客户导向目录分割问题的改进算法①

杜萍萍[1], 陆 可[1], 吴金南[2]

[1](安徽工业大学 管理科学与工程学院, 马鞍山 243002)
[2](安徽工业大学 商学院, 马鞍山 243002)

摘 要: 客户导向目录分割问题假设顾客至少对目录中一定数量的商品感兴趣, 计算目录覆盖的顾客数量, 据此评估目录分割结果. 现有的分割算法为了保证目录尽可能多的覆盖顾客, 而忽略了目录分割结果的效用. 针对该问题, 本文构建一种新的数据存储结构 *CFP-Tree* 用于存储顾客交易数据, 并提出一种新的算法 *Effective-Cover* 解决目录分割问题. 该算法使用树深度遍历法选择目录产品. 实验结果表明, 该算法能够获得更好的目录分割结果.

关键词: 目录分割; *CFP-Tree*; *Effective-Cover* 算法; 客户; 商业智能

Improved Algorithm of Customer Oriented Catalog Segmentation Problem

DU Ping-Ping[1], LU Ke[1], WU Jin-Nan[2]

[1](School of Management Science and Engineering, Anhui University of Technology, Maanshan 243002, China)
[2](School of Business, Anhui University of Technology, Maanshan 243002, China)

Abstract: The customer oriented catalog segmentation problem assumes that one customer is interested in at least a certain number of items in the catalog, and then calculates the number of customers that are covered by the catalog. Hence, the result of catalog segmentation is assessed according to it. In order to ensure that the catalog covers customers as many as possible, the existing segmentation algorithm ignores the effect of the results of the catalog segmentation. Aiming at this problem, this paper constructs a new data storage structure *CFP-Tree* for storing customer transaction data, and presents a new algorithm *Effective-Cover* to solve the problem of catalog segmentation. The algorithm uses tree depth traversal method to select catalog products. The experimental results show that the algorithm can obtain better catalog segmentation results.

Key words: catalog segmentation; *CFP-Tree*; *Effective-Cover* algorithm; customer; business intelligence

客户分类问题[1,2]是一类经典的商业智能优化问题, 客户分类问题实际也属于分割问题的一种. 以此为切入点, 近年来, 多种商业智能理论相继被提出, 其中基于微观经济学的观点尤其引人注目[3-5]. 该理论将商业智能视为大量非聚集数据在商业目标驱动下的优化问题, 当企业使用某个模式使得企业效益增加时, 该模式被认为有效[5].

简单地为所有顾客采取同一商业决策不利于利润最大化. 一个有效的方法是形成 k 个决策, 使每个决策覆盖的顾客数量最大化, 从而使企业获得更多的利润. Ester 研究了微观经济学视角下数据挖掘问题的另一个问题[6]: 客户导向的目录分割问题. 将顾客划分成不同的群体, 并制定针对不同顾客群体的商品目录, 使得商品目录在顾客群体上产生尽可能大的效用. 整体的效用, 由所有目录所覆盖的满足最小兴趣度的顾客数量来度量. 在这样的假设下, 目录分割问题就演化成加入兴趣度约束的目录分割问题, 也称为客户导向目录分割问题[6].

由于现实生活中数据信息量大, 为了提高算法效率, 需要一个有效的数据结构来存储数据. Han 在研究

① 基金项目:国家自然科学基金(7371013);安徽工业大学校青年教师科研基金(QZ201420);安徽省教育厅自然科学基金(KJ2016A087)
收稿时间:2016-07-21;收到修改稿时间:2016-08-22 [doi:10.15888/j.cnki.csa.005690]

关联规则时, 提出了一种新的存储结构--频繁模式树 *FP-Tree*(*frequent pattern tree*), 取得了显著的效果[7,8]. Xu 构建了一种新的数据结构频繁模式树 *TFP-Tree* (*frequent pattern tree with T-layer interest constraint*)来存储顾客交易数据, 并在此基础上提出一种有效的客户导向目录分割算法 *Max-Cover* (*maximal customer cover catalog segmentation algorithm*), 获得了更好的目录分割效用[9,10]. Amiri 提出了两种解决客户导向目录分割问题的算法[11]. 第一种是贪婪算法, 并利用算法的随机性避免了局部最优; 第二种是基于关联规则挖掘的启发式算法. Iraj 将客户导向目录分割问题转化成一种可供替代的方法问题, 提出了一种自适应的遗传算法解决该问题, 并且有效地避免了局部最优[12]. 也有一些关于目录分割问题的算法相继被提出, 比如最大二等分和不相交的目录分割问题的改进算法[13].

针对客户导向目录分割问题, 本文基于树结构改进了客户交易数据的存储方式, 构建一种新的数据结构 *CFP-Tree*(*frequent pattern tree of customer database*), 改进了树的遍历方法, 设计出 *Effective-Cover*(*effective customer cover catalog segmentation algorithm*)目录分割算法, 旨在提高生成目录的整体效用.

1　符号定义与问题描述

1.1 相关符号定义

本文使用二部图 G 代表顾客数据库. 有两类顶点集合, 一类为顾客集合, 另一类为产品集合, 边代表相应的顾客对相应的产品感兴趣的事实. 假设收集到的顾客数据存储在顾客数据库(*Customers DB*)中. 相关的符号定义包括:

二部图 $G=\{P,C,E\}$: 有两个顶点集合 P, C 和一个边集 E, 使用||表示集合的基数, $|P|=m$, $|C|=n$, 其中:

$C=\{c_1,c_2,...,c_i,...,c_n\}$, 表示所有的顾客;

$P=\{p_1,p_2,...,p_j,...,p_m\}$, 表示所有的产品;

$E=\{e_{ij}=(c_i,p_j)\,|\,c_i$ 对 p_j 感兴趣, $\mathrm{c}_i\in \mathrm{C},\mathrm{p}_j\in \mathrm{P}\}$ 对于 $\forall P'\subseteq P, w(P')=$ {至少有一条边连接到 p' 中的顶点 C 的顾客集合}.

对于 $P'\subseteq P$, $t\geq 1$, $\varphi(P',t)$ 表示至少有 t 条边连接到产品集 P' 中的顾客集 C', 即: $\varphi(P',t)=\{C'\subseteq C\,|\,\forall c_i\in C',\exists e_{ij}=(c_i,p_j),e_{ij}\in E',$ $p_j\in P',P'\subseteq P\}$.

1.2 客户导向目录分割问题定义

定义 1. 分割问题: 给出 n 个收益函数 $f_1,...,f_n$ 和参数 k, 求 k 个子集 $X_1,...,X_k\in D$, 使 $\sum_{i=1}^{n}[\max_{1\leq j\leq k}f_i(x_j)]$.

定义 2. 覆盖. 如果顾客 $Customer_i$ 对目录 $Cata\log_j$ 中至少有一个产品感兴趣, 称该目录覆盖了该顾客.

定义 3. 最多顾客覆盖问题. 给出任意的二部图 $G=(P,C,E)$ 和一个正整数 r, 寻找一个大小为 r 的子集 $P'\subseteq P$ 使顾客集合 $w(p')$ 尽可能大.

定义 4. k 目录分割问题. 给出使用二部图 $G=(P,C,E)$ 表示的顾客数据库, 并且 $|p|=m,|C|=n$, 求商品集合 P 的 k 个子集 $p_1,...,p_k$, 其中 $|P_i|=r$, 和对应顾客集合 C 的 k 个子集 $C_1,...C_k$, 使得 $\sum_{i=1}^{k}|\omega(P_i)|$ 尽可能大.

$$s.t.P_1\cup...\cup P_k\subseteq p,$$
$$C_1\cup...\cup C_k=C,C_i\cap C_j=\varnothing,$$
$$|P_1|=|P_2|=...=|P_k|=r.$$

将顾客分成 k 个群体, 每个顾客最多只能属于其中一个群体, 每个群体所对应的目录包含 r 个产品, 而同一产品允许在多个产品目中中同时出现.

定义 1 至定义 4 中的目录分割问题以最大化的覆盖顾客数量为目的. Ester 引入最小兴趣度 t[6], 即在目录分割问题中加入兴趣度约束, 使发送到顾客的产品目录至少有兴趣度 t 的顾客数量来评价目录的整体效用.

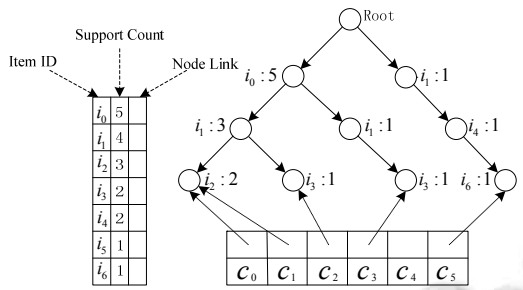
定义 5. 客户导向的目录分割问题. 给出使用二部图 $G=(P,C,E)$ 表示的顾客数据库, $|p|=m,|C|=n$, 制定 k 个目录 $p_1,...,p_k$, 其中 $|P_i|=r$, 和对应的 C 的 k 个子集以 $C_1,...C_k$, 并且顾客在对应的目录中至少有 t 个感兴趣的产品, 使得 $\sum_{i=1}^{k}|\varphi(P_i,t)|$ 最大.

$$s.t.P_1\cup...\cup P_k\subseteq p,$$
$$C_1\cup...\cup C_k=C,C_i\cap C_j=\varnothing,i\neq j,$$
$$|P_1|=|P_2|=...=|P_k|=r.$$

1.3 *CFP-Tree* 结构定义

考虑到顾客数据库数据量庞大的问题, 为了提高算法的运行效率, 我们需要建立一个有效的数据结构, 同时存储事务和其对应的顾客信息. 本研究使用树结构存储数据, 即兴趣度 t 的约束. 在构建树时仅构建

顶上 t 层, 假设顾客感兴趣的产品中支持度最高的 t 个产品就可覆盖其基本需求. 将顾客对应的感兴趣产品按支持度降序排列, 并取前 t 个插入树中. 使用本文提出的 *CFP-Tree* 结构存储数据, 将每条交易对应的顾客信息存入表中. *CFP-Tree* 每个树枝末尾加入顾客信息, 如图 1 所示.



(a)项列表 (b)*CFP-Tree*

图 1 *CFP-Tree* 结构图

定义 6. 短交易. 若顾客感兴趣的产品量小于 t, 则称该顾客交易为短交易.

若将短交易插入树中, 则叶节点到根的路径长度小于 t, 故没有必要将此交易插入树中. 因此, 在数据预处理中, 可删除所有短交易和其对应的顾客信息.

定义 7. 锚节点. 距 *CFP-Tree* 根节点 t 层的节点.

若选择从锚节点到根上所有节点, 则可覆盖全部顾客. *CFP-Tree* 需维护 2 个表: 频繁项列表按频繁计数降序排列; 树下面的顾客表记录每个顾客对应的叶节点.

定义 8. *CFP-Tree*. 设使用树结构表示顾客数据库. *CFP-Tree* 是一种高度为 t 的树结构, 包括三部分, 定义如下:

1) 包含一个标记为 *null* 的根节点、一个项前缀子树作为根的孩子、一个频繁项表、一个顾客项表;

2) 在项前缀子树上的每个节点包含四个域, 项名称、频繁项、节点链和标记标签, 其中, 项名称记录当前项代表哪个节点; 频繁项表示到此节点的频繁度; 节点链链接到 *CFP-Tree* 上有相同名称的下一项, 若无则为 *null*; 标记标签用来标记该节点是否被选中, 被选中为 1, 未被选择则为 0;

3) 在频繁项列表的每个入口包含项名称和节点链头 2 个域. 其中, 节点链头指向 *CFP-Tree* 上有相同项名称的第一个节点;

4) 顾客项表头的每个入口包含顾客名称和节点链头 2 个域. 其中, 节点链头指向 *CFP-Tree* 上的后 1 个节点.

为构建 *CFP-Tree*, 需扫描数据库两次, 第一次读取全部的项, 并将单个项按照频繁度降序排序, 构建 *CFP-Tree* 根节点和左侧链表; 第 2 次扫描数据库, 将事务数据库中的每个交易都插入到 *CFP-Tree* 树结构中.

2 *Effective-Cover*算法

客户导向的目录分割问题已被证明是一个 NP 完全问题, 徐秀娟在研究这类问题时; 提出一种新的数据结构 *TFP-Tree*, 基于此数据结构提出一种新的算法 *Max-Cover*, 获得了比较好的目录分割结果[10].

但是该算法在树的遍历中, 是将树的叶子节点按照支持度进行降序排序, 在遍历树的过程中优先选择支持数大的叶子点, 再从选中的叶子节点到根节点的路径上选择产品加入到目录. 这样做只考虑叶子节点的支持度, 而忽视了节点之间的关联性, 可能会导致生成的目录内部产品耦合性不高,顾客与顾客之间的关联性也不大.

本文在此基础上改进了数据存储结构, 构建了一种新的数据结构 *CFP-Tree*, 并改进了树的遍历方法, 设计出相应的最大顾客覆盖的目录分割算法 *Effective-Cover*, 结合本文提出的基于兴趣度约束的目录分割算法, 可将商品目录分割问题变成树深度搜索问题.

2.1 构造 *CFP-Tree*

构建 *CFP-Tree* 算法描述如算法 1.

算法 1. 构建 *CFP-Tree*.

输入: 顾客数据库;

输出: *CFP-Tree*.

Step1: 扫描数据库, 找到所有频繁项集合和其对应的频繁度. 将所有项按频繁度降序排列, 所得链表记做 L_f.

Step2: 创建 *CFP-Tree* 的根节点并标记为 *null*, 对每个交易 T 进行如下操作: 依据 L_f 将 T 中的每个项排序, 设排序后的链表为 $[p|P|p']$, 其中是 p 第一个项, p' 是最后一项, $|P|$ 是剩余的项. 调用插入操作

$insert-tree([p|P|p'],T)$.

Step3: $insert-tree([p|P|p'],T)$ 操作. 若 T 同名项 N, 则将 N 的计数加 1; 否则, 创建新节点 N, 将计数置为 1, 将 N 的父节点链接到 T, 并将在频繁项表中对应的节点链头引出链接到此节点. 若 $P \neq \phi$, 则继续递归调用 $insert-tree([p|P|p'],T)$.

Step4: 插入操作完成后, 在顾客项表中找到当前交易对应的顾客, 并将其链接到 p'.

表 1 给出顾客交易数据的一个例子. 设此时兴趣度 t 为 3, 其中每条交易中的项按照支持度降序排序, 将修剪后的每条数据写在表的第三列.

扫描表 1 所示的顾客数据集, 删除短交易; 并对所有频繁项按照降序排序, 然后对每条交易中的项按照频繁度从高到低的顺序进行排序, 取出前 t 个产品. 首先得到图 1(a)的项列表. 然后扫描数据库构建相应的 *CFP-Tree*, 如图 1(b)所示. 例如对于顾客 c_1 其对应的交易为 i_0,i_1,i_2,i_4, 将其排序为 i_0,i_1,i_2, 由于 t 为 3, 则将其剪枝, 仅保留前 t 个产品, 即 i_0,i_1,i_2, 再将其插入 *CFP-Tree* 上, 成为最左边的树枝. 对于剩余的顾客交易数据, 也使用类似方法插入.

表 1　示例数据库

customers	*Items*	*Pruned Items*
c_0	i_0,i_1,i_2,i_4	i_0,i_1,i_2
c_1	i_0,i_1,i_2,i_4	i_0,i_1,i_2
c_2	i_0,i_1,i_3	i_0,i_1,i_3
c_3	i_0,i_2,i_3,i_6	i_0,i_2,i_3
c_4	i_0	
c_5	i_1,i_4,i_5	i_1,i_4,i_5

2.2 算法过程

本文使用 *CFP-Tree* 树形结构存储顾客数据, 客户导向的目录分割问题变成了在剪枝后的树上寻找某些产品的组合问题, 并使该组合可最大程度覆盖感兴趣的顾客. 下面给出了本文提出的 *Effective-Cover* 算法的具体过程.

算法 2: *Effective-Cover*

输入: 历史交易记录 *Customer DB*, 目录数量 k, 每个目录包含的产品数量 r, 顾客兴趣度 t.

输出: k 个目录和对应的 k 个顾客簇.

Step1: 将一个顾客的所有交易合并为一个交易, 并删除所有短交易;

Step2: 标记所有顾客为未覆盖顾客;

$For(cat = 1\ to\ k)$ {

(第一次)扫描数据库中所有未覆盖的顾客交易得到相应的频繁项并计算支持度, 按照频繁项的支持度进行降序排序, 将频繁项对应的产品标记为未选状态;

(第二次)扫描预处理过的数据中所有未覆盖的顾客交易, 构建 *CFP-Tree*;

扫描 *CFP-Tree*, 将当前的所有叶子节点加入数组 *Leaf* 中, 将数组 *Leaf* 按照节点计数从高到低排序后得到新数组 *OrderLeaf;*

$while(|cat| < r)$ {

$if(r-|cat| \geq t)$ {

从数组 *OrderLeaf* 选择支持度最大的叶子节点 p, 将 *OrderLeaf* $[p]$ 到 *root* 路径上的未选中节点加入 *cat;*

}

else {

从 $t-(r-|cat|)$ 层中寻找父节点已经被覆盖的子树;

if (从父节点到叶子节点上的存在未选中节点){

将未选中节点加入到 *cat*;

}

else{

在没有被覆盖的叶子节点当中选中支持度最大的节点 p',将 *OrderLeaf* $[p']$ 到 *root* 路径上的未选中节点加入 *cat*;

}

}

将 *cat* 覆盖的顾客标记为已覆盖顾客;

}

2.3 算法分析

根据整个算法运行过程, 我们观察到对算法效率影响最大的是反复扫描数据库读取数据的时间. 与原有的算法相比较, 本算法通过增加节点标记的方法大大减少了数据库的扫描次数, 提高了算法的运行效率. 虽然在算法复杂方面较原有算法没有明显降低, 但在目录产品选择上降低了算法的随机性, 一定程度上提高了目录分割结果的准确性.

3 实验及讨论

3.1 实验设置

为了演示算法运行效果, 本文首先给出一个包含 20 条顾客交易数据的示例数据集. 在本次实验中, 相应的参数设置如下: t=3, r=5, k=3. 其中每条交易中的项按照支持度降序排序, 并得到了数据预处理后的剪枝结果, 如表 2 所示.

表 2 示例数据集

customers	*items*					*Pruned Items*		
c_0	i_4	i_5	i_2	i_7	i_8	i_4	i_5	i_2
c_1	i_4	i_2	i_1	i_6	i_7	i_4	i_2	i_1
c_2	i_4	i_2	i_1	i_3	i_8	i_4	i_2	i_1
c_3	i_5	i_2	i_1	i_3	i_7	i_5	i_2	i_1
c_4	i_5	i_2	i_3	i_8		i_5	i_2	i_3
c_5	i_4	i_5	i_6	i_7		i_4	i_5	i_6
c_6	i_4	i_5	i_3	i_7		i_4	i_5	i_3
c_7	i_4	i_2	i_1	i_6	i_3	i_4	i_2	i_1
c_8	i_5	i_1	i_6	i_3	i_7	i_5	i_1	i_6
c_9	i_4	i_5	i_2	i_1	i_8	i_4	i_5	i_2
c_{10}	i_4	i_5	i_7	i_8		i_4	i_5	i_7
c_{11}	i_4	i_1	i_3	i_7		i_4	i_1	i_3
c_{12}	i_4	i_2	i_6	i_8		i_4	i_2	i_6
c_{13}	i_5	i_1	i_6	i_8		i_5	i_1	i_6
c_{14}	i_5	i_2	i_1	i_7		i_5	i_2	i_1
c_{15}	i_4	i_1	i_6	i_7		i_4	i_1	i_6
c_{16}	i_4	i_5	i_2	i_6		i_4	i_5	i_2
c_{17}	i_4	i_5	i_2	i_6	i_3	i_4	i_5	i_2
c_{18}	i_4	i_1	i_3	i_6		i_4	i_1	i_3
c_{19}	i_4	i_5	i_2	i_6	i_8	i_4	i_5	i_2

3.2 实验结果

根据算法 1 构建 CFP-Tree, 我们首先将数据库的每条交易信息存储到树结构中, 如图 2 所示. 再根据算法 2, 从树中遍历选择节点依次加入到目录中, 得出目录分割结果, 如表 3 所示.

表 3 实验结果

	Effective-Cover	*Max-Cover*
Cat_1	i_1,i_2,i_4,i_5,i_6	i_1,i_2,i_4,i_5,i_6
Cat_2	i_1,i_3,i_4,i_6,i_7	i_2,i_3,i_4,i_5,i_7
Cat_3	i_1,i_2,i_3,i_5,i_6	i_1,i_2,i_4,i_5,i_6
Covered Customers	18	16

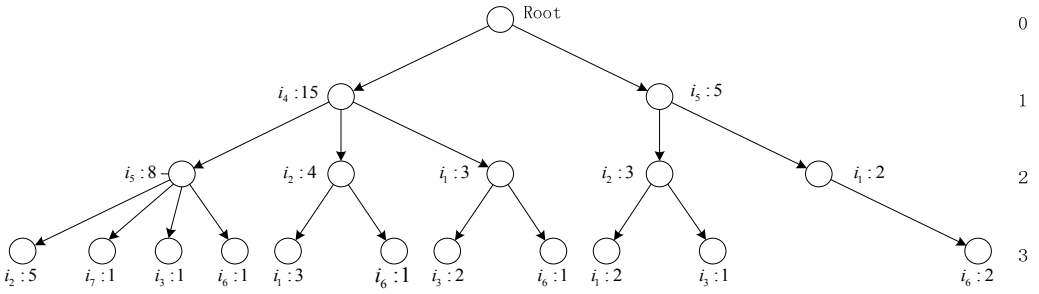


图 2 *CFP-Tree*

3.3 实验结果分析与讨论

从实验结果可以看出, 与文献[10]的算法相比, 本算法在遍历树的过程中, 在选择节点时, 优先选择父亲节点被覆盖的子节点, 否则, 优先选择支持数最大的叶子节点. 这样做既保证了同一目录中, 客户之间兴趣尽可能相似, 同时保证了目录覆盖客户数尽可能多,提高了目录分割结果的效用.

本算法构建一棵 *CFP-Tree* 树只需要扫描数据库 2 次. 其中, 第 1 次获得所有顾客交易中对应商品的支持度, 第 2 次将所有顾客的交易插入树中, 每个顾客对应 1 条交易, 完成了数据预处理. 构建 *CFP-Tree* 树结构后, 按照算法 *Effective-Cover* 中树的遍历方法从树上选择 r 个节点加入. 每个目录构建成功后, 都需要扫描数据库 1 次以便标记所有被当前目录覆盖的产品和顾客. 因此, 构建 k 个录只需扫描 $k+2$ 次数据库.

在示例数据集上, 本文所提出的 *Effective-Cover* 算法的客户覆盖数比文献[10]提出的 *Max-Cover* 算法多. 为了测试本文所提出的算法在大规模数据集上的运行效果, 本文使用IBM数据生成器产生合成数据集, 并在该数据集上比较两种算法. 实验结果显示, 本文所提出的算法在算法运行效率和客户覆盖结果上, 具有更好的表现.

4 总结与展望

商业活动的最终目的是获得尽可能大的利润. 近年来, 微观经济观点指导下的数据挖掘得到了快速的发展, 本文将微观经济观点引入到数据挖掘中, 讨论了目录分割问题. 在未来的工作中, 我们将从以下角度继续深入讨论本文研究的问题:

1) 在商品选择的过程中, 不仅考虑顾客的交易历史, 还考虑现实情境中能反映顾客兴趣度的其他信息, 如商品浏览记录等.

2) 在商品选择的过程中, 考虑单个商品对企业总体利润的贡献度. 譬如现有商品选择算法都不允许负利润商品存在. 然而现实情况下, 为了吸引顾客, 捆绑销售的商品集合中可能包含企业亏本售卖的商品.

参考文献

1 Rezaeinia SM, Rahmani R. Recommender system based on customer segmentation(RSCS). Kybernetes, 2016, 45(6): 946–961.

2 Güçdemir H, Selim H. Integrating multi-criteria decision making and clustering for business customer segmentation. Industrial Management & Data Systems, 2015, 115(6): 1022–1040.

3 Kleinberg J, Papadimitriou C, Raghavan P. Segmentation problems. Journal of the ACM (JACM), 2004, 51(2): 263–280.

4 Kleinberg J, Papadimitriou C, Raghavan P. Segmentation problems. Proc. of the Thirtieth Annual ACM Symposium on Theory of Computing. 1998. 473–482.

5 Kleinberg J, Papadimitriou C, Raghavan P. A microeconomic view of data mining. Data Mining and Knowledge Discovery, 1998, 2(4): 311–324.

6 Ester M, Ge R, Jin W, et al. A microeconomic data mining problem:customer-oriented catalog segmentation. Proc. of the Tenth ACM SIGKDD Int'l Conference on Knowledge Discovery and Data Mining. 2004. 557–562.

7 Han J, Pei J, Yin Y. Mining frequent patterns without candidate generation. ACM Sigmod Record, 2000, 29(2): 1–12.

8 Han J, Wang J, Lu Y. Mining top-k frequent closed patterns without minimum support. IEEE Int'l Conference on Data Mining, ICDM 2002. 2002. 211–218.

9 Xu X, Liu Y, Wang Z, et al. Catalog segmentation with

double constraints in business. Pattern Recognition Letters, 2009, 30(4): 440–448.

10 徐秀娟,王喆,常晓宇,等.一种新的面向顾客的目录分割算法.计算机研究与发展,2008,(z1):310–315.

11 Amiri A. Customer-oriented catalog segmentation: Effective solution approaches. Decision Support Systems, 2006, 42(3): 1860–1871.

12 Mahdavi I, Movahednejad M, Adbesh F. Designing customer-oriented catalogs in e-CRM using an effective self-adaptive genetic algorithm. Expert Systems with Applications, 2011, 38(1): 631–639.

13 Xu Z, Du D, Xu D. Improved approximation algorithms for the max-bisection and the disjoint 2-catalog segmentation problems. Journal of Combinatorial Optimization, 2014, 27(2): 315–327.

14 Kianfar K, Fathi M, Hasanzadeh A, et al. Catalog segmentation with the objective of satisfying customer requirements in minimum number of catalog. 2009 IEEE International Conference on Industrial Engineering and Engineering Management. IEEE. 2009. 1253–1257.