

基于多 Markov 链预测模型的 Web 缓存替换算法

黄学雨, 钟艳青

(江西理工大学 信息工程学院, 江西 赣州 341000)

摘 要: 为了提高 web 缓存的性能, 提出了一种基于多 Markov 链预测模型的 Web 缓存替换算法 PGDSF-AI. 首先将 Web 中具有不同浏览特征的用户分为多类, 为每一类用户建立类 Markov 链, 进一步建立多 Markov 链预测模型, 然后利用该模型对当前的用户请求预测, 进而组成预测对象集. 当缓存空间不足时, 选取键值最小且不在预测对象集中的对象替换, 通过估算对象的平均间隔时间, 避免缓存大量保留长时间没有访问的对象. 实验结果表明, 提出的算法有较好的性能.

关键词: Web 缓存; 替换算法; Markov 链; 预测模型; 浏览序列

中图分类号: TP393

文献标识码: A

文章编号: 1000-7180(2014)05-0036-05

Web Cache Replacement Algorithm Based on Multi-Markov Chains Prediction Model

HUANG Xue-yu, ZHONG Yan-qing

(School of Information Engineering, Jiangxi University of Science and Technology, Ganzhou 341000, China)

Abstract: To improve the performance of web cache, a new cache replacement algorithm named PGDSF-AI is proposed. The Algorithm is based on multi-Markov chains prediction model and characteristics of user browsing is used. Firstly, the Web users are classified according to browsing characteristics and a classified-Markov is built for a classified users, and then multi-Markov chains prediction model is constructed. Then, the prediction model is forecasted based on user access sequence and the target set of prediction is formed. When the cache has not enough space to accommodate new request, replace the key which is the smallest and does not belong to the target set of prediction. The average interval is used to avoid objects long time no access retaining in the cache. Experimental results have showed that the proposed algorithm had good performance.

Key words: web cache; replace algorithm; markov chain; prediction model; access sequence

1 引言

由于用户数量的急剧增加和现有网络速度的双重限制, 人们在获取网络信息时往往需要忍受长期的等待. Web 缓存技术通过保存用户访问过的对象而减少网络通信量和改善响应时间, 成为了简单而有效的解决方法.

缓存的性能很大程度上依赖于缓存替换算法^[1], 虽然经典算法使 Web 缓存的性能得到了一定程度的提高, 但由于缺乏预测机制缓存的性能作用

有限. 文献[2]中通过研究社交网站(SNS)中用户的访问行为模型, 提出了具有预测机制、面向 SNS 用户的缓存替换算法, 但是算法只面向 SNS 用户无法适应通用的网络环境中. 文献[1]中利用 PPM 预测模型根据上下文关系和序列匹配得到预测对象集, 该模型在预测精度与空间复杂度间难以达到平衡. 文献[3]中将预处理后会话得到的序列构建访问模式树, 并根据最小支持度进一步建立频繁访问模式树, 该方法没有考虑各个用户浏览过程中的差异性. 目前还没有一种可以适应任意网络环境表现出较好

收稿日期: 2013-06-23; 修回日期: 2013-08-17

基金项目: 国家自然科学基金资助项目(71061008)

性能的替换算法^[4]. 替换算法的优劣衡量通过多个性能标准综合评估,好的算法需要在多个性能标准间取得平衡.

文献[4]中使用 Markov 链模型描述 Web 中用户的浏览特征,进而对用户的访问进行预测,该类方法取得了较高的预测准确率. 针对已有算法的缺陷,本文充分考虑 Web 中各个用户浏览过程的差异性,将具有相同或相近浏览特征的用户用同一个模型描述,建立多 Markov 链模型^[5],利用该模型对用户的访问对象进行预测,使替换算法保留将要访问的对象. 在 GDSF 算法的基础上引入平均间隔时间,避免缓存保留大量不再具有存储价值的对象,并将新算法命名为 PGDSF-AI (Prediction Greedy Dual Size Frequency-Average Interval) 算法. 通过轨迹实验表明,PGDSF-AI 算法的多个性能指标取得了较好的效果.

2 缓存替换问题描述

缓存替换问题可以描述如下: 假设 O 是有限 Web 访问对象数据集,对任意请求对象 $d \in O$,有相应的对象大小为 s_d 和取回代价 c_d . 假定缓存总容量为 C ,当前请求序列 $R = (R_1, R_2, \dots, R_n)$ 导致缓存的一个状态序列 $(S_1, S_2, \dots, S_n)$,已使用的缓存容量为 C_{used} ,对任意的 $S_k (k = 1, 2, \dots, n)$ 有

$$S_k = \begin{cases} S_{k-1}; & \text{if } R_k \in S_{k-1} \\ S_{k-1} \cup \{R_k\}; & \text{if } R_k \notin S_{k-1}, C_{used} + S_d \leq C \\ (S_{k-1} - E_k) \cup \{R_k\}; & \text{if } R_k \notin S_{k-1}, C_{used} + S_d > C \end{cases} \quad (1)$$

式中, E_k 是移除缓存的对象的集合,且 $E_k \subset S_{k-1}$,缓存的所有状态的迁移均满足式(1).

缓存替换算法就是在所有满足上式的状态序列中,找出一个状态序列使目标函数的数值最小. 通常情况下使用一个目标函数衡量缓存中对象的价值,选择价值最小的对象从缓存中替换出去.

3 多 Markov 链预测模型

3.1 多 Markov 链模型

多 Markov 链模型根据 Web 中各个用户浏览过程表现的差异性和相似性,将 Web 上所有的用户分为多个类别,用四元组表示为: $\langle X, K, P(C), MC \rangle$. 其中, $X = \{x_1, x_2, \dots, x_n\}$ 是一个离散随机变量,每个 x_i 为一个网页,称为模型的一个状态; K 表示用户类别的数目; $C = \{c_1, c_2, \dots, c_k\}$ 表示用户

的类别,其分布函数 $P(C)$ 表示不同类别用户的概率分布; $MC = \{mc_1, mc_2, \dots, mc_k\}$ 为类 Markov 链的集合, mc_k 是描述类别为 c_k 的用户浏览特征的 Markov 链,称为类 Markov 链,其概率转移矩阵 A_k 和初始状态分布 B_k 分别表示为

$$A_k = (p_{kij}) = \begin{pmatrix} p_{k11} & \cdots & p_{k1j} & \cdots & p_{k1n} \\ \vdots & & \vdots & & \vdots \\ p_{k11} & \cdots & p_{kij} & \cdots & p_{kin} \\ \vdots & & \vdots & & \vdots \\ p_{kn1} & \cdots & p_{knj} & \cdots & p_{knn} \end{pmatrix} \quad (2)$$

$$B_k = (p_{ki}) = (p_{k1}, p_{k2}, \dots, p_{kn}) \quad (3)$$

式中, p_{kij} 表示从状态 X_i 到状态 X_j 的转移概率,即属于 c_k 类别的用户访问页面 x_i 后接下来访问页面 x_j 的概率; p_{ki} 表示状态 X_i 的初始状态概率.

3.2 多 Markov 链模型的建立

将用户的访问历史页面以时间顺序排列,获得该用户的浏览序列. 设有学习数据 $D = \{d_1, d_2, \dots, d_m\}$ 是 m 个用户的浏览序列,第 k 类用户有 m_k 个浏览序列. 利用最大似然估计计算用户的类别 C 的概率分布 $P(C = c_k) = m_k/m$. 将初始状态看成特殊的聚类结果,将每个用户的浏览序列看成一个类,为每一类别用户建立类 Markov 链,并使用式(4)和(5)计算概率转移矩阵 A_k 中的每一项 p_{kij} 和初始状态分布 B_k 中的每一维 p_{ki} 的值,从而得到 m 个类 Markov 链.

$$p_{kij} = \frac{S_{kij} + \alpha_{kij}}{\sum_{j=1}^n (S_{kij} + \alpha_{kij})} \quad (4)$$

$$p_{ki} = \frac{\sum_{j=1}^n (S_{kij} + \alpha_{kij})}{\sum_{i=1}^n \sum_{j=1}^n (S_{kij} + \alpha_{kij})} \quad (5)$$

式中, S_{kij} 表示在第 k 类所有用户浏览序列中,状态对 (x_i, x_j) 出现的次数; α_{kij} 是超级参数,取值为 β/n^2 , β 为问题域空间的大小.

同理,计算任意两个类 Markov 链 mc_k 和 mc_l 合并后的类 Markov 链 $mc_{(k+l)}$ 为

$$p_{kij} = \frac{S_{kij} + S_{lij} + \alpha_{(k+l)ij}}{\sum_{j=1}^n (S_{kij} + S_{lij} + \alpha_{(k+l)ij})} \quad (6)$$

$$p_{ki} = \frac{\sum_{j=1}^n (S_{kij} + S_{lij} + \alpha_{(k+l)ij})}{\sum_{i=1}^n \sum_{j=1}^n (S_{kij} + S_{lij} + \alpha_{(k+l)ij})} \quad (7)$$

式(6)和(7)中的所有参数与式(4)和(5)的参数

相同.

任意两个概率转移矩阵 $\mathbf{A}_k$ 和 $\mathbf{A}_l$ 的相似度定义为矩阵中行的交叉熵的平均值:

$$\text{sim}(\mathbf{A}_k, \mathbf{A}_l) = \left(\sum_{i=1}^n \sum_{j=1}^n p_{kij} \log \frac{p_{kij}}{p_{lij}} \right) / n \quad (8)$$

式中, p_{kij} 、 p_{lij} 分别表示概率转移矩阵 $\mathbf{A}_k$ 和 $\mathbf{A}_l$ 中从状态 X_i 到状态 X_j 的转移概率; n 是概率转移矩阵的行数.

类 Markov 链的动态特征由其转移矩阵描述, 两个类 Markov 链 mc_k 和 mc_l 的相似度由它们的概率转移矩阵的相似度定义:

$$\text{sim}(mc_k, mc_l) = 2 / (\text{sim}(\mathbf{A}_k, \mathbf{A}_l) + \text{sim}(\mathbf{A}_l, \mathbf{A}_k)) \quad (9)$$

M 为 Bayes 网络模型^[5], 使用模型 M 的似然函数 $P(D/M)$ 为准则函数评价聚类结果. 不断尝试合并具有较大相似度的类 Markov 链, 找到使得准则函数取极值的聚类结果, 即为最好的聚类结果^[6]. 至此, 多 Markov 链模型完成了建立的整个过程.

3.3 多 Markov 链模型的预测

多 Markov 链模型建立后, 根据当前用户的浏览过程对该用户的接下来的访问进行预测. 假设已知 Web 中用户的浏览序列 $X = (x_1, x_2, \dots, x_l)$, 为了预测用户将要访问的对象, 需要先判定该用户所属的用户类别, 再利用当前的浏览序列预测用户可能处于的状态. 其预测过程分为以下两步:

(1) 判定用户所属类别

在任意类别 c_k 中, 浏览序列 $(x_1, x_2, \dots, x_l)$ 出现的概率为

$$P(x_1, x_2, \dots, x_l | C = c_k) = p_{ki} \prod_{i=2}^l p_{k(i-1)i} \quad (10)$$

式中, p_{ki} 和 $p_{k(i-1)i}$ 分别是初始状态分布 B_k 和概率转移矩阵 $\mathbf{A}_k$ 中的项.

利用式(11) 计算当前用户属于某一类别 c_k 的概率:

$$\frac{P(C = c_k | x_1, x_2, \dots, x_l) = P(x_1, x_2, \dots, x_l | C = c_k) P(C = c_k)}{P(x_1, x_2, \dots, x_l)} \quad (11)$$

式中, $P(x_1, x_2, \dots, x_l)$ 表示序列 $(x_1, x_2, \dots, x_l)$ 的边际概率; $P(C = c_k)$ 表示类别 c_k 用户的概率分布.

如果用户满足式(12), 则将用户所属的类别判定为类别 c_k .

$$P((x_1, x_2, \dots, x_l) | C = c_k) P(C_k) = \max_{j=1,2,\dots,K} (P((x_1, x_2, \dots, x_l) | C = c_j) P(C = c_j)) \quad (12)$$

(2) 利用类 Markov 链进行预测

假设用户在 t 时刻的状态向量为 $\mathbf{H}(t) = (0, 0, \dots, 1, 0, \dots, 0)$, 对应的状态为 X_i , 即 t 时刻访问了页面对象 x_i , 则状态向量的第 i 维值记为 1, 其余各维值记为 0. 当前用户在 t 时刻的状态用一个状态概率向量 $\mathbf{V}(t) = (P(X_t = x_1), P(X_t = x_2), \dots, P(X_t = x_l))$ 表示, $\mathbf{V}(t)$ 中的每一维的值就是在 t 时刻用户处于不同状态的概率. 若经过判定后用户所属类别为 c_k , 已知 $t-1$ 时刻的状态向量 $\mathbf{H}(t-1)$, 则 t 时刻用户处于不同状态的概率为

$$\mathbf{V}(t) = \mathbf{H}(t-1) \times \mathbf{A}_k \quad (13)$$

式中, $\mathbf{A}_k$ 表示类 Markov 链 mc_k 的状态转移矩阵; $\mathbf{H}(t-1)$ 表示 $t-1$ 时刻的状态向量.

4 基于预测的缓存替换算法 PGDSF-AI

4.1 平均间隔时间

对于最近很少访问或者以后也几乎不会被访问的对象, 则认为该类对象不再具有存储价值, 若一直存在缓存中则会降低缓存的命中率. 为了解决这个缺陷, 本文使用平均访问间隔时间^[7] 评估对象接下来的访问变化, 避免缓存保留大量不再具有存储价值的对象. 假设对象在第 $k-1$ 次访问后经过 t_k 的时间被第 k 次访问, 且第 $k-1$ 次访问后已知平均访问间隔时间为 T_{k-1} , 则第 k 次访问后得到的平均间隔时间 T_k 为

$$T_k = \begin{cases} t, & \text{if } k = 1 \\ \lambda t_k + (1 - \lambda) T_{k-1}, & \text{if } k > 1 \end{cases} \quad (14)$$

式中, λ 为调节参数, 取介于 0.5 和 1 之间的值; t 为当前时间与对象第一次访问的间隔时间.

平均间隔时间 T_k 根据历史访问的平均间隔时间, 考虑对象过去的访问频繁程度, 同时, 通过计算最近一次访问后的时间流逝, 充分考虑 Web 对象访问的时间局部性. 刚被访问过的对象以及经常被访问的对象, 对应的平均间隔时间也越小; 对于长时间没有被访问的对象, 由于其流逝时间 t_k 很大, 则其平均间隔时间也较大.

4.2 缓存替换算法 PGDSF-AI

本文提出的 PGDSF-AI 算法通过训练建立多 Markov 链预测模型, 采用 GDSF 算法作为原型计算对象的键值大小, 使用键值函数作为缓存替换问题描述中的目标函数用来衡量缓存中对象的价值, 充分考虑对象的流行度、大小、取回代价等因素对缓存性能的影响, 引入平均间隔时间评估对象访问的变化, 使用式(15) 计算缓存中对象的键值大小.

$$H = M + (f_d / T_d) \times (c_d / s_d) \quad (15)$$

式中, M 表示膨胀因子; f_d 表示对象 d 的流行度; T_d 表示对象 d 访问 f_d 次后的平均间隔时间; c_d 表示取回对象 d 所花费的代价; s_d 表示对象 d 的大小。

当用户的请求到来时,如果缓存中不存在请求的对象且没有足够空间保存该对象,PGDSF-AI 算法将缓存中所有对象按键值大小进行排序.利用多 Markov 链预测模型根据当前用户的请求序列给用户分为类别 c_k ,用描述该类别用户浏览特征的类 Markov 链对用户可能处于的状态进行预测.为了控制预测的对象数目和准确性,设定概率阈值 θ ,获取大于该概率阈值的状态对象,将其组成用户极有可能访问的预测对象集 PSet.如果键值最小的对象出现在 PSet 中,则认为该对象接下来会被访问,选择下一个对象替换出缓存直到缓存有足够空间存放请求对象.PGDSF-AI 算法描述如算法 1 所示.

算法 1: PGDSF-AI 缓存替换算法

输入: 学习数据 D ; 当前请求对象 d ; 概率阈值 θ ;

输出: 缓存对象;

步骤 1: 初始状态时将 D 中每个用户的浏览序列看成一个类,利用式(4)和(5)计算概率转移矩阵 A_k 和初始状态分布 B_k ,为每一类用户建立类 Markov 链;

步骤 2: 用式(8)和(9)计算任意两个类 Markov 链之间的相似度,并按相似度大小排成队列 Q ;

步骤 3: 利用似然函数 $P(D/M)$ 计算准则函数值 P ,利用式(6)和(7)合并 Q 中的两个类 Markov,直到不存在使 P 增大的合并;

步骤 4: If(d 在缓存中时) 更新缓存中 d 的 T_d 和 f_d ;

Else

While(缓存空间不足存放 d)

For(缓存中所有的对象)

利用式(13)计算缓存中所有对象的键值,并按键值从小到大排成 H ;

步骤 5: 利用式(10)~(12)判定当前用户的类别 c_k ,用式(13)对用户访问进行预测,得到状态概率向量 $V(t)$;

For $V(t)$ 中的每一维值 $V(t)[j]$

If ($V(t)[j] > \theta$)

预测对象集 $PSet = PSet + V(t)[j]$;

步骤 6: For $k = 0$ To length(H)

If ($H[k]$ 对应的对象不属于 $PSet$)

剔除 $H[k]$ 对应的对象;

If(缓存空间足够存放 d)

将 d 放入缓存中;

break;

5 仿真实验与分析

5.1 实验设置

实验使用轨迹驱动的方法对算法性能比较和评估,数据集使用 DEC^[8] 和 NLNR^[7] 两个数据集,DEC 数据集记录了 Squid 代理服务器处理的从 1996.9.1 到 1996.9.22 的 3 543 968 个请求,NLNR 包含了 NLNR 中四个主要的代理服务器于 1997.12.22 为期一天的 1 766 409 个请求,实验前先对数据集中的数据进行预处理.

为了验证本文提出的缓存替换算法的性能,实验在 Linux 系统平台下使用 C 语言编写的缓存模拟器对替换算法仿真.引入的平均间隔时间中 λ 参数设为 0.6,即充分考虑对象访问的时间局部性,对最近一次访问后的流逝时间赋予更大的权重,同时考虑过去的平均间隔时间.选取数据集的 3/4 作为训练集建立多 Markov 链预测模型,其余 1/4 数据集对用户状态对象预测.设定概率阈值 $\theta=0.3$ 的条件下,缓存模拟器分别按照缓存容量占总访问规模的 0.05%~20%变化范围运行.

5.2 实验结果与分析

为了验证算法的性能,本文主要从命中率(Hit Rate,HR)和字节命中率(Byte Hit Rate,BHR)两个性能指标上将提出的算法分别与 LRU、SIZE、GD-Size、GDSF 和 LH-MLRU^[2] 算法的实验结果进行对比,实验结果分别如图 1~4 所示.

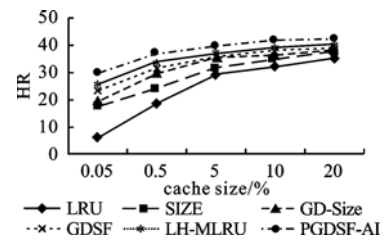


图 1 DEC 数据集上 HR 与缓存大小的关系

从图 1~图 4 可以看出,在命中率(HR)方面 PGDSF-AI 算法始终优于其他所有算法,当缓存大小占总访问对象总大小的 0.05%、0.5% 和 5% 时,与 LH-MLRU 算法相比,在 DEC 数据集上能提高 4%~5% 左右,在 NLNR 数据集上提高约 2%~4%.当缓存大小占总访问对象总大小的 10% 和 20% 时,与其他算法相比有所减少.这种现象产生的

原因有两点:一是 PGDSF-AI 算法利用多 Markov 预测模型对用户的浏览进行预测产生了预测对象集,在当缓存容量不足需要替换缓存中对象时,需要先判断该对象是否在预测对象集中,因此接下来可能被访问的对象可以保留在缓存中而不被替换出去;二是缓存保护平均间隔时间小的对象,使其获得较高的键值,避免缓存过多保留流行度大且长时间不被访问的对象.但当缓存容量不断增大时,缓存有足够的容量保存新的对象,算法的优势也在减弱.

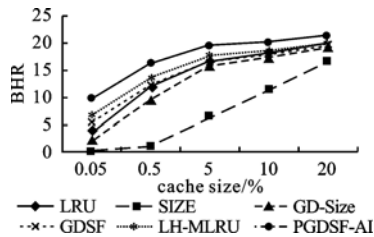


图 2 DEC 数据集上 BHR 与缓存大小的关系

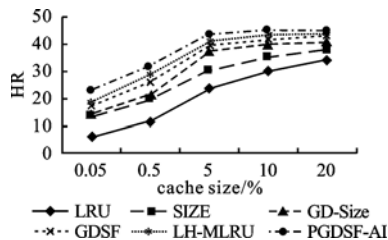


图 3 DEC 数据集上 HR 与缓存大小的关系

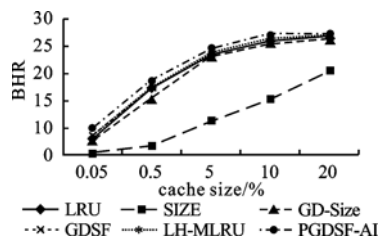


图 4 NLANR 数据集上 BHR 与缓存大小的关系

在字节命中率(BHR)方面,PGDSF-AI 算法在数据集 DEC 和 NLANR 上均优于其他算法,在数据集 DEC 上与 LH-MLRU 算法相比能提高 1%~4%左右,在数据集 NLANR 上略占优势. LH-MLRU 算法在两个数据集上均略微优于 LRU、SIZE、GD-Size 和 GDSF 算法,SIZE 算法的性能表现最不理想.这是因为 PGDSF-AI 算法综合考虑了对象的各个因素,并保护平均间隔时间小的对象,使得经常被访问的对象尽量保留在缓存中. LH-MLRU 算法由于灵活的设置参数使算法在多个性能上达到最

优,而 SIZE 算法每次替换占用空间最大的对象,导致小的对象长时间保存在缓存中.

6 结束语

本文基于多 Markov 链模型使用不同的类 Markov 链描述 Web 中具有不同浏览特征的用户,并对用户的访问对象进行预测,提出了具有预测机制的 Web 缓存替换算法 PGDSF-AI. 经过预测使缓存中保留最近可能会被访问的对象,进一步改善了缓存的性能. 通过在键值计算中引入平均间隔时间,可以快速反映对象的访问变化,从而有效地避免缓存过多地保留长时间没有访问的对象. 通过实验表明 PGDSF-AI 算法与其他算法相比在多个性能指标上均具有较好的效果.

参考文献:

- [1] 石磊,孟彩霞,韩英杰. 基于预测的 Web 缓存替换策略[J]. 计算机应用,2007,27(8):1842-1845.
- [2] Lin Rongheng, Zhang Hui, Zou Hua. Web replacement policy based on user requests for SNS[J]. Journal of Beijing University of Posts and Telecommunications, 2012,35(1):111-114.
- [3] Han Xiangchun, Tian Yugen. Web cache replacement algorithm based on prediction[J]. Computer Engineering and Design, 2010, 31(1):110-113.
- [4] 陈佳,吴军华. 基于混合 Markov 模型的用户浏览预测[J]. 计算机工程与设计,2009, 30(4):903-905.
- [5] Xing YongKang, Ma ShaoPing. Modeling user navigation sequences based on multi-markov chain[J]. Chinese Journal of Computers, 2003, 26(11):1510-1517.
- [6] 林文龙,刘业政,姜元春. Web 浏览预测的 Markov 模型综述[J]. 计算机科学,2008, 35(1):9-14.
- [7] 林永旺,张大江,钱华林. Web 缓存的一种新的替换算法[J]. 软件学报,2001,12(11): 1710-1715.
- [8] Tom M. Kroeger. Digital's web proxy traces [EB/OL]. [2013-06-15]. <http://apotheca.hpl.hp.com/ftp/pub/compaq/traces/proxy/webtraces.html>.

作者简介:

黄学雨 男,(1970—),硕士,副教授. 研究方向为企业信息系统、视觉工程技术.

钟艳青 女,(1989—),硕士研究生. 研究方向为企业信息系统.