维普资讯 http://www.cqvip.com

源关联规则生成算法

Source Association Rules Generation Algorithm

李学明 张 伟 彭 军 刘用国 吴中福 廖晓峰
(重庆大学计算机学院 重庆400044)

Abstract In this paper, we present the source association rules generation algorithm which can reduce the number of association rules, and prove the several properties of association rules which are the base of our algorithm.

Keywords Source association rules, Apriori, Data mining, KDD

一、引言

IBM 科学家 Rakesh Agrawal 于1993年提出了用于交易的关联规则数据挖掘[1]算法，该算法把基于关联规则的数据挖掘分为两大步，第一步，从交易中发现频繁项目集；第二步，从已发现的频繁项目集中生成所需的关联规则。由于第二步相对简单，且 Rakesh Agrawal 已给出了一个有效算法来生成所需的关联规则，因此人们对基于关联规则的数据挖掘进行的大量的研究都集中在第一步，即如何从交易集中快速生成频繁项目集。但研究发现，Rakesh Agrawal 生成算法虽然能正确有效地生成关联规则，但生成的关联规则具有相当大的冗余性。例如：设关联规则 a->(b,c)表示买面包(a)的人中有80%的人买了啤酒(b)和香烟(c)，则按 Agrawal 生成算法，一定会生成如下几条关联规则：

(1)a->b，买面包(a)的人中有85%的人买了啤酒(b)。

(2)a->c，买面包(a)的人中有90%的人买了香烟(c)，可表示为：

(3)(a,b)->c，买面包(a)同时买啤酒(b)的人中有86%买了香烟(c)。

(4)(a,c)->b，买面包(a)同时买香烟(c)的人中有92%买了啤酒(b)。

这几条关联规则从意义上有一些区别，但 a->(b,c)实际上已包含了以上4条规则，也即，有了 a->(b,c)关联规则，无需进行任何计算，则可以自动导出以上四条关联规则，详细论证见第二节。

本文在研究关联规则的冗余特性时，发现了几个关联规则的重要性质，在此基础上给出了在生成过程中如何消去冗余关联规则的改进型关联规则生成算法，即源关联规则生成算法。

二、关联规则性质及其冗余特性

现在给出关联规则的一般形式化描述。一般地，令 $I=\{i_1,i_2,\cdots,i_n\}$是所有项目(item)的集合(I 称为项目集)，$i_p(p=1,2,\cdots,n)$是一个项目。$T=\{t_1,t_2,\cdots,t_m\}$是所有交易(transaction)的集合(T 称交易集或数据库)，$t_p(p=1,2,\cdots,m)$是一个交易，每个交易是某些项目的集合，是 I 的子集，即 $t_p=\{i_{j1},i_{j2},\cdots,i_{jk}\}$，$j1\neq j2\neq\cdots\neq jk$，$j1,j2,\cdots,jk\in[1,n]$。下面给出关联规则相关定义。

定义1(规则) 设 X 是某些项目的非空集合，如果 X 包含于 T 中，称交易集 T 支持 X。则规则可以描述如下：X=>Y，其中 X、Y 都是包含于 T 的非空集，且 $X\wedge Y=\Phi$。其意义在于一些项目 X 的出现可能导致另一些项目 Y 的出现。=>称为关联，X 称为规则的前件或先决条件，Y 称为规则的后件或结果。

定义2(支持度) 设 X 是某些项目的集合，如果 X 包含于 T 中，即交易集 T 支持 X，则支持度记为 S(X)。S(X)=|{t|t∈T,X 包含于 T}|，表示 X 在 T 中的频度或 X 在 T 中出现的次数。若 X 的支持度≥给定的最小支持度 min_sup，则称 X 为频繁项。

对规则 X=>Y，其支持度 Sup(X=>Y)=S(X∪Y)，表示在 T 中交易 X∪Y 所出现的频度或次数为 Sup(X=>Y)，Sup(X=>Y)/|T|表示交易 X∪Y 在 T 中所占的比例。

规则的支持度可以按次数或比例给出。

定义3(信任度) 对规则 X=>Y，其信任度 Conf(X=>Y)=S(X∪Y)/S(X)，$0\leqslant S(X=>Y)\leqslant 1$。其表示在包含 X 的交易集中 Y 出现的比例。

定义4(关联规则) 给定规则最小支持度为 min_sup、最小信任度为 min_conf%。对规则 X=>Y，若 Sup(X=>Y)>=min_sup 且 Conf(X=>Y)>=min_conf%，则把规则 X=>Y 称为关联规则。

下面证明关联规则的几个性质。

性质1[1] 若 X 为频繁项，其支持度为 S(X)；设 X' 是 X 的非空子集，其支持度为 S(X')，显然 X' 也是频繁项，并且有 S(X')≥S(X)。

例如：{A,B,C,D}是频繁项，则其子集如{A,B,C}也是频繁项。并且 S({A,B,C})≥S({A,B,C,D})。

性质2 若 X=>Y 是关联规则，且设 Y' 是 Y 的非空子集，则有 X∪Y'=>Y−Y' 也是关联规则(Y-Y' 不为空)。

例如：若 A=>{B,C,D}是关联规则，则{A,B}=>{C,D}也必定是关联规则，取 Y'={B}。

证明：①设 X 的支持度为 S(X)，Y 的支持度为 S(Y)，最小信任度为 min_conf%。②由于 X=>Y 是关联规则，故有 Conf(X=>Y)=S(X∪Y)/S(X)≥min_conf%，由此得到 S(X∪Y)≥S(X)*min_conf%。③对规则 X∪Y'=>Y−Y'，其信任度 Conf(X∪Y'=>Y−Y')=S(X∪Y'∪Y−Y')/S(X∪Y')=S(X∪Y)/S(X∪Y')≥S(X)*min_sup%/S(X∪Y')≥min_sup%。因为 X 是 X∪Y' 的子集，故 S(X)≥S(X∪Y)，即 S(X)/S(X∪Y)≥1。Conf(X∪Y'=>Y−Y')≥min_sup%。由此，可得规则 X∪Y'=>Y−Y' 是关联规则。

性质3 若 X=>Y 是关联规则，且设 Y' 是 Y 的非空子集，则有 X=>Y−Y' 也是关联规则(Y-Y' 不为空)。

例如：若 A=>{B,C,D}为关联规则，则 A=>{C,D}也

维普资讯 http://www.cqvip.com

必定是关联规则，取 y'={B}。

证明：①设 X 的支持度为 S(X)，Y 的支持度为 S(Y)，最小信任度为 min_conf%。②由于 X=>Y 是关联规则，故有 Conf(X=>Y)=S(X∪Y)/S(X)≥min_conf%，由此得到 S(X∪Y)≥S(X) * min_conf%。③规则 X=>Y-Y'，其信任度 Conf(X=>Y-Y')=S(X∪Y-Y')/S(X)≥S(X∪Y)/S(X)=S(X) * min_sup%./S(X)=min_sup%。因为 X∪Y-Y'是 X∪Y 的子集，故

S(X∪Y-Y')≥S(X∪Y)

Conf(X=>Y-Y')≥min_sup%

由此，可得规则 X=>Y-Y'是关联规则。

性质2、性质3的意义：若 X=>Y 是关联规则，则利用性质2、性质3就可以自动得到若干关联规则，而不用去进行相应的计算。

例如：若 A->{B,C,D}是关联规则，由性质2、性质3可得到如下规则也是关联规则：

(1){A,B}=>{C,D},Y'={B}
(2){A,C}=>{B,D},Y'={C}
(3){A,D}=>{B,C},Y'={D}
(4){A,B,C}=>{D},Y'={B,C}
(5){A,B,D}=>{C},Y'={B,D}
(6){A,C,D}=>{B},Y'={C,D}
(7){A}=>{C,D},Y'={B}
(8){A}=>{B,D},Y'={C}
(9){A}=>{B,C},Y'={D}
(10){A}=>{D},Y'={B,C}
(11){A}=>{C},Y'={B,D}
(12){A}=>{B},Y'={C,D}

关联规则(1)～(6)由性质2所得，其余规则由性质3所得。

由此可以看出，只要得到了关联规则 A->{B,C,D}，上述的12条关联规则都可以自动生成，也即上述关联规则隐含在关联规则 A->{B,C,D}中。

本文后面的关联规则生成算法就是如何产生形如 A->{B,C,D}的关联规则，而消除其他可利用性质2、性质3导出的关联规则。这样，既加快了关联规则的产生速度，又不影响关联规则所蕴含的知识。

定义5(源关联规则) 若 X=>Y 是关联规则，且设 Y'是 Y 的非空子集，则称关联规则 X=>Y-Y'和 X∪Y'=>Y-Y'相对于关联规则 X=>Y 是导出型关联规则，若 X=>Y 不能由其他任何规则导出，则称为源关联规则(Source Association Rules)。

源关联规则的性质如下：

若 X=>Y 是源关联规则，X'是 X 的非空子集，则 X-X'=>Y∪X'一定不是关联规则(X-X'非空)。

证明：X-X'=>Y∪X'要么是关联规则，要么不是关联规则，二者只取其一。若 X-X'=>Y∪X'是关联规则，由性质2可知：X-X'∪X' => Y∪X'- X'即 X=>Y 是关联规则，且是由 X-X'=>Y∪X'导出的，即 X=>Y 是导出型关联规则，与 X=>Y 是源关联规则不符。

该特征的意义：若{A_1,A_2,…,A_m}是频繁项，则计算其包含的源关联规则 X=>Y 时，应该先计算|X|=1的规则，若 X=>Y 是关联规则，则其一定是源关联规则，则相关计算可终止，否则，计算|X|=2的规则，以此类推，直至|X|=m-1。

三、源关联规则的生成算法描述

1. 算法思想

本算法利用了第二节中的关联规则性质2、性质3和源关联规则性质。

2. 算法描述

设频繁集的项目数最大为 m，即频繁集 L=L_1∪L_2∪…∪L_m

算法：源关联规则生成算法

```
1)for(i=m;i<=2;i ++)Do
2)Begin
3)forall X∈Li Do
4)  Begin
5)    LX1={c∈X};
6)    prune_lx(Ht,LX1,X);
7)    forall c∈LX1 c.conf=S(X)/S(c);
8)    C1={c ∈LX1|c.conf≥min_conf%};
9)    forall c ∈C1 Do
10)     Begin
11)       输出关联规则(c->X-c);
12)       create_hash_table(Ht,c,X);
13)     End
14)   LX1=LX1-C1;
15)   for (k=2;k≤i-1;k ++) Do
16)   Begin
17)     LXk= prod_subset(LXk-1);
18)     prune_lx(Ht,LXk,X);
19)     forall c∈LXk Do c.conf=S(X)/S(c);
20)     Ck={c ∈LXk|c.conf≥min_conf%};
21)     Forall c ∈Ck Do
22)     Begin
23)       输出关联规则(c->X-c);
24)       create_hash_table(Ht,c,X);
25)     End
26)     LXk= LXk-Ck;
27)   End
28)  End
29)End
```

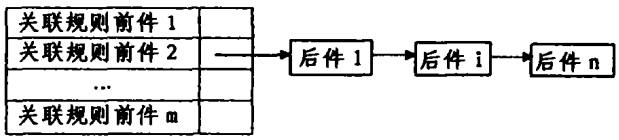
p∈LX_{k-1},q∈LX_{k-1}

```
过程:prod_subset(LXk-1)
1)insert into LXk
  select p[1],p[2],…,p[i-1],q[i-1]
  form LXk-1 p,LXk-1 q
  where p[1]=q[1],p[2]=q[2],…,p[k-2]=q[k-2],p[k-1]
<q[k-1]
2)forall c∈LXk Do /* 修剪过程 */
  forall(k-1)subsets s of c Do
    if s∉LXk-1 then
      delete c from LXk
过程:Create_hash_table(Ht,c,X)
1)addr=hash_func(c);
2)if ht(addr)=c then insert_link(c,X);
3)else create_link(c,X);

过程:prune_lx(Ht,LX,X)
1)forall c∈LX Do
2)begin
3)  addr=hash_func(c);
4)  if search_hash_table(addr,c,X)为真 Do LX=LX-{c}
5)End
```

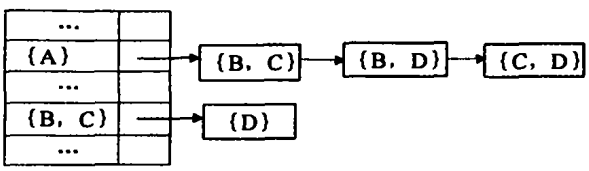
算法说明：

(1)过程 create_hash_table 和 prune_lx 的作用：去掉满足性质3的导出型关联规则。create_hash_table 和 prune_lx 采用的数据结构为 Hash 链表。



如：关联规则{A}->{B,C},{A}->{B,D},{A}->{C,D},{B,C}->{D}

其对应的 Hash 链表如下：



(2)过程 prod_subset(LX_{k-1})的功能：由 k-1项目集 LX_{k-1}生成 k-项目集 LX_k。

举例说明：若 LX_{k-1}={{A B C},{A B D},{A C D},{A

维普资讯 http://www.cqvip.com

C E}，{B C D}}，则由1)insert 操作产生的 LX_k={{A B C D}，{A C D E}}；由2)修剪过程对 LX_k 作修剪得到 LX_k={{A B C D}}，因为{A C D E}的(3－1)子集{C D E}不在 LX_{k-1} 中，故{A C D E}从 LX_k 中被删除，这样处理的依据是性质1。

过程 prod_subset(LX_{k-1})的作用：去掉满足性质2的导出型关联规则。

上述算法中 search_hash_table，hash_func 的描述略。

3 算法分析

对型如 X={A_1，A_2，…，A_n}的频繁项，对应的规则为{A_{i1}，A_{i2}，…，A_{ip}}=>{$A_{i(p+1)}$，$A_{i(p+2)}$，…，A_{in}}，其中 p=1，2，…，n－1，$A_{ij}\in\{A_1,A_2,\cdots,A_n\}$，j=1，2，…，n，$A_{ij}\neq A_{ik}$(j≠k)。

(1)若所有形如{A_{i1}，A_{i2}，…，A_{ip}}=>{$A_{i(p+1)}$，$A_{i(p+2)}$，…，A_{in}}的规则都是关联规则，则 Apriori 生成方法会产生2^n-2条关联规则，因为前件元素个数 p 为1的关联规则有 C_n^1，为2的关联规则有 C_n^2，依次类推，总的关联规则数=$C_n^1+C_n^2+\cdots+C_n^{n-1}=2^n-2$。而本方法仅产生 n 条关联规则，{$A_{ij}$}=>{$A_{i1}$，$A_{i2}$，…，$A_{i(j-1)}$，$A_{i(j+1)}$，…，$A_{in}$}。

(2)若形如{A_{i1}，A_{i2}，…，$A_{i(j-1)}$，$A_{i(j+1)}$，…，A_{in}}=>{A_{ij}}的规则才是关联规则，则两种方法产生的关联规则数目是一样的，因为此种关联规则不能导出任何其他关联规则。除此之外，本文给出的方法产生的关联规则数目都会比 Apriori 少。

(3)本方法产生的关联规则数最多为 $C_n^{[n/2]}$条，如 X={A，B，C，D}，无论支持度如何分布，用本方法产生的关联规则数最多为 C_4^2=6条。因为，若要产生的关联规则多，则必须使形如{A_{i1}，A_{i2}，…，A_{ip}}=>{$A_{i(p+1)}$，$A_{i(p+2)}$，…，A_{in}}的关联规则所导出的关联规则越少，也即前件的元素个数要尽可能多，后件的元素个数尽可能少，极端情况下为(2)所示的形式，但此时总的关联规则数是最少的，因此前件的元素个数不能太多，后件的元素个数不能太少，必须是使 C_n^p(p 为前件的元素个数)达到最大值的 p，从组合规律可知，p=[n/2]时，C_n^p有最大值。该结论的另一作用在于确定 Hash 表的大小。

(4)Hash 表的大小可设为|Ht|=$|L_m|*C_m^{[m/2]}$。Hash 策略可采用如下策略：

addr(A_1，A_2，…，A_p)=(order(A_1)*10^0+ order(A_2)*10^1+…+ order(A_p)*$10^{(p-1)}$) mod |Ht|。

结论 作者在深入研究关联规则特性的基础上，发现了关联规则的冗余特性，本文给出了相关的定义及其若干性质的数学证明，并利用这些性质提出了一个有效地减少关联规则数目的生成算法。该方法生成的关联规则数目虽然减少了，但并没有遗失与之相关的知识，若需要，可以利用这些规则自动生成那些被消除的关联规则。此外，由于产生的关联规则数目显著减少了，因此生成关联规则所耗费的时间也自然地减少了。作者在 Win98环境下利用 vc6.0实现了该算法，运行表明了该算发的正确性和有效性。

参考文献

1 Agrawal R，Imielinski T，Swami A. Mining Association Rules between Sets of Items In Large DataBase. In：Proc. ACM SIGMOD Conf. on Management of Data，Washington，D. C. 1993. 207～216

2 Agrawal R，Srikant R. Fast Algorithms for Mining Association Rules：[IBM Research Report RJ9839]. IBM Almaden Research Center，San Jose，Calif. 1994

3 Agrawal R，Mannila H. Fast Discovery of Association Rules. Advances in Knowledge Discovery and Data Mining. AAAI Press /The MIT Press，1996

4 Jong S P，Philip S Yu. Using a Hash-Based Method with Transaction Trimming for Mining Association Rules. IEEE Transactions on Knowledge and Data Engineering，1997，9(5)

5 欧阳为民，等. 国际上关联规则发现研究综述. 计算机科学，1999，26：41～44

(上接第94页)

系。对每个样本的流量进行标准化处理后，构造流量时间序列相空间。根据分形维的定义实现计算分形维的算法，生成分形维数。之后将数值生成图表进行下一步的处理。数据如表1所示。

将得出的数值递交对比模块进一步地分析得出结论，来决定是否再要计算其它采样间隔时间其它时间段的分形维，如以天为单位，或是否要改变监测采集方式，如监测其它指标。并根据分形维的变化情况来得出流量变化的规律。我们将在环境中利用这些实验得出的数据，根据分形维的变化情况来深入研究流量行为变化规律。

表1 样本分形维数值表

采样时段	采样间隔	样本维数
9:00 -10:00	1s	约5.07
13:00 -14:00	1s	约 4.26
16:00 -17:00	1s	约 7.38
19:00-20:00	1s	约 6.25

结束语 本文根据网络行为研究需要设计并实现了一个网络行为建模环境，有效地融合了网络数据采集监测功能和理论模型分析功能，为网络行为长期性的研究创造了一个有效、互动、稳定的研究环境。我们还在这一环境中，对不同时段所采集的网络流量时间序列进行分析，得到了一些网络流量模型的特征参数分形维数。对网络行为的研究还处于起步阶段，我们希望网络行为研究环境的建立能为实际的研究工作提供方便，同时我们也将不断地应用新的技术和理论来充实和完善现有的研究环境。

参考文献

1 Taqqu L，W. On the Self-similar Nature of Ethernet Traffic(Extended Version). IEEE/ACM Transactions on Networking，1994，2(1)：1～55

2 Huberman B A，Lukose R. Social Dilemmas and Internet Congestion. Science，1997，277：25

3 Veres A. The Chaotic Nature of TCP Congestion Control. IEEE INFOCOM. 2000：1715

4 袁坚，任勇，山秀明. 一种计算机网络的元胞自动机模型及分析. 物理学报，2000，49：398

5 陈式刚. 映象与混沌. 北京：国防工业出版社，1992. 180～181

6 Gao C，Cong S Wu. G. Measurement-based Multifractal Traffic Modeling. In：Proc. of the ISCA 13th Intl. Conf. Las Vegas，Nevada，USA，Aug. 2000：355～360