文章编号:1001 - 9081(2005)02 - 0314 - 03

一种基于 ExLF 日志文件的用户会话识别启发式算法

冯 林[1,2],何明瑞[2],罗 芬[2]
(1. 西南交通大学 计算机与通信工程学院,四川 成都 610031;
2. 四川师范大学 工程技术系,四川 成都 610072)
(mgyfl@163.net)

摘 要:在详细介绍 ExLF 日志文件格式的基础上,定义了会话表;阐述了用户会话识别图的概念;给出了生成此会话识别图的一种启发式算法。最后,用一个例子验证了算法的有效性。

关键词:ExLF 日志文件格式; Web 挖掘;用户会话识别图
中图分类号: TP301 **文献标识码**:A

Heuristic algorithm of user session identification based on ExLF Log file format

FENG Lin[1,2], HE Ming-rui[2], LUO Fen[2]
(1. *College of Computer Science and Communication Engineering, Southwest Jiaotong University, Chengdu Sichuan* 610031, *China*;
2. *Department of Engineering and Technology, Sichuan Normal University, Chengdu Sichuan* 610072, *China*)

Abstract: Based on ExLF Log file format, the Session Tabel was defined, and the conception of user session graph was expounded in detail. A heuristic algorithm of creating graph of user sessions identification was presented. At last, an example illustrated the efficiency of this algorithm.

Key words: ExLF Log file format; Web mining; graph of user session identification

面对互联网海量的数据,如何从中发现有用知识,成为 Web 应用研究的领域之一。而对 Web 用户会话进行有效地识别,可以有针对性地定义用户行为模式及挖掘相应模式,以进一步挖掘更复杂的用户行为结构。目前,用户会话识别有 Cookies、嵌套会话 ID、登记、修改浏览器等方法。

1 Web 服务器日志文件格式概述

Web 服务器日志记录了网站用户的访问浏览行为,它对于系统管理员和 Web 开发人员来说都是至关重要的。不同的 Web 服务器提供的日志文件格式不尽相同。考虑到普遍性和代表性,下面仅以 IIS 服务器日志文件为例,讨论 W3C 扩展日志记录 ExLF 的基本格式。

ExLF 是最复杂的标准日志格式,也是唯一能够定制那些特殊的域写入日志的标准化日志格式,它提供了更详细的可选域的集合。正是这种灵活性使得 ExLF 被很多各种各样的软件使用,包括防火墙、缓存服务器以及其他的一些应用软件。

为了确定什么被写入日志,一个 ExLF 日志文件包括两种不同的记录类型。第一种记录类型是指令型记录,它包括日志文件内容的元数据。第二种记录类型是数据记录,包括被日志文件实际记录的数据域。

日志文件中以"#"开头的表示这一行是指令,这条指令提供有关日志记录或关于日志文件本身的一些信息。有两条指令是必需的,而且一定要在一个日志文件的开头出现,这两条指令是 Version 和 Fields。下面给出 W3C 扩展日志格式指令的使用。

Version:ExLF 使用的版本;Fields:日志文件中出现的空白域列表;Software:生成日志的软件;Start-Date:日志开始的日期和时间;End-Date:日志结束日期和时间;Date:在日志中加入条目的时间;Remark:软件或管理软件人员在日志中的注释部分。

表 1 W3C 扩展日志格式部分域

域标识符	前缀是否必须	描述
date	No	事务完成的日期
time	No	事务完成的时间
ip	Yes	前缀确定的主机的 IP 地址和端口号
dns	Yes	由前缀确定的主机的完整主机名
method	Yes	执行的动作
uri	Yes	被访问的数据源的完整 URL
cookie	Yes	发送的 cookie 的内容
username	Yes, cs only	访问者被网站记录后的用户名
referrer	Yes, cs only	最后访问的页面的 URL
user-agent	Yes, cs only	客户浏览器类型
protocol	Yes	使用的 Internet 协议,如 HTTP, FTP
status	Yes	HTTP 状态码
cached	No	是否缓存了一次点击;0 表示没有缓存

ExLF 确定的日志格式可以被定义成一个域的集合,而不是定义一种固定的格式。但要注意在很多域的前面需加上前缀。下面列出了 W3C 扩展日志格式的可用域前缀。

c - :客户端;s - :服务器端;r - :远程服务器;cs - :客户端到服务器端;sc - :服务器端到客户端;sr - :服务器端到远

收稿日期:2004 - 07 - 28;**修订日期**:2004 - 10 - 15

作者简介:冯林(1972 -),男,四川巴中人,硕士研究生,主要研究方向:数据挖掘; 何明瑞(1963 -),男,四川达州人,副教授,主要研究方向:数据库理论与技术; 罗芬(1970 -),女,重庆人,讲师,硕士,主要研究方向:计算机网络理论与设计.

程服务器;rs - :远程服务器到服务器端;x - : 应用。这些前缀与域指令行中的域标识符结合在一起,用于说明什么样的数据将写入日志。

下面是一个 Web 服务器日志文件 ExLF 格式的例子,其中,被日志文件实际记录的数据域已经过清洗、抽取、转换。

```
# Software: Microsoft Internet Information Services 5.0
#Version: 1.0
#Date: 2001 -03 -01 17:42:15
#Fields: c-ip date time cs-method cs-uri-stem sc-status sc-bytes
  sc-referer cs-user-agent
```

表 2　日志文件实际记录的数据域经清洗、抽取、转换的会话表

IP	Time	RequestURL	ReferrerURL	Agent
169.254.147.157	15:04:41	GET index.html HTTP/1.0	http://www.search.edu?key=try	Mozilla4.2(Win98, I)
169.254.147.157	15:04:41	GET L.html HTTP/1.0	http://www.try.edu/index.html	Mozilla4.2(Win98, I)
169.254.147.157	15:04:41	GET A.html HTTP/1.0	http://www.try.edu/index.html	Mozilla4.2(Win98, I)
169.254.147.157	15:04:44	GET/HTTP/1.0	http://www.search.edu?key=try	Mozilla4.2(IE5.0, winNT)
169.254.147.157	15:04:45	GET L.html HTTP/1.0	http://www.try.edu/index.html	Mozilla4.2(IE5.0, winNT)
169.254.147.157	15:04:45	GET A.html HTTP/1.0	http://www.try.edu/index.html	Mozilla4.2(IE5.0, winNT)
169.254.147.157	15:05:18	GET D.html HTTP/1.0	http://www.try.edu/A.html	Mozilla4.2(IE5.0, winNT)
169.254.147.157	15:05:43	GET Index.html HTTP/1.0	-	Mozilla4.2(Win98, I)
169.254.147.157	15:05:43	GET L.html HTTP/1.0	http://www.try.edu/index.html	Mozilla4.2(Win98, I)
169.254.147.157	15:05:45	GET A.html HTTP/1.0	http://www.try.edu/index.html	Mozilla4.2(Win98, I)
169.254.147.157	15:06:01	GET H.html HTTP/1.0	http://www.try.edu/C.html	Mozilla4.2(Win98, I)
169、254.147.157	15:06:06	GET B.html HTTP/1.0	http://www.try.edu/L.html	Mozilla4.2(Win98, I)
169、254.147.157	15:06:18	GET 2.html HTTP/1.0	http://www.try.edu/B.html	Mozilla4.2(Win98, I)
169.254.147.157	15:06:19	GET F.html HTTP/1.0	http://www.try.edu/B.html	Mozilla4.2(Win98, I)
169.254.147.157	15:06:36	GET L.html HTTP/1.0	http://www.try.edu/C.html	Mozilla4.2(Win98, I)
169.254.147.157	15:07:24	GET G.html HTTP/1.0	http://www.try.edu/2.html	Mozilla4.2(Win98, I)
169.254.147.157	15:07:49	GET 4.html HTTP/1.0	http://www.try.edu/F.html	Mozilla4.2(Win98, I)
169.254.147.157	15:07:49	GET FL.html HTTP/1.0	http://www.try.edu/F.html	Mozilla4.2(Win98, I)
169.254.147.157	15:09:59	GET 5.html HTTP/1.0	http://www.try.edu/G.html	Mozilla4.2(Win98, I)
169.254.147.157	15:10:02	GET G3.html HTTP/1.0	http://www.try.edu/G.html	Mozilla4.2(Win98, I)

2　会话识别图的定义及相应算法

2.1　会话识别图的定义

在利用 IP 地址、Agent 情况和 Web 服务器端用户访问的点击记录,进行用户识别时,常常会遇到以下几种典型问题:

1) 单 IP 地址/多服务会话。互联网服务供应商通常会为用户提供若干代理服务,一个代理服务可能同时有几个用户在访问同一网站。

2) 多 IP 地址/单服务会话。一些互联网服务供应商或一些黑客工具会给用户的每次访问随机赋予若干 IP 地址之一,这样的话,一个服务会话就会有若干个不同的 IP 地址。

3) 多 IP 地址/单用户。一个用户通过若干具有不同 IP 地址的机器在不同的会话期间来访问网站,这使得很难跟踪来自同一个用户的重复访问。

4) 多服务会话/单用户。在一个同时打开若干浏览器进程窗口,并同时访问同一个网站的不同内容时,就会出现这样的情况。

5) 单客户/多用户。当一个以上的用户使用同一台机器时,例如:在家庭或公共场合,就会产生这样的情况。

在识别出用户之后,每个用户的点击序列就必须划分为会话。由于一般无法获得来自其他服务器的网页请求信息,因此很难知道用户当前是否已离开特定的网站。若没有其他明显标识的话,通常用一个时间阈值来帮助将点击序列划分为会话。

通过以上分析可知,当只有 ExLF 日志时,用户和会话通常认为是相同的,因为没有办法知道来自相同 IP 地址/Agent 对的不同会话原来都是同一个用户。为此可以合理地假设同一个 IP 地址不同类型 Agent 就代表不同的会话。在会话识别中,常常需要利用网页文件请求的来源,若请求来源的网页文件不是已知的会话内容,就认为它来自一个新的会话。

定义 1　会话表 SessionTabel 是指对 Web 服务器日志文件经清洗、抽取、转换所生成的数据表。在本文中,数据表关系模式定义如下:

SessionTabel(IP, Time, RequestURL, ReferrerURL, Agent)

其中:IP 表示客户端 IP 地址;RequestURL 表示客户端请求的方法、URL 地址及使用协议;ReferrerURL 表示用户此刻点击进入当前页面的 URL;Agent 表示客户端浏览器类型。

定义 2　最接近网页是指在 SessionTabel 中,在 t_i 时刻请求的页面文件为 f_i,如果 f_i 的 ReferrerURL 值在相同 IP/Agent 对 $\cdots t_l \cdots t_m \cdots t_n \cdots$ 时均出现在 RequestURL 中,则称满足 $\min\{t_i - t_l, \cdots, t_i - t_m, \cdots, t_i - t_n\}$ 成立的时间 t_k 所对应的网页文件为 f_i 最接近网页。

定义 3　会话识别图定义描述为:$SessionGraphs = \{G_1, G_2, \cdots G_n\}$

$G_i = (V,E)(i = 1,2,\cdots,n)$ 为一有向图,且 $G_i \cap G_j = \varnothing(i \neq j)$,$V$ 为顶点集合,E 为边集合。G_i 代表一会话,须满足以下条件:

1) G_i 的顶点集 V 由某会话所访问的所有网页文件构成,每个链接是 G_i 的一条有向边。

2) 设当前请求页面 f_i 的 ReferrerURL 为 r_j,如果 r_j 不存在

于 SessionGraphs 集合或 r_j 为符号“-”，则以 f_i 为顶点生成新的会话图 G，并把 G 加入 SessionGraphs 集合；否则，则存在一条从 r_j ®f_i 的有向弧。

2.2 会话识别图算法描述

输入：ExLF 日志文件。

输出：用户会话识别图（如图 1）。

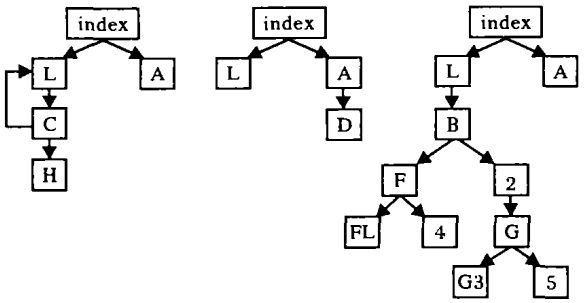

图 1 用户会话识别图

1) 对 Web 服务器的日志文件进行清洗、转化，生成 SessionTabel 数据表；
2) 以 IP、Time、Agent 对 SessionTabel 排序；
3) SessionGraphs = ∅, count = 0 及域值 T；
4) while not SesionTabel. EOF() do
for each unique IP/Agent
if $t_j - t_{j-1} < T$
f_i = SessionTabel. RequestURL;
r_j = SessionTabel. ReferrerURL;
if r_j not exist in SessionGraphs or r_j = ' - '
$G = (f_i, \varnothing)$;
SessionGraphs = SessionGraphs ∪{G};
count++;
else if r_j in SessionGraphs and r_j 在 SessionGraphs 中出现次数 > 1,
$r_k = f_i$ 最接近网页;
G. V = G. V ∪{f_i};
G. E = G. E ∪{$r_k f_i$};
else
G. V = G. V ∪ {f_i};
G. E = G. E ∪{$r_j f_i$};
End

3 结语

本文在介绍 ExLF 日志文件格式的基础上，阐述了会话识别图的概念，并给出了生成此会话识别图的启发式算法。我们将在以下相关领域进一步探讨和完善：

1）在用户会话识别图基础上，研究会话图之间的相似性算法，可以对用户进行聚类分析。

2）在用户会话识别图的基础上，以浏览页面的时间作为图权值，可以挖掘用户在一个网页上所花时间，来确定用户对感兴趣的页面。

3）在用户会话识别图基础上，研究用户访问网页的进程，可以进一步挖掘用户行为模式。

参考文献：

[1] 朱明. 数据挖掘[M]. 合肥：中国科学技术大学出版社，2002.
[2] SWEIGER M. 点击流数据仓库[M]. 陆昌辉，张光剑，陈佐，等译. 北京：电子工业出版社，2004.
[3] SLINOFF G. Web 数据挖掘[M]. 沈钧毅，宋擒豹，燕彩蓉，等译. 北京：电子工业出版社，2004.
[4] 史忠植. 知识发现[M]. 北京：清华大学出版社，2002.

（上接第 313 页）

2) x86 Family 6 Model 7 Stepping 3 GenuineIntel ~548 Mhz

这台机器配置有 WebSphere Application Server 5.0.25。

Test Client：

操作系统：Microsoft Windows 2000 Server

内存：512M 处理器：x86 Family 15 Model 2 Stepping 4 GenuineIntel ~1804 Mhz

这台机器装有 Microsoft 的 Web Application Stress 负载测试工具，作为测试客户端运行测试。用单个线程单 Socket 对以下两个方案进行无间歇压力测试，测试的持续时间 3 分钟 。

方案 1 由 Server 2 上的 WebSphere Application Server 服务静态内容，并且打开了 file serving enabler。

方案 2 由 Server 1 上的 IBM HTTP Server 服务静态内容，并将所有内容都传递回在 Server 上的 Application Server。

表 1 两个方案的测试结果

case	Number of test clients	Test length (h/min/sec)	Number of hits	Requests per second
1	1	0:03:00	293	1.63
2	1	0:03:00	441	2.45

尽管这些测试运行是最低限度的，但它们提供了一些有用的数据。首先，如表 1 所示，用分离的 Web Server 服务静态内容（这些静态内容来自与 Application Server 分离的机器）使性能有了明显的提高。如果 Application Server 服务静态内容，那么 Application Server 机器的 CPU 利用率比其他几个方案的都高。当 Application Server 不服务静态内容时，CPU 利用率会较低并且在测试运行期间错误也较少。因此，Application Server 机器一般应执行紧急任务，最好不要让 Application Server 机器服务静态内容[6]。

5 结语

本文提出了将 Web Server 独立配置，和 Application Server 的功能分离来提高性能的思想，并且通过测试已经取得了良好的效果。目前，这种思想越来越受到重视，越来越多的企业正在采用这种发式，IBM 的官方网站上也建议将动态、静态分离来提高性能，这说明这种新的部署方式有着十分广阔的前景。

参考文献：

[1] 范国闯，钟华，黄涛. Web 应用服务器研究综述[J]. 软件学报，2003，14(10)：1728 - 1739.
[2] COCASSE S，KULKARMI M . IBM WebSphere Execution Team . Understanding the WebSphere Application Server Web sever plug_in [EB/DL]. http://www - 3. ibm. com/software/webservers/appserv/doc/lates/prereq. html, 2004.
[3] 刘慧，李玉忱苏鹏、基于 J2EE 架构的分布式 Web 应用的研究[J]. 计算机应用研究，2003，20:(9)：47 - 49.
[4] 郝永芳，舒云，王德军. Web 应用环境中的系统可用性设计[J]. 计算机应用研究，2003，20:(6)：123 - 126.
[5] SADTLER C，HEYWARD J，IWAMOTO A，*et al*. IBM WebSphere Application Server V5. 0 System Management and Configuration [EB/DL]. http://www. ibm. com/redbooks, 2004.
[6] 姜昌华，朱敏，陈优广、Web 应用程序压力测试[J]、计算机应用，2003，23:(10)：75 - 77.