基于Markov链的Web访问序列挖掘算法研究及性能分析

肖 哲[1,2]，任胜兵[1]

（1.中南大学 信息科学与工程学院，湖南 长沙 410083；2.湖南工业大学 科技学院，湖南 株洲 412008）

摘 要：引入正向、逆向Markov一步状态转移概率矩阵构造序列数据库，并将逐层投影的PrefixSpan序列挖掘算法改为伪投影和隔层投影算法结合，以改进经典序列算法中存在的时间或空间开销太大的缺陷。性能分析表明，与经典算法相比，这种基于Markov链的Web访问序列模式挖掘新算法能够通过较少的计算量和空间复杂度获得较优的Web访问序列模式。

关键词：数据挖掘；Web访问；Markov链；序列模式

中图分类号：TP311.13 **文献标识码**：A **文章编号**：1673-9833(2008)03-0053-04

Research and Performance Analysis on Web Accessing Sequence Data Mining Algorithm Based on Markov's Chain

Xiao Zhe[1,2]，Ren shengbing[1]

(1. School of Information Science and Engineering，Central South University，Changsha 412083, China；
2. College of Science and Technology，Hunan University of Technology，Zhuzhou Hunan 412008, China)

Abstract：The transition probability matrix to building sequence database of forward and backward Markov one-step state is introduced. Using pseudo projection combined with bi-level projection algorithm instead of level-by-level projection, it can improve classical sequence mining algorithms that usually have much time and space complexity properties. Performance shows that the proposed one can decrease time and space overhead while searching for Web accessing sequence pattern when compared with classical algorithm.

Key words：data mining；Web accessing；Markov's chain；sequence pattern

0 引言

当前网站的发展速度飞快，网站管理者如何更好地优化网站成为竞争的关键。随着网站内容的丰富和用户数量的增加，Web日志的规模正呈几何级数递增。经典的序列模式挖掘算法（如Apriori all算法[1,2]、GSP[3]算法、FreeSpan[4]算法）虽然计算的准确度高，但需要多遍扫描数据库且会产生数量庞大的候选序列，因此存在计算量高、耗时长、所需存储空间大的问题。这导致经典算法在效率和及时性上都达不到要求。

目前，对于Web访问序列模式挖掘的研究大多集中在对挖掘算法本身的改进上。本文提出了一种基于Markov链和改进PrefixSpan序列模式挖掘算法的新算法。该算法通过Markov的一步正向、逆向转移概率矩阵来构造序列数据库，并通过隔层投影和伪投影相结合的PrefixSpan算法对此数据库进行挖掘，从而以较小的时间和空间复杂度获得较为准确的结果。

收稿日期：2008-03-24

作者简介：肖 哲（1977-），女，湖南澧县人，湖南工业大学讲师，中南大学硕士生，主要研究方向为人机交互；
任胜兵（1969-），男，湖南岳阳人，中南大学副教授，博士，硕士生导师，主要从事软件演化，软硬件协同设计研究.

1 Web日志的预处理

我们在进行Web访问序列挖掘前，首先要对所有的Web访问日志进行筛选。因为这些日志记录并不是都对研究用户行为特征有用，所以需要将Web日志中的有用部分提取出来。

1.1 Web访问日志数据的选择和预处理

原始日志记录并不适于挖掘，需要剔出无用信息和将信息进行必要的整理，其处理步骤通常包括：

1）将日志文件中后缀为gif、jpg、cgi、js等的记录删除；

2）删除访问失败的记录；

3）删除访问时间过短或过长的用户记录，时间的长短可通过设定一个门限值来判断（本文中将过短的时间阈值设为10 s，过长的时间阈值设为1 h）；

4）区分不同的用户：不同的IP地址或IP地址相同但操作系统不同，则认为是不同的用户。

1.2 会话识别

当用户的页面请求在时间上跨度较大时，用户可能多次访问同一个网站，会话识别的目的就是将用户的访问记录分为单个会话。较常用的方法是时间阈值法，通过设定一个时间阈值，若两次请求的时间间隔超过了规定的阈值，则将其当作是两个会话来处理。

1.3 浏览顺序的处理

为提取有用的用户浏览顺序的数据，将浏览顺序数据序列做一定的精简，即：若用户对页面的访问顺序是：$A->B->A->C$，那么可以将用户的行为视为：$A->B->C$。

2 基于Markov链的Web访问分析

Web网站可看作是若干页面URL的集合，因此，对Web网站访问序列的分析，实际上就归结为对用户访问的所有URL序列的分析，从而找到其中的频繁序列。而Markov链的状态转移概率矩阵，是反映链接与链接之间转换频繁度的最好方式。

下面以一中学网站为例，来对这一过程进行详细说明。该网站共包含网页12个，分别为“首页”、“学校简介”、“领导班子”、“学校新闻”、“高考信息”、“招生信息”、“办学成果”、“名师风采”、“家校直通”、“学生天地”、“雁过留声”和“校友录”。将这12个页面分别命名为A，B，…，L，作为Markov分析的状态集合R，Markov的分析过程如下。

2.1 构造正向、逆向状态转移概率矩阵

初步研究每一个网页到其余所有网页的转移可能性 以“首页”为例，首页包含到其余11个网页的全部链接，因此，从A到其余状态的一步转移概率大于等于0；而“校友录”没有到其他任何网页的链接，因此，从状态L到其余状态的转移概率一定为0。

建立正向一步转移概率矩阵 首先添加一个会话起始状态S，因此状态集共有13个状态，为$R=\{S, A, B, C, \cdots, L\}$。而正向一步转移概率矩阵为13*13阶，如下式所示：

$$\boldsymbol{P}=\left(P_{ij}\right)_{i,j\in R}=\begin{array}{c} \\ S \\ A \\ B \\ \vdots \\ L \end{array}\begin{array}{c} \begin{array}{ccccc} S & A & B & \cdots & L \end{array} \\ \begin{bmatrix} P_{SS} & P_{SA} & P_{SB} & \cdots & P_{SL} \\ P_{AS} & P_{AA} & P_{AB} & \cdots & P_{AL} \\ P_{BS} & P_{BA} & P_{BB} & \cdots & P_{BL} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ P_{LS} & P_{LA} & P_{LB} & \cdots & P_{LL} \end{bmatrix} \end{array}。$$

上式中，定义状态X到状态X（$X\in R$）的转移概率为0；定义状态S到其余12个状态的转移概率为这12个状态本身作为起始访问页面的概率。例如，可通过搜索引擎或收藏夹等途径直接访问“学校新闻”，而不需要通过“首页”进入。经过统计，所有有效日志访问记录中，以“首页”作为起始页面的访问概率为0.27，以“校友录”作为起始页面的访问概率为0.44。

对经过预处理的日志文件进行统计，得到除S状态外两两状态之间的正向状态转移概率。例如，经过对所有的有效访问日志文件进行的分析表明，从“首页”状态A到其余所有状态的转移中，到“学校新闻”状态D的转移次数所占比例最高，概率约为0.445。

建立逆向一步转移概率矩阵 首先添加一个会话结束状态T，此时，状态集共包含13个状态，为$R'=\{T, A, B, C, \cdots, L\}$。逆向一步转移概率矩阵为：

$$\boldsymbol{P}'=\left(P_{ij}\right)_{i,j\in R}=\begin{array}{c} \\ T \\ A \\ B \\ \vdots \\ L \end{array}\begin{array}{c} \begin{array}{ccccc} T & A & B & \cdots & L \end{array} \\ \begin{bmatrix} P_{TT} & P_{TA} & P_{TB} & \cdots & P_{TL} \\ P_{AT} & P_{AA} & P_{AB} & \cdots & P_{AL} \\ P_{BT} & P_{BA} & P_{BB} & \cdots & P_{BL} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ P_{LT} & P_{LA} & P_{LB} & \cdots & P_{LL} \end{bmatrix} \end{array}。$$

上式中，定义状态X到状态X（$X\in R'$）的转移概率为0；定义状态T到其余12个状态的转移概率为以这12个状态结束访问的概率；定义状态X到状态Y（$X, Y\in R'$）的转移概率为在所有状态到Y的转移中，X状态到Y状态转移所占的比例。

例如，经过统计，所有有效日志访问记录中，以“校友录”作为结束页面的概率为0.47，以“高考信息”作为结束页面的概率为0.218。

对经过预处理的日志文件进行统计，得到除T状态外两两状态之间的逆向状态转移概率。例如，A至K的所有状态都有到“校友录”状态L的链接，而在这11个状态到状态L的转移中，状态A到状态L的转移占到了所有转移的9.4 %。

2.2　构造序列数据库

2.2.1　构造正向序列集合

定义 1：正向序列　　根据正向转移概率矩阵 $\boldsymbol{P}$，从初始状态 i 出发，经过 N 步状态转移，到达最终状态 j，每一步状态转移都对应到状态空间中的一个状态，由这些状态按照出现的先后顺序组成的序列就称为正向序列。

如图 1 所示，A 为起始状态，则正向序列为 $\{ABC\}$、$\{ABE\}$ 和 $\{AC\}$。

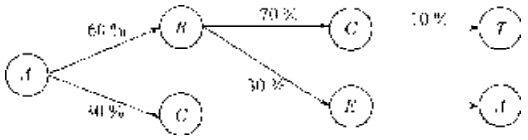

图 1　正向序列示意图

Fig. 1　The diagram of forward sequence

在这里，第二个正向序列的构造基于如下定义：若某一状态在下一步又回到了其父状态，则认为该序列的构造结束。具体的正向序列构造算法如下：

1）构造能够作为起始状态的状态集合 R_s；

2）从 R_s 中选择某一个状态作为起始状态，选择正向转移概率矩阵 $\boldsymbol{P}$ 中概率最大的 s %（这里的 s % 是根据构造需要设定的阈值）个状态作为当前状态的子状态，并以每一个子状态作为当前状态；

3）重复第 2 步操作，直到当前状态在父状态中出现过，那么当前状态的父状态就是最终的状态，本序列构造结束；

4）将已在构造好的序列中出现的所有状态从 R_s 中剔除。若此时 R_s 为空，则序列构造过程结束；否则选择 R_s 中剩余的某个状态作为开始状态，转第 2 步。

正向序列构造完成后，需要对这些序列做筛选，选择算法如下：

1）将序列集合中重复的序列剔除掉；

2）若以 A 页面开始的会话个数只占到整个会话数的 p %，那么以该页面开始的正向序列就可以从序列集合中剔除掉（本文中设 $p = 2$），这样做的目的是将影响小的序列从数据库中删除，以降低下一步筛选时的复杂度。

2.2.2　构造逆向序列数据库

定义 2：逆向序列　　根据逆向转移概率矩阵 $\boldsymbol{P}'$，从最终状态 j 出发，经过 N 步状态转移，到达最初状态 i，每一步状态转移都会对应到状态空间中一个状态，并将上述状态以逆序收集，按照这个方法组成的序列就称为逆向序列。其构造方法与正向序列相同。

2.2.3　剔除正向序列和逆向序列中重复的部分

构造好正向序列集合和逆向序列数据库后，还需将这两个序列中的重复部分剔除。

经过以上 3 步操作后，剩下的 m 个序列构成序列数据库 C，进行序列模式的挖掘。

3　序列模式挖掘

目前较为常用的序列模式挖掘算法为基于逐层投影的 PrefixSpan 算法[5]。与经典的模式挖掘算法相比，该算法虽然大幅度降低了计算复杂度，并缩减了检索空间，但是需要不断地构造投影数据库，这成为该算法的主要开销。

该算法有两个改进算法：隔层投影的 PrefixSpan 算法和伪投影 PrefixSpan 算法。本文为了同时降低时间和空间复杂度，提出了将隔层投影和伪投影算法相结合的办法，这样做能够最大程度地节省存储空间和执行时间。

4　性能比较

下面将以上述中学网站为例，对比 GSP 算法和 PrefixSpan 算法来对本文算法的时间和空间执行效率进行分析。

4.1　时间复杂度比较

图 2 是采用 3 种算法进行序列挖掘所需的执行时间对比图。

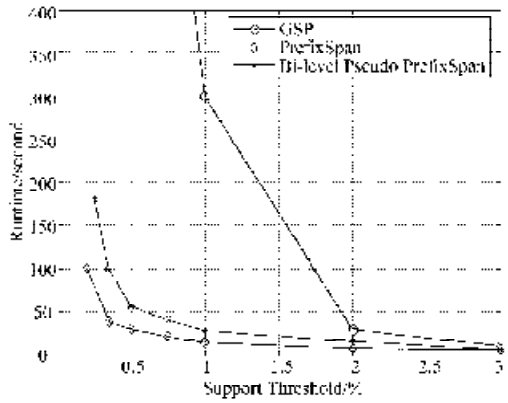

图 2　算法复杂度对比图

Fig. 2　The Comparison Chart of algorithm complexity

从图 2 可以看出，本文提出的算法在时间复杂度方面比 GSP 算法和 PrefixSpan 算法都有很大的减少。这主要是因为：GSP 算法需要对原始数据进行反复多次的扫描来计算支持度，因此需要占用很多的运行时间。而当支持度下降的时候，频繁序列的数量会呈指数上升，这样一来，所花费的扫描时间也会呈指数级上升，因此，相对而言，GSP 所需要的执行时间是相当长的。而 PrefixSpan 由于需要在挖掘时不断地构造投影数据库，而这一过程也需要反复地扫描数据库，这也是比较耗时的。本文提出的算法由于采用了 Markov 理论来构造序列数据库，使待扫描数据库的容量大大降低，同时使用了隔层投影的 PrefixSpan 算法，减少了构造投影数据库的次数，因此，执行时间有了一定程度的降低。

4.2 空间复杂度比较

表1给出了在支持度阈值为2～6的情况下，对经典的GSP算法、PrefixSpan算法及本文提出的算法占用内存的情况进行比较的结果。

表 1 3种算法空间占用情况对比表

Tab. 1 Comparative table of three kinds of algorithm space occupancy

kB

算 法	支持度阈值				
	2	3	4	5	6
GSP	10 016	11 296	11 835	12 956	13 678
PrefixSpan	6 108	7 007	7 426	8 068	8 781
Our Algorithm	2 560	2 892	3 354	3 819	4 221

从表1中可以看出，GSP算法由于需要产生大量的候选序列，因此空间占用最大；PrefixSpan算法由于不需要产生候选序列模式，从而大大减小了存储空间；本文提出的算法由于采用了伪投影的方法，只存储序列指针和偏移，因此能够有效地减少序列挖掘过程中占用的内存。

4.3 准确度分析

准确度比较必须有一个比对标准，本文将GSP算法计算出来的序列模式作为一个基准，其他算法计算出来的序列模式与这个基准对比，以覆盖基准序列模式的范围作为衡量算法准确度的方法。

定义序列模式准确度时，先分别计算待比较算法和GSP算法支持度为 i 的序列模式集，并将计算结果进行比较。设GSP算法的支持度为 i 的序列模式集为 $S(i)$，待比较算法的支持度为 i 的序列模式集为 $S'(i)$，设

$$\eta_i = \frac{S'(i)\text{中属于}S(i)\text{中的序列模式个数}}{S(i)\text{中序列模式总数}},$$

$$\lambda_i = \frac{S'(i)\text{中属于}S(i)\text{中的序列模式个数}}{S'(i)\text{中序列模式总数}},$$

那么准确度的计算公式为：$\phi = \sum_{i=1}^{n} \omega_i \eta_i \lambda_i, r \geq 1$，其中 ω_i 为各个支持度的权重值，ω_i 的计算公式如下：

设 $\sigma_i = \frac{\text{序列模式总个数}}{\text{支持度为 } i \text{ 的序列模式个数}}, i \geq 1$，那么

$\omega_i = \frac{\sigma_i}{\sum_{j=1}^{r} \sigma_j}, i \geq 1$。将本算法与PrefixSpan算法进行准确度比较（其中 r 取6），结果见图3。

图3是根据上述定义给出的准确度计算公式，以经典的GSP算法作为基准的比对，画出的两种算法（PrefixSpan算法和本文提出的算法）的准确度比较图。可以看出，Markov正向、逆向状态转移矩阵的引入，使得扫描数据库的容量大幅度降低，因此带来了一定程度的准确度降低。但是，此结果与PrefixSpan算法相比还是十分接近的。

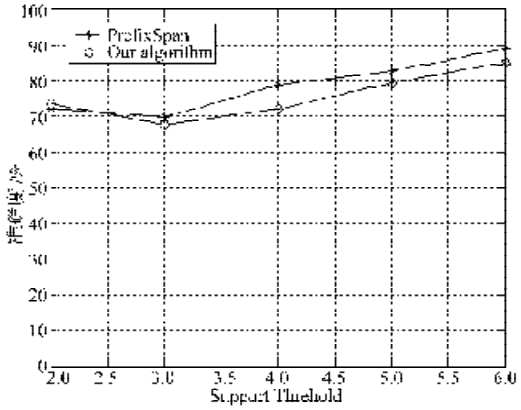

图 3 算法准确度对比图

Fig. 3 The Comparison Chart of algorithm accuracy

5 结语

通过仿真可以看出，本文提出的算法，使用Markov模型构造出了具有代表性的候选序列，采用伪投影和隔层投影相结合的方式来进行数据挖掘，从而能够在保持较高准确度的同时，极大地减少了候选序列的数量，使时间执行效率大为提高，同时使所需空间存储大为降低。在某中学的网站优化中，此算法能够以较高的准确度快速及时地对Web日志进行分析，提取用户的访问行为特征，从而对网站的改进和优化起到了指导作用，使用户访问网站的满意度和方便程度有了明显的提高。

参考文献：

[1] Agrawal R，Srikant R. Mining sequential patterns[C]//In Proc. of 1995 Int. Conf. Data Engineering(ICDE '95). Taibei: IEEE. Computer Society，1995：3-10.

[2] Hong Tzung-pei，Kuo Chan-Sheng，Chi Sheng-Chai，et al. Mining fuzzy sequential patterns from quantitative data[J]. Systems, Man and Cybernetics，1999，3：962-966.

[3] Ren Jia-dong，Cheng Yin-Bo，Yang Liang-Liang， et al. An algorithm for mining generalized sequential patterns[J]. Machine Learning and Cybernetics，2004，2：1288-1292.

[4] Pei Jian，Han Jiawei，Mortazavi-Asl B，et al. Mining sequential patterns by pattern-growth: the PrefixSpan approach [J]. Knowledge and Data Engineering，2004，16(11)：1424-1440.

[5] Pei Jian，Han Jiawei，Wang Jianyong，et al. PrefixSpan: Mining sequential patterns efficiently by prefix-projected pattern growth[C]// In Proc. 2001 Int Conf Data Engineering (ICDE. 2001). Heidelberg：IEEE. Computer Society，2001：215-224.

（责任编辑：廖友媛）