HAL
archives-ouvertes.fr

Robustness of a routing tree for the Push Tree Problem

Frédéric Havet

▶ To cite this version:

Frédéric Havet. Robustness of a routing tree for the Push Tree Problem. RR-4464, INRIA. 2002. inria-00072124

HAL Id: inria-00072124
https://hal.inria.fr/inria-00072124

Submitted on 23 May 2006

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

INRIA

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Robustness of a routing tree for the Push Tree Problem

Frédéric Havet

N° 4464

May 2002

THÈME 1

ISSN 0249-6399 ISRN INRIA/RR--4464--FR+ENG

INRIA
SOPHIA ANTIPOLIS

Robustness of a routing tree for the Push Tree Problem

Frédéric Havet

Thème 1 — Réseaux et systèmes
Projet Mascotte

Rapport de recherche n° 4464 — May 2002 — 8 pages

Abstract: The Push Tree problem contains elements from both the Steiner Tree and Shortest Path problem. It deals with the trade-offs between the push and pull mechanism used in information distribution and retrieval. In [HW01], a two step approach for the Push Tree Problem was proposed. In the first step, a "good" spanning tree (called routing tree) is constructed and then the problem is solved in this particular tree. Finding a routing tree is NP-hard but the second step may be performed easily, thus the idea is to use the routing tree as a (semi-) stable infrastructure and to perform adaptations to changing information patterns inside the routing tree.

In this paper, we study the robustness of a routing tree when the information requests disappear at some nodes.

Key-words: Data replication, network optimisation, robustness

Unité de recherche INRIA Sophia Antipolis
2004, route des Lucioles, BP 93, 06902 Sophia Antipolis Cedex (France)
Téléphone : +33 4 92 38 77 77 — Télécopie : +33 4 92 38 77 65

Robustnesse d'un arbre de routage pour le problème de l'arbre savant

Résumé : Le problème de l'arbre savant contient des éléments à la fois du problème de l'arbre de Steiner et du problème du plus court chemin. Il modélise le compromis entre les deux mécanismes de distribution de l'information consistant à pousser ou à tirer l'information. Dans [HW01], est proposée une approche en deux étapes pour le problème de l'arbre savant. Tout d'abord, un "bon" arbre couvrant (appelé arbre de routage) est construit et le problème est ensuite résolu à l'intérieur de cet arbre. Trouver un arbre de routage est NP-difficile mais la seconde étape s'effectue facilement. Ainsi une idée est d'utiliser l'arbre de routage comme infrastructure (semi-) stable et d'opérer les adaptations à l'intérieur de celui-ci.

Dans ce rapport, nous étudions la robustesse d'un arbre de routage lors de la disparition de requêtes en certains nœuds du réseau.

Mots-clés : Replication de données, optimisation dans les réseaux, robustesse

1 Introduction

Information distribution occurs in many different contexts, but the ways in which distribution is achieved tend to be variations of only two basic mechanisms: push and pull. Push is a proactive mechanism: a source of information sends the information to (or in the direction of) points where requests for that information possibly will occur in the near future. Pull is a reactive mechanism: when a request occurs, a message is sent to the information source, and the source replies by sending the information to the request point.

The push mechanism is particularly useful if the information is static. In that case, the information has to be sent to all possible request points only once, and all future requests can then be handled locally. If the information changes, however, requests will result in information that is out of date. To ensure that this does not happen, any information update will have to be propagated to all possible request points. This update traffic is potentially wasteful, especially if large numbers of updates are sent to points where hardly any request occurs. Therefore, if the request rate is very low (relative to the update rate), it makes sense to apply the pull mechanism.

To study this trade off between pull and push mechanism Havet and Wennink [HW01, HW] introduced the Push Tree Problem that formulated below:

Let $G = (V(G), E(G))$ be an undirected graph, and let $l(e) = l(u, v) > 0$ be the length of any edge $e = (u, v) \in E(G)$. Moreover, let $l(H) = \sum_{e \in E(H)} l(e)$ be the length of any subgraph H of G.

Push Tree Problem : *Given a graph G, a* source node $s \in V(G)$*, an* update rate $\mu \geq 0$*, a* request set $R \subseteq V(G)$*, and* request rates $r(v) \geq 0, \forall v \in R$*. Find a subgraph PT of G, with $s \in V(PT)$, and paths P_v from v to any node in $V(PT)$, $\forall v \in R$, such that*

$$\mu l(PT) + \sum_{v \in R} r(v) l(P_v)$$

is minimal.

The source node s contains a piece of information that changes μ times per time unit. After every change, the updated information is *pushed* from s to all nodes in $V(PT)$ using the edges in $E(PT)$. Multicast is used, which means that only one update message is sent over every edge. We measure the size of the traffic associated with a message as the number of bits times the distance covered. The total amount of update traffic per time unit is therefore equal to $\mu l(PT)$.

Requests for the information stored at s occur $r(v)$ times per time unit at node $v \in R$. Each request results in a request message to be sent over the *pull* path P_v from v to a node in PT which contains an up to date copy of the information. This node then sends a reply message to v using the same path, but now in reverse direction. The total amount of request traffic per time unit is therefore equal to $\sum_{v \in R} r(v) l(P_v)$.

As noted in [HW01], the Push Tree problem is easy if G itself is a tree. Indeed, let $T = (V, E)$ be a tree with a source $s \in V$ and a request set $R \subset V$. Removing any edge $e \in E$ from T result in the tree breaking down into two components, V_e^1 that contains s, and V_e^2. Then the *load of* e in the push tree is $\min\{\mu; \sum_{v \in V_e^2} r(v)\}$. and the traffic on the edge e is $w(e) = l(e).\lambda(e)$. The total traffic size then is $w(T) = \sum_{e \in E} w(e)$. We denote by w^* the amount of traffic in the optimal solution. If the request function that we are considering needs to be identified, we use the notations $w_r(T)$ and w_r^*.

Therefore Havet and Wennink [HW] studied a particular class of heuristics for the Push Tree problem which uses a two step approach. In the first step, a *routing tree* T is constructed and it connects the source node and all the request nodes. In the second step, the optimal push tree in T is obtained by solving the problem in the routing tree. Each heuristic in this class is thus characterized by the used method for obtaining the routing tree.

Such heuristics are attractive because they generate an infrastructure, while still allowing rapid adaptations to changing traffic patterns. In [HW], the sensitivity of solutions obtained in this way to changes in update and request rates was investigated. Suppose that a routing tree T has been obtained that results in an optimal or near-optimal (with approximation ratio ρ) push tree for a particular request function r. How good is this routing tree if the request function is modified?

Havet and Wennink discussed two different scenarios. In the first one, the set of request nodes R remains unchanged, and all request rates change almost uniformly, that is the new request function r' satisfies $(1 - \xi)r \leq r' \leq (1 + \varepsilon)r$. They show $w_{r'}(T) \leq \rho \frac{1+\varepsilon}{1-\xi} w_{r'}^*$.

In the second scenario, R changes. This corresponds to $\xi = 1$ or $\varepsilon = \infty$. They show that adding a single request node can have a dramatic effect on the quality of a routing tree : even if we have an optimal routing tree for a particular request function, adding a single request node may result in this routing tree becoming arbitrarily bad (in relation to the optimal routing tree for the new request function).

However routing tree seems to be more robust to removal of vertices. In [HW], it is conjectured that if the request function is stable except in k vertices where it decreases (and maybe disappear) then $w_r'(T) \leq (2k+1-\frac{r_m}{\mu}) w_{r'}^*$ where r_m is the minimum non zero request rate over all the nodes. This conjecture was proved it for $k = 1$ [HW].

In this paper, we establish this conjecture for every k :

Theorem 1 *Consider two instances of the Push Tree problem that are identical except for the request rates at nodes $v^1, \ldots, v^k$. We denote the first request function by r, and the second by r', and we assume $r'(v^i) < r(v^i), i = 1, \ldots, k$. Let T be the optimal routing tree for request function r. Then,*

$$w_{r'}(T) \leq (2k + 1 - k\frac{r_m}{\mu}) w_{r'}^*.$$

Theorem 1 generalizes a result of Irlande & al [AIL00], who prove it for the special case when $r_m = \mu$: the Steiner Tree problem.

The upper bound $(2k + 1 - k\frac{r_m}{\mu})$ is tight. Consider the cycle $(s, v_1, v_2, \ldots, v_k, u, s)$ with (s, u) and (u, v_k) of length 1 and all other edges of length $2 - \frac{r_m}{\mu}$. Let $r(v_i) = \mu$ and

$r'(v_i) = 0$ for $1 \leq i \leq k$ and $r(u) = r'(u) = r_m$. The path $T = (s, v_1, v_2, \ldots, v_k, u)$ is an optimal routing tree for request function r, but $w_{r'}(T) = (2k + 1 - k\frac{r_m}{\mu})w^*_{r'}$.

This result shows us that removing request nodes have a small impact whereas adding request nodes may have a dramatic one. Hence, if we want to use the routing tree as a stable infrastructure in which the Push Tree Problem is solved dynamically, it is far better to overestimate the request node sets than to underestimate it.

But if we want to use the routing tree as a semi-stable infrastructure, we will need to compute it from time to time. Since it is NP-hard to find the optimum routing tree, we may decide to use an almost optimal routing tree produced by some heuristic. But Theorem 1 may not be extended to almost optimal routing :

Proposition 1 *For any $\rho > 1$ and $M > 0$ there are two instances I and I' of the Push Tree problem and a tree T such that:*
I and I' are identical except for the request rates at node v where $0 = r'(v) < r(v)$; $w_r(T) \leq \rho w^*_r$ and $w_{r'}(T) \geq M w^*_{r'}$.

Proof. Consider the following the network with node s, t, u and v with $l(s, v) = \frac{M-1}{\rho-1}$, $l(s, u) = M - 1$, $l(u, t) = l(s, t) = 1$. Let s be the source, $\mu = 1$ and $r(t) = r'(t) = 1 = r(v)$ and $r(s) = r'(s) = r(u) = r'(u) = 0 = r'(v)$.

Let T be tree induced by the tree edges (t, s), $s, u)$ and (u, v). Then $w_r(T) = L + M$ and $w_{r'}(T) = M$ and it is easy to see that $w^*_r = L + 1$ and $w^*_{r'} = 1$. Hence $w_r(T) \leq \rho w^*_r$ and $w_{r'}(T) \geq M w^*_{r'}$.

However, the tree T in the above proof is very particular and is very unlikely to be produced by an heuristic. Indeed it is very bad on a subnetwork (that is the one induced by s, u and t). But usually heuristics yield solutions that guarantee an approximation ratio on every subnetwork.

Therefore, it would be interesting to obtain Theorem 1 like results for almost optimum trees that are produced with some particular heuristic.

2 Proof of Theorem 1

In order to prove Theorem 1, we need the following result which holds for all pairs of instances of the Push Tree problem that are identical except for their request function.

Proposition 2 (Havet and Wennink [HW]) *Let $\kappa \geq 1$ and let r and r' be two request functions such that $r'(v) \leq \kappa . r(v)$ for all $v \in R$. For any routing tree T, $w_{r'}(T) \leq \kappa w_r(T)$.*

Let us first prove that it suffices to prove Theorem 1 when $r(v^i) = \mu$, for $1 \leq i \leq k$. Indeed suppose that the set I of integers such that $r(v^i) \neq \mu$ is not empty. For every $i \in I$, let v^i_1 be the closest vertex of the r-push tree of T. And let r_1 be the request function coinciding with r on every vertices except v^i and v^i_1 for $i \in I$ where $r_1(v^i) = 0$ and $r_1(v^i_1) = \mu$. (Note

that it may happen that $v_1^i = v_1^{i'}$.) It is easy to see that T is an r_1-optimal tree. Since r_1 and r' differ only in the v_1^i, $i \in I$, then if the result holds for r_1 to r' then

$$w_{r'}(T) = w_{r_1}(T) \leq (2k + 1 - k\frac{r_m}{\mu})w^*_{r'}$$

Let us now consider that $r(v^i) = \mu$, for $1 \leq i \leq k$.

Let S be the $(s, v^1, v^2, \ldots, v^k)$-Steiner tree in T and let P^i be the path joining s to v^i in T. Clearly, $S = \bigcup_{i=1}^k P^i$ and S is a subtree of the push tree. Let us look to the loads of the edge of T under r' and r. If an edge does not belong to the tree S then its load is obviously the same under r' and r. Set $A := E(T) \setminus E(S)$ and let $w(A) := \sum_{e \in A} w_r(e, T) = \sum_{e \in A} w_{r'}(e, T)$. Since $r(v) = \mu$ all the edges of P have load μ under r. Thus

$$w^*_r = \mu l(S) + w(A) \tag{1}$$

For $i \in \{1, 2\}$ and each vertex $t \in R'$, let $p^i(t)$ be the closest vertex of P^i to u in T.

Let $1 \leq i \leq k$. Let us order the vertices $u_1^i, u_2^i, \ldots, u_{m^i}^i$ of R' such that $d_T(p^i(u_j^i), v_i) \leq d_T(p^i(u_{j+1}^i, v^i)$.

Let $\tau = \min(\mu, \sum_{u \in R'} r'(u))$.

One can find sequences of vertices of R', $(t_1^i, t_2^i, \ldots, t_m^i)$ $1 \leq i \leq k$ and functions r^i, $1 \leq i \leq k$, such that:

- for every $1 \leq i \leq k$ and $1 \leq j \leq m$, $t_j^i \in R'$;
- for every $1 \leq i \leq k$ and $1 \leq j \leq m$, if $t_j^i = u_l^i$ then for every $j' < j$ then $t_{j'}^i \in \{u_1^i, u_2^i, \ldots, u_l^i\}$, and for every $l' < l$, setting $U^i(l', j)$ the set of vertices $t_{j'}^i$ with $j' \leq j$ such that $t_{j'}^i = u_{l'}^i$, $\sum_{t \in U^i(l', j)} r^i(t) = r(u_{l'}^i)$;
- for every $1 \leq i \leq k$, $\sum_{j=1}^m r^i(t_j^i) = \tau$;
- for all $1 \leq j \leq m$ and all pair (i_1, i_2), $r^{i_1}(t_j^{i_1}) = r^{i_2}(t_j^{i_2}) = r_j$.

For $1 \leq j \leq m$, let F_j be the forest that is the union of the paths Q_j^i, $1 \leq i \leq k$, joining $p^i(t_j^i)$ to v^i in T.

$$w_{r'}(T) = \tau l(S) - \sum_{j=1}^m r_j l(F_j) + w(A) \tag{2}$$

We will now obtain upper bounds of $\tau l(S) - \sum_{j=1}^m r_j l(F_j)$ and $w(A)$ by bounding $w_r(T)$ by the traffic under r in some appropriate trees constructed from the r'-optimal tree T'.

Let T_1 be a tree obtained from T' by adding a $(s, v^1, v^2, \ldots, v^k)$-Steiner tree S_1 in G. Let us look the loads in the routing tree T_1 when the request function changes from r to r'.

It may only change for edges of S_1 where the load was μ and may decrease. Thus $w_r(T_1) \leq w^*_{r'} + \mu l(S_1)$. Since S_1 is a Steiner tree in G, $l(S_1) \leq l(S)$. Thus $w_r(T_1) \leq w^*_{r'} + \mu l(S)$. Hence $\mu l(S) + w(A) = w^*_r \leq w_r(T_1) \leq w^*_{r'} + \mu l(S)$.

$$w(A) \leq w^*_{r'} \tag{3}$$

Let T_j be a tree obtained from T' by adding a shortest forest D'_j in G connecting t^i_j to v^i for $1 \leq i \leq k$. Clearly $w^*_{r'} = w_{r'}(T_j)$. Let us look now of the loads of the edges of T_j under r and r'. Let S'_j be the $(s, t^1_j, t^2_j, \ldots, t^k_j)$-Steiner tree in T'. If e is not in $S'_j \cup D'_j$ then $\lambda_{r'}(e, T_j) = \lambda_r(e, T_j)$. If e is in $S'_j \cup D'_j$, then $\lambda_r(e, T_j) = \mu$. And if $e \in S'_j$, then $\lambda_r e, T_j \geq r_m$ since $r(t^i_j) \geq r_m$ and if $e \in D'_j$, $\lambda_r e, T_j = 0$. Thus, $w_r(T_j) \leq w_{r'}(T_j) + \mu l(D'_j) + (\mu - r_m) l(S'_j)$.

Now, let D_j be the shortest forest in T connecting t^i_j to v^i for $1 \leq i \leq k$. By definition, $l(D'_j) \leq l(D_j)$. And $l(D_j) \leq l(F_j) + \sum_{i=1}^k d_T(p^i(t^i_j), t^i_j)$. Hence,

$$w_r(T_j) \leq w^*_{r'} + \mu l(F_j) + \mu \sum_{i=1}^{k} d_T(p^i(t^i_j), t^i_j) + (\mu - r_m) l(S'_j) \tag{4}$$

Let us now consider the index j for which $d_j = l(F_j) + \sum_{i=1}^k d_T(p^i(t^i_j), t^i_j) + (1 - \frac{r_m}{\mu}) l(S'_j)$ is minimum. $w^*_r \leq w_r(T_j)$ thus by Equations 1 and 4

$$\begin{aligned}
\mu l(S) + w(A) &\leq \mu d_j + w^*_{r'} = \frac{\mu}{\tau} \tau d_j + w^*_{r'} \\
&\leq \frac{\mu}{\tau} \sum_{j=1}^{m} r_j d_j + w^*_{r'} \\
&\leq \frac{\mu}{\tau} \left(\sum_{j=1}^{m} r_j l(F_j) + \sum_{j=1}^{m} \sum_{i=1}^{k} r_j d_T(p^i(t^i_j), t^i_j) \right) + \frac{\mu - r_m}{\tau} \sum_{j=1}^{m} r_j l(S'_j) + w^*_{r'}
\end{aligned} \tag{5}$$

By definition, for every $1 \leq i \leq k$, $w(A) \geq \sum_{j=1}^m r_j d_T(p(t^i_j), t^i_j)$, so

$$\sum_{j=1}^{m} \sum_{i=1}^{k} r_j d_T(p^i(t^i_j), t^i_j) \leq k w(A) \tag{6}$$

Now $l(S'_j) \leq \sum_{i=1}^k d_{T'}(s, t^i_j)$ thus

$$\sum_{j=1}^{m} r_j l(S'_j) \leq \sum_{i=1}^{k} \sum_{j=1}^{m} r^i(t^i_j) d_{T'}(s, t^i_j)$$

And because $\sum_{j=1}^m r^i(t^i_j) = \tau \leq \mu$, we have $\sum_{j=1}^m r^i(t^i_j) d_{T'}(t^i_j, s) = w_{r^i}(T')$. And by Proposition 2, $w_{r^i}(T') \leq w_{r'}(T')$, since $r^i(v) \leq r'(v)$ for every node v. Thus, $\sum_{i=1}^k \sum_{j=1}^m r^i(t^i_j) d_{T'}(s, t^i_j) \leq$

$kw^*_{r'}$. These two inequalities give:

$$\sum_{j=1}^{m} r_j l(S'_j) \leq k w^*_{r'} \tag{7}$$

Equations 5, 6 and 7 yield

$$\begin{aligned} \mu l(P) - \frac{\mu}{\tau}\left(\sum_{j=1}^{m} r_j l(F_j)\right) &\leq \left(\frac{k.\mu}{\tau} - 1\right) w(A) + \left(k.\frac{\mu - r_m}{\tau} + 1\right) w^*_{r'} \\ \tau l(P) - \sum_{j=1}^{m} r_j l(F_j) &\leq \left(k - \frac{\tau}{\mu}\right) w(A) + \left(k(1 - \frac{r_m}{\mu}) + \frac{\tau}{\mu}\right) w^*_{r'} \end{aligned} \tag{8}$$

Now Equations 2 and 8 yield $w_{r'}(T) \leq \left(k + 1 - \frac{\tau}{\mu}\right) w(A) + \left(k(1 - \frac{r_m}{\mu}) + \frac{\tau}{\mu}\right) w^*_{r'}$. Because $k + 1 - \frac{\tau}{\mu} \geq 0$, by Equation 3, we obtain:

$$w_{r'}(T) \leq \left(2k + 1 - k\frac{r_m}{\mu}\right) w^*_{r'}$$

References

[AIL00] J. C. König A. Irlande and C. Laforest. Incrémentalité pour l'arbre de steiner,. In 2^e *rencontres francophones sur les Aspects Algorithmiques des Telecommunications (ALGOTEL'2000)*, pages 173–178, La Rochelle, France, Mai 2000.

[HW] F. Havet and M. Wennink. The push tree problem. *Networks*, Submitted, Extended version of [HW01].

[HW01] F. Havet and M. Wennink. The push tree problem. In *SPAA'01: 13th ACM Symposium on Parallel Algorithms and Architectures*, pages 318–319, 2001.

Unité de recherche INRIA Sophia Antipolis
2004, route des Lucioles - BP 93 - 06902 Sophia Antipolis Cedex (France)

Unité de recherche INRIA Lorraine : LORIA, Technopôle de Nancy-Brabois - Campus scientifique
615, rue du Jardin Botanique - BP 101 - 54602 Villers-lès-Nancy Cedex (France)
Unité de recherche INRIA Rennes : IRISA, Campus universitaire de Beaulieu - 35042 Rennes Cedex (France)
Unité de recherche INRIA Rhône-Alpes : 655, avenue de l'Europe - 38330 Montbonnot-St-Martin (France)
Unité de recherche INRIA Rocquencourt : Domaine de Voluceau - Rocquencourt - BP 105 - 78153 Le Chesnay Cedex (France)

Éditeur
INRIA - Domaine de Voluceau - Rocquencourt, BP 105 - 78153 Le Chesnay Cedex (France)
http://www.inria.fr
ISSN 0249-6399