第3章
在线购物行为分析

随着网络及相关技术的发展，越来越多的人接受了网上购物。在对用户线下在实体店的购物行为的分析中，频繁模式和关联规则的挖掘是常用的分析方法，发现的关联规则可以被用于辅助零售运营商进行运营管理，例如货品的布局安排、促销策略的制定、进货管理等。用户的在线购物行为是否存在新型的行为模式呢？如果存在，如何定义和发现呢？本章将回答这些问题，介绍新型的跨网站在线购物模式以及交易行为模拟方法。

3.1 挖掘跨网站购物模式

本节将介绍一种新型的频繁模式挖掘问题及其新颖的挖掘方法。

3.1.1 什么是跨网站购物模式

频繁项集的挖掘最早源自于超市购物篮分析。它用于分析顾客一次购买的商品之间的关联性，即哪些商品经常被一起购买，这从一定程度上反映了顾客的购买行为模式。例如，购买婴儿尿不湿的同时购买啤酒是从美国中西部的超市交易数据中发现的一种频繁购物行为，通常我们将{尿不湿，啤酒}这个商品组合称为一个频繁项集。频繁项集又称为频繁模式。

传统的频繁模式的挖掘局限在于，发现的频繁模式及规则仅限于在一家超市一次购物的商品之间的关联，在电子商务环境下，运用该方法可以发现客户在一个购物网站一次所购物品之间的关联，然而在一个网站所购物品的数量通常非常少，而可能在不同网站同时进行购物，为此本书提出了一种新型的跨网站购物模式的挖掘问题。

设 S 代表电子商务网站的集合，即 $S=\{s_1,s_2,\cdots,s_m\}$，$P=\{p_1,p_2,\cdots,p_n\}$ 是所有商品的集合，每个商品又称为项(item)。下面是在线购物交易的

定义。

定义 3.1(在线交易)　一个顾客在一个网站 $w_i \in S$ 上所购物品表示为 $w_i(p_i^1, p_i^2, \cdots, p_i^{a_i})$,称为**网站内交易**,其中 $\{p_i^1, p_i^2, \cdots, p_i^{a_i}\} \subseteq P$ 称为**项集**,$a_i \in [1..n]$。一个**在线交易**是一个顾客一定时间段内在不同网站上的网站内交易的集合,表示为

$$T_k = \{w_{k1}(p_{k1}^1, p_{k1}^2, \cdots, p_{k1}^{a_1}), w_{k2}(p_{k2}^1, p_{k2}^2, \cdots, p_{k2}^{a_2}), \cdots, w_{kl}(p_{kl}^1, p_{kl}^2, \cdots, p_{kl}^{a_l})\} \tag{3.1}$$

其中 $a_i \in [1..n]$,$p_{ki}^q \in P$ $(q \in [1..n])$,$w_{ki} \in S$。若 $i \neq j$,且 $i, j \in [1..l]$,满足 $w_{ki} \neq w_{kj}$。这说明一个在线购物交易内的各个网站是不同的,如果一个顾客在给定时间段内在同一个网站多次购物,则合并多次购物的物品到一个网站内交易。

定义 3.2(在线交易)　数据库给定一个时间段,在线交易数据库 D 是在该时间段内所有在线交易的集合,即 $D = \{T_k \mid k \in [1..u]\}$,$u$ 为在线交易的总个数。

表 3.1 是在线交易数据库的一个示例。

表 3.1　在线交易数据库

交易号(TID)	在 线 交 易
T_1	$\{s_1(p_1, p_2), s_2(p_3, p_4), s_3(p_5)\}$
T_2	$\{s_2(p_3, p_4), s_3(p_5, p_6), s_4(p_1)\}$
T_3	$\{s_1(p_1, p_4), s_2(p_1, p_3)\}$

定义 3.3(在线购物模式)　给定一个时间段,一个**在线购物模式**是一个形如$<(p_{d1}^1, p_{d1}^2, \cdots, p_{d1}^{f1}), (p_{d2}^1, p_{d2}^2, \cdots, p_{d2}^{f2}), \cdots, (p_{de}^1, p_{de}^2, \cdots, p_{de}^{fe})>$的项集的组合,满足如下条件:项集$(p_{di}^1, p_{di}^2, \cdots, p_{di}^{fi})$中的每一个项都来自同一个网站,且不同的项集来自不同的网站。一个在线购物模式包含的项集的个数称为其大小。由于一个在线购物模式中项集来自不同网站,因此又称为跨网站购物模式。

一个项集$\{p_{di}^1, p_{di}^2, \cdots, p_{di}^{fi}\}$被一个在线交易 $T_k = \{w_{k1}(p_{k1}^1, p_{k1}^2, \cdots, p_{k1}^{a_1}), w_{k2}(p_{k2}^1, p_{k2}^2, \cdots, p_{k2}^{a_2}), \cdots, w_{kl}(p_{kl}^1, p_{kl}^2, \cdots, p_{kl}^{a_l})\}$所包含,意味着存在 T_k 中的项集$\{p_{kj}^1, p_{kj}^2, \cdots, p_{kj}^{fj}\}$,使得$\{p_{di}^1, p_{di}^2, \cdots, p_{di}^{fi}\} \subseteq \{p_{kj}^1, p_{kj}^2, \cdots, p_{kj}^{fj}\}$。一个在线购物模式 $o = <(p_{d1}^1, p_{d1}^2, \cdots, p_{d1}^{f1}), (p_{d2}^1, p_{d2}^2, \cdots, p_{d2}^{f2}), \cdots, (p_{de}^1, p_{de}^2, \cdots, p_{de}^{fe})>$被一个在线交易 $T_k = \{w_{k1}(p_{k1}^1, p_{k1}^2, \cdots, p_{k1}^{a_1}), w_{k2}(p_{k2}^1, p_{k2}^2, \cdots, p_{k2}^{a_2}), \cdots, w_{kl}(p_{kl}^1, p_{kl}^2, \cdots, p_{kl}^{a_l})\}$所包含,意味着 o 中的每个项集$\{p_{di}^1, p_{di}^2, \cdots, p_{di}^{fi}\}$都存在 T_k 中的一个不同的网站内交易 $w_{kj}(p_{kj}^1, p_{kj}^2, \cdots, p_{kj}^{fj})$,使得$\{p_{di}^1,$

$p_{di}^{2},\cdots,p_{di}^{fi}\}\subseteq\{p_{kj}^{1},p_{kj}^{2},\cdots,p_{kj}^{fj}\}$，即 o 中的任一个项集都被该交易中的一个项集包含，且 o 中不同的项集应该由 T_k 中不同的项集包含。

定义 3.4（支持度） 一个在线购物模式 $o=<(p_{d1}^{1},p_{d1}^{2},\cdots,p_{d1}^{f1}),(p_{d2}^{1},p_{d2}^{2},\cdots,p_{d2}^{f2}),\cdots,(p_{de}^{1},p_{de}^{2},\cdots,p_{de}^{fe})>$的支持度 sup($o$)定义为包含该模式的在线交易的个数或比例（以下用个数）。一个项集的支持度定义为包含该项集的在线交易的个数或比例。

定义 3.5（频繁在线购物模式） 若一个在线购物模式的支持度不小于给定的最小支持度阈值 minsup，则称为频繁在线购物模式或频繁跨网站购物模式。一个项集的支持度若不小于给定的最小支持度阈值，则称该项集为频繁项集。

以表 3.2 为例，若最小支持度阈值设为 2，$<(p_1),(p_3,p_4),(p_5)>$是一个频繁在线购物模式，它被在线交易 T_1、T_2 所包含。在 T_1 中，$\{p_1,p_2\}\supset\{p_1\}$，$\{p_3,p_4\}\supseteq\{p_3,p_4\}$，$\{p_5\}\supseteq\{p_5\}$。

3.1.2 跨网站购物模式的无候选集挖掘方法

基于层次的挖掘方法需要生成候选集，在数据密集的情况下，候选集的个数会很大，影响算法效率。在本节中将介绍一种无候选集的挖掘方法，名为 OSP-tree，其主要步骤如下。

算法：OSP-Tree

输入：在线交易数据库 D，最小支持度阈值 minsup
输出：频繁在线购物模式集合
主要步骤：
1. 发现 D 中满足最小支持度阈值的网站内频繁项集；
2. 将数据库 D 中的项用频繁项集进行替换，D 转换为数据集 DT；
3. 调用过程 BuildTree(DT) 为数据库 DT 构建树 T；
4. 调用 findOSP(T，minsup)发现所有频繁在线购物模式集合；
5. 将每个频繁在线交易模式中的项集用产品代替

第 1 步是发现所有支持度不小于 minsup 的项集。显然，一个在线购物模式如果是频繁的，则其包含的每一个项集必定也是频繁的。可以修改传统的频繁项集挖掘算法，以发现在线交易数据库中的频繁项集。然后将发现的每个频繁项集用一个唯一的标识符进行标识。以表 3.1 中数据库为例，假设最小支持度阈值为 2，则发现的所有频繁项集及其识别符的映射如表 3.2 所示。

表 3.2　频繁项集(minsup=2)

项　集	p_1	(p_3,p_4)	p_3	p_4	p_5
支持度	3	2	3	3	2
标识符	i_1	i_2	i_3	i_4	i_5

第 2 步是数据库的转换,根据第 1 步的结果将数据库 D 中每个网站内交易进行转换,将一个网站内交易用其包含的所有频繁项集的标识符替代。例如,T_1中的网站内交易 $s_2(p_3,p_4)$包含了 3 个频繁项集$\{p_3,p_4\}$、$\{p_3\}$和$\{p_4\}$,因此该网站内交易变为 $s_2(i_2,i_3,i_4)$。转换后的数据集为 DT,如表 3.3 所示。

表 3.3　数据集 DT

交易号(TID)	在 线 交 易
T_1	$\{s_1(i_1),s_2(i_2,i_3,i_4),s_3(i_5)\}$
T_2	$\{s_2(i_2,i_3,i_4),s_3(i_5),s_4(i_1)\}$
T_3	$\{s_1(i_1,i_4),s_2(i_1,i_3)\}$

第 3 步主要是基于 DT 构建一种成为 OSP 树的结构。以表 3.3 中所示的 DT 表为例,其 OSP 树如图 3.1 所示。树中每条路径对应一个在线购物模式。除了根结点外,其余每个结点代表一个在第 1 步中发现的项集。其中部分结点中除了记录项集外,还记录包含所在路径对应的在线购物模式的所有交易号。例如,树中左边第二条路径对应的在线购物模式为$\{i_1,i_2,i_5\}$(即$<p_1(p_3,p_4)p_5>$),包含该模式的交易号的集合为$\{T_1,T_2\}$,称为交易集。头表中记录了树中出现的每个不同的项集及指向树中该项集所在结点的指针,相同项集的结点通过链表链接在一起,便于在树中找到所有的相同项集。

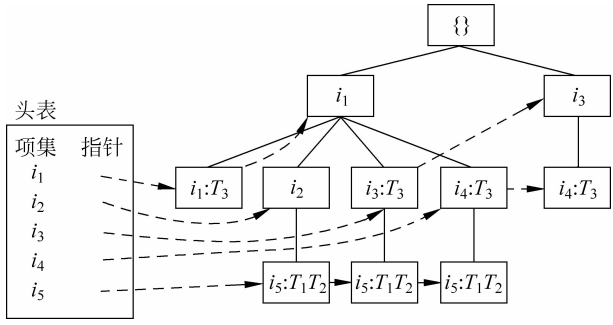

图 3.1　表 3.3 对应的 OSP 树

为了构建该树，首先将步骤 1 中发现的所有频繁项集，如表 3.2 中所示的$\{i_1,i_2,i_3,i_4,i_5\}$，按照某种顺序排序，例如可以按照支持度的大小或者字母序。此处假设按照字母序排序，即 $i_1\rightarrow i_2\rightarrow i_3\rightarrow i_4\rightarrow i_5$。在树的构建过程中，对于 DT 表中的每一行，从其中每个网站中取一个项集，按照前面指定的顺序排序，然后插入到树中。例如，对于表 3.3 中的第一行，可以从 $s_1(i_1)$取 i_1，从 $s_2(i_2,i_3,i_4)$中取 i_2，从 $s_3(i_5)$中取 i_5，构成在线购物模式 $\{i_1,i_2,i_5\}$，插入树中，对应的是图 3.1 中左边的第二条路径。

第 4 步通过 OSP 树发现所有频繁的在线购物模式。这个过程与从传统的交易数据库对应的 FP-tree 中发现频繁项集的过程(Han，*et al.*，2000)相似，区别在于支持度的计算方式。由于一个在线交易对应 OSP 树中的多条路径，例如对于表 3.3 中的第一行对应 3 条路径：$\{i_1,i_2,i_5\}$、$\{i_1,i_3,i_5\}$、$\{i_1,i_4,i_5\}$。这 3 条路径中 i_1 和 i_5 重复了 3 次，但是其支持度只能计数 1 次，因为仅被 T_1 这一个在线交易包含。为此，不能如同在 FP-tree 一样在插入每条路径时将每个结点的计数增 1，为了便于计算支持度，将包含该路径对应的在线交易模式的在线交易号记录在树中相应路径的最后一个结点中。利用这些结点中交易集可以计算在线交易模式的支持度。

下面通过一个例子说明从 OSP 树中发现所有频繁在线购物模式的主要过程。以如图 3.1 所示 OSP 树为例，从头表的最后一个项集 i_5 开始，首先发现所有包含该项集的在线交易模式，然后依次发现包含项集 i_4 不包含项集 i_5 的所有在线交易模式，包含项集 i_3 不包含项集 i_4 和 i_5 的所有在线交易模式，包含项集 i_2 不包含项集 i_3、i_4 和 i_5 的所有在线交易模式，最后是仅包含 i_1 的所有在线交易模式。这样所有的频繁在线交易模式都会被发现。在发现包含 i_5 的所有频繁在线交易模式的过程中，首先将 i_5 输出。然后，头表中的指针链表找到包含此项集的所有 OSP 树的路径，即路径 $i_1i_2i_5$、$i_1i_3i_5$ 和 $i_1i_4i_5$，据此可以构建包含项集 i_5 的条件数据库。将其中不频繁的项集删除之后，可以构建条件 OSP 树，如图 3.2 所示。

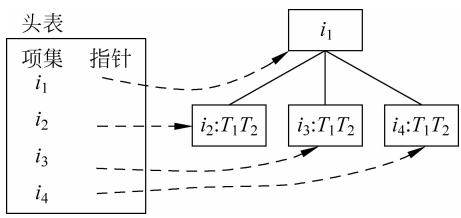

图 3.2 i_5 条件 OSP 树

从如图 3.1 所示的树构建如图 3.2 所示的树的过程中可以看到，结点 i_5 交易集被复制到了其上层结点 i_2、i_3 和 i_4 中，原因是它们同时与 i_5 出现在这些在线交易中。这样按照与 OSP 树同样的过程处理各个条件 OSP 树，可以输出所有满足最小支持度阈值的在线交易模式。注意，在处理如图 3.2 所示的条件 OSP 树时，输出的每个在线交易模式都要加上项集 i_5。例如在处理 i_4 时，首先输出的是 $i_5 i_4$。$i_5 i_4$ 的条件 OSP 树仅含有一个结点 $(i_1 : T_1 T_2)$。输出模式 $i_5 i_4 i_1$ 后就停止继续往下迭代而返回到处理图 3.2 中的项集 i_3，结果将输出 $i_5 i_3$ 和 $i_5 i_3 i_1$。同样，处理图 3.2 中的项集 i_2，结果将输出 $i_5 i_2$ 和 $i_5 i_2 i_1$。处理图 3.2 中的项集 i_1，结果将输出 $i_5 i_1$。此时图 3.2 中的所有项集处理完毕，返回至图 3.1 中的 OSP 树，接着处理项集 i_4。

在如图 3.1 所示的 OSP 树中处理项集 i_4 时，首先可以找到包含该项集的两个路径，如图 3.3 中的(a)和(d)所示。对于图 3.3 (a)中所示的第一条路径，很明显模式 $i_1 i_4$ 不仅出现在交易 T_3 中还同时出现在交易 T_1 和 T_2 中，因此，i_4 的交易集应该是其本身的交易集和其孩子结点交易集的并集，如图 3.3(b)所示。对于图 3.3(d)中所示的第二条路径，由于项集 i_3 与 i_4 仅共同出现在一个交易 T_3 中，不符合最小支持度 2 的阈值，因而被删除。最终 i_4 条件 OSP 树如图 3.3(c)所示。

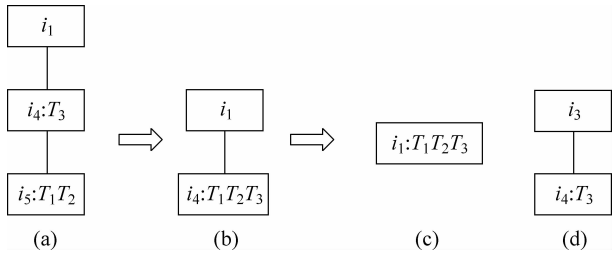

图 3.3　构建 i_4 条件 OSP 树

按照此过程继续处理项集 i_3、i_2 和 i_1，即可发现所有频繁在线交易模式。

最后在第 5 步中，将在第 4 步中发现的所有频繁在线交易模式参照第 2 步的映射过程将项集用原来的产品集替换，如用 (p_3, p_4) 替换 i_2。

下面说明利用算法 OSP-Tree 基于 OSP 树可以发现所有频繁的在线交易模式。

定理 3.1　给定在线交易数据库 DB 和最小支持度阈值 minsup，基于 OSP 可以发现所有满足 minsup 的频繁在线交易模式。

证明：首先，频繁的在线交易模式包含的每个项集是频繁的，因此，在

第 1 步中发现了所有频繁的项集。另外，每个在线交易模式的项集来自不同的网站，网站内交易中的商品用频繁项集进行替换。在第 3 步中将包含在一个在线交易中的不同网站中的项集的所有组合都抽出存放在 OSP 树中。这种组合针对每个在线交易都是最长的，最终频繁的在线交易模式来自于这些组合，因此所有信息都包含在 OSP 树中了。所以从原来的交易数据库 DB 发现频繁模式转变为从 OSP 树中发现频繁的在线交易模式。

在遍历 OSP 树的过程中，每个路径的项集的所有组合都通过一种递归的分而治之的方法进行处理了。以图 3.1 中的 OSP 为例，项集 i_1、i_2、i_3、i_4 和 i_5 的所有组合可以分为五类：所有包含项集 i_5 的在线交易模式，包含项集 i_4 不包含项集 i_5 的所有在线交易模式，包含项集 i_3 不包含项集 i_4 和 i_5 的所有在线交易模式，包含项集 i_2 不包含项集 i_3、i_4 和 i_5 的所有在线交易模式，最后是仅包含 i_1 的所有在线交易模式。这样所有可能存在的在线交易模式都被检查是否符合阈值，频繁的在线交易模式均会被发现。

表 3.4 是在如表 3.1 所示的数据中发现的所有频繁的在线购物模式。

表 3.4 所有频繁的在线购物模式（minsup＝2）

在线购物模式	p_1	p_3	p_4	p_5	(p_3,p_4)		
支持度	3	3	3	2	2		
在线购物模式	$p_1(p_3p_4)$	p_1p_3	p_1p_4	p_1p_5	$(p_3p_4)p_5$	p_3p_5	p_4p_5
支持度	2	3	3	2	2	2	2
在线购物模式	$p_1p_3p_5$	$p_1p_4p_5$	$p_1(p_3p_4)p_5$				
支持度	2	2	2				

3.1.3 挖掘其他类型的购物模式

前面介绍的基于 OSP 树的算法和基于层次的算法不仅可以用于发现频繁的在线购物模式，还可以用于发现其他类型的购物模式。下面列出的是三种不同类型的模式。

（1）类型 1：形如 $\{(p_{d1}^1, p_{d1}^2, \cdots, p_{d1}^{f1}), (p_{d2}^1, p_{d2}^2, \cdots, p_{d2}^{f2}), \cdots, (p_{de}^1, p_{de}^2, \cdots, p_{de}^{fe})\}$ 购物模式，与定义 3.3 定义的模式相同。

（2）类型 2：形如 $\{s_{d1}(p_{d1}^1, p_{d1}^2, \cdots, p_{d1}^{f1}), s_{d2}(p_{d2}^1, p_{d2}^2, \cdots, p_{d2}^{f2}), \cdots, s_{de}(p_{de}^1, p_{de}^2, \cdots, p_{de}^{fe})\}$ 的购物模式，其中 $s_{di} \neq s_{dj}(i \neq j, i, j \in \{1,2,\cdots,e\})$ 是带有网站信息的购物模式。

（3）类型 3：对于形式如下的购物模式

$$\{s_{d1}, s_{d2}, \cdots, s_{dk}, (p_{d(k+1)}^1, \cdots, p_{d(k+1)}^{f(k+1)}), \cdots, (p_{dl}^1, \cdots, p_{dl}^{fl}),$$

$$s_{d(l+1)}(p^1_{d(l+1)},\cdots,p^{f(l+1)}_{d(l+1)}),\cdots,s_{de}(p^1_{de},p^2_{de},\cdots,p^{fe}_{de})\}$$

其中模式的每个构成元素可以是网站、项集或带网站的项集。

对于类型 3，如果一个模式的所有元素均为网站如$\{s_{d1},s_{d2},\cdots,s_{de}\}$或均为产品如$\{p_{d1},p_{d2},\cdots,p_{de}\}$（与类型 1 不同，它不区分是否来自不同网站），则可以利用传统的频繁项集的挖掘方法进行发现（Agrawal，*et al.*，1993；Han，*et al.*，2000）。如果只需发现类型 2 的模式，则可以通过修改已有的多层次关联规则的挖掘方法（Han and Fu，1999）或交易间关联规则的挖掘方法（Tung，*et al.*，1999；Feng，*et al.*，2002）来实现，可以将网站和商品进行组合，如将 s_1p_1 当作一个项。但是这种方法无法同时发现类型 1 和类型 3 的模式。而且，在多层次关联规则的挖掘算法中，商品之间的关系是树状结构，而我们的方法可以处理如图 3.4 所示的二部图关系结构，其中，一种商品可以在多个网站上购买，在一个网站上也可以购买多种商品。并且发现模式对应的关联规则的左右两边可以包含相同的商品，如（book，CD）→（book，movie）。

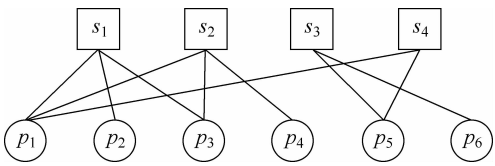

图 3.4　商品和网站之间的二部图关系结构

尽管前面介绍的 OSP-Tree 和 OSP-Level 算法可以直接用来发现类型 1 在线购物模式，只需通过如下的改动，利用本书介绍的方法可以同时发现上述三种类型的模式，将网站 s_i 与在此网站上购买的所有商品同等对待，作为一个项处理，即把表 3.1 变为表 3.5 即可。

表 3.5　更新后的交易数据库 DB

TID	OS 交易
T_1	$\{(s_1,p_1,p_2),(s_2,p_3,p_4),(s_3,p_5)\}$
T_2	$\{(s_2,p_3,p_4),(s_3,p_5,p_6),(s_4,p_1)\}$
T_3	$\{(s_1,p_1,p_4),(s_2,p_1,p_3)\}$

基于此表，利用 OSP-Tree 或 OSP-Level 算法即可发现各种可能类型的在线购物模式。注意，如果最终输出的是模式中包含如$(s_i,p_{i1},p_{i2},\cdots,p_{ik})$的项集，则只需将其转换为 $s_i(p_{i1},p_{i2},\cdots,p_{ik})$即可。

3.1.4 实验及案例分析

为了验证所提算法的性能，分别在实际的用户网购数据集和合成数据集上进行实验。下面先介绍在实际数据集上发现的频繁在线购物模式，然后介绍在合成数据集上对两种算法效率和空间上的比较。

用户网购数据集是由 comScore 公司提供。数据集包括 50 000 个消费者在 2004 年 1 月到 10 月间的 10 个月内的网购数据，涉及商品种类 61 种。50 000 个人中的 24 824 人在此期间进行了网购，其中至少 62% (15 416 人)在两个和两个以上的网站上进行过购物。每个被调查的消费者对应一台被追踪的计算机，因此，我们从中挑选了家庭人数只有一人的进行实验。我们将一个消费者一个月内的购物合并为一个在线交易。经过这样处理之后的交易数据库包含 75 053 个在线交易。设最小支持度阈值 minsup=10，用产品类型代替产品进行挖掘。这样在最终的数据集中，一个用户购物的不同网站个数最大是 40，最小为 1，平均值为 3。在一个月内至少在 2 个网站上网购过的用户的个数是 4771。图 3.5 是有关消费者人数和网站个数的关系图，图中每个点表示的是在给定个数的网站上进行过购物的消费者人数。图 3.6 则是消费者人数与商品类型个数的分布图，即购买过相应商品种类数的消费者人数。平均来说，一个消费者在 10 个月中购买了 3.35 种不同商品。一个消费者一个月内所购商品不同种类的最大值是 14。

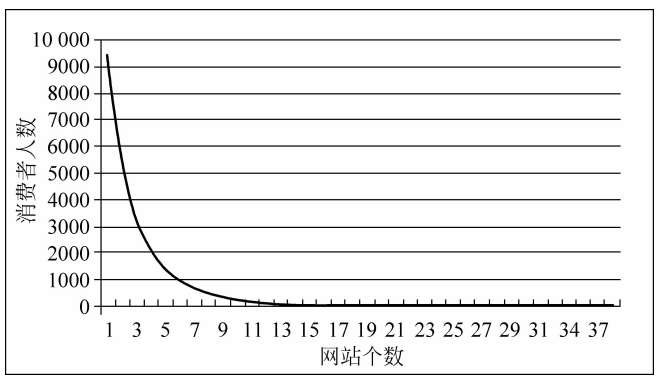

图 3.5 消费者人数与网站个数关系

使用 OSP-Level 和 OSP-Tree 算法发现的频繁在线购物模式是相同的。在发现在线购物模式中，有 116 个包括 2 个项集，3 个包括 3 个项集。表 3.6 列出了其中一些有意义的模式及其支持度。

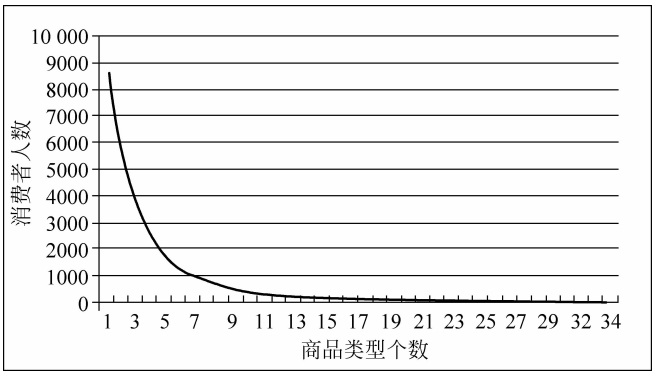

图 3.6　消费者人数与产品类型个数关系

表 3.6　发现的频繁在线购物模式例子

频繁在线购物模式	支　持　度
{Air Travel,Hotel Reservations,Car Rental}	21
{Hotel Reservations,(Air Travel,Car Rental)}	19
{Air Travel,Hotel Reservations}	167
{Air Travel,Car Rental}	201
{Hotel Reservations,Car Rental}	134
{Event Tickets,Hotel Reservations}	38
{Music,Movies & Videos}	70
{Books & Magazines,Movies & Videos}	74
{Health & Beauty,Books & Magazines}	45
{Apparel,Shoes}	85
{Apparel,Jewelry & Watches}	41
{Apparel,Kitchen & Dining}	26
{Apparel,Bed & Bath}	44
{Apparel,Health & Beauty}	76
{Apparel,Books & Magazines}	95

从表 3.6 可以看到，人们常常在一个网站上购买机票，然后在其他网站上预订旅店和租车；同样，在一个网站上购买衣物，在另外的网站上购买配饰。商家如果要拓展销售商品的种类的话，根据这种模式便于商家决定增加哪些商品，或者投放何种商品的广告更能提高点击率或购买率。例如，在销售门票的网站上投放当地旅馆的广告等。

按照 3.1.4 节中介绍的方法，也可以发现其他类型的模式，下面是几个例子：

{bmgmusic.com (Music),columbiahouse.com(Movies & Videos)}sup=49

{Hotel Reservations,southwest.com (Air Travel)}sup=45

{ebay.com,amazon.com (Books & Magazines)}sup=80

{Books & Magazines,ebay.com}sup=110

{ebay.com,amazon.com}sup=105

尽管这些模式并不那么出乎人的预料，但是至少通过提出的方法可以找到存在的各种频繁模式，也可以验证一些可能存在的猜想。

下面介绍在合成数据集上的性能比较。由于真实数据集的大小限制，为了便于验证所提方法在各种不同类型数据集上的性能，我们首先利用数据生成器生成合成数据集。我们采用的是 IBM 研究员开发的生成器(Agrawal and Srikant,1995)，生成数据原本用于序列模式的研究。为了适合本书所提问题，我们加上了网站信息，即为一个序列中的每个交易加上网站信息。表 3.7 列出了该数据生成器的主要参数以及与生成器参数的映射关系。

表 3.7 数据生成器参数

原有生成器参数	本书中对应参数	符号
消费者个数	在线交易个数	T
每个消费者的平均交易个数	一个在线交易包含的网站的平均个数	S
一个交易包含的项的平均个数	一个网站上所购商品的平均个数	P
不同项的总个数	不同商品的个数	I

为了评估各种参数的取值对算法性能的影响，分别对表 3.7 中的 4 个参数以及最小支持度阈值变换其取值，生成不同的数据集，比较两种算法在这些数据集上的运行时间和空间消耗。一个数据集各参数的取值用符号 T、S、P、I、minsup 表示，如 T5000、S2、P2、I100、minsup0.01 表示 $T=5000$、$S=2$、$P=2$、$I=100$、minsup=0.01(相对支持度)，即 5000 个在线交易，每个交易平均包含 2 个网站，每个网站平均购买 2 种商品，不同商品(或种类)的个数为 100，最小支持度阈值为 0.01。

图 3.7 告诉我们，算法 OSP-Tree 在支持度阈值比较小时效率更高，图 3.8、图 3.10、图 3.12 和图 3.14 均显示 OSP-Level 消耗的内存空间更少。从图 3.9 可以看到，随着在线交易个数的增长，算法 OSP-Level 的运行时间增长更快。

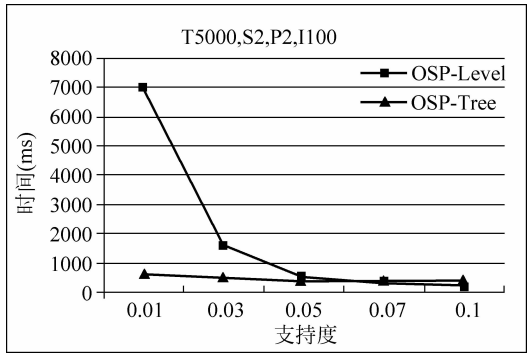

图 3.7　运行时间随支持度的变化

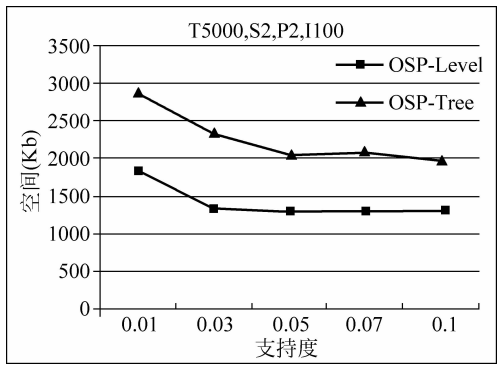

图 3.8　空间随支持度的变化

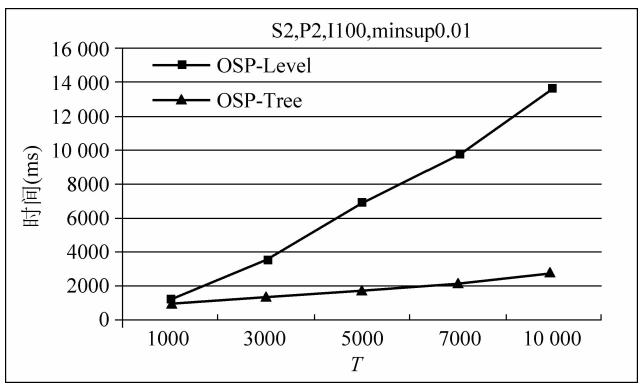

图 3.9　在线交易个数对运行时间的影响

图 3.11 和图 3.13 显示，随着 P 和 S 的增长，算法 OSP-Tree 的运行时间增长更快。

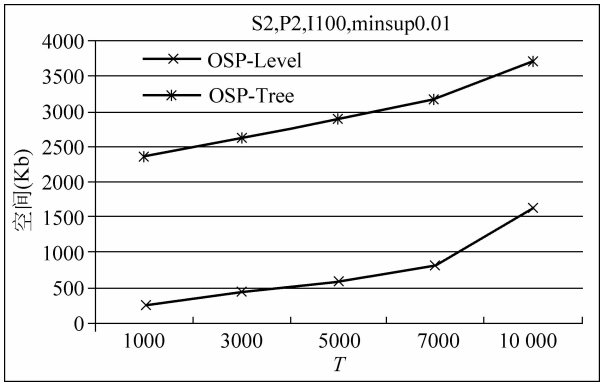

图 3.10 在线交易个数对空间的影响

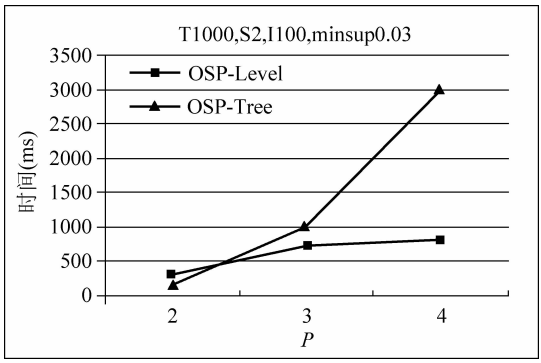

图 3.11 平均商品个数 P 对运行时间的影响

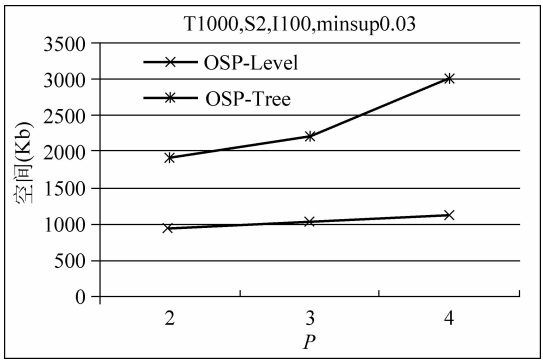

图 3.12 平均商品个数 P 对空间的影响

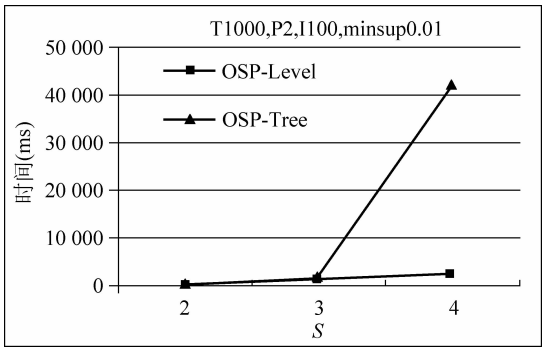

图 3.13　平均网站个数 S 对运行时间的影响

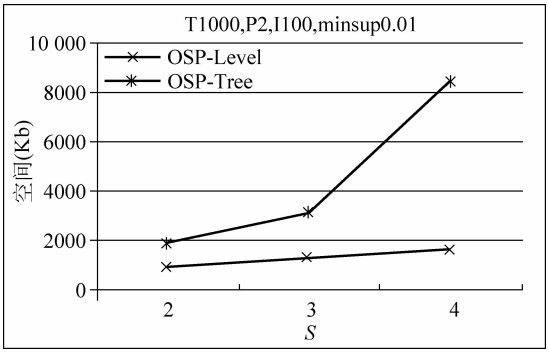

图 3.14　平均网站个数 S 对空间的影响

从图 3.15 可以看出，随着不同商品个数的增大，运行时间下降，这是由于在 T 固定的情况下，I 增大使得每个项的支持度降低，满足最小支持度阈值的频繁模式减少。这些实验结果也显示，多数情况下，算法 OSP-Tree 比 OSP-Level 快。

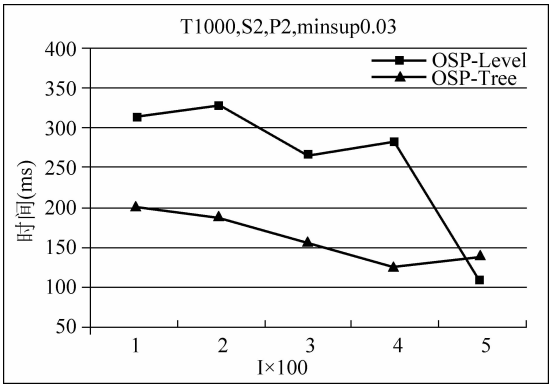

图 3.15　不同商品个数对运行时间的影响

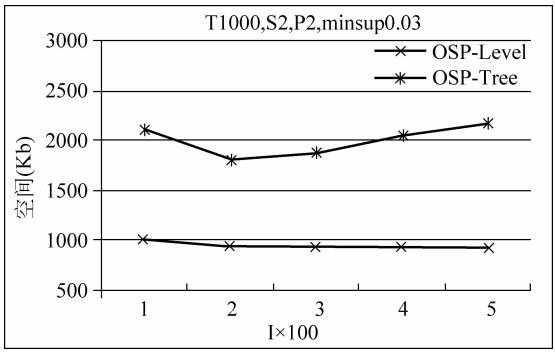

图 3.16 不同商品个数对空间的影响

3.1.5 相关工作

与此相关的工作主要分为四类：关联规则的挖掘、多层次关联规则的挖掘、交易间关联规则的挖掘以及序列模式的挖掘，下面分别介绍。

关联规则的挖掘最早用于进行购物篮分析，由 IBM 的研究者于 1993 年提出(Agrawal，Imielinski and Swami，1993)。关联规则是形如 $X \rightarrow Y$ 的规则，其中 X 和 Y 分别称为项集，是项(item)的集合。在购物篮分析中每个项就是一件商品。一个项集是一个顾客一次所购买商品的一个子集，因此关联规则是发现商品之间的关联关系。用于发现关联规则的数据集通常称为交易数据库(transaction database)，其中每一行(对应一个交易)记录了一个顾客一次所购买的所有商品。一条规则的显著性通过支持度和置信度衡量。规则 $X \rightarrow Y$ 的支持度是同时包含项集 X 和 Y 的行数的比例，而其置信度则是包含 X 的行中同时包含 Y 的行所占的比例。关联规则的概念和基础算法一经提出便引发了众多研究者的关注和后续研究，其中一些有代表性的挖掘算法包括 Apriori (Agrawal and Srikant，1994)、Eclat (Zaki，2000)、FP-growth (Han，Pei and Yin，2000；Han，*et al.*，2004)、TreeProjection (Agarwal，*et al.*，2000) 及 PRICES (Wang and Tjortjis，2004)等。这些算法的细节可以参考文献(Kotsiantis and Kanellopoulos，2006)。本书提出的在线购物模式的定义与传统的关联规则的定义有很大区别，利用关联规则的发现算法无法发现所有的在线购物模式。

多层次关联规则是对传统关联规则的一种扩展 (Han and Fu，1999；Srikant and Agrawal，1995)，用于发现有层次关系的项之间的关联关系。例如，牛奶→面包、牛奶→脱脂面包等。本书所提出的频繁模式与此种关联

规则是不同的。首先，在多关系关联规则中，项之间的关系是层次关系，通常用树的数据结构表示，而本书所提出的方法可以利用商品和网站之间的关系发现复杂的频繁模式，这种关系是用二部图的数据结构表示的，如图 3.4 所示，其中一种商品可以在多个网站上购买，在一个网站上也可以购买多种商品。其次，最重要的是多层次关联规则对应的频繁模式的结构是不同的。

传统的关联规则发现的是交易内的项集之间的关联关系，交易间关联规则用于发现包含于不同交易的项集之间的关联关系。发现交易间关联规则的问题最早是由香港科技大学的陆宏钧等学者(Lu，*et al.*，1998)。之后多名学者对此进行了进一步的研究，提出了许多高效的挖掘算法，如 E-Apriori 和 EH-Apriori (Lu，*et al.*，2000)、FITI (Tung，*et al.*，1999) 及 ITP-Miner (Lee and Wang，2007) 等。另外，学者们也研究了用于减少所发现规则数目的基于模板的挖掘算法(Feng，*et al.*，1999；Feng，*et al.*，2002)以及交易间闭合频繁模式的挖掘方法(Chen，*et al.*，2005)。交易间关联规则有很多应用场合，例如用于股票价格波动的预测 (Lu，*et al.*，1998；Lu，*et al.*，2000；Tung，*et al.*，1999) 以及天气预报等 (Feng，*et al.*，2001)。与本书介绍的方法相比，交易间关联规则的挖掘方法无法发现所有类型的在线购物模式。

序列模式(sequential pattern)的挖掘也与本书介绍的工作有关。序列模式也是频繁模式的一种，它是序列数据库中的频繁子序列(Agrawal and Srikant，1995；Srikant and Agrawal，1996；Pei，*et al.*，2004)。序列模式最早提出时也用于进行购物分析。例如，<(床单，枕头)，床罩，窗帘>是一个典型的序列模式，它代表了用户购物的一个有代表性的行为模式，即在一次购买了床单和枕头之后的一段时间内顾客会接着购买床罩，过一段时间又会买窗帘。序列模式的一个重要特点是其时序性，在线购物模式不强调时序性，更具有概括性。

传统的关联规则涉及两个维度的信息：用户以及项(物品)。当涉及多个维度时不能直接使用已有的关联规则发现方法。例如，当要分析用户(user)、标签(tag)以及资源(resource)之间的关联关系时，需要对三维信息进行转化(Schmitz，*et al.*，2006)。转化方法为将其中两个维度组合在一起作为一个维度，例如，将用户与标签组合或者与资源组合。这种方法只能发现在线模式所包含的部分类型的模式。

3.2 交易行为模拟

在研究关联规则的挖掘算法的过程中，为了测试和检验算法的有效性，除了使用真实的数据集如超市购物数据集之外，通常还要用到合成数据集，以便产生具有各种特点的数据集，测试算法的各方面特性，例如，对密集数据集和稀疏数据集的处理，或者对短交易或者长交易的处理特性等。为此Agrawal等人在文献(Agrawal and Srikan，1994)中首先提出了一个用于模拟人的交易行为的非层次交易数据流生成器，主要用于关联规则研究工作。后来又在文献(Srikant and Agrawal，1995)中扩展了这个交易数据流生成器，使其能够从一个用户定义的概念层次中抽取数据项，而最终的输出仍然是为多层次关联规则算法使用的交易数据流，这种数据流在下面被称做不定维数据流(解释详见3.2.2节)。

数据流(data stream)是一种只能被读取一次或少数几次的点的有序序列(S. Guha，*et al.*，2003)。例如，如果把超市中每次交易当作一个点，按时间先后不断完成的交易组成一个数据流。数据流是一种现实生活中普遍存在的与静态数据相对应的数据存在形式，对数据流进行管理以及从中提取有用信息成为数据管理和数据挖掘领域的一个重要的发展方向。与一般数据集相比，数据流通常具有长度无限、取值范围不可预期、特征随时间变化等特点。由于数据流研究的特殊性，在算法测试过程中通常也要用到大量的人工生成数据。数据的层次信息最早由广义关联规则(Srikant and Agrawal，1995)的概念引入，随后出现了一些相关的研究(Han and Fu，1995)，对数据流的研究也逐渐将层次信息列入考虑范围之内(Cormode，*et al.*，2004；Hershberer，*et al.*，2005)。随着对多层次数据流研究的发展，传统算法研究如文献(Metwally，*et al.*，2005；Manku and Motwani，2002)介绍的数据流生成器，由于不能构造数据项的概念层次结构以及改变数据流格式而变得不再适用。本书提出一种新的人工层次数据流生成器HIDS(HIerarchical Data Stream)，用于解决这些问题，它可以按照用户要求随机构造出贴近现实的概念层次结构，既可以生成固定维数的数据流，也可以生成维数可变的数据流，同时兼容了一般数据流生成器的功能。HIDS可以满足目前大多数数据流研究的要求。

和以往的人工数据生成器不同的是，本书提出的层次数据流生成器是具有多种功能的，而不仅仅用于生成类似超市商品交易这样的不定维非层

次数据流，它还能够生成固定维数的数据流。在生成不定维数据流的时候，考虑到这种形式的数据流主要用于模拟人的交易行为，我们借鉴了文献(Srikant and Agrawal，1995)中的思路，同时加入了若干针对概念层次定制的新策略。虽然其中使用的生成器也可以产生层次数据流，但是我们的生成原理及使用方式与其有着很显著的差别。

下面先简要介绍现实中存在的几种常见的概念层次结构，接着将介绍本书提出的人工层次数据流生成器的原理，最后将用一个实际的例子来比较说明该生成器产生的概念层次的合理性。

3.2.1　数据的层次结构

数据流中的数据项通常会带有层次信息。例如 IP 地址就是典型的带有层次信息的数据，这种层次关系是由表达 IP 地址的数字客观决定的；而对于地点这样的属性，尽管其层次关系也是客观存在的，却通常不直接表示出来；另外，还有一些如超市中的商品的层次信息，主要是靠人为因素来决定的。下面分别讨论各种不同的层次信息携带方式。

1. 自然层次

前面把 IP 地址看成显性携带层次信息，把地点等属性看成隐性携带层次信息，两者本质上的共同点是数据所在的域有着可以准确定义的相对固定的概念层次结构，本书将其称作自然层次，它不由人的意志来决定。

例如，在 32b 的 IP 地址中，从左至右每 8b 作为一层，总共为五层，最上层的 * 为根结点，包含了整个 IP 域，如图 3.17 所示。

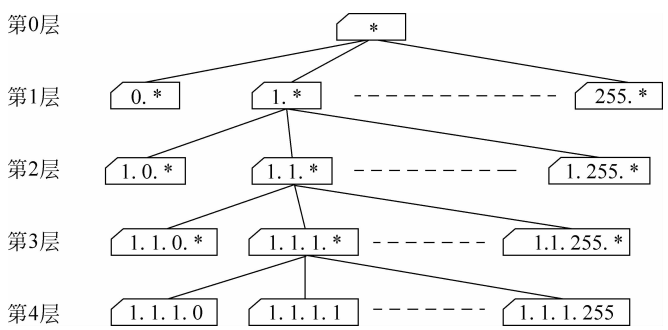

图 3.17　IP 地址的层次结构

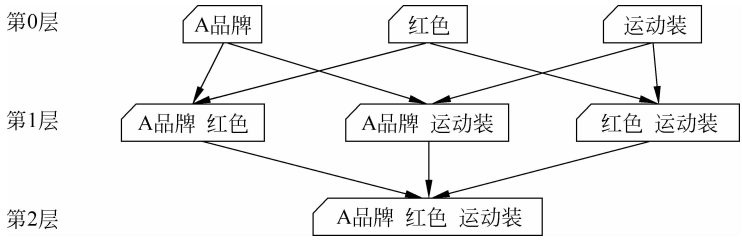

图 3.18 完全层次示例

值得注意的是，像地点这样的属性虽然是隐性地带有层次信息，但这种层次还是有着固定结构的，例如国家、省、市、区等就是这个层次结构中的一部分。与 IP 地址不同的是，我们实际接触到的地点层次结构通常只是其完整结构的一个子集，因为通常使用完整结构是没有必要的。

2. 人为层次

诸如超市商品那样的数据层次结构也是常见的，如表 3.8 所示是超市常用的层次分类方案。这种层次结构得到了广泛应用，是因为它更便于管理，并且更容易被人的直觉所理解和接受。

表 3.8 超市应用的商品层次结构

目录	大分类	中分类	小分类	单品(包含所有细节特征)
10000 百货	11000 服装	11100 男装	11110 西服	11111 A 品牌 XF3523 型号西服
				11112 B 品牌 S533 型号西服
			11120 运动服	11121 C 品牌 Y531 型号运动服
		11200 女装	11210 女裙	11211 A 品牌 Q75 型号女裙
				11213 C 品牌 C43 型号女裙
			11220 衬衫	11221 B 品牌 CH66 型号衬衫
	12000 鞋帽	12100…	…	…

然而抛开直觉的束缚，我们会发现，商品本身(指表 3.8 中的单品一级)的各个属性本来是处于同一个级别的，并没有高低之分，我们完全可以构造出如表 3.9 所示的层次分类，尽管这种层次结构并不适合超市的应用环境。也就是说，现实中使用的概念层次有些并不是唯一的构造形式，适合应用的结构也并不一定适合数据挖掘操作，有时根据自己的需要来重新构造层次则可能得到更有用的知识。

表 3.9　一种合理的商品层次结构

目录	大分类	中分类	小分类	单品(包含所有细节特征)
10000 百货	11000 品牌 A	11100 男式	11110 西服	11111 A 品牌 XF3523 型号西服
		11200 女式	11210 女裙	11211 A 品牌 Q75 型号女裙
				11212 A 品牌 Q67 型号女裙
	12000 品牌 B	12100 男式	12110 西服	12111 B 品牌 S533 型号西服
			12120 衬衫	12121 B 品牌 CH66 型号衬衫
				12122 B 品牌 CH67 型号衬衫
	…	…	…	…

3. 完全层次

这种层次结构衍生于人工层次，不是现实当中可以使用的结构。事实上，它太全面因而带有过多信息，对于任何一种现实应用都是有冗余的，然而它真正揭示了全部的挖掘可能性。任何一个人工层次都是完全层次的一个子集。

仍然以超市的商品为例，如图 3.18 所示，“A 品牌红色运动装”是一个叶子结点，完全层次就是把它的所有属性平等对待，用组合的方式得到每一个可能的上层结点，形成网络结构。其中“A 品牌红色”、“A 品牌运动装”和“红色运动装”都是潜在的上层结点，在现实应用中通常只会选择其中一个。

3.2.2　人工层次数据流生成器

该生成器可以生成如下多种数据流，涵盖了常见的数据流结构。

定义 3.6(一维数据流)　即每次只到达一个数据项的数据流，即点对应一个数据项。

定义 3.7(n 维数据流)　即每次到达 n 个数据项，是一维数据流的一般形式。

定义 3.8(不定维数据流)　每次到达的项数是随机的，例如超市收银台的交易数据流。

与以往的数据流生成器不同，该生成器产生的数据项来自于一个根据用户参数随机生成的层次结构，确切地说，数据流中的每一项都是这个层次结构的一个叶子结点。在生成器产生的数据项中，我们以字典序的方式用数字表示每一层的结点信息。从图 3.19 中可以清楚地看到这种表现方式，其中结点[1,4,1,2]的含义即结点[1,4,1]的第 2 个子结点。

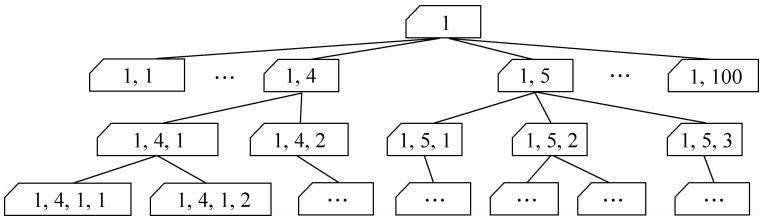

图 3.19 生成器构造的概念层次表示方法举例

本书提出的层次数据流生成器 HIDS 支持多种层次结构。从层次的深度上看，可以生成所有叶子结点等深的结构和满足给定偏差的平衡结构（与平衡二叉树类似的概念）；从结点的扇出度上看，可以生成等扇出度的结构和扇出度满足特定分布的结构。

当然，该生成器是向下兼容的，也可以生成不含有层次信息的数据流。

与非层次数据流生成器不同，这个生成器算法主要有两个步骤：一是构建概念层次结构；二是生成任意长度的数据流。

1. 建立概念层次结构

此生成器在生成数据流的时候只建立一棵树形的层次结构，数据流中任何维度都从这一棵树中提取叶子结点作为取值。对于那些应用于多维数据流的挖掘算法，其核心在于用怎样的数据结构来表达多维的复杂关系，算法本身只要知道各个维度都是来自相应的树结构即可，并不会辨别两棵树是否相同，因此从这个角度讲，生成多棵树给不同的维度使用的想法暂时不用考虑。

图 3.20 显示出了生成器在建立层次结构及生成数据流时的参数设置过程。表 3.10 和表 3.11 分别列举了构建层次结构及生成数据流时所需的参数。

表 3.10 建立概念层次的参数

参　数	描　述	默 认 值
H	树的最大深度	5
DH	树的最大偏差	0
FU	结点扇出度的取值上界	15
FL	结点扇出度的取值下界	5
N	叶子结点总数	10 000
P	zipf/正态分布偏差参数	1/3

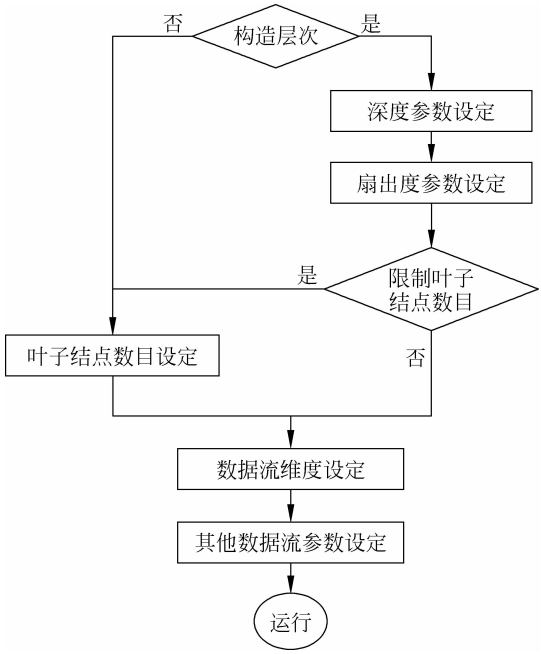

图 3.20　生成器的参数设置过程

表 3.11　生成数据流的参数

参　数	描　述	默 认 值
D	数据流长度	1 000 000
T	同时到达的数据项数目(均值)	10
L	潜在数据项集个数	10 000
I	潜在数据项集大小的均值	4
P	zipf 分布偏差参数	0.9

首先要进行选择的是是否构建层次结构,如果不需要,则直接进入生成数据流的过程中,即与一般数据流生成器无异。

第二步是进行层次树的深度设定。在用户给定参数 H 及 DH 之后,生成器将构造最大深度为 H,最大深度偏差为 DH 的平衡树,即树中任何一个中间结点的所有分支的深度最大相差不超过 DH。当 DH＝0 的时候,树中所有叶子结点将处于同一层,如超市的商品层次结构。

第三步是进行中间结点的扇出度设定。在这一步用户除了需要输入 FU 与 FL 两个参数以外,还需要选定一种分布。当前版本的生成器提供

zipf 分布(Zipf,1949)、正态分布及均匀分布，用户需要给定 zipf 分布的不均匀参数及正态分布的标准差参数。设定完成之后生成器会对每一个中间结点按照指定分布参数计算扇出度。当 FU＝FL 时，将构造出如 IP 地址那样的等扇出度层次结构。最后要选择是否对叶子结点数目进行限制，设计这一步的目的是有些实验会倾向于使用比较规范的取值个数，如 10k。由于建树的过程带有随机性，初次构建的结果通常不会达到设定值，所以一旦选择对叶子结点数目进行限制，生成器会自动对最终的叶子结点数量进行调整。然而在调整之前，生成器会根据用户设定的深度及扇出度参数对叶子结点的数目做估计，如果用户设定的限制个数偏离太多则会被要求调整参数。

2. 生成数据流

首先介绍不定维数据流的生成方法。由于不定维数据流主要用于进行频繁模式的分析，所以我们根据人的行为模式来设计随机变量的组合。以零售交易的模式为例：人们在购物的时候总是倾向于同时购买一些东西，可以把这些通常被同时购买的商品看成若干个商品集。例如，一个可能的商品集是床单、枕套、被褥等，而另一个商品集是牛奶、面包、火腿等。这样划分商品并不是说一个商品集中的商品总是都被购买，人们可能只会在一个商品集中选择部分商品。在一次交易中，人们通常会在若干个商品集中挑选商品。交易的大小即交易包含的商品个数围绕一个均值波动，只有少数交易波动会很大。同样，商品集的大小也是围绕一个均值波动，只有少数商品集会很大。具体的生成过程包含两个主要步骤，构造商品集和生成数据流，如下所示。

算法：HIDS

构造 L 个商品集

1. 随机构造第一个商品集
2. 循环开始(1≤i≤L－1)
3. 　由均值为 I 的泊松分布生成当前商品集尺寸 p
4. 　由指数分布生成关联度 r
5. 　根据 r 从上一个商品集中随机抽取商品
6. 　从整个商品集中随机抽取知道总数达到 p
7. 　由正态分布生成冲突系数 c
8. 　由指数分布生成权重系数 w
9. 循环结束

生成长度为 D 的数据流

1. 循环 1 开始(1≤i≤D)
2.　由均值为 T 的泊松分布生成当前交易尺寸 p
3.　循环 2 开始(直到当前交易尺寸达到 p)
4.　根据 w 随机抽取商品集 s
5.　将 s 全部放入当前交易中
6.　循环 3 开始(直到将 s 中的商品全部删掉)
7.　生成随机数 u
8.　如果 u<c,则从 s 中随机删掉一个商品
9.　否则退出循环 3
10.　循环 3 结束
11.　循环 2 结束
12.　输出当前交易
13. 循环 1 结束

对于一维数据流,由于数据流每次只有一个数据项到达,故不需要复杂的行为模拟。这里按照用户设定的 zipf 分布来抽取数据项,对层次结构中的所有叶子结点同等看待。

生成 n 维数据流可以看作是生成不定维数据流的一个子集,只要每一行同时输出的数据项数目恒定为 T 即可,而不需要通过均值为 T 的泊松分布来确定。其他方面都和不定维数据流生成过程一致,故其过程不再详述。

该生成器的输出结果为一个描述概念层次结构的文件以及一个数据流文件,前者包含每个层次上的结点数目以及每个中间结点的子结点数目。

3.2.3　测试

除了利用各种参数控制数据流的特性外,由于生成器的随机性及现实的多样性导致了无法定义精确的测度来衡量一个人工数据生成器的性能,为此我们找到一个真实数据流的数据库,一个音乐网站的文件目录信息及歌曲下载日志,如下所示:

```
58.66.121.67 -- [12/Nov/2006:04:02:29 + 0800] "GET  /music/Songs/Chinese/
Female/张靓颖/2006.10 % 20The % 20One/04.如果爱下去.mp3 HTTP/1.1" 206
20074 " - " "Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1; Maxthon)"
```

其中,文件目录信息隐含歌曲的概念层次,而每一条下载日志就是一个一维数据流中的元素。经过对目录信息的抽取,我们提取出一个 8 层 57 000 个叶子结点的层次结构,其结点扇出度近似符合参数为 1.3 的 zipf 分布,并且与 HIDS 在相同参数下的结果非常接近,如图 3.21 和图 3.22 所示。图 3.21

是两者结点扇出度分布的截取图，图中横轴表示按扇出度排序的结点编号，纵轴表示结点扇出度值。主要的区别是生成器的结果与理想的分布比较接近，而真实数据偶尔会有较大的波动。图 3.22 显示的是下载日志与生成器产生的 zipf 参数为 0.8 的一维数据流的局部比较，可以看到生成器的模拟很接近真实数据。

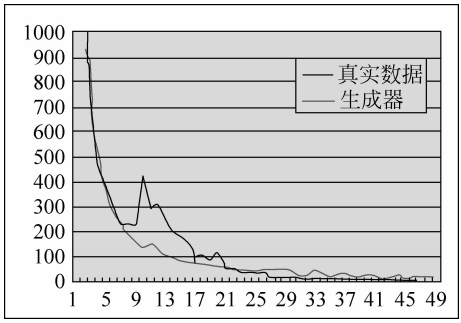

图 3.21 层次结构比较

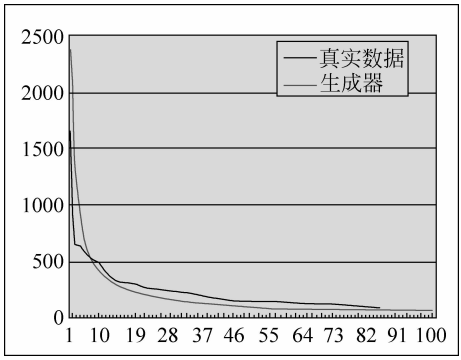

图 3.22 数据流分布比较

3.2.4 结论

本节提出一种新的人工层次数据流生成器 HIDS，它可以按照用户对层次深度及扇出度的要求随机构造出概念层次结构，包括特殊的等深度层次和等扇出度层次以及一般的不等深或者扇出度依赖某种特定分布的层次结构。HIDS 既可以生成固定维数的数据流也可以生成维数可变的数据流，涵盖了常见的数据流结构，可以满足目前大多数数据流研究的要求。同时该生成器是向下兼容的，可以生成不含有层次信息的数据流。

根据特定的需要，该生成器很容易扩展为可以针对各个层次分别设定扇出度分布，也可以扩展为每个数据流维度拥有不同的层次结构，其基本实现原理相似。

参考文献

Agarwal，R.，C. Aggarwal，V. Prasad. 2000. A tree projection algorithm for generation of frequent itemsets. *J. Parallel and Distributed Computing*.

Agrawal，R.，T. Imielinski，A. N. Swami. 1993. Mining association rules between sets of items in large databases. *Proc. 1993 ACM SIGMOD Int. Conf. on Management of Data*，207-216.

Agrawal，R.，R. Srikant. 1994. Fast algorithms for mining association rules. *Proc. 20th VLDB Conference*，487-499.

Agrawal R. and R. Srikant. 1995. Mining sequential patterns. *Proc. 1995 Int. Conf. Data Engineering*，Taipei，Taiwan，Mar. 3-14.

Chen，J.，L. Ou，J. Yin，J. Huang. 2005. Efficient mining of cross-transaction web usage patterns in large database，*Proc. of 3rd Int. Conf. on Networking and Mobile Computing*，519-528.

Cormode，G.，F. Korn，S. Muthukrishnan，D. Srivastava. Diamond in the Rough：Finding Hierarchical Heavy Hitters in Multi-Dimensional Data. *Proceedings of the ACM SIGMOD Conference on Management of Data*，2004，June：155-166.

Feng，L.，T. Dillon，J. Liu. 2001. Inter-transactional association rules for multi-dimensional contexts for prediction and their application to studying meteorological data. *Data & Knowledge Engineering* 37(1) 85-115.

Feng，L.，H. Lu，J. Yu，J. Han. 1999. Mining inter-transaction associations with templates. *Proc. 1999 ACM CIKM*，225-233.

Feng，L.，J. X. Yu，H. Lu，J. Han. 2002. A template model for multidimensional inter-transactional association rules，*The VLDB Journal* 11(2) 153-175.

Guha，S.，N. Mishra，R. Motwani，L. O'Callaghan 2003. *IEEE Transactions on Knowledge and Data Engineering*，Volume 15 Issue 3，March 2003. Page 515-528.

Han，J.，Y. Fu. 1995. Discovery of Multiple Level Association Rules from Large Databases，*Proceedings of the 21st VLDB Conference Zurich*，Switzerland，1995.

Han，J.，Y. Fu. 1999. Mining multiple-level association rules from large databases. *IEEE Transactions on Knowledge and Data Engineering*，11(5) 798-805.

Han，J.，J. Pei，Y. Yin. 2000. Mining frequent patterns without candidate generation. *Proc. 2000 ACM-SIGMOD Int. Conf. Management of Data*，Dallas，TX，1-12.

Han，J.，J. Pei，Y. Yin，R. Mao. 2004. Mining frequent patterns without candidate

generation：a frequent-pattern tree approach. *Data Mining and Knowledge Discovery*，8，53-87.

Hershberger，J.，N. Shrivastava，S. Suri，C. D. Toth 2005. Space Complexity of Hierarchical Heavy Hitters in Multi-Dimensional Data Streams. *Principles of Database Systems* (*PODS*)，2005，June：13-15.

Kotsiantis，S. D. Kanellopoulos. 2006. Association rules mining：a recent overview. *GESTS International Transactions on Computer Science and Engineering*，32 (1)，71-82.

Lee，A. J. T.，C. S. Wang. 2007. An efficient algorithm for mining frequent inter-transaction patterns. *Information Sciences* 177(17) 3453-3476.

Lu，H.，L. Feng，J. Han. 2000. Beyond intratransaction association analysis：mining multidimensional intertransaction association rules，*ACM Transactions on Information Systems* 18(4) 423-454.

Lu，H.，J. Han，L. Feng. 1998. Stock movement prediction and n-dimensional inter-transaction association rules，*Proc. of the 3rd ACM-SIGMOD Workshop on Research Issues on Data Mining and Knowledge Discovery*，Seattle，WA，USA，12：1-12：7.

Manku，G.，R. Motwani 2002. Approximate Frequency Counts over Data Streams. *Proceedings of the 28th International Conference on Very Large Data Bases* (*VLDB*)，2002，August：346-357.

Metwally，A.，D. Agrawal，A. E. Abbadi 2005. Efficient Computation of Frequent and Top-k Elements in Data Streams. *Proceedings of the 10th International Conference on Database Theory* (*ICDT*)，2005，January：398-412.

Palmeri，C. 1997. Believe in yourself，believe in the merchandise. *Forbes* 160(5) (Sep 8，1997) 118-124.

Pei，J.，J. Han，B. Mortazavi-Asl，J. Wang，H. Pinto，Q. Chen，U. Dayal，M. C. Hsu. 2004. Mining sequential patterns by pattern-growth：the PrefixSpan approach，*IEEE Transactions on Knowledge and Data Engineering* 16 1424-1440.

Schmitz，C.，A. Hotho，R. Jäschke，G. Stumme. 2006. Mining association rules in folksonomies. *Data Science and Classification*：*Proc. of the 10th IFCS Conf.*，*Studies in Classification*，*Data Analysis*，*and Knowledge Organization*，pp. 261-270.

Srikant，R.，R. Agrawal. 1995. Mining generalized association rules. *Proc. 21st VLDB Conference*，Zurich，Switzerland，September：409-419.

Srikant，R.，R. Agrawal. 1996. Mining sequential patterns：generalizations and performance improvements，*Proc. Int'l Conf. Extending Database Technology*，3-17.

Tung，A. K. H.，H. Lu，J. Han，and L. Feng. 1999. Breaking the barrier of transactions：

mining inter-transaction association rules, *Proc. SIGKDD-1999*, 297-301.

Wang, C., C. Tjortjis. 2004. PRICES: An efficient algorithm for mining association rules, *Lecture Notes in Computer Science*, Volume 3177, 352-358.

Zaki. M. J. 2000. Scalable algorithms for association mining. *IEEE Transactions on Knowledge and Data Engineering*, 12(3) 372-390.

Zipf, G. K. 1949. *Human Behavior and the Principle of Least Effort*. Addison-Wesley, 1949.