MODELING THE WEB AS A FOREST OF TREES

Fathi Tenzakhti
Computer Science Department, University of Nizwa, Oman

Keywords: Internet, Web, Replication, Tree model, Response time, Dynamic Programming.

Abstract: This study tries to demonstrate that the World Wide Web (the Web for short) could be modeled as a forest of trees. Each Web site has its own tree for which it is the root. Since trees were extensively studied in the literature, many problems related to performance, fault-tolerance and availability of the Web could be understood more easily and the existing body of knowledge about trees could be applied to solve these problems.

1 INTRODUCTION

The World Wide Web (Web for short) is a very important source of information, and Web services are being promoted as the next generation technology for Business to Business (B2B) and Business to Consumer (B2C) E-commerce. Unfortunately, the Web suffers from some drawbacks from which we state slow response time and high rate of unavailability especially in the EAST. Solving these problems in an efficient and simple way requires understanding the topology of the Web and how information travels from a Web site to its clients.

In this study, we try to show that it is safe and sound to model the Web as a forest of trees. Each Web site is the root of a tree along which information from the Web site to its clients travels. The argument is that in a given period of time θ (around 6 hours), each Web site has a set of clients accessing it. Given that the Internet routes are stable (V. Paxon, 1997), the information traveling from the Web site to its clients take the same root during this period of time θ .

Consequently, each Web site is the root of a tree whose leafs are the clients of this Web site during the period θ . The internal nodes of the tree are intermediary nodes that could be clients to the Web site. Information from the Web site to its clients travels along the path of this tree. Since the Web is a collection of Web sites, it is safe to say that the Web could be modeled as a forest of trees. Each Web site being the root of its tree.

In the rest of the paper, section 2 presents studies that have assumed that the Web is modeled as a forest of trees. Section 3 presents the organization of the Web and tries to show that the tree modeling is safe and accurately describes how the information flows between the Web site and its clients. Section 4 presents the benefits of such modeling and how it affects the way the replica placement problem in the Web is solved. Finally section 5 concludes the paper.

2 RELATED WORK

To the best of my knowledge, the first study that has assumed the Web as a set of trees is the one in (B.Li et al., 1997) The authors assumed the Web as a forest of trees (the tree model for short); each tree is rooted at the target Web server. This tree model reduced the problem of placing Web proxies to placing copies on a tree and helped optimize a performance measure for the target Web server subject to system resources and traffic pattern. Specifically, the study was interested in finding the optimal placement of multiple web proxies (M) among potential sites (N) under a give traffic pattern.

The study in (F. Tenzakhti, 2006) has assumed that the routes along witch information flows from Web servers to clients form a tree rooted at the Web server. It then considered the problem of finding a minimum cost residence set that optimizes the cost of servicing access requests in a read-only environment taking into account the capacity constraint of the links.

Tenzakhti F. (2008).
MODELING THE WEB AS A FOREST OF TREES.
In *Proceedings of the Fourth International Conference on Web Information Systems and Technologies*, pages 216-219
DOI: 10.5220/0001513302160219

3 WWW AS A FOREST OF TREES

The tree model that we advocate for the Web depends on the stability of the Internet routing methods. If the routes on the Internet are stable, the routes used by clients to access the Web server will form a shortest path tree (or routing tree) rooted at the server. Existing studies in (V. Paxon, 1997) have pointed out that in practice most routes in the Internet are stable. It has been found that 80% of routes change at a frequency lower than once a day. (Krishnan et al, 2000) have traced the routes from Bell Lab's Web server to 13,533 destinations. They have found that almost 93% of the routes are stable during their experiments. Therefore the stability of Internet routing is a realistic assumption that can reduce the Web arbitrary topology to a forest of trees.

Given the stability of Internet routing (V. Paxon, 1997) , an object requested by a client c and located at server s travels through a path $s \to r_1 \to r_2 \ldots \to r_n \to c$, called a *preference path* and is denoted by $\pi(s,c)$. The preference path consists of a sequence of nodes with the corresponding routers. Routes from s to the various clients form a routing tree along which requests are propagated. Consequently, for each server s, a tree T_s rooted at s could be constructed to depict the routing tree, and the entire Web could be represented as a collection of such routing trees, each routed at a given Web server. Formally, a routing tree is the union of the preference paths.

Each server s knows the preference path from itself to any client c. This information can be extracted and periodically refreshed from the routing database kept by the routers (Anne Benoit et al, 2006). The routing information allows the comparison of network distances (e.g. number of hops) among servers within a given platform. A user issues one request at a time for a Web page, which is fetched to the user as a single unit.

In the tree T_s, when a client c sends a request to access a server s, the request is always sent to the root along the preference path. If the server is replicated, the request meets a replica on its way and the requested object is available, it is served by the replica. Otherwise it has to travel all the way to the root where it is serviced by s. Note that if there is a replica closer to the client c but not enroot between c and s, it is ignored.

According to this tree model, the network topology is thus represented by a graph $G = (V, E)$ where $m = |V|$ is the number of nodes and E is the set of edges and represent physical links connecting these nodes. Nodes are routers, Web servers or a combination of both (servers provide the information a client is looking for). Routers are connected via wide-area links to form the communication network. Some routers, called gateways, provide connections to the outside Internet. These are the gateways through which all requests enter the system.

4 BENEFITS OF TREE MODELLING

Replication is a technique of storing copies of shared objects on servers where they are frequently accessed. It is used to address the scalability problem of popular sites (Anne Benoit et al, 2006). Replication improves efficiency by allowing operations to use local replicas instead of remote ones (Anne Benoit et al, 2006, F. Tenzakhti et al, 2003).

The replica placement problem deals with many issues. It tries to find how many replicas are needed in the replicated system, where to place these replicas, how to route requests to the appropriate replica etc… (F. Tenzakhti et al, 2003**,** B.Li et al., 1997). In this study, we are mainly interested in where to place a given number of replicas. Along the study, we propose to use the word proxy to mean a replica of the whole site. The proxies discussed in this paper are transparent proxies. They are located along the routes from clients to a Web server and are transparent to the clients. A proper placement of proxies would lead most client requests to be served at proxies, without letting them travel further to the server. Since the access patterns of clients and the sizes of the trees are different, the allocation and placement of proxies have significant impact on the overall system performance. To formally define our problem, we introduce the following notations. Let $d(u,v)$ be the distance between any two vertices u and v in the tree graph, T_s $d(u,v)$ is equal to the length of the shortest path $\pi(u,v)$.

$$d(u,v) = \sum_{(x,y)\in\pi(u,v)} d(x,y) \qquad (1)$$

Let $p(v,s)$ be the first proxy met while traveling

from v to s in T_s. This will be referred to as the *optimal* proxy. This could be v itself if v is a proxy or s if no proxy is met on the way to the root server. Let $f(v)$ be the access frequency from client v to server s during a period of time θ and $\beta(v)$ the load that node v imposes on the proxy $p(v,s)$. If P is the replication scheme (the set of proxies for the tree T_s associated with the $p(v,s)$ function), then the total distance to access the proxies is $d(P) = \sum_{v \in T_s} d(v, p(v,s))$ and the total cost of accessing the data is given by:

$$C(T_s, P) = \sum_{v \in T_s} f(v) d(v, p(v,s)) \quad (2)$$

Any node v whose optimal proxy is $u = p(v,s)$ imposes a load $\beta(v)$ on u. As a constraint, the set of nodes whose optimal proxy is u should not impose a load that is greater than the capacity κ_u of u. Consequently, if $P_u = \{x \in T_s : p(x,s) = u\}$, then the following equality must be satisfied $\sum_{P_u} \beta(x) \le \kappa_u$. Now, for a fixed number of proxies $k \ge 1$, let us find the optimal replication scheme P that minimizes the total access cost $C(T_s, P, \kappa)$ over the tree T_s, taking into consideration the capacity constraints κ of the proxies (κ being a vector storing the node capacities). The problem thus reduces to finding:

$$C(T_s, k, \kappa) = \min_{P \subseteq T_s, |P|=k} \{C(T_s, P, \kappa)\}, \quad (3)$$

$$\text{subject to } \sum_{P_u} \beta(x) \le \kappa_u, \forall u \in P \quad (4)$$

We will use dynamic programming to compute the above recurrence, and therefore find the optimal *k*-replication scheme. Consider tree T_s rooted at s with a set V of vertices. Assume that the children of each non-leaf vertex are ordered from left-to-right so that given any two siblings u and v, we are able to determine that u is to the left of v or vice versa. For $v \in T_s$, let T_v be the subtree of T_s rooted at v. For any $u \in T_v$, we can partition T_v into 3 subtrees:

1) Subtree $L_{v,u}$ containing all nodes to the left of u.
2) Subtree containing all nodes in T_u.
3) Subtree $R_{v,u.}$ containing the rest of the nodes.

Formally, we write:

- $L_{v,u} = \{x : x \in T_v : x \text{ is to the left of } u\}$
- $T_u =$ subtree of T_v rooted at u
- $R_{v,u} = \{x : x \in T_v, x \notin T_u \cup L_{u,v}\}$.

The central issue here is to divide the problem into small-scale sub problems. For this reason, we need to further partition $R_{v,u}$ into smaller subtrees. For any $u' \in R_{v,u}$, we introduce

$$L_{v,u,u'} = \{y : y \in R_{v,u} \text{ and } y \text{ is to the left of } u'\}$$

Given the recursive nature of the solution, equation 3 applied to T_v yields

$$C(T_v, k, \kappa_v) = \min_{P \subseteq T_v, |P|=k} \{C(T_v, P)\} \quad (5)$$

and

$$\sum_{P_u} \beta(x) \le \kappa_u, \forall u \in P \quad (6)$$

where $C(T_v, k, \kappa_v)$ is the minimum access cost obtained by placing k proxies in T_v given the load capacity κ_u of each node $u \in T_v$. When $k = 1$, the only proxy is always placed at root v. When $k > 1$, we can always find a node u, $u \in T_v$ and $u \ne v$ which satisfies:

1. A proxy is placed at u;
2. No proxy is placed in $L_{v,u}$;
3. No proxy is placed in $\pi(u,v) - \{u,v\}$. Assuming T_v is partitioned at the node u, and that k' proxies are placed in T_u, $1 \le k' \le k-1$, then $k - k'$ proxies are placed in $R_{v,u}$.

For all k proxies, we need to find all possible partitioning points $u \in T_v$ and all possible values k'. Recursively, the proxies are allocated to T_u and $R_{v,u}$ the same way as in T_v. The dynamic programming approach can thus be formulated by the equations (8).

We can therefore write

$$C(T_v,k,\kappa_v)=\ell(L_{v,u},\kappa_v)+C(T_u,k',\kappa_u)+C(R_{v,u},k-k',\kappa'_v)\text{ , where }\kappa'_v=\kappa_v-\sum_{x\in L_{v,u}}\beta(x) \quad (7)$$

$$C(T_v,k,\kappa_v)=\begin{cases}\sum_{x\in T_v} f(x)d(x,v) & \text{if } \sum_{x\in T_v}\beta(x)\le\kappa_v \text{ and } k=1\\ \min_{u\in T_v,1\le k'\le k}\{\ell(L_{v,u},\kappa_v)+C(T_u,k',\kappa_u)+C(R_{v,u},k-k',\kappa'_v)\} & \text{if } \sum_{y\in L_{v,u}}\beta(y)\le\kappa_v \text{ and } k>1\\ \text{Undefined Otherwise}\end{cases} \quad (8)$$

In equation (8), $\ell(L_{v,u},\kappa_v)$ is a constant that could be undefined if the total load of the nodes in $L_{v,u}$ is higher then the capacity κ_v of the node v. $C(T_u,k',\kappa_u)$ is recursively defined in T_v with a capacity constraint κ_u of node u and κ_x of all $x\in T_u$. $R_{v,u}$ is further partitioned into $L_{v,u,u'}$, $T_{u'}$ and $R_{v,u'}$ around the node u' where a proxy is put. The capacity constraint of $R_{v,u}$ with respect to proxy v is the remaining capacity κ'_v obtained by subtracting from the capacity κ_v of v the total loads imposed on v by the nodes in $L_{v,u}$.

5 CONCLUSIONS

In this study, we have showed that it is safe and sound to assume that the Web is a forest of trees. Each tree is rooted at a Web site. The leaves of the tree are the client of the Web site during a given period of time θ and the interior nodes are either clients or routers used to route the requests and information from and to the client. With this simple topology, many of the problems related to the performance and availability of the Web could be studied easily and simple algorithms based on the tree topology could be easily developed to solve these Web problems. In this study, we have shown how and algorithm for replication in the Web for performance could be easily developed once the tree structure is assumed.

REFERENCES

Anne Benoit, V. Rehn, and Y. Robert., 2006. Impact of QoS on Replica Placement in Tree Networks. *Research Report 2006-48*, LIP, ENS Lyon, and France. Available at graal.ens-lyon.fr/~yrobert/.

B. Li, M.J. Golin, F. Italiano, X. Deng, K. Sohraby., 1999. On the Optimal Placement of Web Proxies in the Internet, *Proc. IEEE INFOCOM*, pp. 1282-1290.

F. Tenzakhti, 2006. Optimal Placement of Web Proxies in the Internet with Link capacity constraints, *Journal of Digital Information Management*, Vol.4. No. 4.

F. Tenzakhti, M. Ould Khaoua, K. Day, 2003. On the Availability of Replicated Content in the Web. *International Journal on computing and information Sciences*, Vol 1. No.1, pp. 51-60.

P. Krishnan, D. Raz, and Y. Shavitt, 2000. The cache Location Problem, IEEE/ACM Transactions on Networking, vol. 8, no. 5, pp. 568-582.

V. Paxson, 1997 End-to-End Routing Behavior in the Internet, *IEEE/ACM Transactions Networking*, vol. 5, no. 5, pp. 601-615.