第27卷第1期 杭州电子科技大学学报 Vol.27,No.1
2007年02月 Journal of Hangzhou Dianzi University Feb.2007

网络中代理服务器最优放置算法

黄孙琴,陈光亭,杨建芳
(杭州电子科技大学理学院,浙江 杭州 310018)

摘要:该文研究在发生数据读取和更新的网络中,代理服务器个数不限的情况下,如何放置多少个代理服务器使得网络中数据访问的总花费最小的问题,包括数据读取和更新,即最优放置代理服务器问题。利用动态规划的方法在时间内给出该问题的最优解。

关键词:代理服务器;最优放置;动态规划;算法
中图分类号:O221.7 **文献标识码**:A **文章编号**:1001-9146(2007)01-0081-03

0 引 言

随着互联网络的迅速发展,使用网络的人越来越多,致使大多数网站遭受网络堵塞的困扰,而用户在使用网络时对网络的浏览速度和效果却愈加重视。如何才能让各地的用户都能够进行高质量的访问,并尽量减少由此而产生的费用。Caching作为解决网络堵塞、带宽瓶颈,减少用户访问延迟的一项重要技术,已为大家广泛采用。代理服务器是Caching的一种类型,它存储了总服务器的全部(或部分)数据,相当于总服务器的一个拷贝。当用户发送请求到总服务器,如果在传送的路径上遇到了存储该部分请求数据的代理服务器,那么这个代理服务器接受该用户的请求,并把请求的数据发送给此用户。放置代理服务器的目的就在于减少用户的访问延迟,均衡服务负载,降低网络带宽消耗。本文讨论的就是代理服务器的放置问题。网站的日常维护靠服务器定期更新数据,有时甚至需要高频率的更新,如占线服务,占线拍卖等。在这种情况下,在网络中放置代理服务器,可以减少客户读取数据的时间,但却增加了代理服务器和总服务器保持数据同步的管理费用。本文研究在代理服务器个数不限的情况下,如何放置多少个代理服务器使得网络中数据访问的总花费(包括数据读取和更新)最小。

1 问题描述

给定一个网络 $G(V,E)$,其中V代表节点的集合,E代表网络链路的集合,对于链路$(v,u)\in E$,$d(v,u)$表示链路的距离。令S表示Web总服务器。$w(v)$表示客户节点访问S的频率,是在一段特定时间内访问服务器的数据量,已把读取的数据对象的大小考虑在内。μ表示总服务器S的更新频率,是在一段特定时间内更新的操作数,已考虑更新数据对象的大小在内。研究的问题是:在网络中选取一组代理服务器P,$P\subseteq V$,并确定放置代理服务器的位置,使得访问S的总成本 $C(T_s,P)$最小(包括数据的读取和更新)。

由于在网络中数据的传输是采用最短路的方法,所以先对原问题描述中的条件进行一点改进,T_s表示以S为根的树,整个网络转换为树结构T_s。令P_v表示从v到S的路径上客户v遇到的第一个代理服务器,P_v可能为v本身或者S。

收稿日期:2006-09-22
基金项目:国家自然科学基金资助项目(10371028),浙江省教育厅科研项目(20050494)
作者简介:黄孙琴(1981-),女,浙江余杭人,在读研究生,组合优化.

定义数据的读取成本为数据量乘以传输的距离，则客户 v 的读取成本为 $w(v)d(v,p_v)$，那么 T_s 上所有客户读取 S 的成本为 $\sum_{v\in T_s} w(v)d(v,p_v)$。

上载数据对象到代理服务器或更新代理服务器上的数据对象也有成本。假定上载到或更新代理服务器的数据采用组播方式，组播机制使用最短路径树传送数据到 P。令 SPT(S,P)表示以 S 为树根到所有代理服务器 P 的最短路径树，则数据更新的费用 $\mu_{(x,y\in SPT(S,P))} d(x,y)$。那么 T_s 上数据访问的总成本为：

$$C(T_s,P)=\sum_{v\in T_s} w(v)d(v,p_v)+\mu_{(x,y\in SPT(S,P))} d(x,y) \tag{1}$$

问题就是找出 P，即在 T_s 上放置多少个代理服务器及它们所在的位置，使得式 1 最小。

2 放置问题的解决方案

为有效地设计算法，首先将树 T_s“等效”地转换为二叉树 BT_s。那么原问题就等价地转换为在二叉树上寻找 P，使得式 1 最小。已知在最优放置中，代理服务器不可能放置在任何一个虚拟节点处。把代理服务器放置在该虚拟节点向根节点过程中遇到的第一个非虚拟节点处可以得到更小的目标值。

不失一般性，可将原树状图直接看成是二叉树。以下将从叶子节点出发向上到根节点利用动态规划的思想给出算法[1,2]。

对于 V 中的每一节点 v，令 T_v 表示 v 以及 v 的所有后代节点生成的子树。记 $C(v,u)$是在 T_v 上适当放置代理服务器后得到的最小花费，u 是 v 到 S 的路径上 v 遇到的第一个代理服务器，即 $C(v,u)=\min_{P\subseteq T_v} C(v,u,P)$。$C_0(v,u)$表示 T_v 中没有放置代理服务器时数据访问的总花费。$C^+(v,u)$表示在 T_v 中放置了一些代理服务器后数据访问的总花费。问题就是求 $C(S,S)$。

定理 1 给定 T 中一个节点 v，对于从 v 到 S 的路径上的任意节点 u，$C(v,u)$可以由以下规则计算：

$$C(v,u)=\min\{C^0(v,u),C^+(v,u)\} \tag{2}$$

(1) 如果 v 是叶子：

$$\begin{cases} C^0(v,u)=w(v)d(v,u) \\ C^+(v,u)=\mu d(v,u) \end{cases} \tag{3}$$

(2) 如果 v 不是叶子，那么 v 有两个子节点 v_l 和 v_r：

$$\begin{aligned} C^0(v,u)&=\sum_{x\in T_v} w(x)d(x,u) \\ &=C^0(v_l,u)+C^0(v_r,u)+w(v)d(v,u) \end{aligned}$$

$C^+(v,u)=\min\{\alpha_1,\alpha_2,\alpha_3,\alpha_4\}$，其中：

$$\begin{cases} \alpha_1=C(v_l,v)+C(v_r,v)+\mu d(v,u); \\ \alpha_2=C^+(v_l,u)+C^0(v_r,u)+w(v)d(v,u); \\ \alpha_3=C^0(v_l,u)+C^+(v_r,u)+w(v)d(v,u); \\ \alpha_4=C^+(v_l,u)+C^+(v_r,u)+w(v)d(v,u). \end{cases} \tag{4}$$

证明 现利用递归的方法证明该定理。

如果 v 是叶子，$C^0(v,u)$就表示在点 v 不放置代理服务器，那么 v 就要把请求发送到 u，就产生一个读取数据的费用，所以 $C^0(v,u)=w(v)d(v,u)$。$C^+(v,u)$表示在点 v 放了代理服务器，显然 v 不用向外发送请求，但 v 这个代理服务器的数据更新需要费用 $\mu d(v,u)$，所以 $C^+(v,u)=\mu d(v,u)$。

如果 v 是个一般的非叶子节点，那么它就有两个子节点 v_l 和 v_r。由归纳假设，$C(v_l,u)$，$C(v_r,u)$已经算得，即已知 $C^0(v_l,u)$，$C^+(v_l,u)$，$C^0(v_r,u)$，$C^+(v_r,u)$。$C^0(v,u)$表示 T_v 中不放置代理服务器，那么 T_v 中所有点都要向 u 发送请求，数据访问的费用为 $C^0(v,u)=\sum_{x\in T_v} w(x)d(x,u)=C^0(v_l,u)+C^0(v_r,u)+w(v)d(v,u)$。

对于 $C^+(v,u)$根据 v 是否放置代理服务器,分两种情况:

(1)在 v 处放置代理服务器,那么 T_{v_l}和 T_{v_r}上点的请求就发送到 v 处,且 v 有数据更新的费用,即 $C^+(v,u)=C(v_l,v)+C(v_r,v)+\mu d(v,u)$;

(2)在 v 处不放置代理服务器,那么 T_{v_l}和 T_{v_r}上的点以及 v 的请求就发送到 u 处,因为 T_v 中须放有一些代理服务器,而代理服务器可能放在 T_{v_l}或 T_{v_r}上,或两棵子树上都有。

可细分为以下 3 种情况:

$$C^+(v,u)=C^+(v_l,u)+C^0(v_r,u)+w(v)d(v,u);$$
$$C^+(v,u)=C^0(v_l,u)+C^+(v_r,u)+w(v)d(v,u);$$
$$C^+(v,u)=C^+(v_l,u)+C^+(v_r,u)+w(v)d(v,u);$$

故 $C^+(v,u)=\min\{\alpha_1,\alpha_2,\alpha_3,\alpha_4\}$。

现在可以针对每一个节点 v,每一个 v 到 S 的路径上的节点 u,来计算 $C(v,u)$,顺序是从叶子到根节点。在计算的过程中需要记录每个算得的 $C(v,u)$,$C^0(v,u)$,$C^+(v,u)$以及放置的代理服务器的个数和位置。

定理 2　利用上述算法可在 $O(n^2)$时间内得到问题的最优解。

证明　用预处理的方法计算任意两节点间的距离 $d(v,u)$并存储这个邻接矩阵,需 $O(n^2/2)$的时间。后面的计算可在 $O(1)$时间内直接调用 $d(v,u)$。设|V| = n,那么叶子节点 v 有$\lfloor n/2\rfloor+1$个。当 v 是叶子的时候,$C^0(v,u)$可在 $O((\lfloor n/2\rfloor+1)n)$,即 $O(n^2/2)$的时间内计算得到。同样 $C^+(v,u)$也是 $O(n^2/2)$可得。存储这些数据,$C(v,u)$在 $O(n^2/2)$时间内可得。当 v 不是叶子的时候,计算 $C^0(v,u)$需做 1 次乘法 2 次加法,$O(3n^2/2)$可得。$C^+(v,u)$需做 4 次乘法 8 次加法 3 次比较,$O(15n^2/2)$可得。存储这些数据,$C(v,u)$可在 $O(n^2/2)$时间内得到。所以上述算法可在 $O(n^2)$时间内得到问题的最优解。

4　结束语

本文讨论了在发生数据读取和更新的网络中,代理服务器个数不限的情况下,如何放置多少个代理服务器使得网络中数据访问的总花费最小的问题,并给出了 $O(n^2)$时间的动态规划算法。

参考文献

[1] Tamir A. An O(pn2) algorithm for the p－median and related problems on tree graphs[J]. Operations Research Letters, 1996, 19(2):59－64.

[2] Jia Xiaohua, Li Deying, Hu Xiaodong, etal. Optimal placement of Web proxies for replicated Web servers in the Internet[J]. The Computer Journal, 2001, 44(5):378－390.

A Placement Algorithm of Web Proxy in the Internet

HUANG Sun-qin, CHEN Guang-ting, YANG Jian-fang

(*School of Science, Hangzhou Dianzi University, Hangzhou Zhejiang* 310018, *China*)

Abstract: With the consideration of both read and update operations to the data on the internet, we study the problem of optimal placement of proxies. For an unconstrained number of proxies, find the optimal number of proxies and their placement, such that the overall access cost is minimized. Using a dynamic programming method can solve it in time.

Key words: proxy; optimal placement; dynamic programming; algorithm