面向不确定感知数据的频繁项查询算法

王 爽[1),2)] 王国仁[2)]

[1)](东北大学软件学院 沈阳 110819)

[2)](东北大学信息科学与工程学院 沈阳 110819)

摘 要 随着计算机网络技术的快速发展,无线传感器网络产生了大量的感知数据流.同时,传感器自身的特点使得感知数据具有不确定的特征,因此需要对传感器网络中不确定感知数据流处理技术进行研究.在传感器网络中,频繁项查询在环境监控和关联规则挖掘等方面具有重要意义.文中首先提出了基本算法,用以连续维护传感器网络中的概率阈值频繁项查询结果.针对基本算法需要维护所有元素的问题,又提出了一种优化算法,算法在两方面进行了优化:(1)设计了一种通过预测元素概率上界的方法进行候选集的构造,仅维护必要信息从而提高查询效率;(2)设计了一种新的cp-list结构,可以压缩不同窗口候选集中的重复元素,降低存储开销.实验结果表明文中提出的算法可以减少连续维护传感器网络中频繁项查询的计算代价和存储空间.

关键词 无线传感器网络;不确定数据流;频繁项;概率阈值;过滤;物联网

中图法分类号 TP311 **DOI号** 10.3724/SP.J.1016.2013.00571

Frequent Items Query Algorithm for Uncertain Sensing Data

WANG Shuang[1),2)] WANG Guo-Ren[2)]

[1)](*Software College, Northeastern University, Shenyang* 110819)

[2)](*School of Information Science and Engineering, Northeastern University, Shenyang* 110819)

Abstract With advances in technology, large amounts of streaming data can be generated continuously by sensors. Due to the inherited limitation of sensors, these continuous sensing data can be uncertain. This calls for stream mining of uncertain sensing data. Frequent items query is valuable in wireless sensor networks (WSNs) and it can be widely used in environmental monitoring, association rules mining, and so on. A basic algorithm which continuously maintains sliding window frequent items over WSNs is proposed. However the basic algorithm needs to maintain all items in the window. Due to this, an improved algorithm is further proposed by optimizing in two aspects: (1) the pruning rules by predicting the upper bound of items probability is developed, which can reduce the candidate set and improve the query efficiency; (2) the large amount of the same items in different window can be compressed by cp-list structure in order to minimize the memory utilization. Finally, experimental results and detailed analysis demonstrate that the high efficiency and low memory cost of the proposed algorithms in WSNs.

Keywords wireless sensor network; uncertain data stream; frequent items; probabilistic threshold; filtering; Internet of Things

---

收稿日期:2011-08-31;**最终修改稿收到日期**:2012-11-17.**本课题得到国家自然科学基金重点项目**(60933001)、**国家杰出青年科学基金项目**(61025007)、**国家自然科学基金青年科学基金项目**(61100022)**资助**.**王 爽**,**女**,1980**年生**,**博士**,**讲师**,**主要研究方向为不确定数据处理、数据流查询**.E-mail:wangsh@mail.neu.edu.cn;wangshuang_neu@163.com.**王国仁**,**男**,1966**年生**,**博士**,**教授**,**博士生导师**,**主要研究领域为**XML**数据管理、生物信息学、分布数据库、并行计算等**.

1 引 言

随着通信技术、嵌入式计算技术和传感器技术的飞速发展和日益成熟，传感器网络已经成为数据采集与传输的重要手段之一，并且被广泛地应用到国防和国民经济各个领域，典型的应用有地震监测、农业监测、矿井环境监测以及战场态势感知等[1]. 这些应用通常都产生较大的实时流式感知数据，由于每个传感器节点只有少量的计算资源，难以处理如此巨大的实时数据流，因此在传感器网络中，普遍采用近期(滑动窗口内)数据进行相关信息获取.

传感器网络，由于受到硬件设备、传感技术、网络传输延迟、无线传感器网络的通信质量和感知环境等因素的影响，会产生不精确的数据或缺失的数据[2]. 同时，数据的不确定性在网络通信和网内处理的过程中会得到进一步的传播和放大. 因此，如何处理和分析无线传感器网络中的这种数据不确定性是一个非常重要的问题.

在传感器网络中，频繁项查询在环境监测、关联规则挖掘等领域有非常重要的应用. 例如在温度监测系统中，某温度值频繁出现就会被认为是异常情况从而报警. 由于传感器自身的限制，检测的记录是不确定的，每一个监控记录都会附加一概率信息，表示该记录的可信程度. 以表 1 中的 R_1 为例，表示传感器 S1 在 L 处监控的温度值为 v_1，该信息的可信度为 0.6.

表 1 温度检测记录

ID	位置	温度	概率
R_1	L	v_1	0.6
R_2	L	v_1	0.4
R_3	L	v_2	0.8
R_4	L	v_1	0.3

由于传感器网络产生的数据不确定，可能世界模型[3]被广泛地应用于不确定数据的建模. 表 1 有 4 条传感器记录，共有 $2^4=16$ 个可能世界实例，如表 2 所示. 例如，可能世界实例 $PW_1=\{R_1\}$ 中，仅 R_1 记录存在，R_2，R_3，R_4 均不存在，v_1 出现 1 次，v_2 出现 0 次，概率为 $0.6\times(1-0.4)\times(1-0.8)\times(1-0.3)=0.0504$.

表 2 表 1 的可能世界模型

ID	可能实例	频次统计	概率
PW_0	$\{\}$	$\{v_1=0,v_2=0\}$	0.0336
PW_1	$\{R_1\}$	$\{v_1=1,v_2=0\}$	0.0504
PW_2	$\{R_2\}$	$\{v_1=1,v_2=0\}$	0.0224
PW_3	$\{R_3\}$	$\{v_1=0,v_2=1\}$	0.1344
PW_4	$\{R_4\}$	$\{v_1=1,v_2=0\}$	0.0144
PW_5	$\{R_2,R_3\}$	$\{v_1=1,v_2=1\}$	0.0896
PW_6	$\{R_1,R_3\}$	$\{v_1=1,v_2=1\}$	0.2016
PW_7	$\{R_3,R_4\}$	$\{v_1=1,v_2=1\}$	0.0576
PW_8	$\{R_1,R_2\}$	$\{v_1=2,v_2=0\}$	0.0336
PW_9	$\{R_2,R_4\}$	$\{v_1=2,v_2=0\}$	0.0096
PW_{10}	$\{R_1,R_4\}$	$\{v_1=2,v_2=0\}$	0.0216
PW_{11}	$\{R_1,R_2,R_3\}$	$\{v_1=2,v_2=1\}$	0.1344
PW_{12}	$\{R_1,R_2,R_4\}$	$\{v_1=3,v_2=0\}$	0.0144
PW_{13}	$\{R_2,R_3,R_4\}$	$\{v_1=2,v_2=1\}$	0.0384
PW_{14}	$\{R_1,R_3,R_4\}$	$\{v_1=2,v_2=1\}$	0.0864
PW_{15}	$\{R_1,R_2,R_3,R_4\}$	$\{v_1=3,v_2=1\}$	0.0864

尽管在传统的确定数据库领域，频繁项查询得到了深入的研究，但是由于传感器网络产生了大量不确定流式感知数据，现有的算法都不能直接应用到传感器网络中. 因此本文提出了一种适用于不确定感知数据的频繁项查询算法，旨在减少每次数据更新的计算代价和存储开销. 主要工作包括：(1) 提出了一种基本的不确定频繁项查询算法，该算法利用现有计算结果进行增量更新，提高查询效率；(2) 提出了一种优化算法，在每个窗口中利用剪枝策略构造候选集合，同时利用提出的 cp-list 结构压缩不同窗口中的候选元素，减少检测数据的数量，提高查询效率，降低存储开销；(3) 通过实验结果证明新算法的有效性和高效性.

2 相关工作

在确定数据中，已有一些基于传感器网络的频繁项查询工作. Considine 等人[4]提出了基于树结构的频繁项查询算法，并设计了一种 sketch 摘要结构，将大值域的数据集映射到小值域的目标数据集中，用目标数据集中的频繁项作为整个数据集的近似解. 但是基于树结构的算法在网络数据丢失情况严重时，会带来巨大的传输开销，因此 Manjhi 等人[5]提出了综合使用树和多路径路由结构的 Tributary-Delta 算法，并利用抽样方法得到近似查询结果. Ren 等人[6]提出了一种分布式查询算法 D-FIMA，通过建立聚集树，用有限个计数单元进行协作计数来维护整个数据集中频繁项的方法，得到近似频繁项.

但是上述研究工作均基于确定数据流，没有考虑传感器数据的不确定性. 近年来开展了不确定频繁项查询方面的工作，研究内容主要分为两类：基于期望的方式[7]和基于概率的方式[8].

定义 1[7]. 期望频繁项. 对于一组传感器记录 R，PW 为 R 所有可能世界实例集合，满足不等式(1)的项 v 称为期望频繁项.

$$\sum_{\omega\in PW} C^v(\omega)\times p(\omega)>esup \tag{1}$$

$C^v(\omega)$为项 v 在 w 实例空间中出现的次数，$p(\omega)$为 w 实例空间的概率，$esup$ 为阈值.

从定义 1 可以看出，计算期望需要列出所有可能实例(如表 2 所示)，其复杂度为指数级. 文献[7]指出期望 $exp(v)$可以采用更简单的线性复杂度方法进行计算，如式(2)所示.

$$exp(v)=\sum p(v\in R_i) \tag{2}$$

虽然使用线性复杂度就可以计算期望值，但是期望并不能真正反映数据本身分布的差异，如表 3 所示数据库.

表 3　不确定数据库

ID	数据	概率
1	a	1/2
2	a	1/2
3	b	1

通过计算，$exp(a)=1/2+1/2=1$，$exp(b)=1$. a，b 两者期望相同，但是频次分布并不相同. 项 a 频次分布如表 4 所示，项 b 频次分布如表 5 所示.

表 4　a 频次分布

次数	0	1	2
概率	1/4	1/2	1/4

表 5　b 频次分布

次数	1
概率	1

针对上述问题，文献[8]提出了一种更复杂的概率频繁项定义.

定义 2[8]. 概率频繁项. 满足不等式(3)的项称为概率频繁项.

$$FP(v)=Pr\{sup^v(R)\geq minsup\}>minprob \tag{3}$$

其中 $sup^v(R)$为项 v 的频次分布，$minsup$ 为频次阈值，$minprob$ 为概率阈值，$FP(v)$为项 v 成为频繁项的概率，如果 $FP(v)$ 大于 $minprob$，则项 v 为频繁项.

项 v 在确定数据库中，其出现频次为确定值. 但在不确定数据库中，其出现频次为随机变量，用 $sup^v(R)$表示. 表 4 为项 a 出现频次的概率分布. 由定义 2 可知，计算频繁概率 $FP(v)$的核心工作是计算频次分布函数 $sup^v(R)$. $sup^v(R)$的计算可以采用基于动态规划的方法[8]，计算公式如(4)所示.

$$sup^v(R)=A^v_{i,|R|},\ i=0,\cdots,num^v(R) \tag{4}$$

其中

$$A^v_{i,j}=\begin{cases}A^v_{i-1,j-1}\times p_i+A^v_{i-1,j}\times(1-p_i), & v_i=v\\ A^v_{i-1,j}, & v_i\neq v\end{cases},$$

$A^v_{i,j}$表示在前 i 个记录中项 v 出现 j 次的概率，p_i为第 i 个记录存在的概率. 利用 $A^v_{i,j}$进行递归计算，就可以得到整个数据集 R 上项 v 的频次分布信息.

在文献[8]的基础上，文献[9]提出了一种基于采样的近似聚集查询算法来处理不确定流式数据. 除了不确定频繁项查询研究之外，在不确定频繁项集挖掘方面也存在一些研究工作. 文献[10-11]分别基于 Apriori 和 FP-Growth 算法解决了期望频繁集项挖掘问题，文献[12-13]解决了概率频繁项集挖掘问题，但是上述文献均基于静态数据，没有考虑数据流环境. 文献[14-16]解决了不确定数据流中频繁项集挖掘问题，但采用的是更简单的期望频繁项定义. 根据上述分析可知，概率频繁项比期望频繁项更能反映频次的分布情况，因此本文采用更复杂的概率频繁项定义，针对传感器网络会产生大量实时数据流的特性，提出了一种精确的概率阈值频繁项查询算法，主要包括使用过滤策略构造候选集，对候选集压缩和候选集更新三方面内容，可以动态地更新结果并过滤掉大量不需要计算的元素.

3　问题定义

由于传感器网络产生无限的数据流，从理论上讲它的数据量也是无限大的，所以，要在所有的数据都采集后再计算传感器网络中的频繁项是不实际的. 因此，本文主要考虑最新采集的数据，也即是针对滑动窗口中的数据进行频繁项查询. 本文讨论的滑动窗口是基于元组的滑动窗口(tuple-based sliding window)，但是也可以扩展到基于时间戳的滑动窗口(timestamp-based sliding window).

目前许多滑动窗口采用每次仅更新一个元组的方式，但是当数据更新量较大时，此种方式效率不高. 因此本文采用批量更新的策略，即每次更新相同数目的记录，这些记录称为基本窗口. 本文以基本窗口作为更新处理单位，模型如图 1 所示.

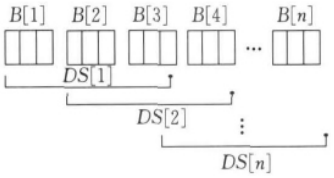

图 1 基于基本窗口的滑动窗口模型

在图 1 中每个基本窗口包含 3 个监测记录，每个滑动窗口由 3 个基本窗口组成. 其中 $DS[i]$表示第 i 个滑动窗口. 表 6 为本文使用符号的相关说明.

表 6 相关符号定义

符号	说明
R	不确定数据监测记录集
R_i	第 i 个监测记录
v_i	第 i 个监测记录值
p_i	第 i 个监测记录的概率
$minsup$	支持度阈值
$minprob$	概率阈值
$sup^v(R)$	v 在 R 中出现频次的概率分布
$num^v(R)$	v 在 R 中出现的次数($p_i>0$)
$exp^v(R)$	v 在 R 中的期望支持度
$DS[i]$	第 i 个滑动窗口
$B[i+j]$	第 i 个滑动窗口中的第 j 个基本窗口
BW_L	基本窗口长度
W_b	滑动窗口中包含基本窗口的数目
W_L	$W_L=BW_L\times W_b$

定义 3. 不确定感知数据流.

传感器网络连续不断地产生感知数据流 $DS=\{R_1,R_2,\cdots,R_n,\cdots\}$，$R_i$ 为第 i 个不确定检测记录，形式为二元组$\langle v_i,p_i\rangle$，v_i表示检测的记录值，p_i为概率. $DS[i]$窗口的有效记录为 $\{R_i,R_{i+1},\cdots,R_{i+W_{L-1}}\}$.

定义 4. $(minsup,minprob)$不确定感知数据流频繁项.

$$Pr\{sup^v(DS[i])\geq minsup\}>minprob \quad (5)$$

在当前窗口 $DS[i]$中满足公式(5)的项 v 即为$(minsup,minprob)$不确定感知数据流频繁项.

4 基于流的概率阈值频繁项查询算法

随着时间的变化，新数据不断地被传感设备采集，它们会进入到滑动窗口中，而一些旧数据将过期，不再参与频繁项运算，此时的查询结果将发生变化. 求解新频繁项的最简单办法就是采用静态频繁项查询算法[8]重新计算，这样当进行连续查询时，上面的过程将重复地执行. 显然，这样的策略是不可取的. 因为在连续时刻的窗口之间有相当大的一部分数据是相同的，所以两个时刻的查询结果之间也会有很多数据是相同的，有效地利用现有结果进行增量更新从而避免重复计算是提高算法性能的关键问题之一.

4.1 增量查询算法

由于本文以基本窗口作为更新处理单位，因此在计算频次分布函数时，也以基本窗口为单位进行计算. 基于该思想，本节提出了一种增量的概率阈值频繁项查询算法 bs-UFI，可以避免重复计算.

对每个基本窗口计算其频次分布$sup^v(B[i+j])$，进行频繁项查询时，需要将当前滑动窗口中所有基本窗口的计算结果进行合并，计算总的频次分布$sup^v(DS[i])$，然后计算频繁概率 $FP(v)$，计算总的频次分布函数如式(6)所示.

$$sup^v(DS[i])=\coprod_{j=0}^{W_b-1}sup^v(B[i+j]) \quad (6)$$

可以采用快速傅里叶变换的方法计算式(6)，时间复杂度为 $O(W_b m\log m)$，m 为基本窗口中项 v 出现的次数.

当前窗口 $DS[i]$数据过期时，无法在现有的$sup^v(DS[i])$计算结果上进行增量更新，必须重新计算. 为了避免重新计算，每到达一个基本窗口，只计算并保存该基本窗口的频次分布 $sup^v(B[i+j])$. 仅当窗口滑动时，才使用式(6)合并 W_b个频次分布函数. 尽管在当前窗口中需要保存 W_b个基本窗口的频次分布，但是其存储代价与保存总的频次分布函数相同，并未增加额外的存储空间，并且可以将计算的复杂度提升至与初始化阶段相同，均为 $O(W_b m\log m)$，具体算法如算法 1 所示.

算法 1. bs-UFI.

输入：DS；$minsup$；$minprob$；BW_L；W_b

输出：$Result$

```
1.  Result=NULL;
2.  FOR 每个新到达的基本窗口 B[i+j]
3.      FOR ∀v∈B[i+j]
4.          计算 sup^v(B[i+j]) (j=0… W_b-1);
5.    //滑动窗口
6.      FOR 每个项 v
7.          计算 sup^v(DS[i]), FP(v);
8.          IF (FP(v)>minprob)
9.              将⟨v,FP(v)⟩加入 Result 集合中;
10. //数据过期处理
11.     FOR 过期基本窗口中的每个项 v
12.         删除 sup^v(B[i+0]);
13. //数据插入处理
14.   FOR 新基本窗口的每个项 v
15.         计算 sup^v(B[new]).
```

对于每个到达的基本窗口，计算该窗口中所有项的频次分布(1～4 行). 当窗口滑动，合并所有基本窗口的频次分布函数，依据当前窗口的频次分布，计算概率频繁项并输出结果(5～9 行). 数据过期仅将当前窗口中最老基本窗口的频次分布删除即可(10～12 行). 插入数据时，计算新基本窗口的频次分布函数(13～15 行).

4.2　基于压缩候选集的增量查询算法

与静态数据不同，传感器产生的数据是不断更新的. 当前的非频繁项随着滑动窗口的前移可能会变为频繁项，为了避免重复计算，在基本算法中保留了所有项的信息，显然这种方法会保留大量低频度的项从而浪费了大量的存储空间. 针对该问题，本节对基本算法进行改进，提出了一种优化的查询算法，仅维护一些重要的项信息即候选集. 在候选集中大量的元素被剪枝，因此大幅度地降低了存储耗费. 算法共包括 3 部分内容：(1) 候选集构造；(2) 候选集压缩；(3) 候选集更新.

4.2.1　候选集构造

滑动窗口之间数据重复，在处理当前窗口时，可以预知哪些数据会参与到未来的滑动窗口中. 如图 1 所示，当窗口滑动，$B[1]$窗口元素过期，现有 $DS[1]$的结果不能直接用于计算新的 $DS[2]$窗口结果，但是当计算 $DS[1]$结果时，可以预知当前窗口中的部分数据会参与到未来的滑动窗口中，例如 $B[2]$，$B[3]$可以参与到 $DS[2]$窗口中，而 $B[3]$可以参与到 $DS[3]$窗口中. 基于不同窗口间的这部分重复数据可以对未来窗口中的查询结果进行预测，得到的预测结果称为 $pUFI(DS[i])$.

定义 5.　$pUFI(DS[i])$. $DS[i]$窗口的预测候选集.

预测候选集($pUFI(DS[i])$)构造的基本思想是在传感器节点设置一个过滤器，提前判断出那些不属于最终结果的项，并将它们过滤掉，仅维护必要的候选项集合.

由分析可知，频次分布函数 $sup^v(R)$满足泊松二项分布的定义，而泊松二项分布可以用更简单的泊松分布 $P(\lambda)$近似替代，近似的频繁概率可以使用式(7)计算，其中 F_p表示泊松分布的累积分布函数.

$$FP^v(DS[i])\approx 1-F_p(minsup-1,exp^v(DS[i])) \tag{7}$$

从式(7)可以看出在泊松分布中，仅利用期望就可以得到近似概率值. 由第 2 节可知，计算基本窗口的期望仅需要 $O(m)$的时间复杂度，而计算频繁概率需要 $O(m^2)$的时间复杂度. 因此利用期望剪枝掉部分项，避免进行复杂的概率计算会极大地提高查询效率.

由 $pUFI(DS[i])$的定义可知，对未来窗口候选集的构造依据的是当前窗口中与未来窗口中的重复数据，这部分仅仅是未来窗口的部分数据，因此无法计算未来窗口中项的精确期望值，但可以估算其上界. 由于传感器产生的数据是随机的，没有任何分布信息，因此只能采用最简单的方法进行预测，即假设后续数据全部与 v 取值相同. 利用该方法可以推测项 v 在未来 $DS[i]$窗口的期望上界，如式(8)所示，其中 j 表示未来窗口 $DS[i]$与当前窗口重复的基本窗口的个数.

$$up\_exp^v(DS[i])=BW_L*(W_b-j)+\sum_{k=0}^{j}exp^v(B[i+k]) \tag{8}$$

由泊松分布性质可知式(8)随期望单调递增. 因此可以通过项 v 的期望上界利用式(7)计算近似频繁概率的上界. 由文献[13]可知使用泊松分布计算的近似概率与精确概率之间的误差为 $\delta=\min((exp^v(DS[i]))^{-1},1)\sum_{i=1}^{|DS[i]|}p_i^2$，因此项 v 成为频繁项的概率上界如式(9)所示.

$$up\_FP^v(DS[i])=1-F(up\_exp_j(v))+\delta \tag{9}$$

若概率上界 $up\_FP_j(v)<minprob$，则项 v 可以过滤掉.

在图 2 中分别列出了图 1 所示 3 个窗口的候选集. 假设 $DS[1]$根据基本窗口 $B[1]$，$B[2]$和 $B[3]$当中的数据按照上述剪枝策略得到候选元素 2，3，则 $pUFI(DS[1])=\{2,3\}$. $DS[2]$根据基本窗口 $B[2]$和 $B[3]$当中的数据得到候选元素 1，2，3，$pUFI(DS[2])=\{1,2,3\}$. $DS[3]$根据基本窗口 $B[3]$当中的数据得到候选元素 1，2，3，4，$pUFI(DS[3])=\{1,2,3,4\}$.

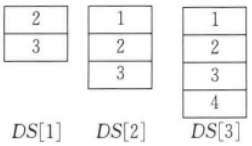

图 2　基于当前窗口 $DS[1]$的预测候选集

根据剪枝规则，随着数据的不断到达，可以逐步精化元素的期望值，从而求得更精确的频繁概率上界，过滤掉更多的元素，而不用采用基于动态规划的方法进行概率计算，降低了算法的开销. 实验结果表明，随着数据的到达，该方法可以过滤掉大量的

元素.

由于不同窗口候选集独立,窗口滑动时可以容易地对候选集进行维护.首先,基于当前窗口的候选集,通过基本算法得到概率频繁项.当数据过期时,如图 3 所示,将最老窗口中的候选集删除.当有新数据插入时,分别更新每个窗口中的候选集,对 $DS[2]$, $DS[3]$窗口中的候选集进行更新,并对最新到达的窗口构建新的候选集 $DS[4]$.

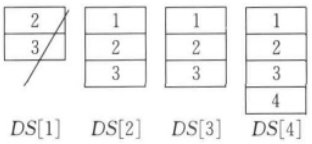

图 3 候选集更新

4.2.2 候选集压缩

从上述分析可以看出,对每个窗口独立维护候选集,可以很容易地得到查询结果,但是不同窗口的候选集存在大量的重复元素,如图 2 所示,项{1,2,3}同时出现在窗口 $DS[2]$, $DS[3]$中,如果将不同窗口中的相同数据压缩,只保留一个备份,会减少存储空间.因此,本文设计了一种带窗口标记的保存候选集元素的数据结构 cp-list,如图 4 所示.

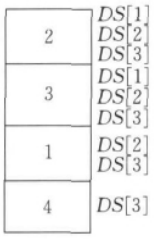

图 4 带有全部窗口标记的候选集

如图 4 所示,将图 2 中不同窗口中的重复元素压缩,并且增加窗口标识,用来标注该元素隶属于哪个窗口.元素的排列顺序按照起始窗口的标识从小到大排列,具有相同起始窗口标识的元素按照期望上界($up\_exp^v(DS[i])$)的值从大到小排列.

虽然通过使用窗口标记的方法可以减少存储空间,但是该方法需要保留所有窗口标记,算法需要对每个候选集中的项维护 W_b 个窗口标记,当 W_b 值较大时,维护窗口标记的代价也是不可忽视的.通过分析不同窗口中的预测候选集,发现了定理 1.

定理 1. $\forall v \in pUFI(DS[i]) \wedge v \in pUFI(DS[j])$,则对于 $\forall l \in [i,j]$, $v \in pUFI(DS[l])$,其中 $i<j$, $j=i+1,\cdots,i+W_b-1$.

证明. v 是 $DS[i]$ 的候选元素,说明 $up\_FP^v(DS[i]) > minprob$,计算概率上界 $up_FP^v(DS[i])$采用的期望值为 $up\_exp^v(DS[i])$.因为 $i<j$,即假设在未来窗口 $DS[l]$中,项 v 全部出现且概率为 1,所以 $up\_exp^v(DS[i]) \leq up\_exp^v(DS[l])$.因为概率随期望单调递增,$up\_FP^v(DS[i]) \leq up\_FP^v(DS[l])$,所以 $up\_FP^v(DS[l]) > minprob$, $v \in pUFI(DS[l])$.

证毕.

如果项 v 是 $DS[i]$窗口的候选元素,且 v 是 $DS[j]$窗口的候选元素,则 v 一定是所有 $DS[i]$和 $DS[j]$之间所有连续窗口中的候选元素.因此仅记录起始窗口标记(W_{start})和其结束窗口标记(W_{end})即可,如图 5 所示.

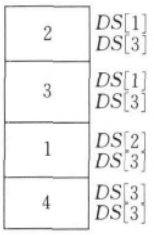

图 5 仅带有起始和结束标记的压缩候选集

通过使用压缩的 cp-list 存储结构,且仅记录起始和结束窗口标记,可以极大地减少存储空间,这对于数据流处理是非常重要的.

4.2.3 候选集更新

采用压缩的 cp-list 存储结构可以减少存储空间,但是当窗口更新时,由于不同窗口的候选集都存放在同一个候选集列表中,则候选集的更新比独立存放的方法要复杂.下面从删除过期数据和插入新数据两方面描述候选集列表的维护算法.

(1)删除过期数据.

在没有使用压缩方法的算法中,元素过期仅删除当前最老窗口中的候选集即可.而采用压缩的 cp-list 结构,所有窗口的候选集压缩存放在一起,不能简单地删除.具体方法为先查询起始窗口标记,找到最老窗口中的候选元素,并将起始窗口标记加 1.如果起始窗口标记大于结束窗口标记,则真正删除该元素.

由于不同窗口中的候选元素都保存在如图 5 所示的 cp-list 中,因此如何快速的找到最老窗口的候选元素也是关键问题之一.本算法在构造 cp-list 时,按照起始窗口标记从小到大对元素进行排列,因此在查询时,仅需将 cp-list 中最前面的元素"删除"即可,直到遇到第一个次老窗口中的元素为止.如图 5 所示,从头开始对 cp-list 中的元素进行处理,首先处理元素 2,得到起始窗口标记 $DS[1]$,因为 $DS[1]$

是当前最老的窗口，将其起始标记加 1，变为 $DS[2]$，继续处理下一个元素，当处理到元素 1 的时候，发现起始窗口为 $DS[2]$，则不需继续处理，因为已经将所有 $DS[1]$窗口中的候选元素处理完毕.

以图 1 所示滑动窗口为例，当 $B[1]$基本窗口数据过期后，图 5 更新后的处理结果如图 6 所示.

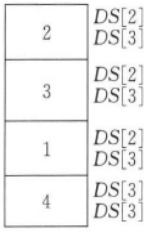

图 6　压缩候选集更新(数据删除)

(2) 插入新数据.

首先对不同窗口中的候选集进行更新. 随着新基本数据的到达，可以进一步精化每个未来窗口中元素的期望值，从而得到更准确的概率上界，进而得到更准确的候选集. 因为元素按起始窗口标记从小到大排列，因此从头对元素依次处理，即可以更新每个窗口的候选集.

对于新基本窗口的数据，同样按照 4.2.1 节的方法构造候选集，若新窗口的候选元素已经在 cp-list 中存在，则将该元素结束窗口标记加 1；否则在 cp-list 末尾插入该元素，起始标记和结束标记均为该窗口的标号.

以图 1 滑动窗口为例，当 $B[4]$基本窗口到达时，$DS[2]$成为当前窗口，包括 $B[2]$，$B[3]$和 $B[4]$基本窗口中的数据，可以进一步缩小候选元素的范围，假设更新后的候选集为{2,3}，按照同样方法更新 $DS[3]$候选集，假设结果为{2,3,4}，利用 $B[4]$当中的数据对 $DS[4]$窗口进行预测，假设结果为{2,3,4,5}，更新后的 cp-list 结构如图 7 所示.

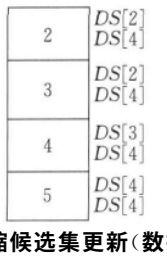

图 7　压缩候选集更新(数据插入)

4.2.4　优化算法(is-UFI)

由于基本算法存在查询效率不高的问题，因此本节在基本算法 bs-UFI 的基础上提出了一种优化算法 is-UFI，如算法 2 所示. 该算法使用泊松分布过滤策略构造候选集，并使用 cp-list 压缩结构动态地维护候选集合.

算法 2.

输入：DS；$minsup$；$minprob$；BW_L；W_b

输出：概率频繁项集合 $Result$

```
1.  Result=NULL;
2.  FOR 每个新到达的基本窗口 B[i+j]
3.    FOR ∀v∈B[i+j]
4.      计算 exp^v(j), up_exp^v(j), up_FP^v(j);
5.      IF up_FP^v(j)<minprob
6.        过滤掉项 v;
7.      ELSE
8.        IF v∉ cp-list
9.          /*按照 v.start 窗口标记升序排列，具有相
            同 v.start 值的元素按照 up_exp(i+j)值降
            序排列的方式加入 cp-list*/
10.         将v加入cp-list; v.start=i+j; v.end=i+j;
11.       ELSE
12.         v.end=i+j;
13.       计算 sup^v[j];
14.   //窗口滑动
15.   FOR 每个没有被过滤掉的项 v
16.     根据 sup^v(DS[i])计算频繁概率 FP(v);
17.     IF FP(v)>minprob
18.       将项v以〈v,FP(v)〉形式加入Result 集合中;
19.   //数据过期处理
20.   FOR 过期基本窗口的每个项 v
21.       删除 sup^v(B[i+0]);
22.       v.start++;
23.       IF v.start>v.end
24.         将项 v 从 cp-list 中删除;
25.   //数据插入处理
26.   创建新的预测候选集 pUFI(B[new]);
27.   将 pUFI(B[new])加入 cp-list;
28.   按照式(4)计算 sup'(B[new]).
```

对于每一个新到达的基本窗口 $B[i+j]$，计算该基本窗口中所有项的期望，并将该期望值与前面基本窗口的期望进行累加，得到真实的 $exp^v[j]$，期望上界 $up_exp^v(j)$和概率上界 $up\_FP^v(j)$(1～4 行)，若 $up_FP^v(j)$小于阈值，该项被过滤掉(5～6 行). 对于没有过滤掉的项 v，如果该项不在 cp-list 中，按照起始窗口标记升序排列的方式将其加入到 cp-list 中，标记起始窗口和结束窗口(8～10 行). 如果该项在 cp-list 中，将项 v 结束窗口标记改为当前窗口标记值(11～12 行). 计算每个基本窗口的频次分布(13 行). 当窗口滑动时，根据 $sup^v(DS[i])$，计算当前滑动窗口的频繁项(14～18 行). 数据过期

时，将起始窗口标记增 1，如果起始窗口标记小于结束窗口标记，则删除该项(19～24 行)．当新数据到达，计算新的基本窗口的候选元素，并将该候选集加入 cp-list 中(25～27 行)，最后计算新基本窗口的频次分布函数(28 行)．

时间复杂度．对于每一个候选元素需要计算期望及出现次数进行过滤条件的判断，因此需要扫描滑动窗口中的数据，时间复杂度为 $O(W_L)$，对于没有过滤掉的元素，需要利用动态规划的方法计算概率，每一个基本窗口中计算概率的时间复杂度为 $O(m^2)$(m 为项 v 在基本窗口中出现的次数)．在整个滑动窗口计算概率的时间复杂度为 $O(W_b m\log m+W_b m^2)$．因此，对于单个候选元素，总的时间复杂度为 $O(W_L)+O(W_b m\log m+W_b m^2)$．

空间复杂度．对于每一个候选元素，需计算期望，空间复杂度为 $O(1)$．另外，记录 $sup^v(DS[i])$ 的空间复杂度为 $O(num^v(DS[i]))$，所以，对于单个元素总的空间复杂度为 $O(num^v(DS[i]))$．

5 实 验

由于目前尚没有采用滑动窗口模型的概率频繁项查询算法，本文将单窗口下的 UFI 算法[8](PHH)应用到滑动窗口下，作为与本文提出的两个算法的比较基准．因此本文共对 3 个算法进行评测：单窗口频繁项查询算法(PHH)、滑动窗口频繁项查询基本算法(bs-UFI)和滑动窗口频繁项查询优化算法(is-UFI)．

实验环境．实验的硬件环境为 Pentium 3.0GHz CPU，4GB 内存，操作系统为 Windows XP，并采用 Microsoft VC++6.0 编程环境开发了模拟测试程序．

数据集．实验采用两组数据集，一组是真实数据集 accidents，来自 Frequent Itemset Mining(FIMI)，另一组数据使用 IBM Data Generator 发生器，分别生成 10 万个元组．由于这两个数据产生的都是频繁项集，本文对其简单修改，只保留其中一组属性的值，构造频繁项．对于第 1 组真实数据集，通过分析数据取值的分布情况，按照相应的分布为每个元组分配概率值．第 2 组数据采用平均分布．其它参数的缺省值是 $minsup$ 为 100，$minprob$ 为 0.3，基本窗口的元组数为 1000，滑动窗口的元组数为 10000．

剪枝能力分析．剪枝能力考查了优化算法 is-UFI 过滤元组的能力．因为在 bs-UFI 算法中，没有使用过滤策略，因此剪枝掉的元素数目为 0，故没在图中对该算法进行表示．图 8、图 9 表示了剪枝能力随 $minsup$，$minprob$ 变化的情况．从图 8、图 9 可以看出，随着 $minsup$，$minprob$ 的逐渐增加，剪枝能力不断地增强，由于 IBM 数据相对稀疏，因此剪枝掉的元组数目较多．

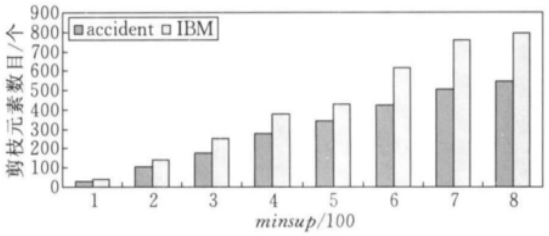

图 8 剪枝元素个数($minsup$)

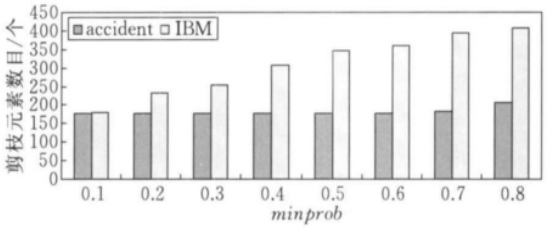

图 9 剪枝元素个数($minprob$)

运行时间分析．图 10 显示参数 W 对时间的影响．W 表示滑动窗口中元组的总个数，取值范围为 5000～10000．(a)是真实数据集的结果，(b)是 IBM 数据的结果．如图 10 所示，随着数据量增加，算法运行时间也会增加．但是，静态的频繁项查询算法(PHH)时间增加明显，因为需要重复计算．而我们提出的算法 bs-UFI 与 is-UFI 增加相对平缓．is-UFI 的算法性能要优于 bs-UFI，因为 is-UFI 算法中仅使用期望就可以过滤掉部分元素，不需要再使用动态规划方法进行概率计算，因此节省了大量的时间．

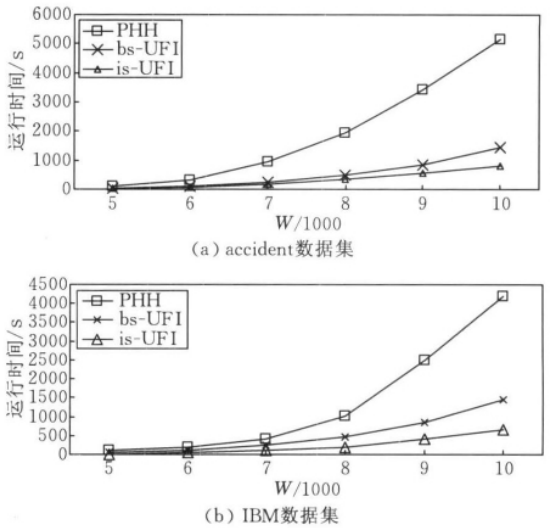

图 10 参数 W 对时间的影响

图 11 显示参数 $minsup$ 对时间的影响. 参数 $minsup$ 为频次阈值，取值范围为 100～1000. 如图 11 所示，is-UFI 算法总是优于 bs-UFI. 因为，在窗口更新的过程中，会剪枝掉一些元组，且计算概率可以利用之前的结果，所以节省了时间. 另外，随着 $minsup$ 的增加，剪枝元组数目增多，时间递减，而没有使用剪枝策略的算法对所有的元素都要计算概率，因此时间不变.

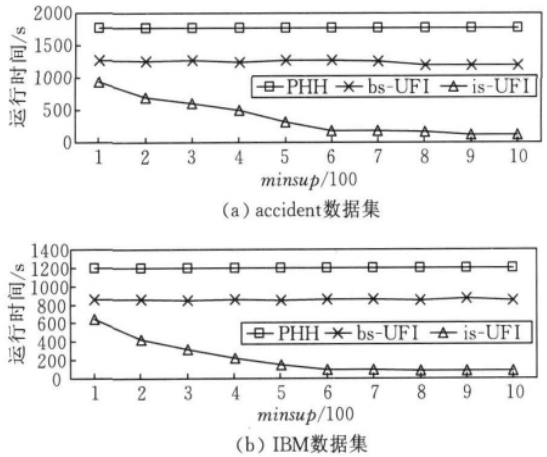

图 11　参数 $minsup$ 对时间的影响

图 12 显示参数 $minprob$ 对时间的影响. 参数 $minprob$ 为概率阈值，取值范围为 0.1～0.8. 如图 12 所示，我们的算法总是优于基本算法. 原因与 $minsup$ 相同. 但是与 $minsup$ 相比，随着 $minprob$ 的增加算法时间下降平缓，说明 $minprob$ 的过滤能力比 $minsup$ 弱.

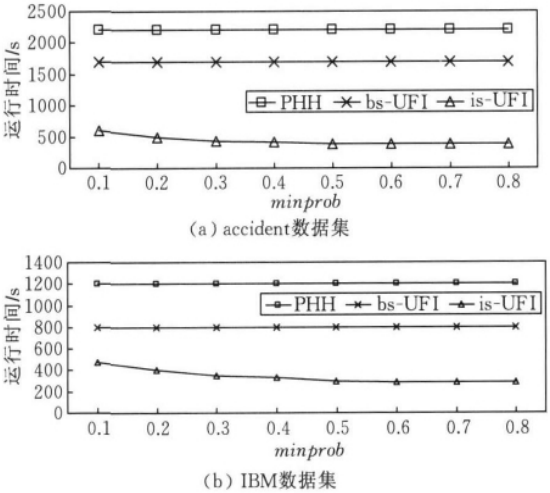

图 12　参数 $minprob$ 对时间的影响

图 13 显示基本窗口 BW_L 对时间的影响. 参数 L 为基本窗口的尺寸. L 的取值从 100～1000. 如图 13 所示，随着 L 的增加，基本窗口的元组数目增加，因此计算基本窗口频次分布函数的代价增大. 采用快速傅里叶算法计算总的分布函数时，需要合并的基本窗口数目减少，但是由于每个基本窗口的项数目增加，所以在合并时计算代价仍然增加. 对于 PHH 算法，每次窗口滑动，都需要重新处理，所以时间代价不变.

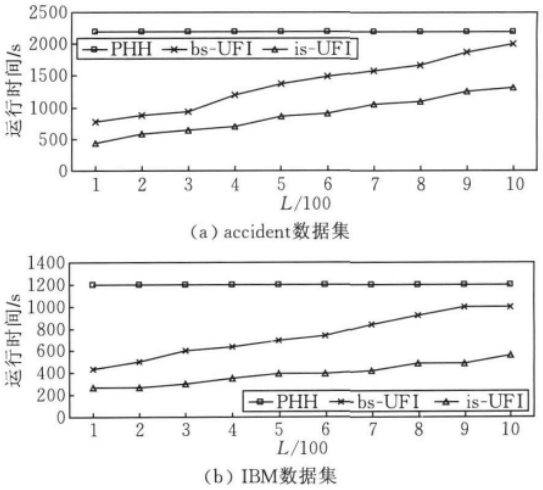

图 13　参数 BW_L 对时间的影响

内存使用. 图 14～图 16 分析了不同参数对 bs-UFI 和 is-UFI 算法内存使用的影响. 这里只采用了真实数据集进行测试. 图 14 测试了滑动窗口尺寸对内存的影响. 由于 bs-UFI 没有使用剪枝，需要对所有数据都进行概率计算，其空间与元素数目成线性增加. 而使用了剪枝的 is-UFI 算法，有大量元素被剪枝，因此空间增加相对平缓.

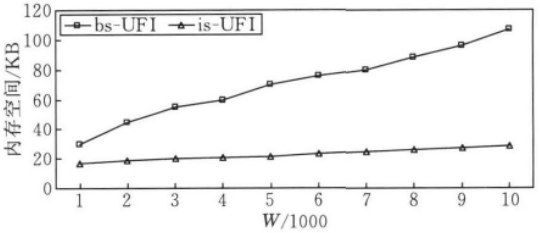

图 14　窗口尺寸对内存的影响

图 15、图 16 测试了参数 $minsup$，$minprob$ 对内存的影响. 没有使用剪枝的算法每次处理元素数目不变，因此内存消耗不变. 而使用剪枝的算法随 $minsup$，$minprob$ 的增加，剪枝数目增多，内存消耗逐渐递减.

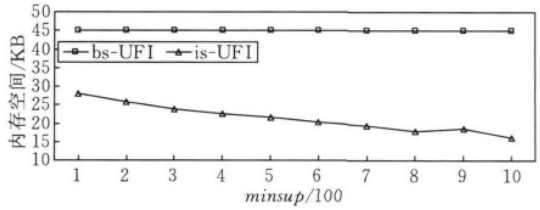

图 15　$minsup$ 对内存的影响

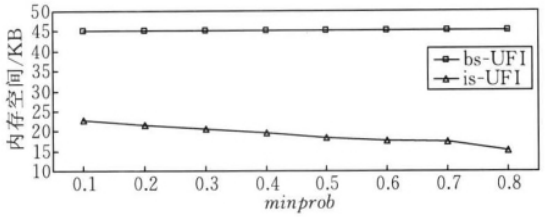

图 16 *minprob* 对内存的影响

扩展性. 最后测试了 bs-UFI 和 is-UFI 算法改变数据库尺寸对时间和空间性能的影响. 这里只采用 IBM 数据集进行测试. 图 17 表明随着数据数目的增加,两个算法的运行时间均呈现线性增加的趋势. 但是,使用了剪枝的 is-UFI 算法较 bs-UFI 增加相对平缓. 因为 is-UFI 算法使用了剪枝策略,有大量元素被剪枝,因此处理时间较少. 相似地,从图 18 可以看出算法的空间性能随元组数目的增加同样呈现线性增加的趋势.

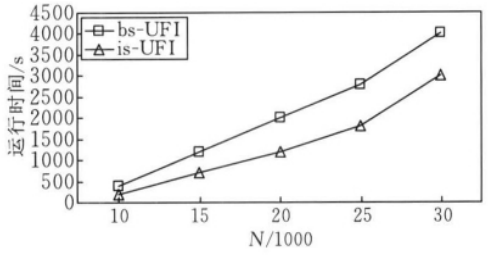

图 17 数据库尺寸对时间的影响

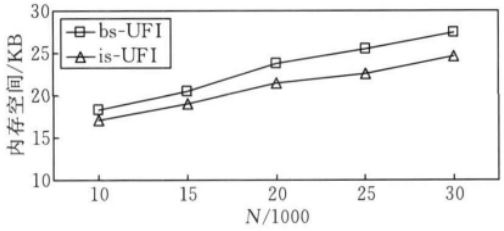

图 18 数据库尺寸对内存的影响

6 结 论

传感器网络产生大量的感知数据流,由于自身特性,使得数据带有不确定性. 本文针对这种不确定数据流,提出了基于滑动窗口的概率阈值频繁项查询算法,为此,首先提出了增量查询算法 bs-UFI,以避免重复计算,进而提出了基于泊松分布的剪枝策略,以有效地过滤掉大量不必计算的项,最后提出了 cp-list 数据结构,以压缩不同窗口之间的候选集,减小存储空间. 实验证明算法在时间和空间均具有高效性.

由于引入了不确定性,使得频繁项查询问题变得非常复杂,因此本文仅考虑了集中式的算法. 但传感器网络实际是一个分布式的数据流系统,对于分布式情况下设计一种旨在减少网络通信量的不确定流频繁项查询算法还有待于进一步研究. 另外,有些应用只需要查询近似频繁项,并不需要精确结果,因此如何在精度和查询代价之间进行折中,使之既能满足用户需求,又能提高查询效率,也是下一步工作的重点.

参 考 文 献

[1] Li Jian-Zhong, Li Jin-Bao, Shi Sheng-Fei. Concepts, issues and advance of sensor networks and data management of sensor networks. Journal of Software, 2003, 14(10): 1717-1727(in Chinese)
(李建中, 李金宝, 石胜飞. 传感器网络及其数据管理的概念、问题与进展. 软件学报, 2003, 14(10): 1717-1727)

[2] Zhou Ao-Ying, Jin Che-Qing, Wang Guo-Ren, Li Jian-Zhong. A survey on the management of uncertain data. Chinese Journal of Computers, 2009, 32(1): 1-16(in Chinese)
(周傲英, 金澈清, 王国仁, 李建中. 不确定性数据管理技术研究综述. 计算机学报, 2009, 32(1): 1-16)

[3] Green T J, Tannen V. Models for incomplete and probabilistic information. IEEE Date Engineering Bulletin, 2006, 29(1): 17-24

[4] Considine J, Li F, Kollios G, Byers J. Approximate aggregation techniques for sensor databases//Proceedings of the 20th International Conference on Data Engineering. Boston, USA, 2004: 449-460

[5] Manjhi A, Nath S, Gibbons P B. Tributaries and deltas: Efficient and robust aggregation in sensor network streams//Proceedings of the 2005 ACM SIGMOD International Conference on Management of Data. Baltimore, USA, 2005: 287-298

[6] Ren Meirui, Guo Longjiang. Mining recent approximate frequent items in wireless sensor networks//Proceedings of the 6th International Conference on Fuzzy Systems and Knowledge Discovery. Tianjin, China, 2009: 463-467

[7] Cormode G, Garofalakis M. Sketching probabilistic data streams//Proceedings of the 2007 ACM SIGMOD International Conference on Data Management. Beijing, China, 2007: 281-292

[8] Zhang Qin, Li Feifei, Yi Ke. Finding frequent items in probabilistic data//Proceedings of the 2008 ACM SIGMOD International Conference on Data Management. Vancouver, Canada, 2008: 810-832

[9] Wang Qiu-Tang, Wang Peng, Zhou Hao-Feng, Wang Wei. Estimating sliding window-based aggregate queries over probabilistic data streams. Journal of Computer Research and Development, 2008, 45(Suppl.): 169-174(in Chinese)

(王秋棠，王鹏，周皓峰，汪卫. 基于滑动窗口的概率数据流上的聚集查询. 计算机研究与发展，2008，45(Suppl.)：169-174)

[10] Chui Chun Kit, Kao Ben. A decremental approach for mining frequent itemsets from uncertain data//Proceedings of the 12th Pacific-Asia Conference on Knowledge Discovery and Data Mining. Osaka, Japan, 2008: 64-75

[11] Leung C K S, Mateo M A F, Brajczuk D A. A tree-based approach for frequent pattern mining from uncertain data//Proceedings of the 2008 Pacific-Asia Conference on Knowledge Discovery and Data Mining. Osaka, Japan, 2008: 653-661

[12] Bernecker Thomas, Kriegel Hans-Peter, Renz Matthias, Verhein Florian, Zuefle Andreas. Probabilistic frequent itemset mining in uncertain databases//Proceedings of the 15th ACM SIGKDD Conference on Knowledge Discovery and Data Mining. Paris, France, 2009: 119-128

[13] Wang Liang, Cheng Reynold, Lee Sau Dan, Cheung David W. Accelerating probabilistic frequent itemset mining: A model-based approach//Proceedings of the 19th ACM Conference on Information and Knowledge Management. Toronto, Canada, 2010: 429-438

[14] Leung Carson Kai-Sang, Jiang Fan. Frequent itemset mining of uncertain data streams using the damped window model//Proceedings of the 2011 ACM Symposium on Applied Computing. Taichung, China, 2011: 950-955

[15] Leung Carson Kai-Sang, Hao Boyu. Mining of frequent itemsets from streams of uncertain data//Proceedings of the 25th International Conference on Data Engineering. Shanghai, China, 2009: 1663-1670

[16] Liao Guo-Qiong, Wu Ling-Qin, Wang Chang-Xuan. Frequent patterns mining over uncertain data streams based on decay window model. Journal of Computer Research and Development, 2012, 49(5): 1105-1115(in Chinese)
(廖国琼，吴凌琴，万常选. 基于概率衰减窗口模型的不确定数据流频繁模式挖掘. 计算机研究与发展，2012，49(5)：1105-1115)

WANG Shuang, born in 1980, Ph.D., lecturer. Her research interests include uncertain data management, data stream query, etc.

WANG Guo-Ren, born in 1966, professor, Ph.D. supervisor. His research interests include XML data management, bioinformatics, distributed database, and parallel computing.

Background

The research of management on uncertain data starts from the late 80's last century, and becomes a very hot field today. Data uncertainty widely appears in various applications, inclusive of economy, military, finance and telecommunication, et al. Computing statistical information on uncertain data stream has attracted a lot of attention recently.

In this paper, we study how to efficiently discover frequent items from large uncertain sensing data. This is technically challenging, since: (1) an uncertain database induces an exponential number of possible worlds; and (2) the mining results may be different while new tuples are being inserted into and old tuples are expired. For uncertain data stream, some algorithms have proposed for mining frequent items, but the definition is based on expectation and not for sliding window model. This paper focus on the probabilistic frequent items mining in sliding window model. Then, an exact incremental mining algorithm is presented based on our framework for maintaining threshold-based frequent items. In order to reduce the computation cost, the algorithm incrementally updates the results upon each window slide and thus eliminates the need for complete recomputation from scratch. The pruning rules are proposed, which can reduce the candidate set and improved the query efficiently.

This work was supported by the State Key Program of National Natural Science Foundation of China (Grant No. 60933001), the National Science Foundation for Distinguished Young Scholars of China (Grant No. 61025007), the National Science Foundation for Young Scientists of China (Grant No. 61100022).