第35卷 第23期 计 算 机 工 程 2009年12月
Vol.35 No.23 Computer Engineering December 2009
·软件技术与数据库· 文章编号：1000—3428(2009)23—0047—03 文献标识码：A 中图分类号：TP311

基于Web日志的实时推荐系统

刘敏娴[1,2]，夏 阳[2]
(1. 徐州师范大学现代教育技术中心，徐州 221116；2. 中国矿业大学计算机科学与技术学院，徐州 221008)

摘 要：针对个性化实时推荐系统的不足，提出通过构造BP树的方法压缩访问事务集。采用一个实时推荐的系统模型，将耗时的数据预处理放在离线模块，实时推荐采用动态修剪BP树的方法，穿过访问模式树的相关部分，利用网页推荐算法得到频繁访问集，生成推荐集。结果表明该算法只需扫描数据库一次，得到的频繁模式可满足页面实时推荐的快速需求。

关键词：数据挖掘；数据预处理；关联规则；实时推荐

Real-time Recommendation System Based on Web Log

LIU Min-xian[1,2], XIA Yang[2]
(1. Modern Educational Technology Center, Xuzhou Normal University, Xuzhou 221116;
2. School of Computer Science and Technology, China University of Mining and Technology, Xuzhou 221008)

【Abstract】Aiming at the insufficiency of personalized real-time recommendation system, this paper proposes the theory through constructing the BP tree method, and compressing visit business collection, using a Real-Time Recommendation System(RTRS) model, putting time-consuming data pre-processing module on the off-line one, recommending the use of dynamic BP tree pruning method in real-time, passing through the visit of the relevant parts of the pattern tree, obtaining the frequent visit collection by way of the homepage recommendation algorithm so as to produce recommendation collection. Result indicates that this algorithm only needs to scan the database one time, the frequent pattern obtained can meet the rapid demands of the Web page recommendation in real-time.

【Key words】data mining; data preprocessing; association rules; real-time recommendation

1 概述

Web使用挖掘技术能捕捉到用户的浏览行为和行为的方式作为智能的Web站点，一些研究就是使用的关联规则挖掘作推荐系统，其中多数研究也试图在生成推荐或为实时的动态推荐系统之前发现所有的关联规则。

WebWatcher系统[1-2]采用跟踪用户浏览Web站点的行为或访问路径方法，学习用户的访问模式，将用户可能感兴趣的Web页在线推荐给用户。还有利用Apriori 算法用于预处理之后的数据，挖掘出频繁项集，同时把用户最近访问的几个页面和挖掘出频繁项集相匹配，关联规则挖掘计算复杂性高，不易增量挖掘。目前已经存在很多的个性化信息推荐系统，主要存在以下一些问题：

(1)多数推荐系统对新用户和访问站点较少的用户的信息推荐考虑不够，因为新用户和浏览站点较少的用户被系统收集的用户信息较少，采用的一些推荐算法并不合适。

(2)算法的伸缩性，推荐系统在处理大规模数据量时面临着越来越严重的问题，难以满足系统实时性要求。

本文研究的问题是利用 Web 使用挖掘动态的引导用户选择适当的网页，基于以往的访问记录，立即推荐给下次合适的网页。通过对当前网站推荐系统以及相关技术的分析，设计一个实时推荐系统(Real-Time Recommendation System, RTRS) 结构，主要解决：(1)对Web日志进行挖掘时，对Web日志文件进行有效的预处理。(2)在预处理的基础上，采用构造 BP 树的算法发现用户的频繁访问路径，产生推荐集进行实时推荐。

2 RTRS模型

本文给出的模型是根据Web以往的访问日志，应用数据预处理的技术，形成事务文件，然后构造 BP 树，利用实时推荐算法对事务文件进行挖掘，从而挖掘出用户浏览的频繁访问路径，形成频繁项集，基于这些关联规则设计了一个实时推荐的模块，为用户提供推荐服务，使浏览的过程成为自适应的过程。RTRS 模型如图 1 所示。

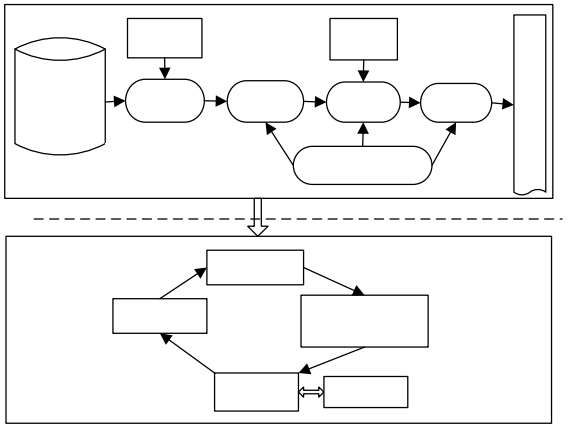


图 1 RTRS 模型

基金项目：徐州师范大学校级基金资助项目(07XLB21)
作者简介：刘敏娴(1967－)，女，讲师，主研方向：数据仓库，数据挖掘；夏 阳，副教授
收稿日期：2009-04-06 **E-mail：**liumx@xznu.edu.cn

离线模块首先需要对原始的 Web 日志文件进行数据预处理，小型的网站每天的日志文件也就几兆或几十兆，而大型的网站每天的日志文件则会超过几百兆甚至几 G，所以对于如此海量的日志文件进行预处理会非常耗时，必须把这一步工作放在离线模块中来进行，否则会大大影响实时推荐的速度。

在线模块模块，其任务就是根据跟踪用户当前点击流及应用推荐算法计算生成推荐集，实现实时推荐在增量构造 BP 树时，只需扫描一次序列事务数据库即可，同时挖掘算法也仅仅挖掘相关部分的结构，减少计算时间，因此，整个框架的效率很高。

3 RTRS关键技术

3.1 离线模块数据的预处理

离线模块数据的预处理过程包括：

(1)数据收集，可以从服务器端数据、客户端数据、代理服务器端进行，这些数据不仅意味着存放的位置不同，其中还包含了 Web 世界中不同的浏览模式，本文使用的数据来自于服务器端。

(2)数据清理，日志中记录的原始数据通常非常大，存在很多对于挖掘任务没有意义的数据记录和数据项。要删除以下几类：首先是图片、动画等，要除去 URL 中包含后缀为：gif, jpg, jpeg, swf, png 等，其次是搜索引擎，这些蜘蛛程序和其他蠕虫(bot)程序，还有用户请求访问失败的记录，这类访问的返回代码为 404(请求的页面没有找到)、301(永久删除)、500(内部服务器错误)，以及由 GET 以外的方式完成的服务和其他无关的日志。例如：后缀为：css, map, js 等文件。

(3)用户识别，目的是对用户唯一性的识别。由于用户端高速缓存、代理服务器、防火墙以及动态地址池的存在，使得这一过程的实现较为困难。如：通过代理服务器上网的用户在日志文件中的 IP 地址相同；由于防火墙的存在，在服务器日志中多个用户的 IP 地址相同，动态地址分配又将同一个用户赋予不同的 IP 地址。

(4)会话识别，就是用户在规定时间内(或称一次浏览内)对服务器的一次有效访问，通过其连续请求的页面，可以获得他在网站中的访问行为和浏览兴趣，有 4 种识别会话的模型[3]：页面类型模型(page type model)，参引长度模型(reference length model)，最大前向参引模型(maximal forword reference model)和时间窗口模型(time window model)。最常采用的是时间窗口模型，以用户访问时间作为划分会话的分界，一般间隔时间取 30 min。

(5)路径完整，指完善访问路径，由于存在着客户端的缓存，用户浏览页面时很可能使用到浏览器的后退功能，这时要根据用户访问路径的前后页面进行推理，将其中疏漏的页面补在路径里。

3.2 在线模块实时推荐

在线模块根据用户的访问，增量构造 BP 树，增量的模式可以很快影响下一次的推荐结果。

3.2.1 基本概念

定义 1 设对数据进行预处理后可以得到页面集合 $E=\{e_1, e_2,\cdots, e_n\}$和序列事务集合 $S=\{s_1, s_2,\cdots, s_m\}$，其中，$S\in E$ 是包含一系列 E 中的一个序列事务，模式 X 代表浏览行为，将项集 X 中所含的项目个数称为项集的维数或长度，记为：$|X|$。

定义 2 设有项目集 $X\in E$，设 $count(X)$为 X 在 S 出现的次数，$count(S)$为 S 中事务的总数，则 X 的支持度定义为 $Support(X)=count(X)/count(S)$，如果 $Support(X)$大于用户给定的最小支持度 minsp，则称 X 为频繁项目集[4]。

3.2.2 算法描述

(1)访问模式树 BP 树的定义

访问模式树 BP 树由项头表(header_object)和前缀树(prefix_tree)组成。

项头表由 3 个域组成：名称(Item)，支持度计数、节点链，其中，支持度计数是指该节点的访问数目，节点链指向相同名字的前缀树的第一个节点。

前缀树包含 4 个子域：名称，计数，子节点链，节点链，其中，计数代表的是访问这个节点的数目，子节点链是指向前缀树中的下一个节点，节点链是指向前缀树中相同名字下一节点的链接，如果没有下一连接，用“Null”表示。

(2)BP 树的构建

首先建立根节点，用 Root 表示，然后扫描事务数据库 S，为第一个事务创建一个分枝，然后将后面的事务插入到树中，当处理一个事务时，首先查看当前事务与树已有的分枝之间是否有相同的共享前缀，如果有，则沿共同路径的前缀上的每个节点的计数增加 1，为跟随在前缀之后的项创建节点并与调整相应的项头表中相同节点的链接，如果没有则创建一个新的分枝，计数置为 1。

本文通过一个实例来构建 BP 树，用户会话如表 1 所示，BP 树的建立过程如图 2 所示。

表 1 用户会话

TID	Item
01	*A*, *B*, *C*, D
02	*A*, *C*
03	*A*, *B*, *D*, *C*
04	*B*, *A*, *C*
05	*A*, *B*, *C*, *D*, *E*

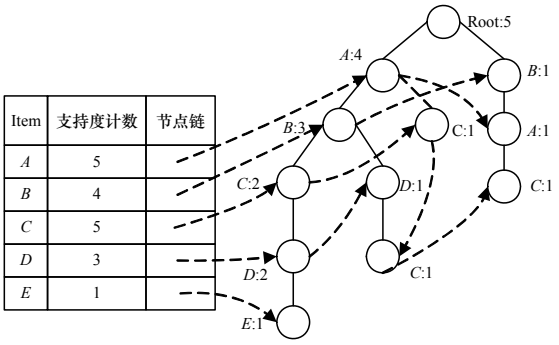


图 2 BP 树的建立过程

(3)页面推荐算法

页面推荐算法由 2 个算法组成：构造候选子集的算法，在候选子集的基础上产生推荐集的算法。

推荐算法首先遍历项头表，找到与浏览行为 X[1]相同的节点，通过发现其节点链一直查找下去，直到节点链为 Null 的节点，只需访问相关部分，对树进行修剪，得到第一个候选节点，再通过 X 的长度 L 来控制挖掘的深度，返回与 X[$index$]相同的下一级候选节点，一直持续到项集的长度，再挖掘所有的候选子项后，合并相同的节点，计数求和，假设这组计数满足之前定义的最小支持阈值 ξ，记录它的名字作为推荐结果组，否则剪切。

构造候选集的算法如下：

输入 BP 树，项头表，访问事务 X

输出 候选集

```
for each 项头表中的每一个节点 do
    if 项头表的名称=X[1] then
        发现第一个候选节点，通过遍历头结点一直到尾节点修剪节点树获得第一个候选集
            for index=2 to |X|
        在第一候选集的基础上，由 X[index]构造从长度为 2~|X|的下一级候选集
Return 候选集 candidate
```

产生推荐集算法如下：

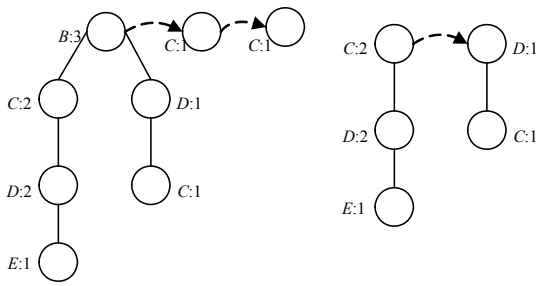
输入 候选集，最小支持阈值 ξ

输出 推荐集

```
i=1
l=|candidate|
while i<=l then
        s=0
      j =i+1
      while j <=l then
      if 候选子集[i] = 临时子集[j] then
          s=s+临时子集[j].count  //合并相同的节点，然后修剪
          从临时子集中修剪临时子集[j]
          L=l – 1  //因为修剪了子集，所以长度-1
      if  s>ξ*根节点的数目 then
          推荐集=推荐集∪候选子集[i]
return 推荐集
```

4 实验结果与分析

本文通过一个例子来说明挖掘过程，BP 树如图 3 所示。X=(A, B)是浏览行为，ξ=0.1 最小支持阈值，使用 X 去挖掘它的候选集。BP 挖掘使用遍历项头表找到与 X[1](即 A)相同的目标，根据项头表，发现有 3 个 A，一个是(A, B, C, D)中的 A，一个是(A, C)中的 A，另一个是(B, A, C)中的 A，它们的子节点都成为下一层的候选节点，把所有的候选节点做成一个链，只须关心候选集和它们的子链图 3(a)，减少了挖掘的时间，然后比较下一个子项 B，发现只有一个节点相匹配，如图 3(b)所示。



(a)选集及其子链图　　(b)匹配节点

图 3　BP 挖掘实例

它们的子节点对应不同的推荐集，然后检查支持阈值，剪切不频繁的分枝，对于每一个推荐结果，它出现的计数必须满足最小支持度阈值，最小支持度等于 ξ 乘以在 S 中事务的数目(根节点的总数)，C(2>0.1×5), D(1>5×0.1)满足阈值，因此，获得推荐集 C 和 D。

5 结束语

本文给出一个基于 Web 日志的实时推荐的模型，使用 BP 树压缩访问信息以便更好地发现频繁序列。为改进搜寻的性能，BP 树挖掘实现分而治之的方法，只需搜寻树的相关分枝，而不是遍历整个树，基于该技术，BP 挖掘能实时推荐，而且每一个推荐都在短时间内实现。但它也存在可扩展性较差的问题，当事务数据量很大，节点较多时，内存可能无法容纳整个 BP 树，从而降低算法的性能。

参考文献

[1] Joachims T, Freitag D, Mitchell T. Web Watcher: A Tour Guide for the World Wide Web[C]//Proc. of IJCAI'97. Nagoya, Japan: [s. n.], 1997.

[2] 戴军湘. 基于Web日志挖掘的自适应网站推荐系统框架研究[D]. 长沙: 湖南大学, 2005.

[3] 余轶军. Web 访问信息挖掘若干关键技术研究[D]. 杭州: 浙江大学, 2006.

[4] 郭　维. Web日志挖掘中GITC算法的改进[J]. 计算机工程, 2008, 34(4): 60-62.

编辑　金胡考

（上接第 46 页）

参考文献

[1] 邓志鸿, 唐世渭, 张　铭, 等. Ontology研究综述[J]. 北京大学学报: 自然科学版, 2002, 38(5): 730-738.

[2] Gruber T R. A Translation Approach to Portable Ontology Specifications[J]. Knowledge Acquisition, 1993, 5(2): 199-200.

[3] Studer R, Benjamins V R, Fensel D. Knowledge Engineering: Principles and Methods[J]. Data and Knowledge Engineering, 1998, 25(12): 16l-197.

[4] Perez A G, Benjamins V R. Overview of Knowledge Sharing and Reuse Component: Ontologies and Problem-solving Methods[C]// Proc. of IJCAI'99. Stockholm, Sweden: [s. n.], 1999.

编辑　金胡考