一种改进的PrefixSpan算法及其在Web用户行为模式挖掘中的应用

姬浩博[1] 王俊红[1,2]
(山西大学计算机与信息技术学院 太原 030006)[1]
(山西大学计算机智能与中文信息处理教育部重点实验室 太原 030006)[2]

摘 要 序列模式挖掘是从序列数据库中挖掘相对时间或其他模式出现频率高的模式。针对PrefixSpan算法构造投影数据库时开销巨大、扫描效率不高的问题,通过以序列扩展代替项集进行扩展、放弃挖掘序列数小于阈值min_support的投影数据库以及直接递归局部频繁项等方式进行改进,并将改进方法应用于Web用户行为模式挖掘中,对日志记录中的规律进行分析和研究。实验分析表明,相比PrefixSpan算法,该改进算法在算法效率方面有一定的提高。

关键词 序列模式挖掘,Web日志挖掘,PrefixSpan算法
中图法分类号 TP181 **文献标识码** A **DOI** 10.11896/j.issn.1002-137X.2016.1.006

Research on Improved PrefixSpan Algorithm and its Application in Web User Behavior Patterns Mining

JI Hao-bo[1] WANG Jun-hong[1,2]
(School of Computer & Information Technology, Shanxi University, Taiyuan 030006, China)[1]
(Key Laboratory of Computational Intelligence and Chinese Information Processing of Ministry of Education, Shanxi University, Taiyuan 030006, China)[2]

Abstract Sequential pattern mining is mining relative time or other mode of high frequency from sequence databases. Based on the PrefixSpan algorithm, the paper proposed an improved adaptive algorithm to improve the problem of expensive construction projection database and low scanning efficiency, through the methods such as using sequence expanding instead of item expanding, abandoning the project databases that the number of sequence is less than min_support and so on. Then the new method was used for Web user behavior pattern mining to analyze and research log records law. Experimental results show that, compared to the PrefixSpan algorithm, the improved algorithm has been improved in the algorithm efficiency.

Keywords Sequence pattern mining, Web log mining, PrefixSpan algorithm

1 引言

随着互联网宽带化、大众化、个性化、移动化的不断发展,新技术层出不穷。带有Web2.0特征的服务已经越来越多地进入网民的视野,RSS、SNS、Tag、Blog、P2P等这些一度只在专业人士的小圈子里出现的概念已经为众多网络用户所应用。Web技术的日新月异、信息的飞速增长,使得Web不仅仅是信息共享和发布的平台。对于无时无刻不断更新数据信息的Web日志文件来说,如何利用这些海量的日志信息进行挖掘,提供智能化、语义化、个性化的信息服务,以更好地满足用户需求,从而带来经济效益,便是Web用户行为模式挖掘的意义所在。

序列模式挖掘是从序列数据库中挖掘相对时间或其他模式出现频率高的模式,由Agrawal和Srikant于1995年提出[1],其关键是对关联模型加入时间属性。序列模式的挖掘主要有apriori算法、广义序列模式(Generalized Sequential Pattern,GSP)挖掘算法[2]、基于模式增长策略的Freespan算法和PrefixSpan算法。Freespan算法和PrefixSpan算法比传统的Apriori算法和GSP算法有效。而PrefixSpan算法是从FreeSpan算法中推导演化而来的,由于收缩速度比FreeSpan更快,因此PrefixSpan比FreeSpan更有效。

序列模式挖掘在Web用户行为模式中有着广泛的应用,研究者们对其开展了很多学术研究和探索。基于频繁模式增长的PrefixSpan算法[3]是现有的序列模式挖掘算法中性能最好的算法之一,它不需要产生候选的序列模式,大大缩减了算法运行需要检索的存储空间,从而有效地减少了构造投影数据库需要的开销。但是,在构造和扫描投影数据库时,PrefixSpan算法还是需要花费大量的时间与空间[4],特别是在挖掘长序列模式和密数据集的过程中,算法可能会生成大量的投影数据库,其中许多投影数据库是重复相同的,从而导致了算

到稿日期:2015-04-30 返修日期:2015-06-26 本文受国家自然科学基金(61202018,61305057,61303008),山西省青年科技基金(2013021018-1),山西省高等学校科技创新项目(2013102)资助。
姬浩博(1989—),男,主要研究方向为数据挖掘、概念格、序列模式挖掘,E-mail:1749742029@qq.com;王俊红(1979—),女,博士,硕士生导师,主要研究方向为数据挖掘。

法性能的下降。

本文将现有的序列模式挖掘 PrefixSpan 算法应用于 Web 日志序列模式挖掘中，并针对此算法存在的构造投影数据库开销巨大、扫描效率不高的问题，通过对投影数据库引入索引、以序列扩展代替项集进行扩展、放弃挖掘序列数已小于阈值 min_support 的投影数据库以及直接递归局部频繁项等方式进行了改进。实验结果表明，改进的算法比 PrefixSpan 算法在效率上有了一定的提高。

2 相关工作

针对 PrefixSpan 算法在挖掘密数据集和长序列模式过程中存在的问题，提出了一些改进的算法。文献[4]提出了一种序列模式更新算法(ISPBP)来解决序列模式挖掘中的增量挖掘问题，该算法利用增量挖掘中形成的频繁项集来减小投影数据库，节省了扫描投影数据库所需要消耗的时间，从而提高了算法的效率。文献[5]提出了一种无重复投影数据库扫描的序列模式挖掘算法(SPMDS)，通过计算投影数据库的伪投影的 Hash 值，然后根据该值判断是否存在重复的投影数据库，并使用一些简化手段来减少对投影数据库的搜索，节省了算法的扫描时间。文献[6]针对现有的序列模式 PrefixSpan 算法提出了一种增量式数据流双重加权序列模式算法 DWC-SPMDS，设计了一种带权重的前缀序列树 WFPS-Tree 和带权重的模式表 WPattern Table，利用增量式方法更新挖掘结果，并通过实验证明算法在运行效率、内存使用、算法可扩展性、模式数量等方面的优越性。文献[7]采用投影和分离数据库等技术来提高 PrefixSpan 算法的挖掘效率。

这些学术研究和探索主要集中在用户访问日志、访问习惯等方面。其中用户访问日志方面主要表现在：对用户访问 Web 时在服务器留下的访问记录进行挖掘，从而挖掘出用户浏览路径和行为倾向，以快速地在海量的 Web 日志数据中自动地发现用户访问页面的相关性，从而通过在相关网页之间增加连接，方便用户使用；用户访问习惯方面主要表现在：对用户浏览习惯加以分析，从而挖掘出用户访问 Web 页面的兴趣模式，以优化站点结构，增加电子商务的潜在用户。Web 用户行为模式挖掘仍有许多未解决的问题，依旧是人们研究的热点问题之一。

3 Web 用户行为模式挖掘

行为模式是指用户操作过程中所体现出来的某种规律性[8]。Web 用户行为模式挖掘是指在用户浏览 Internet 信息的过程中，服务器会在日志文件中记录用户访问及其与客户端之间的交互信息(包括访问的页面、时间、用户 ID 等信息)，通过分析 Web 服务器中的日志文件，发现相似用户群体、访问模式、频繁路径等用户访问站点的浏览规律。分析不同 Web 站点的访问日志可以帮助人们理解用户行为模式和 Web 的结构，给网站管理员提供有利于 Web 站点改进或能够带来经济效益的信息[9]。目前，Web 行为模式挖掘研究领域主要包括：形式化描述网络访问行为、自动获取行为特征以及发现行为规律，研究的数据主要包括 URL 页面请求、页面间链接的拓扑结构、注册用户特征等[6]。

针对用户行为模式的特点，结合 Web 应用的需求，Web 用户行为模式挖掘的处理过程可以分为 4 个阶段：数据预处理、模式挖掘、模式分析及应用，其工作流程如图 1 所示。

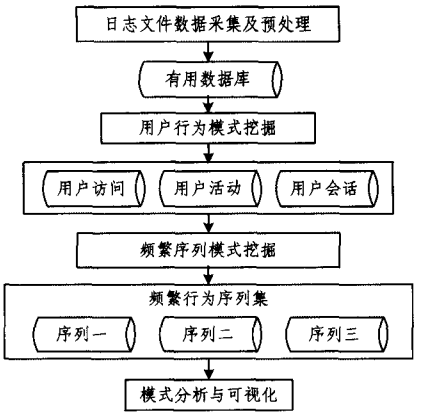

图 1 Web 用户行为模式挖掘的工作流程

(1)数据预处理

对用户行为模式进行挖掘时，首先要对日志文件数据进行预处理，常见的日志文件包括 Server logs、Error logs、Cookie logs 这 3 种类型。由于日志文件记录的是系统在不同时间所发生的事件，因此对日志文件进行预处理得到事务数据库后，再对事物数据库进行数据挖掘。

(2)模式挖掘

这一阶段是 Web 日志挖掘的核心。发现的模式一般有关联规则、序列模式、用户聚类等。统计分析是指获取用户行为的统计信息，如访问时间、频率等[10]；关联规则分析可获取用户页面访问行为间的关系；聚类分析是指通过聚类将特征相似用户的访问行为特点归并分组；频繁序列模式分析可以获取用户的访问习惯、爱好及趋势等[11]。

(3)模式分析

此阶段是利用合适的工具和技术进行用户访问模式的分析、解释，从而筛选出有价值的、用户感兴趣的模式。

(4)可视化

将发现的有价值的模式以适当的形式可视化，为决策层的决策提供依据。

4 PrefixSpan 算法

针对行为序列数据进行分析可以发现，序列的每个元素具有单向性，同时以活动为单位的序列模式比较长。对现有序列模式挖掘算法的适用场合进行分析，选择合适的用户行为频繁序列模式挖掘算法———PrefixSpan 算法[3]。

PrefixSpan 算法是一种使用数据库投影技术的深度优先搜索算法，其性能优于 GSP 与 Apriori，且拥有能够处理非常大的序列数据库的能力。PrefixSpan 算法由于在挖掘的过程中不产生候选序列，且相对原始序列数据库包含较少的投影库，极大缩减了算法运行时需要检索的存储空间，从而减少了构造投影数据库需要的开销，因而其性能更优。本文将 PrefixSpan 算法应用到 Web 用户行为模式的挖掘中，并对此算法进行改进，改进的算法减小了投影数据库的规模，提高了扫描数据库的效率。

4.1 基本定义

基于 PrefixSpan 算法的 Web 日志序列模式的相关定义如下[3]。

定义 1(Web 日志访问序列) 定义为：$s=(p_1, p_2, \cdots, p_m)$，其中 m 为访问序列的长度，$p_k(1\leqslant k\leqslant m)$ 为一个页面

项，长度为 m 的访问序列即为 m-序列。

定义 2(前缀及后缀) 设 Web 日志访问序列 $s=(p_1,p_2,\cdots,p_m)$，a 是页面项，$s<>a=\langle P_1,P_2,\cdots,P_m\rangle$ 表示 s 连接 a，即序列扩展。如果 $s'=s\diamond a$，那么 s 是 s' 的前缀，a 是 s' 的后缀。

定义 3(子序列与超序列) 假设 $s_a=\langle a_1a_2\cdots a_n\rangle$，$s_b=\langle b_1b_2\cdots b_n\rangle$，如果存在整数 $1\leqslant j_1<j_2<\cdots<j_m\leqslant m$，使得 $a_1=b_{j1}$，$a_2=b_{j2}$，$\cdots$，$a_n=b_{jn}$，则称序列 s_a 为 s_b 的子序列，序列 s_b 是 s_a 的超序列。

定义 4(Web 日志序列数据库) 数据库 S 是元组 $\langle uid,s\rangle$ 的集合，其中 uid 是用户 ID，S 是 Web 访问序列。

定义 5(投影数据库中的支持度) 在数据库中包含 s_a 的元组个数为：

$$supports()(s_a)=|\{\langle SID,s_a\rangle|(\langle SID,s_a\rangle\in S\wedge(s_a\subseteq S)\}|$$

定义 6(阈值 min_support) 如果访问序列 S 在 Web 日志序列数据库 S 中的支持度不低于 min_support，则称访问序列 S 为频繁 Web 日志序列模式。

4.2 PrefixSpan 算法描述

PrefixSpan 算法的基本思想是使用频繁前缀划分搜索空间和投影序列数据库，并搜索相关序列，检查前缀子序列，将其相应的后缀子序列投影到数据库中。该算法同时采用分治的策略，不断产生更多、更小的投影数据库，然后在各投影数据库上进行序列模式挖掘。PrefixSpan 算法的主要步骤如下：

Step 1 在原始序列数据库中找出所有的频繁 1-模式，频繁 1-模式组成的集合记为 f_list，假设集合 f_list 的元素个数为 m；

Step 2 把原始序列数据库划分为 m 个子空间，以Step 1 中的 1-模式为前缀，构造相应的数据库，即对应前缀的后缀集合，称此后缀集合为 1-闭模式；

Step 3 以 Step 2 中的 1-闭模式为前缀，在投影数据库中继续 Step 2 的过程，直到不再产生频繁 1-闭模式为止。

下面以表 1 给出的序列数据库 S 及 min_support=2 为例描述 PrefixSpan 算法挖掘的过程。

表 1 序列数据库

序列 ID	序列
1	⟨a(abc)(ac)d(cf)⟩
2	⟨(ad)c(bc)(ae)⟩
3	⟨(ef)(ab)(df)cb⟩
4	⟨eg(ae)cbc⟩

Step 1 得到长度为 1 的序列模式。扫描 S 一次，找出序列中所有长度为 1 的序列模式。它们是⟨a⟩:4，⟨b⟩:4，⟨c⟩:4，⟨d⟩:3，⟨e⟩:3 和⟨f⟩:3，其中符号"⟨模式⟩:计数"表示模式和它的支持度计数。

Step 2 划分搜索空间。序列模式的完全集可根据 6 个前缀划分成下面的 6 个子集：前缀分别为⟨a⟩、⟨b⟩、⟨c⟩、⟨d⟩、⟨e⟩、⟨f⟩的子集。

Step 3 找出序列模式的子集。通过构建相应的投影数据库，并在其中递归地进行挖掘，可以得到步骤 Step 2 提到的序列模式的子集。投影数据库和从中得到的序列模式如表 2 所列。

表 2 序列模式

前缀	投影数据库	序列模式
⟨a⟩	⟨(abc)(ac)d(cf)⟩ ⟨(_d)c(bc)(ae)⟩ ⟨(_b)(df)eb⟩ ⟨(_e)cbc⟩	⟨a⟩⟨aa⟩⟨ab⟩⟨a(bc)⟩⟨a(bc)a⟩⟨aba⟩⟨abc⟩ ⟨(ab)⟩⟨(ab)c⟩⟨(ab)d⟩⟨(ab)f⟩⟨(ab)dc⟩ ⟨ac⟩⟨aca⟩⟨acb⟩⟨acc⟩⟨ad⟩⟨adc⟩⟨af⟩
⟨b⟩	⟨(_c)(ac)d⟩ ⟨(_c)(ae)⟩ ⟨(df)cb⟩,⟨c⟩	⟨b⟩⟨ba⟩⟨bc⟩⟨(bc)⟩⟨(bc)a⟩⟨bd⟩⟨bdc⟩⟨bf⟩
⟨c⟩	⟨(ac)d(cf)⟩ ⟨(bc)(ae)⟩ ⟨b⟩⟨bc⟩	⟨c⟩⟨ca⟩⟨cb⟩⟨cc⟩
⟨d⟩	⟨(cf)⟩ ⟨c(bc)(ae)⟩ ⟨(_f)cb⟩	⟨d⟩⟨db⟩⟨de⟩⟨dcb⟩
⟨e⟩	⟨(_f)(ab)(df)cb⟩ ⟨(af)cbc⟩	⟨e⟩⟨ea⟩⟨eab⟩⟨eac⟩⟨eacb⟩⟨eb⟩⟨ebc⟩⟨ec⟩ ⟨ecb⟩⟨ef⟩⟨efb⟩⟨efc⟩⟨efcb⟩
⟨f⟩	⟨(ab)(df)cb⟩,⟨cbc⟩	⟨f⟩⟨fb⟩⟨fc⟩⟨fcb⟩⟨fbc⟩

5 PrefixSpan 算法的改进

5.1 PrefixSpan 算法与 Web 日志数据库分析

在构造和扫描投影数据库时，PrefixSpan 算法需要花费大量的空间和时间，具体体现在：递归地构建大量投影数据库是执行 FreeSpan 算法的主要开销；而且还需要反复扫描投影数据库，因而减少投影数据库的规模和扫描时间是改进 PrefixSpan 算法[5]的主要途径；且基于 Web 日志数据的如下特性[12,13]，算法进行了扩展和改进。

(1)Web 日志文件无时无刻不在进行着数据的更新。Google 公布的数据显示，每天其数据的索引量就超过了几百亿。对于整个万维网来说，其数据量更是一个无法估计的数字，因而 Web 日志数据库很庞大，其投影数据库同样如此。对投影数据库引入索引，只给定一个初始的 Web 日志序列数据库 A 用于记录所有中间过程的下标的位置，仅需要利用下标去原始库 A 中查找该字符序列即可。此方法无疑提高了查找效率。

(2)由于同一个用户在同一时刻不可能同时去访问两个不同的页面，因此在 Web 日志序列模式的挖掘中，可以通过序列扩展代替项集扩展。

(3)同一个用户的访问序列中出现重复访问页面的概率很小，即 Web 日志数据较为稀疏，这会产生大量的非频繁项，使得算法为扫描不可能出现的序列模式而花费大量不必要的挖掘时间。通过比较投影数据库所包含的序列数和阈值min_support，放弃挖掘序列数已小于阈值 min_support 的投影数据库，既可以减少保存投影数据库所需内存空间，也将减少扫描不可能出现频繁序列的投影数据库的时间，提高挖掘效率。

(4)由于用户对网页内容的关注千差万别，因此用户的访问序列长短不均匀。假如用户对某些页面不感兴趣，他可能访问一个页面后就不再访问与此相关的页面；如果用户对某方面的页面非常感兴趣，那么他很可能访问许多类似的页面。因此在递归挖掘投影数据库的过程中，可以直接递归局部频繁项，以避免构造大量重复的投影数据库，减少构造投影数据库的时间。

5.2 PrefixSpan 算法的改进

针对 PrefixSpan 算法存在构造投影数据库开销巨大、扫描效率不高的问题，主要通过以序列扩展代替项集进行扩展、放弃挖掘序列数已小于阈值 min_support 的投影数据库以及

直接递归局部频繁项等方法进行改进。改进的 PrefixSpan 算法的主要步骤如下：

Step 1　扫描 Web 日志序列数据库 S 一次，得到长度为 1 的序列模式 patterns，删除模式 patterns 中的非频繁项，并将频繁项按其支持度从大到小排序。

Step 2　依次对频繁项进行投影，通过构建相应的投影数据库，并对每一个投影数据库递归地进行挖掘。其中对每一个频繁项，构建相应的投影数据库，对投影数据库扫描一次，删除其局部非频繁项，得到其局部频繁项。如果局部频繁项与 Step 1 得到的除当前频繁前缀记为 α 以外的频繁项相同，则记为 β；对 Step 1 中的投影数据库作关于 β 前缀的投影，判断是否可以减少 β 重复投影及其挖掘。如果条件满足，则只挖掘 β 前缀的序列模式一次，即得到 β 为前缀的序列模式的同时可以得到 α 连接 β 列模式。在找到原投影数据中对应于该频繁项的所有后缀集合后，保存该频繁项的投影数据库，之后只保存在扫描原投影数据库时得到的支持数 min_support 的项，若该频繁项的投影数据库所含序列数小于 min_support，则结束在投影数据库中继续挖掘。

5.3　改进的 PrefixSpan 算法实例

对表 1 中的序列数据库 S 及 min_sup=2：

Step 1　扫描 S 找出长度为 1 的序列模式：⟨a⟩:4，⟨b⟩:4，⟨c⟩:4，⟨d⟩:3，⟨e⟩:3 和⟨f⟩:3。

Step 2　找出以⟨a⟩为前缀的序列模式。在以⟨a⟩为前缀的后缀集合中，⟨a⟩的支持度为 2，故可直接得到序列模式⟨aa⟩。对以⟨a⟩为前缀的投影数据库进行扫描，可得除⟨aa⟩外的序列模式：⟨ab⟩，⟨(ab)⟩，⟨f⟩，⟨ad⟩，⟨af⟩。然后以这些长度为 2 的序列模式为前缀继续进行挖掘，得到的序列模式如表 2 所列。同理，找出以⟨b⟩、⟨c⟩、⟨d⟩、⟨e⟩为前缀的序列模式。其中在以⟨b⟩、⟨d⟩、⟨e⟩、⟨f⟩为前缀后缀的集合中，⟨b⟩、⟨f⟩的支持度为 1，⟨d⟩、⟨e⟩的支持度为 0，因而都舍弃；以⟨c⟩为前缀后缀的集合中，⟨c⟩的支持度为 3，故可直接得序列模式⟨cc⟩。对以⟨c⟩为前缀的投影数据库进行扫描，可得除⟨cc⟩外的序列模式：⟨ca⟩，⟨cb⟩。然后以这些长度为 2 的序列模式为前缀继续进行挖掘，最后得到如表 2 所列的结果。

6　实验结果及性能分析

本文实验在 PC 机上进行，机器的配置为：CPU 2.27 GHz，2GB 内存，500MB 硬盘；操作系统为 Windows XP；编译平台为 Eclipse。算法代码采用 JAVA 语言编写实现。实验选用真实数据集 Rearch 来进行算法的对比，Rearch 数据集的来源为清华大学的 PORTAL 站点某一天的日志集。首先对日志进行数据预处理，得到所有的用户事务，用户序列总数为 10000，平均序列项数为 2.42。

图 2、图 3 分别为在 Rearch 数据集上，分别取最小支持度为 1%、2%、3%、4%、5%时，PrefixSpan 算法和改进的 PrefixSpan算法的执行效率及内存使用的对比结果。由图 2 可以看出：与 PrefixSpan 算法相比，改进的 PrefixSpan 算法执行效率得以提高；由图 3 可以看出：随着最小支持度的增大，改进的 PrefixSpan 算法从整体而言所占用的空间相对较小。由于 Rearch 数据集存在大量非频繁项，PrefixSpan 算法扫描大量不可能产生的模式需要花费很多不必要的时间。而改进的 PrefixSpan 算法采用序列扩展来生成频繁序列，删除频繁度小于最小支持度 min_support 的非频繁项，直接定位投影序列位置进行局部频繁项的递归挖掘，减少了投影数据库的构造数量，从而减少了保存投影数据库所需内存空间，不用再为扫描不可能出现的序列模式而花费不必要的挖掘时间，因而提高了内存的使用效率和算法的挖掘效率。

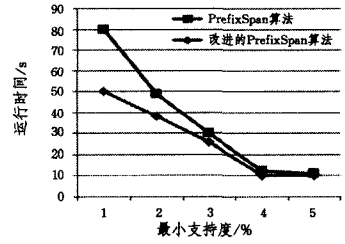

图 2　算法运行时间比较

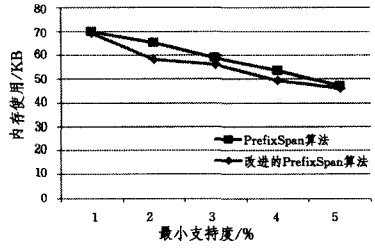

图 3　算法内存使用比较

结束语　本文针对现有的序列模式挖掘 PrefixSpan 算法存在的构造投影数据库开销巨大、扫描效率不高的问题，提出一种改进算法，并将其应用于 Web 日志序列模式挖掘中。实验结果表明，改进的算法不仅比 PrefixSpan 算法在效率上有了一定的提高，还可以有效地获取用户访问 Web 页面的行为模式信息，对日志记录中的规律进行分析和研究，从而优化网站性能，提高网站的知名度。下一步工作将进一步探讨改进的 PrefixSpan 算法的优化问题，同时考虑将其应用至其它应用领域。

参 考 文 献

[1] Agrawal R, Srikant R. Mining sequential patterns [C]// Proceedings of the Eleventh International Conference on Data Engineering (ICDE'95). Washington, DC: IEEE Computer Society, 1995: 3-14

[2] Srikant R, Agrawal R. Mining sequential patterns: generalizations and performance improvements [C]// Proceedings of the 5th International Conference on Extending Database Technology: Advances in Database Technology (EDBT'96). Berlin: Springer-Verlag, 1996: 3-17

[3] Han J, Pei J, Mortazviasl B, et al. FreeSpan: Frequent pattern projected sequential pattern mining[C]// Proceedings of the 6th ACM-SIGKDD International Conference on Knowledge Discovery and Data Mining. New York: ACM Press, 2000: 355-359

[4] Lu J P, Liu Y B, Ni W W, et al. Incremental updating algorithm for mining sequential patterns based on the projection database [J]. Journal of Southeast University, 2007, 36(3): 457-462 (in Chinese)
陆介平，刘月波，倪巍伟，等. 基于投影数据库的序列模式挖掘增量式更新算法[J]. 东南大学学报，2007，36(3)：457-462

[5] Zhang K, Zhu Y Y. Algorithm of sequence pattern mining without repeated database scan projection[J]. Journal of Computer Research and Development, 2007, 44(1): 126-132 (in Chinese)
张坤，朱扬勇. 无重复投影数据库扫描的序列模式挖掘算法[J]. 计算机研究与发展，2007，44(1)：126-132

[6] Han G W. Research on sequential pattern algorithm for data streams based on prefix sequence tree[D]. Qinhuangdao: Yanshan University, 2013(in Chinese)
韩高伟. 基于前缀序列树的数据流序列模式算法研究[D]. 秦皇岛:燕山大学, 2013

[7] Saputra D, Rambli D R A, Foong O M. Sequential Pattern Mining using PrefixSpan with Pseudoprojection and Separator Database[C]// International Symposium on Information Technology, 2008. Kuala Lumpur, Malaysia, 2008: 1-7

[8] Dai Y X, Lian Y F, Wang H. System security and intrusion detection[M]. Beijing: Tsinghua University Press, 2002 (in Chinese)
戴英霞, 连一峰, 王航. 系统安全与入侵检测[M]. 北京:清华大学出版社, 2002

[9] Guo L, Xiang X, Shi Y C. Use Web usage mining to assist online E-Learning assessment[C]// IEEE International Conference on Advanced Learning Technologies. 2004: 912-913

[10] Xing G, Shen J. Efficient data mining for web navigation patterns [J]. Information and Software Technology, 2004, 46(1): 55-63

[11] Xu H Q, Wang Y C. The path Webpage pre fetching model based on analyzing user access[J]. Journal of Software, 2003, 14(6): 1142-1147(in Chinese)
许欢庆, 王永成. 基于用户访问路径分析的网页预取模型[J]. 软件学报, 2003, 14(6): 1142-1147

[12] Yang Z L, Wang Y T, Kitsuregawa M M. An Effective system for mining web log[C]// Proceedings of the 8th Asia-Pacific Web Conference on Frontiers of WWW Research ad Development(APWeb'06). Harbin, China, 2006: 40-52

[13] Mabroukeh N R, Ezeife C I. A Taxonomy of Sequential Pattern Mining Algorithms[J]. Journal of ACM Computing Surveys, 2010, 43(1): 1-41

[14] Hu Y H, Wu F, Liao Y J. An efficient tree-based algorithm for mining sequential patterns with multiple minimum supports[J]. Journal of Systems and Software, 2013, 86(5): 1224-1238

[15] Hu Y H, Huang C K T, Kao Y H. Knowledge discovery of weighted RFM sequential patterns from customer sequence databases[J]. Journal of Systems and Software, 2013, 86(3): 779-788

[16] Salehi M, IKamalabadi N. Hybrid recommendation approach for learning material based on sequential pattern of the accessed material and the learner's preference tree[J]. Knowledge-Based Systems, 2013, 48: 57-69

[17] Hu Y H, Tsai C F, Tai C T, et al. A novel approach for mining cyclically repeated patterns with multiple minimum supports [J]. Applied Soft Computing, 2015, 28: 90-99

[18] Zhang B B, Lin C W, Gan W S, et al. Maintaining the discovered sequential patterns for sequence insertion in dynamic databases [J]. Engineering Applications of Artificial Intelligence, 2014, 35(10): 131-142

(上接第 17 页)

[5] Yan Chao-kun, Hu Zhi-gang, Li Xi, et al. Reliability-Cost Optimization Scheduling Model and Algorithm in Grid [J]. Computer Science, 2013, 40(3): 136-141(in Chinese)
阎朝坤, 胡志刚, 李玺, 等. 面向可靠性-费用优化的网格任务调度模型及算法研究 [J]. 计算机科学, 2013, 40(3): 136-141

[6] Li Xin, Jia Zhi-ping, Ju Lei, et al. Energy Efficient Scheduling and Optimization for Parallel Tasks on Homogeneous Clusters [J]. Chinese Journal of Computers, 2012, 35(3): 591-602(in Chinese)
李新, 贾智平, 鞠雷, 等. 一种面向同构集群系统的并行任务节能调度优化方法 [J]. 计算机学报, 2012, 35(3): 591-602

[7] Cotronis Y, Konstantinidis E, Louka M A. A comparison of CPU and GPU implementations for solving the Convection Diffusion equation using the local Modified SOR method [J]. Parallel Computing, 2014, 40(7): 173-185

[8] Konstantinidis E, Cotronis Y. Graphics processing unit acceleration of the red/black SOR method[J]. Concurrency and Computation, 2013, 25(8): 1107-1120

[9] Epicoco I, Mocavero S, Aloisio G. The performance model for a parallel SOR algorithm using the red-black scheme [J]. International Journal of High Performance Systems Architecture, 2012, 4(2): 101-109

[10] Satoa Y, Hino T, Ohashi K. Parallelization of an unstructured Navier-Stokes solver using a multi-color ordering method for OpenMP [J]. Computers & Fluids, 2013, 88(12): 496-509

[11] Iwashita T, Nakashima H, Takahashi Y. Algebraic Block Multi-Color Ordering Method for Parallel Multi-Threaded SparseTriangular Solver in ICCG Method [C]// Proceeding of IEEE 26th International Parallel and Distributed Processing Symposium. Shanghai, China, 2012: 474-483

[12] Wang Chuan-long, Meng Guo-yan, Yong Xue-rong. Modified parallel multisplitting iterative methods for non-Hermitian positive definite systems [J]. Advances in Computational Mathematics, 2013, 38(4): 859-872

[13] Meng Guo-yan, Wang Chuan-long, Yan Xi-hong. Self-adaptive Non-stationary Parallel Multisplitting Two-Stage Iterative Methods for Linear Systems [J]. Data and Knowledge Engineering, 2012, 7696: 38-47

[14] Zhang Cheng-yi, Luo Shuang-hua, Xu Zong-ben. On parallel multisplitting block iterative methods for linear systems arising in the numerical solution of Euler equations [J]. Journal of Computational and Applied Mathematics, 2015, 279: 249-260

[15] Zhang Li-tao, Li Jian-lei, Gu Tong-xiang, et al. Convergence of Relaxed Matrix Parallel Multisplitting Chaotic Methods for Matrices [J]. Journal of Applied Mathematics, 2014, 2014: 1-9

[16] Zhang Li-li. Two-Stage Multisplitting Iteration Methods Using Modulus-Based Matrix Splitting as Inner Iteration for Linear Complementarity Problems [J]. Journal of Optimization Theory and Applications, 2014, 160(1): 189-203

[17] Zheng Ning, Yin Jun-feng. Accelerated modulus-based matrix splitting iteration methods for linear complementarity problem [J]. Numerical Algorithms, 2013, 64(2): 245-262

[18] Bai Zhong-zhi, Zhang Li-li. Modulus-based synchronous two-stage multisplitting iteration methods for linear complementarity problems [J]. Numerical Algorithms, 2013, 62(1): 59-77

[19] Butrylo B, Tudruj M, Masko L. Parallel SSOR preconditioning implemented on dynamic SMP clusters with communication on the fly [J]. Future Generation Computer Systems, 2010, 26(3): 491-497

[20] Xu Qiu-yan. A New Parallel Successive Overrelaxation Iterative Algorithm for Poisson Equation [C]// Proceeding of 2011 International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery. Beijing, China, 2011: 295-300