doi:10.11959/j.issn.1000-436x.2017286

基于规则前件发生树匹配的数据流预测方法研究

尤涛，李廷峰，杜承烈，钟冬，朱怡安
（西北工业大学计算机学院，陕西 西安 710129）

摘 要：现有基于规则匹配的数据流预测算法存在前件发生定义不准确、前件相关性未考虑、预测结果描述不严谨等不足，造成预测过程效率较低、精度不高等问题。提出基于前件发生树的概率叠加预测算法，定义区间最小非重叠发生，避免前件的错误匹配；通过前件的合并构建前件发生树，提高前件发生的搜索效率；基于概率叠加的思想计算后件的发生区间和发生概率，使预测精度进一步提高。理论分析和实验结果表明，该算法具有较高的时空效率和预测精度。

关键词：数据流；情节规则；区间最小非重叠发生；前件发生树；概率叠加预测
中图分类号：TP311 **文献标识码**：A

Data stream prediction based on rule antecedent occurrence tree matching

YOU Tao, LI Ting-feng, DU Cheng-lie, ZHONG Dong, ZHU Yi-an
(School of Computer Science and Engineering, Northwestern Polytechnical University, Xi'an 710129, China)

Abstract: There are some shortages in the existing rule-based data stream prediction algorithm, such as inaccurate definition of antecedent occurrence, ignoring the correlation between rules and imprecise description of prediction accuracy. These make low forecasting process efficiency and low prediction accuracy. The superposed prediction algorithm was proposed based on antecedent occurrence tree, and interval minimal non-overlapping occurrence was defined to avoid the problem of excessive matching antecedent. The efficiency was improved for searching antecedent's occurrence by merging rule's antecedents in antecedent occurrence tree, and the succedent occurrence based on superposed probability was predicted to enhance prediction accuracy. The theoretical analysis and experimental evaluation demonstrate the algorithm is superior to the existing prediction algorithms in terms of time and space efficiency and prediction accuracy.

Key words: data stream, episode rule, interval minimal non-overlapping occurrence, antecedent occurrence tree, prediction based on superposed probability

1 引言

数据流是由一系列值对（事件类型、发生时间）构成的序列。由于历史流数据中蕴含着大量信息，研究历史流数据的潜在规律并应用这些规律作出预测，能够为许多现实应用提供重要的决策支持[1~3]。随着信息技术的不断发展，数据流预测模型被广泛应用到众多领域，如网络安全监控、金融证券管理、事务日志分析、无线射频识别与传感器数据处理等。

数据流高速、无界、连续、时变等特性，给数据流预测带来了时空复杂性、实时性、自适应性等方面的要求。目前，针对数据流预测方面的研究，诞生了许多预测方法，如典型的基于回归分析方法[4,5]、规则匹配[6~8]的数据流预测方法等。在数据流的众多预测方法中，回归分析预测方法不能预测数据流上的非线性数据（如事件等），具有一定的应用局限性。

规则匹配方法首先在历史流数据中抽取反映情节之间因果联系的情节规则，再利用这些情节规

收稿日期：2017-03-06；**修回日期**：2017-06-11
基金项目：2017 航空科学重点基金资助项目；2016 复杂产品智能制造系统技术国家重点实验开放研究基金资助项目
Foundation Items: 2017 Aviation Science Key Foundation of China, 2016 Open Fund of State Key Laboratory Intelligent Manufacturing System Technology

则在线匹配前件，从而实现对未来流数据的预测。但是现有算法存在前件匹配不精确、忽略规则间的相关性、预测精度描述不严谨等问题。这促成了本文的研究动机，详述如下。

首先，一个情节规则的前件可能在当前数据流上发生多次，但并非所有的发生对预测产生意义。针对数据流 *DS*=<(A,1), (A,2), (B,3), (A,4), (A,5), (A,6), (B,7), (A,8), (B,9), (A,10), (B,11)>，根据最小发生、非重叠发生、最小非重叠发生、最近最小非重叠发生、区间最小非重叠发生的定义，某规则<AAB,C,5, 90%,9>的前件 AAB 分别有以下 5 个发生集。

1) 最小发生集[9]{[1,3],[5,7],[6,9],[8,11]}：该发生集的基数最大，均用于预测的话，会造成过匹配现象，例如，发生[6,9]与[5,7]存在重叠，对预测没有意义。

2) 非重叠发生集[10]{[1,3],[4,7],[8,11]}：该发生集的每个发生未必是最小发生，例如，区间[4,7]上存在<AAB>的另一次发生[5,7]([4,7]并非最小发生)，影响预测精度。

3) 最小非重叠发生集[11]{[1,3],[5,7],[8,11]}：该发生集虽然保留了最小发生和非重叠发生的优点，但根据规则窗口宽度，有些发生可能对未来某时刻没有预测意义，造成过匹配现象。例如，发生[1,3]的起始时刻 1 与截止时刻 11 的间隔为 10，已超过规则的窗口宽度 9，没有预测意义。

4) 最近最小非重叠发生集[12]{[8,11]}：该发生集只考虑离截止时刻最近的有效发生，忽略了有效窗口内的发生[5,7]，造成欠匹配现象。

5) 区间最小非重叠发生集{[5,7],[8,11]}：考虑规则窗口宽度是 9，有效区间的开始时刻是从截止时刻往前推进 8 个（窗口宽度减去后件长度）时刻，即[5,12]。该发生集在最小非重叠发生的基础上，结合规则的有效窗口宽度限制，避免了前件匹配的“过”“欠”现象。

其次，前件的重复匹配。运用情节规则进行数据流预测时，往往会出现前件完全相同或前件部分相同的情况。已有方法往往忽略这些信息而直接分别进行匹配，效率低下。

再次，已有的预测方法未考虑各有效发生对预测的影响（例如，有效预测区间内非最近的最小发生对预测的影响）。由于每个前件发生对应一个后件预测区间，若在有效窗口内存在前件多次发生，则后件的多个预测区间需要综合考虑。

基于上述分析，本文提出了基于前件发生树的预测（SPAOT, superposed prediction based on antecedent occurrence tree）算法。该算法基于前件发生树结构高效查找多规则前件的区间最小非重叠发生，并将区间内各有效发生的影响作用有效体现在预测结果中。理论分析和实验结果表明，SPAOT 算法具有较高的预测效率和精度。

2　相关工作

现有的相关研究依据其采用预测模型的不同可分为隐马尔可夫模型方法、情节规则方法。

隐马尔可夫模型方法中，EGH 算法[1]分为 2 个阶段。1) 训练阶段，基于非重叠发生的情节支持度定义，考虑频繁模式间的混合度，争取将多个频繁模式对应到一个隐马尔可夫模型（HMM, hidden Markov model）；2) 预测阶段，查找频繁情节的最近一次非重叠发生，完成预测。HMM_BW 算法[13]针对数据流的增量特性，使用在线 HMM 训练算法，根据频繁模式动态改变 HMM 参数。HMM 模型的匹配预测方法有以下不足。首先，HMM 模型待匹配形式严格，很难进行模式间的合并导致匹配效率较低；其次，预测阶段旨在查找模式发生，并非所有的发生对预测均有意义，容易导致预测区间过时等问题；最后，预测结果只能是预测未来下一个时间点的数据，无法同时预测多个目标事件类型的发生。

与 HMM 模型预测方法不同，DeMO[6]、CBS-Tree[3]、ToFel[7]、Predictor[14]等算法依赖于历史数据蕴含的情节规则，通过规则前件的匹配完成后件发生的预测。DeMO 算法引入 2 个优化技术来修剪前件的非最小发生区间、超过规则窗口宽度的发生以避免不必要的队列维护。但 DeMO 算法也存在一些不足。1) 匹配规则形式严格，限定了规则的前件必须为单映射情节。2) 规则过匹配，因为它考虑的是前件的最小发生，数据流上这样的发生次数要远远多于非重叠的发生。3) 预测区间可能过时，因为它们在找到规则前件最近的最小发生时，给出的预测区间是规则的窗口宽度减去该次发生的起始时间，显然，这个预测区间可能出现在截止时刻之前，达不到预测的目的。

CBS-Tree 算法采用后向检索规则前件策略，用叶子节点个数固定的完全二叉树来存储和维护数据流，按照与规则前件拓扑结构相反的顺序在该完全二叉树中查找前件此区间内最近的最小发生以

实现预测。算法所需的存储空间只与所有规则中最大的前件窗口宽度有关，而与待匹配的规则个数无关，空间复杂度较低；但是在 CBS-Tree 中查找前件的发生非常耗时，时间代价较大。

ToFel 算法将待匹配规则的前件看作是一个拓扑图，并将数据流的演化看成是一个由符合拓扑关系的所有事件组成的队列，通过在队列上查找规则前件最近的最小发生来进行预测。由于队列维护时记录了规则前件的多次不完整发生，所以 ToFel 的时间代价也比较大。

Predictor 算法为每个待匹配情节规则分别使用了一个规则自动机，通过单遍扫描数据流来同时跟踪这些自动机的状态变迁，以搜索每个规则前件的最近最小非重叠发生。算法虽没有采用规则合并的策略，但进行了事件扫描操作的合并，提高了前件的搜索效率。然而，算法定义的最近最小非重叠发生仍会出现过匹配（最近的也有可能超出预测区间）、欠匹配（忽略其他有效匹配）问题，影响预测效率。

另外，与规则匹配相关的研究中，复杂事件处理技术得到了广泛的关注。已有的研究主要可以分为 NFA[15]、Petri 网[16]和匹配树[17]3 种。NFA 将一个复杂事件表达式构造为自动机，事件到达将触发自动机的状态迁移，自动机进入可接受状态表明检测出一个满足条件的复杂事件实例；Petri 网是一种适合于描述异步并发现象的系统模型，它根据复杂事件表达式构造其对应的 Petri 网实例，以 Petri 网输入节点为基本事件，输出节点为所要检测的复杂事件，通过输入令牌，计算跃迁函数，如果成立则产生跃迁并标记此处的节点，当最后一个节点被标记出说明产生对应的复杂事件；匹配树的基本思想是由复杂事件表达式构造出相应识别树，其中，基本事件为树的叶节点，各个层次的复杂事件为树的中间节点，如果到达根节点，就认为检测到了一个复杂事件。但是，它们各有优缺点：NFA 简单，易于理解和实现，但只有到达最终状态才回溯匹配状态，同时由于其自身结构的限制，在表达否定算子和并发事件方面存在不足；Petri 网功能强大，但表达和执行比较复杂，难以支持复杂判断，并且对 Petri 网的性能分析也很复杂；匹配树方法侧重研究通过树方式来支持丰富的查询语义，但是由于树的构建方式多样，不同的构建方式的效率差别很大，同时树结构如何充分利用中间节点找寻最优构建方式以及丰富语义仍有待研究。可见，复杂事件处理的研究成果并不能直接解决问题。

3 定义

定义 1 事件、事件序列。给定事件类型集 $\varepsilon=\{E_1, E_2,\cdots,E_n\}$，一个事件就是一个二元组$(E,t)$，其中，$E\in\varepsilon$，$t$ 表示事件发生时间。事件序列是由若干ε中的事件按发生时间先后排列的序列，表示为 $ES=<(E_1,t_1),(E_2,t_2),\cdots,(E_s,t_s)$。

定义 2 情节。一个情节是由若干事件组成的序列，表示为 $ep=<(E_1,t_1),(E_2,t_2),\cdots,(E_k,t_k)>$，简记为 $\alpha=<E_1, E_2,\cdots,E_k>$。

定义 3 发生。给定事件序列 ES 和情节 $ep=<E_1, E_2, \cdots,E_k>$，若 ES 在时间$[t_1,t_k]$上按 ep 的事件排列顺序出现了 ep 所代表的所有事件，则称 ES 上发生了情节 ep，时间$[t_1,t_k]$称为 ep 在 ES 上的一次发生。

定义 4 区间最小非重叠发生。在给定$[t_s,t_e]$内找到情节 ep 在事件序列 ES 上的一次发生$[t_1,t_k]$，若在 ES 上不存在 ep 在此区间内的另一次比它还小的发生，则称$[t_1,t_k]$是 ep 在 ES 上的一次区间最小非重叠发生。可见，该发生可能有多个。

定义 5 情节规则。一个情节规则γ是一个五元组(l, r, s, c, w)。其中，l、r、s、c、w 分别称为γ的前件、后件、支持度、置信度和窗口宽度。$\gamma.s$ 用于衡量γ的统计特性，它等于情节 $concat(\gamma.l,\gamma.r)$的支持度；$\gamma.c$ 用于衡量γ的可信程度，它等于情节 $concat(\gamma.l,\gamma.r)$的支持度与情节 $concat(\gamma.l)$的支持度的比值；$\gamma.w$ 用于衡量γ的时间特性，表示情节 $concat(\gamma.l,\gamma.r)$发生时终止时间与起始时间的最大差值，亦即规则的前件和后件必须在指定的窗口宽度内发生。

定义 6 过匹配与欠匹配。在基于规则的预测过程中，若前件匹配次数超过实际有效发生次数的匹配，称为过匹配；若前件匹配次数少于实际有效发生次数的匹配，称为欠匹配。

定义 7 预测偏离度。预测偏离度是指后件在预测子区间内实际发生频率和预测概率之间的偏差程度。$realOccr$ 记为后件在某子区间的实际发生频率，$probOccr$ 记为后件在某子区间的预测概率，将 $\frac{1}{n}\sum_{i=1}^{n}\frac{|realOccr[i]-probOccr[i]|}{realOccr[i]}$ 定义为预测偏离度，其中，n 表示预测区间的子区间数。

4　SPAOT 算法

SPAOT 算法的主要数据结构是前件发生树(AOT, antecedent occurrence tree)。根节点为空，在第 0 层；其余每个节点代表事件，从根节点到各子节点的路径代表情节规则的前件。每个节点需维护以下信息：事件类型 E，前一事件指针以及后一事件指针。同时，第 1 层节点还包括前件所属组 *group*。每组维护一个公共部分索引表(CPT, common part table)，并指向相关节点。同时，在各前件的结束位置还需保留其对应的后件(组)预测信息。当使用 AOT 结构进行前件发生匹配时，还需为每个节点维护发生时刻 *Time* 队列。

SPAOT 算法将数据流预测分为 3 个阶段：第 1 阶段为构建 AOT 树，即根据情节规则集前件的合并情况构建 AOT，得到精简的待匹配前件集；第 2 阶段为搜索前件发生，按序扫描数据流上的事件，填充 AOT 各节点的 *Time* 队列，依据区间最小非重叠发生定义，逆序扫描 *Time* 队列搜索前件，通过各规则前件间的匹配信息共享提高搜索效率；第 3 阶段为后件预测，考虑前件多次发生对预测的叠加作用，计算后件在未来的发生时刻和发生概率。如图 1 所示。

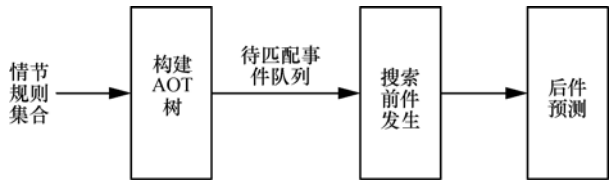

图 1　SPAOT 算法总体流程

4.1　构建 AOT 树

传统方法往往以待匹配事件队列为中心，随着数据流长度增长，所需存储空间也在增长。SPAOT 算法以待匹配规则为核心，有效避免算法存储空间随数据流变化而增长的问题。

AOT 树型结构是在充分考虑规则前件关系的基础上建立的，前件间关系可以分为以下 3 种。

1) 前件完全相同（Same）。有 2 个情节规则α、β，其前件可分别表示为 $\alpha.l$=<E_1,E_2, … ,E_k>、$\beta.l$=<E'_1,E'_2,…,E'_k>，若 E_i=E'_i($1\leqslant i\leqslant k$)，则α和β是前件完全相同的规则。例如，2 个情节规则 α=(<AB>,<D>,5,80%,10)，β=(<AB>,<E>,2,90%,9)。

2) 前件部分相同（PartSame）。可以细分为以下情况：2 个情节规则α、β，其前件可分别表示为 $\alpha.l$=<E_1,E_2 … E_p>、$\beta.l$=<E'_1,E'_2 … E'_k>，若 E_i=E'_i(i=1…n, n<min(k,p))，则α和β是前件左部分相同的规则；与左部分相同定义类似的右部分相同；若 E_i=E'_i($m\leqslant i\leqslant n$, m,n<min(k,p))，则α和β是前件中间部分相同的规则；若 E_i=E'_i($1\leqslant i\leqslant$min(k,q))即 $\alpha.l\subset\beta.l$ 或 $\beta.l\subset\beta.l$，则α和β是前件互相包含的规则。例如，前件左部分相同的 2 个情节规则 α=(<ADC>,<D>,5,80%,10)，β=(<ADF>,<E>,2, 90%,9)。

3) 除此之外，就是完全不同的前件类型。

根据规则前件集的分类情况对前件进行合并，得到精简的待匹配规则前件集，并依次加入 AOT，最终为后续的高效匹配打下基础。AOT 的构建过程如算法 1 所示，步骤如下。

Step1　(第 1)行)，合并规则前件集。因为完全相同和完全不同的均只留下前件的单例，所以将前件集 A 中存在完全不同和完全相同的前件合并到集合 *Sm*；部分相同的前件放入集合 *PSm*。当然也存在完全相同的前件和其他前件间存在部分相同的情况，均衡考虑数据结构的复杂性，这种情况不予处理。

Step2　(第 2)~11)行)，构建 AOT。首先，使用广义后缀树(GST, generalized suffix tree)算法得到 *PSm* 中前件的公共部分放入 *ComPart*，依据公共部分 *cp* 不同进行分组，组号为 *gn*；其次，根据所有组的前件构建 AOT 分支，建立公共部分索引表 *CPT*，将其连接到 AOT 的对应节点。此时，各前件均已记录在 AOT 中，并在各前件的结束位置保留了其对应的后件(组)预测信息。

算法 1　*buildAOT*(Set A)

输入　规则前件集 A

1) 根据定义将前件集 A 分配给 *Sm* 和 *PSm*

2) for each　$l\in Sm$

3) *addNodes* (*root*,*l*,0)//在 *root* 下增加分支 *l*，分组为 0

4) for each　$l\in PSm$

5) 使用 GST 得到 *ComPart*set

6) for each　$cp\in ComPart$

7) for each　$l\in PSm$

8)　　*addNodes* (*root*, *l*, *gn*)

9) 令 $gn=gn+1$

10) 建立组 *gn* 对应的 *CPT*

11) return

例 1 构建 AOT。设给定的数据流为 DS，*min_sup*=2，现用算法 SPAOT 对表 1 建立 AOT。

表 1 预测情节规则

编号	前件	后件	支持度	置信度	窗口宽度
1	<AAB>	<DE>	2	0.8	8
2	<AAB>	<C>	4	0.8	9
3	<AC>	<F>	4	0.8	9
4	<BC>	<D>	2	0.6	9
5	<CDB>	<E>	3	0.6	8
6	<CDF>	<A>	5	1.0	7
7	<DBC>	<AE>	2	1.0	6

表 1 中，前 2 个规则是前件完全相同的规则，第 3 个规则是完全不相同规则，*group* 置为 0；第 5、第 6 个规则是前件左相同，公共部分为 CD，*group* 置为 1；第 4、第 7 个规则满足前件包含关系，公共部分为 BC，*group* 置为 2。构建结果如图 2 所示。

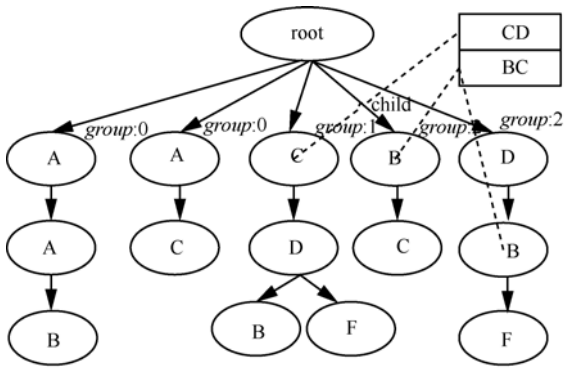

图 2 表 1 对应的前件发生树

4.2 搜索前件发生

SPAOT 算法的关键是如何查找前件的区间最小非重叠发生。在 AOT 中，对每分支自底向上遍历节点的 *Time* 队列，将 *Time* 队列逆序搜索直至发生时刻小于前继节点的发生时刻，在满足区间要求的情况下找到所有最小非重叠发生。搜索过程中，依据 *CPT*，先搜索前件公共部分，再拓展到其他各前件情况。这样做避免了搜索时重复状态造成的冗余。区间最小非重叠发生则保证了所有的前件发生对预测都有意义，解决了过匹配和欠匹配问题。搜索过程如算法 2 所示，步骤如下。

Step1 (第 1)~9）行)，对 *Sm* 集的前件，直接搜索各节点发生时刻，填充 *Time* 队列。使用 *findOccurrence* 函数搜索发生。搜索结束，将前件发生放入集合 *intvlMinOccr* 并输出。

Step2 (第 10)~23）行)，对 *PSm* 集的前件，首先搜索 *CPT* 中的前件公共部分的发生时刻，将其拷贝到 AOT 各分支的对应节点，再向前或向后搜索分支的其他节点。最后，按照第一步搜索方式搜索区间最小非重叠发生，存储并输出 *intvlMinOccr* 集合。

对于 *findOccurrence* 函数，其步骤如下。

Step1 (第 1)~4)行)，对前件长度为 1 的情况，则认为搜索区间内的每次出现都是一次有效发生。

Step2 (第 5)~17)行)，其余情况，利用各节点 *Time* 队列的前后关联查找区间内的有效发生。

算法 2 *findOccurrenceOfAOT*(Node* *root*, Set *DS*, Interval *intvl*).

输入 根节点 *root*, 数据流 *DS*, *intvl* 表示搜索区间

输出 前件区间最小非重叠发生

```
1) Set intvlMinOccr
2) for each branch of AOT with group=0
3)     令 node=root
4)     while node->child!=NULL
5)         范围内使用 E 填充 node 的 Time
6)         令 node = node -> child
7)      intvlMinOccr=findOccurrence(node,intvl)
8)     输出 intvlMinOccr
9)     删除 Time 队列
10)    for each branch of AOT with group!=0
11)    while node->child!=NULL
12)     if CPT 中的公共部分被填充
13)         把该节点的 Time 值，赋给它的分支
14)         向前或向后搜索分支的其他节点
15)     else
16)         填充 CPT 中的公共部分
17)         把该节点的 Time 值赋给它的分支
18)         向前或向后搜索分支的其他节点
19)     令 node = node -> child
20)     intvlMinOccr=findOccurrence(node,intvl)
21)    输出 intvlMinOccr
22)    删除 Time 队列
23) return
```

算法 3 *findOccurrence* (Node **end*, Interval *intvl*)

输入 *end* 为分支末节点，*intvl* 为搜索区间 [*intvl.s*,*intvl.e*]

输出 前件区间最小非重叠发生集

```
1) 定义 occrSet//记录区间最小非重叠发生集
```

2) if *end ->father=root*

3)　将 *intvl* 区间内的 *end.Time* 加入 *occrSet*

4)　return *occrSet*

5) else while *end ->father*!=*root*

6) for *i=tail* to 0//从当前节点 *Time* 队尾至队首

7)　令 *begin=end.Time*(*i*)//*begin* 为发生开始时刻

8)　*flag=true*//标记是否被用于之前发生

9)　if *end.Time*(*i*)> father nodes' *Time*(*i*)

10)　令 *stop*= father nodes'*Time*(*i*)//*stop* 为发生结束时刻

11)　else

12)　for *j=i*to 0//从当前时刻至队首

13)　if *end.Time*(*i*)> *end ->father.Time*(*j*)

14)　令 *stop= end ->father.Time*(*i*)

15)　if *begin* > *stop* &&*flag=true*&& *begin*> *intvl.s*&& *stop*<*intvl.e*

16)　将[*begin*, *stop*] 集加入 *occrSet*

17) return *occrSet*

例 2　基于 AOT 搜索前件分支。对于数据流 *DS*'=<(A,1), (B,2), (D,3), (C,4), (A,5), (C,6), (B,7), (E,8), (C,9), (B,10), (F,11)>。通过图 3 的 AOT 结构，可以发现 AOT 右边 2 个分支存在着前件包含关系，故对于 DBC 前件来说，可以共享 BC 前件中的 *Time* 队列，缩短前件的匹配时间。按照 AOT 从底向上，逆向搜索各节点的 *Time* 队列，可以得到 BC 的区间最小非重叠发生集是{[2,4],[7,9]}，DBC 的区间最小非重叠发生集是{[3,9]}。

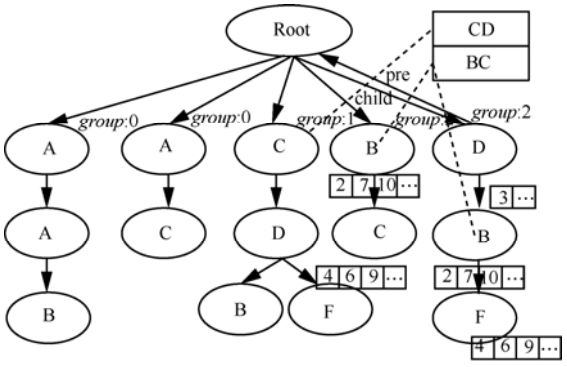

图 3　AOT 前件搜索范例

4.3　后件预测

在 SPAOT 算法的第 2 个阶段搜索了前件的多次区间最小非重叠发生，但如何有效利用还有待解决。现有的预测方法均认为后件的发生服从均匀分布，这在区间内前件单一有效发生情况下是适用的；而当前件有多次有效发生情况下预测效果较差。统计发现当前件多次发生造成预测区间重叠时，后件的发生并不遵循均匀分布，而是如图 4 所示的阶梯状，越早的预测区间发生概率越大。

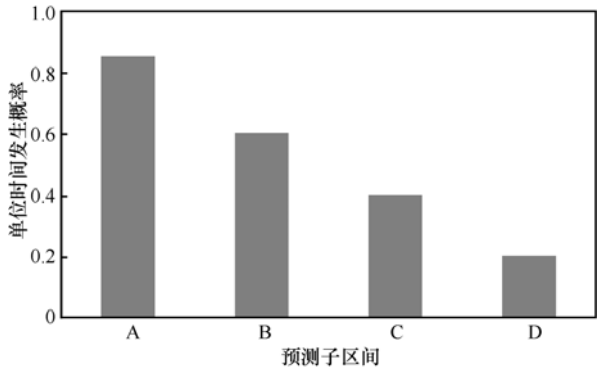

图 4　前件多次有效发生对应的后件实际发生统计

据此，在预测阶段，SPAOT 算法需考虑前件的多次区间最小非重叠发生对预测的累计作用，进行叠加预测。预测过程如算法 4 所示，步骤如下。

Step1　(第 1)~10)行)，为了保证前件发生集 *intvlMinOccr* 各发生均对预测有意义，需判断搜索区间内是否出现规则后件：若没有后件发生，则在规则窗口宽度 $\gamma \cdot \omega$ 限制下，计算前件各发生对应的完整后件预测区间，用 *I*[*i*]表示第 *i* 个预测区间；若后件的一部分发生，则计算前件各发生对应的剩余部分后件的预测区间；若后件完整发生，无论随后是否存在后件的其他完整或不完整发生，都应立即停止后件的匹配，因为后件的一次完整发生意味着前件的当前发生对预测不会产生任何意义。

Step2　(第 11)~15)行)，考虑前件发生对后件预测的累计作用，将区间进一步划分，用 *subI*[*i*]表示子区间，计算各预测子区间的发生概率 $P_{subi[i]}$，并输出。

$$P_{subi[i]} = \frac{1}{n}\sum_{j=1}^{n} c\frac{length(subI[i])}{length(I[i])} \tag{1}$$

根据预测阶段伪代码易知，当没有预测重叠区间时，其退化为一般的概率发生预测算法。

算 法 4　*prediction*(Rule *γ*,Set *intvlMinOccr*, int*stop*)

输入　预测规则*γ*，*intvlMinOccr* 为前件有效发生集，*stop* 为搜索区间结束时刻

输出　发生区间、发生概率

1) 令 *i*=0//*i* 为标记预测区间

2) for each *Occ*∈*intvlMinOccr*

3) for *Occ.e* to*stop*// *Occ.e* 表示发生结束时刻

4) if *γ.r* not occurs

5) 令 *I*[*i*].*s*=*stop*+1//*Pr*[*i*].*s* 为 *i* 预测区间开始时刻

6) 令 *I*[*i*].*e*= $\gamma\cdot\omega$ −length(*stop*−*Occ.e*)+ *stop* //*Pr*[*i*].*e* 为 *i* 预测区间结束时刻

7) else if part of *γ.r*occurs//*γ.r* 部分发生

8) 令 *I* [*i*].*s*= part of *γ.r*'s ending time+1

9) 令 *I* [*i*].*e*= $\gamma\cdot\omega$ −length(*stop*-*Occ.e*)+ *stop*

10) 令 *i*=*i*+1

11) for each*poccr*[*j*]∈*I*

12) 令 *subI*[*j*].*s* =*poccr* [*j*−1].e+1

13) 令 *subI*[*j*].*e* =*poccr* [*j*].*e*

14) 计算 $P_{subi[j]}$ of *subI* [*j*]

15) 输出 *subI*[*j*], $P_{subi[j]}$

例 3 概率叠加预测。对于例 2 的 *DS*'来说，通过算法 2 得到规则<BC,D,2,60%,9>前件的 2 个区间最小非重叠发生[2,4]、[7,9]。发生[2,4]的预测[5,10]没有后件 D 发生，因此，D 在区间[5,10]的预测概率为 60%；同样的，发生[7,9]在[10,15]上的预测 D 发生概率为 60%。此时预测区间被分为 2 个部分：[11,15]与重叠预测区间[10]。根据概率叠加思想，每个子区间的发生概率为

$$P_{[11,15]}=\frac{0+\frac{5\times0.6}{6}}{2}=0.25$$

$$P_{[10]}=\frac{0.6+\frac{1\times0.6}{6}}{2}=0.35$$

而按照传统的预测方法得到的 2 个预测子区间的发生概率为

$$P'_{[11,15]}=\frac{5\times0.6}{6}=0.5$$

$$P'_{[10]}=\frac{1\times0.6}{6}=0.1$$

从 SPAOT 算法的处理过程不难发现：在构建 AOT 阶段，充分利用规则间已有的关联关系建立数据结构，节省存储空间；在搜索前件发生时，通过区间最小非重叠发生的判断，避免了前件的过匹配与欠匹配，而且对 AOT 的前件公共部分，只进行了一次搜索，提高匹配精度和效率；在后件预测过程中，充分考虑了发生的重叠区间对预测的叠加影响，使预测精度进一步提高。

5 算法分析

设 *R* 为给定的情节规则集，*DS* 为给定的数据流，*m* 为前件的平均长度，*l* 是搜索区间长度，*r* 为合并系数即合并后的规则个数与原来规则个数的比值，$0<r\leqslant1$。则 SPAOT 算法的正确性与复杂度分析如下。

SPAOT 算法是正确的。算法在构建 AOT 阶段，利用规则间已有的关联关系精简树结构，得到的 AOT 仍包含规则的全集；在搜索前件发生时，利用区间最小非重叠发生和 AOT 的前件公共部分提高匹配精度和效率，并未遗漏任何有效的前件匹配；在后件预测过程中，充分考虑发生的重叠区间对预测的叠加影响，给出了更加精确的预测。整个过程保证了所有的合法待匹配数据与所有的规则前件进行了匹配，并给出了精确的预测结果，因而是正确的。

时间复杂度。SPAOT 算法由构建 AOT 结构、搜索前件发生、后件预测 3 个部分组成。当规则集 *R* 给定后，构建 AOT 结构只需要进行一次，因此，时间复杂度记为常数；后件预测虽然在预测过程中全程参与但是其复杂度小于搜索前件发生的复杂度，因此，本文主要分析搜索前件发生的时间复杂度。按照规则合并策略减少需搜索的规则前件，即需要在 *DS* 上匹配 *r*|*R*|个规则前件，此时，算法的时间复杂度为 *O*(*r*|*R*||*DS*|)。

空间复杂度。AOT 中节点的数量是固定的，即 *R* 中各规则合并后的前件长度之和 *r*|*R*|*m*。由于只对 AOT 的每一分支建立 *Time* 队列，最坏情况是分支的所有节点需记录所有时刻，即 *Time* 队列长度为 *l*，则算法的空间复杂度是 O(*r*|*R*|*m*+*l*)。

表 2 为 SPAOT 算法与各经典预测算法时空复杂度比较。DeMO 算法、Predictor 算法的时间复杂度都是 O(|*R*||*DS*|)。CBS-Tree 算法的时间复杂度是 *O*(|*R*||*DS*|$S_t$log*L*)，其中，S_t 为匹配一个事件所需时间，*L* 为 CBS-Tree 的叶子节点数，CBS-Tree 算法匹配事件时至多需搜索 2 条路径，故其搜索时间为 log*L*。由于 *L* 代表 CBS-Tree 的叶子节点数，肯定大于 2，即 log*L*>1 且 $S_t$log*L*>1，故 CBS-Tree 算法的时间复杂度大于 DeMO 和 Predictor 算法。SPAOT 算法时间复杂度小于这 3 个算法。

表 2 中，DeMO 算法、CBS-Tree 和 Predictor 算法的空间复杂度均是 *O*(|*R*|*m*+|*DS*|)。由于 SPAOT 按照规则合并策略将规则合并后构建 AOT，故 AOT

中节点个数为$|R|m \leqslant |R|m$；只需要在预测区间内进行搜索，$l \leqslant |DS|$。因此 SPAOT 算法的空间复杂度优于这 3 个算法。

表 2　SPAOT 算法与各经典预测算法时空复杂度比较

算法	时间复杂度	空间复杂度
DeMO 算法	O(|R||DS|)	O(|R|m+|DS|)
CBS-Tree 算法	O(|R||DS|$S_t$log$L$)	O(|R|m+|DS|)
Predictor 算法	O(|R||DS|)	O(|R|m+|DS|)
SPAOT 算法	O(r|R||DS|)	O(r|R|m+l)

6　实验

通过 4 组实验对比了 SPAOT 与 DeMO、CBS-Tree、Predictor、EGH[1]算法的预测精度和时空性能。实验采用的硬件环境为 2.65 GHz Intel(R) Core(TM) i5 CPU，内存 6 GB，操作系统为 Windows 7，程序采用 Java 实现。

6.1　实验数据集

实验采用 2 种类型的数据集。

1) 道路交通数据集。从美国交通管理中心 TMC（traffic management center）在双城地铁（twin cities metro freeways）网络的传感器上收集的真实数据，选取了 2013 年 1 月 1 日至 2014 年 1 月 1 日的数据组成事件序列。每个记录包括发生时刻、车辆速度和道路占有率 3 个属性，这些记录按照发生时刻排序。每个事件可以看成是车辆速度和道路占有率的结合，表示此时刻道路的拥挤程度，得到的事件序列包含 303 个不同事件和 1 503 680 个记录。由于某时刻道路的拥挤情况可以根据车辆速度和道路占有率结合得到的车辆流动速度来度量，且不同时刻的道路间拥挤情况之间存在着相互影响，研究道路之间的影响关系可以为交通及公交运行状况的分析以及线路选择等提供支持。

2) 新闻事件数据集。来源于匈牙利新闻门户网站，选取了 2014 年 3 月 1 日至 2014 年 5 月 1 日的鼠标点击事件流。该事件流包含 3 340 个不同事件和 990 000 次新闻事件序列。由于新闻事件之间是相互影响的，因此，可以分析出不同的新闻事件之间的因果关系，以此发现新闻事件之间存在的潜在规律，能够为新闻门户网站的更加合理化的设计等提供支持。

本文从 1~6 月的道路交通数据集（支持度和置信度分别为 7 和 60%）和 3 月的新闻事件数据集（支持度和置信度分别为 8 和 60%）上抽取了 398 个道路交通情节规则和 954 个新闻事件情节规则作为待匹配的情节规则，并将剩余数据集作为验证数据集。按照 SPAOT 算法构建 AOT 时，道路交通情节规则间的事件共享率为 37.6%、新闻事件为 41.3%。

6.2　实验结果

实验 1　各算法预测精度比较。为了考察算法在数据流上的预测精度。实验时采用交叉验证方法，从 2 种数据集中选取特定的事件序列作为当前数据流，而原数据集上后面的剩余序列将作为判断算法预测是否正确的验证序列。

根据定义 7，在 2 个实验数据集上，得到了 DeMO、CBS-Tree、Predictor、EGH、SPAOT 平均预测偏离度的比较。从图 5 可以看出，DeMO、CBS-Tree、Predictor 算法由于发生定义不同而导致前件查找略有不同，因而预测精度也不尽相同；但是均未考虑多次有效发生对预测的累计作用，造成了偏离度高的问题。EGH 算法预测的是未来数据流上一个时间点的发生情况，HMM 过程在预测中不自觉地起到了一定的概率累计作用，因此，平均偏离度较前 3 个算法低。SPAOT 算法充分考虑了前件的各有效发生对后件的影响，所以平均偏离度最低。

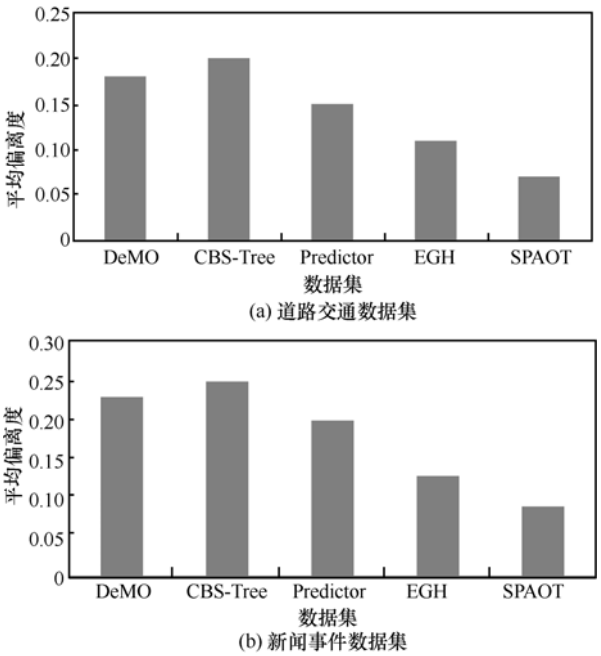

图 5　预测偏离情况比较

另外，各算法在道路交通数据集上的平均偏离程度都要比在新闻事件数据集上低，这是因为道路交通数据集的数据间因果性比新闻事件数据集更强。

EGH 算法所采用 HMM 模型，中间状态计算量较大，代价较高，与其他基于规则匹配的算法不具有可比性，故将不参与时空性能方面的实验比较。

实验 2 运行时间与数据流长度的关系。通过改变数据流长度（道路交通数据从 7~12 月，新闻事件点击数据从 40~60 天）得到图 6 所示的 4 个算法在 2 个数据集上的运行时间结果。从图 6 可以看出，Predictor 优于 DeMO 和 CBS-Tree。这是由于 DeMO 通过查找待匹配规则前件最近的最小发生来预测后件，一方面，在当前数据流上可能存在规则前件许多重叠的最小发生，因而导致了对该情节的过于匹配；另一方面，只要存在规则前件最近的最小发生，DeMO 就给出了一个预测结果，而这个预测结果可能无效。CBS-Tree 虽然考虑了区间因素，但由于其采用的不是以规则为核心的数据结构，且采用的也不是非重叠最小发生。另外，DeMO 与 CBS-Tree 均未考虑查找多个前件公共项的优化，因此效率较低。Predictor 是根据待匹配规则前件最近的最小非重叠发生（避免了对规则前件的过于匹配）进行预测，且进行了查找多前件发生的优化工作，因此较 DeMO、CBS-Tree 效率高。

SPAOT 算法要优于其他 3 个算法。这是由于 SPAOT 算法通过前件发生树，搜索到规则合并后的所有前件发生，比其他 3 个算法要更节省搜索前件发生的时间。随着数据流长度的增加，4 个算法的运行时间都在增加，SPAOT 算法的增长率低于其他算法，这是由于 SPAOT 算法考虑情节规则的窗口宽度，只搜索在有效区间内的发生，因此受数据流长度的影响较小。

实验 3 运行时间与规则个数的关系。首先，分别从 398 个道路交通情节规则和 954 个新闻事件情节规则中随机地选择各自的子集作为待匹配的情节规则，从而得到如图 7 所示的运行时间和规则个数的比较结果。从图 7 可以看出，SPAOT 算法所花时间要少于其他算法（主要原因同实验 2），且随着规则个数增长优势明显。这是由于 SPAOT 算法基于规则合并策略，建立前件发生树，搜索所有前件的发生，规则越多，可合并的规则也会越多，故随着规则个数的增长，SPAOT 算法的时间增长率也较低。

实验 4 内存开销与数据流长度的关系。通过改变数据流长度（道路交通数据累计测试 6 个月（从 7~12 月），新闻事件点击数据从 40~60 天）得到如图 8 所示的内存开销结果（单位 KB）。从图中可以看出，CBS_Tree 与 DeMO 较 Predictor 花费内存大，这是由于 2 个算法查找的前件发生集比 Predictor 大，而 DeMO 只记录前件发生集的始末时刻，因此较 CBS_Tree 略小。但这 3 个算法都在数据流上搜索前件发生，故随着数据流长度的增加，其内存开销也会增加。由于 SPAOT 算法基于规则合并策略，同时该算法只维护一个 AOT 分支节点的 Time 队列，而且 SPAOT 搜索的是有效区间内的发生，其内存开销最优且几乎不受数据流长度的影响。

另外，内存开销与规则个数的关系也有类似实验结果，这里不再赘述。

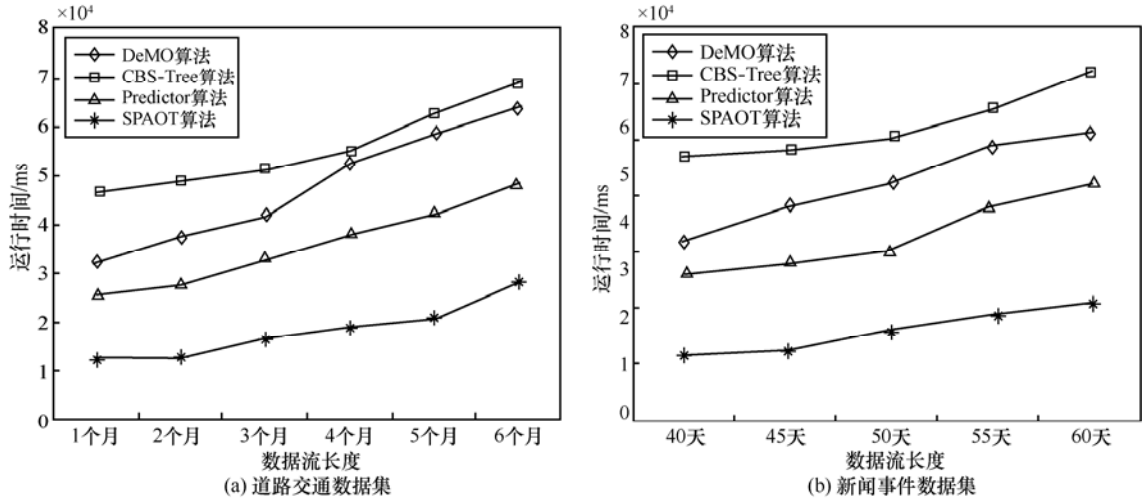

图 6 运行时间与数据流长度的关系

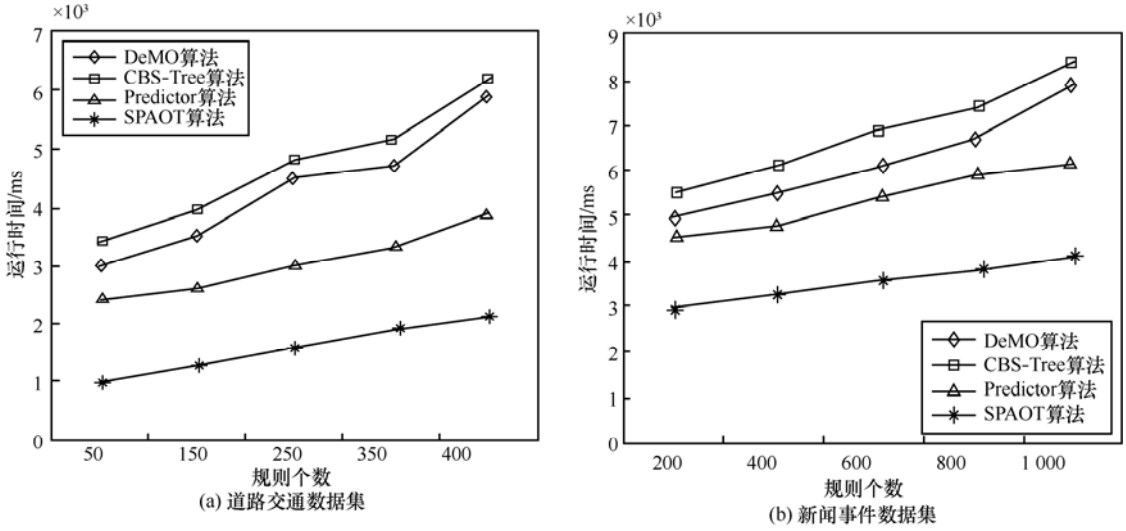

图 7　运行时间与规则个数的关系

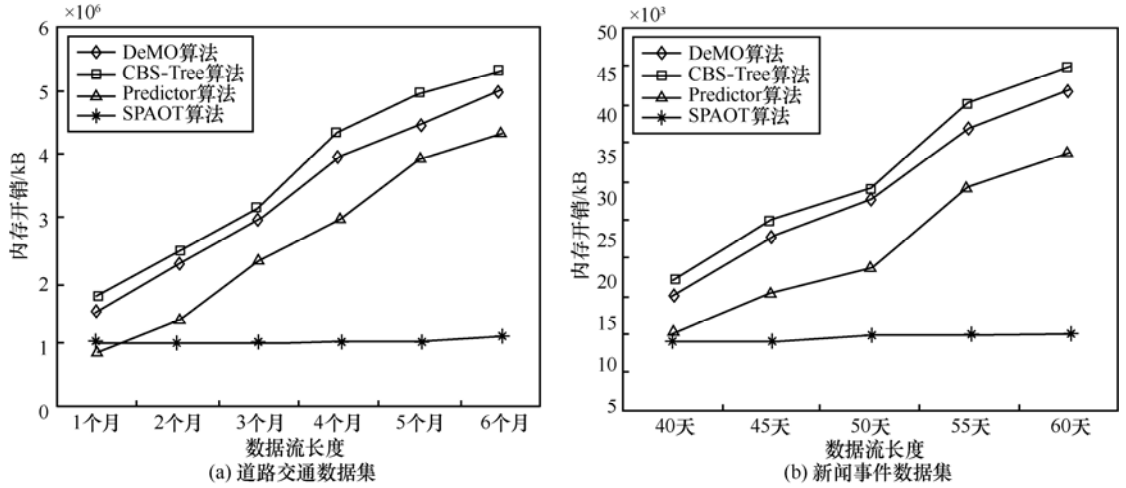

图 8 内存开销与数据流长度的关系

7　结束语

针对现有数据流预测算法效率较低、预测精度不高等问题，本文提出基于前件发生树的概率叠加预测算法。定义区间最小非重叠发生，保证搜索到的前件发生均是对预测有效的发生；按照规则合并策略合并情节规则前件，避免在数据流上所有规则前件的匹配；考虑发生的重叠区间对预测的叠加影响，基于概率叠加的思想，精确计算后件的发生区间和发生概率。理论分析和实验评估证明算法 SPAOT 能够有效地基于情节规则匹配进行数据流预测。

参考文献：

[1] LAXMAN S, TANKASALI V, WHITE R W. Stream prediction using a generative model based on frequent episodes in event sequences[C]//The 14th ACM SIGKDD International Conference on Knowledge Discovery and data mining. New York, US, 2008: 453-461.

[2] KHAN M S, COENEN F, REID D, et al. A sliding windows base dual support framework for discovering emerging trends from temporal data[J]. Knowledge-Based Systems, 2010, 23(4):316-322.

[3] CHO C W, ZHENG Y, WU Y H, et al. A tree-based approach for event prediction using episode rules over event streams[C]//Database and Expert Systems Applications. Berlin, Germany. 2008: 225-240.

[4] CHAVENT M, GIRARD S, KUENTZ-SIMONET V, et al. A sliced inverse regression approach for data stream[J]. Computational Statistics, 2013: 1-24.

[5] MONTGOMERY D C, PECK E A, VINING G G. Introduction to linear regression analysis[M]. Manhattan: John Wiley & Sons, 2012.

[6] CHO C W, ZHENG Y, CHEN A L P. Continuously matching episode

rules for predicting future events over event streams[M]//Advances in Data and Web Management.Berlin: Springer, 2007.

[7] CHO C W, WU Y H, YEN S J, et al. On-line rule matching for event prediction[J]. The VLDB Journal, 2011, 20(3): 303-334.

[8] YANG Q, LI T Y, WANG K. Building association-rule based sequential classifier for Web-document prediction[J]. Data Mining and Knowledge Discovery,2004,8(3):253-273.

[9] MANNILA H, TOIVONEN H. Discovering generalized episodes using minimal occurrences[C]//The Second International Conference on Knowledge Discovery and Data Mining. Portland, USA,1996: 146-151.

[10] LAXMAN S, SASTRY P S, UNNIKRISHNAN K P. Discovering frequent episodes and learning hidden Markov models: a formal connection[J]. IEEE Transactions on Knowledge and Data Engineering,2005,17(11):1505-1517.

[11] ZHU H, WANG P, HE X, et al. Efficient episode mining with minimal and non-overlapping occurrences[C]// 2010 IEEE 10th International Conference on. Sydney, Australia, 2010: 1211-1216.

[12] ZHOU W, LIU H, CHENG H. Mining closed episodes from event sequence efficiently[C]//The 14th Pacific-Asia Conf on Advances in Knowledge Discovery and Data Mining. Berlin, Germany,2010: 310-318

[13] WAKABAYASHI K, MIURA T. Data stream prediction using incremental hidden Markov models[M]. Springer Berlin Heidelberg, 2009.

[14] 朱辉生, 汪卫, 施伯乐. 基于情节规则匹配的数据流预测[J]. 软件学报, 2012, 23(5), 1183-1194.
ZHU H S, WANG W, SHI B L. Data stream prediction based on episode rule matching[J]. Journal of Software, 2012, 23(5), 1183-1194.

[15] GYLLSTROM D, AGRAWAL J, DIAO Y, et al. On supporting Kleene closure over event streams[C]//The IEEE International Conference on Data Engineering. Cancun, Mexico, 2008: 1391-1393.

[16] GIANNOZZI P, BARONI S, BONINI N, et al. Quantum ESPRESSO: a modular and open-source software project for quantum simulations of materials[J]. Journal of Physics: Condensed Matter, 2009, 21(39): 395-502.

[17] MEI Y, MADDEN S. Zstream: a cost-based query processor for adaptively detecting composite events[C]//The 2009 ACM SIGMOD International Conference on Management of data. Providence, USA, 2009: 193-206.

作者简介：

尤涛（1983-），男，河南陕县人，博士，西北工业大学副教授，主要研究方向为分布式数据处理。

李廷峰（1992-），男，黑龙江鸡西人，西北工业大硕士生，主要研究方向为数据库与数据挖掘。

杜承烈（1970-），男，陕西西安人，博士，西北工业大学教授，主要研究方向为军用软件工程。

钟冬（1979-），男，陕西西安人，博士，西北工业大学副教授，主要研究方向为计算机网络技术。

朱怡安（1961-），男，陕西西安人，博士，西北工业大学教授，主要研究方向为并行计算。