetics. Xi'an,China,2003:1735-1740

[6] Mahamed G H,Omran,Engelbrecht P A,et al. Dynamic Clustering Using Particle Swarm Optimization with Application in Unsupervised Image Classification[J]. Trans on Engineering,Computing and Technology,2005,9:199-204

[7] Van der Merwe D W,Engelbrecht A P. Data clustering using particle swarm optimization[C]//Proceedings of IEEE Congress on Evolutionary Computation 2003. Canbella, Australia, 2003: 215-220

[8] Venkatalakshmi K,Shalinie S M. Classification of multispectral images using support vector machines based on PSO and K-Means clustering. Intelligent Sensing and Information Processing[C]//Proceedings of International Conference. 2005:127-133

[9] Sousa T,Silva A,Neves A. Particle Swarm-based Data Mining Algorithms for Classification Tasks[J]. Special issue: Parallel and nature-inspired computational paradigms and applications, 2004,30(5/6):767-783

[10] De Falco I,Della Cioppa A,Tarantino E. Facing Classification Problems with Particle Swarm Optimization[J]. Applied Soft Computing,2007,7(3):652-658

[11] Han Jiawei,Kamber M. Data Mining: Concepts and Techniques [M]. Morgan Kaufmann,2000

[12] Agrawal R,Imielinski T,Swami A. Mining Association Rules Between Sets of Items in Large Database[C]//Proc. of ACM-SIGMOD on Management of Data. Washington DC,1993:207-216

[13] Agrawal R,Srikant R. Fast Algorithms for Mining Association Rules[C]//Proc. of VLDB. Santiago,Chile,1994:487-499

[14] Han J, Yin Y. Mining Frequent Patterns Wthout Candidate Generation[C]// Proc. 2000 ACM-SIGMOD Int. Conf. on Management of Data (SIGMOD'00). Dallas,TX,May 2000:1-12