The optimal placement of replicated items in distributed databases on tree-like networks

E.M. Bakker, J. van Leeuwen

RUU-CS-91-23
June 1991

Utrecht University

Department of Computer Science

Padualaan 14, P.O. Box 80.089,
3508 TB Utrecht, The Netherlands,
Tel. : ... + 31 - 30 - 531454

The optimal placement of replicated items in distributed databases on tree-like networks

E.M. Bakker, J. van Leeuwen

Technical Report RUU-CS-91-23

June 1991

Department of Computer Science
Utrecht University
P.O.Box 80.089
3508 TB Utrecht
The Netherlands

ISSN: 0924–3275

Here the increase of reliability and availability that results from replication is not taken into account, i.e., is not "weighted" in the cost function. The cost function that we defined was proposed in [MW87], where it is shown that computing a scheme that determines the number of copies of each data item and the location of each copy in a general network such that the cost function is minimized is likely to take too much time in practice. For networks that have a tree-structure, on the other hand, we will show that it is possible to determine an optimal allocation scheme in linear time. This improves the quadratic time-bound given in [MW87].
The paper is organized as follows. In Section 2 some preliminaries are given. In Section 3 the problem is studied for the case of tree networks. Finally, in Section 4 we state some conclusions and suggestions for future research.

2 Preliminaries

As usual we model a communication network by a graph $G = (V, E)$. We take G to be undirected and assume that the nodes in G host the files of a distributed database. Let $d_G(v, w)$ be the length of the shortest path between v and w in G. If U is a subset of V, then $d_G(v, U)$ is equal to $min_{w \in U} d_G(v, w)$. Consider the placement of the replicas of a data item in G. We call the set $R \subseteq V$ consisting of all nodes where a copy of a certain data item is located the *residence set*. If we want to read the data item, it is obvious that we like to access the nearest copy of it. Clearly, for a node $v \in V$ the nearest copy is at distance $d_G(v, R)$. If we want to update the data item, the update information is sent to all nodes that contain a copy of the data item, i.e., to all elements of the residence set R. (In many concurrency control schemes updates are batched before they are broadcast, but we ignore such optimizations here.) Another parameter that determines the quality of a residence set is the ratio of the *frequency of the retrieval operations* and the *frequency of the update operations*. To model the update and retrieval cost of a residence set, we define a cost function as follows.

Definition 2.1 *Let $G = (V, E)$ be a graph and α a positive real number. Let $R \subseteq V$ be a residence set. Let $v \in V$, and $P_v = R \cup \{v\}$.*

- *A* write-instance *is a weighted directed graph $I_v = (P_v, A_v)$ with the property that for every $w \in R$ there exists a path from v to w in I_v. Furthermore, every edge $e = (v', w') \in A_v$ has weight $w(e)$ equal to $d_G(v', w')$. The cost of the write-instance I_v is equal to $W(I_v) = \sum_{e \in A_v} w(e)$.*

- *The* write-cost *$wr(v)$ for a node v is equal to the minimum of $W(I_v)$ taken over all possible write-instances I_v.*

- *The* read-cost *$r(v)$ for a node v is equal to $d_G(v, R)$.*

- *The* cost function *$c(R)$ is defined by $c(R) = \sum_{v \in V} wr(v) + \alpha . \sum_{v \in V} r(v)$.*

Note that each edge in a write-instance I_v represents a "channel" for the transfer of update information between two participants, i.e., nodes that are element of $R \cup \{v\}$. The cost of an edge is equal to the length of the shortest path between the two nodes. If in a write-instance $I_v = (P_v, A_v)$ update information is transfered from one node, say v', to two different nodes, say w_1 and w_2, then the cost of this transfer is equal to the sum of the length of the shortest path from v' to w_1 and the length of the shortest path from v' to w_2, regardless the fact that some common sub-path may be followed. If there exists a $w' \in V$ such that $d_G(v', w') + d_G(w', w_1) + d_G(w', w_2)$ is less than $d_G(v', w_1) + d_G(v', w_2)$, then clearly it would be better to first transfer the update information from v' to w' and from there on transfer it to w_1 and w_2. However, in the definition of the write-instance it is assumed that update information is only transfered between nodes that are element of P_v. Hence there only exists an alternative write-instance $I'_v = (P_v, A'_v)$ where update information is transfered from v' to w' and from there on to w_1 and w_2, if w' is element of P_v. If w' is not an element of P_v, this optimization will not be taken into account.
Despite the simplicity of the cost function the following problem is NP-complete [MW87] :

> **Problem:** REPLICA RESIDENCE SET
> **Instance:** A graph $G = (V, E)$, and positive n-bit real numbers W and α (n a positive integer).
> **Question:** Is there a residence set $R \subseteq V$ such that $c(R) = \sum_{v \in V} wr(v) + \alpha . \sum_{v \in V} r(v) \leq W$?

This is the main motivation to restrict the problem to the more special class of trees.

3 The problem restricted to trees

Let us recall the following useful lemma. For the proof of the lemma the reader is referred to [MW87].

Lemma 3.1 **[MW87]** *Let $G = (V, E)$ be a connected graph and let $R \subseteq V$ be a residence set. If R induces a connected subgraph of G, then $wr(v) = d(v, R) + |R| - 1$ for every $v \in V$.*

We will denote the cost function as defined in Definition 2.1 by c. The problem of finding a residence set on a tree that minimizes c, is efficiently solvable. The following lemmas and theorems help us to devise a linear-time algorithm that solves the REPLICA RESIDENCE SET problem for trees. The first lemma we need is another result from [MW87].

Lemma 3.2 **[MW87]** *Let $T = (V, E)$ be a tree. If $R \subseteq V$ is a residence set such that $c(R)$ is minimal, then R induces a connected subgraph of T (thus a subtree of T).*

Observe that from Lemma 3.2 it follows that Lemma 3.1 applies to every residence set that minimizes c on a tree. Now recall some notions concerning central structures in trees.

Definition 3.3 *Let $T = (V, E)$ be a tree and let $v \in V$. The* branch-weight *of v, denoted by $bw(v)$, is equal to the largest number of vertices in a component of $T - v$. The* branch-weight centroid *of T is the collection of vertices of T with minimum branch-weight.*

Definition 3.4 *Let $G = (V, E)$ be a graph. The* median *of G is equal to the collection of vertices v' with minimal distance sum $= \sum_{v \in V} d_G(v, v')$.*

The following two theorems are due to B. Zelinka [Z68] and C. Jordan [J69], respectively.

Theorem 3.5 [Z68] *For any tree T the median of T equals the branch weight centroid of T.*

Theorem 3.6 [J69] *Let $T = (V, E)$ be a tree. If $C \subseteq V$ is equal to the median of T, then $|C| \leq 2$.*

The branch-weight centroid of a tree and a residence set that minimizes the cost function c are closely related, as shown in the following theorem.

Theorem 3.7 *Let $T = (V, E)$ be a tree. Let $R \subseteq V$ be a residence set of T such that $c(R)$ is minimal. If $C \subseteq V$ is the branch-weight centroid of T, then $C \subseteq R$.*

Proof. If $|R| = 1$, then it is easy to verify that R is equal to the median of T and the theorem easily follows from Theorem 3.6. Assume that $|R| > 1$. Suppose, to the contrary, that the theorem does not hold, i.e., there exists a node $v_c \in C$ such that $v_c \notin R$. Remark that by Lemma 3.2 R induces a connected subgraph, i.e., a subtree T_R. Hence there exists a unique path P from v_c to R. Let v be the node on P that is adjacent to a node $\in R$. By applying Zelinka's and Jordan's theorem it follows that $|C| \leq 2$. Furthermore, $|R| > 1$. It follows that there exists a node $w \in R$ such that w is a leaf of T_R, and w is not adjacent to v. As $v_c \in C$, it is clear that the branch-weight of v_c and hence the size of every component of $T - v_c$ is less than or equal to $|V|/2$ (Jordan's lemma, see [K36]). w is a leaf of T_R hence w has exactly one neighbor $w' \in R$. Let T_w be equal to the component of $T - (w, w')$ that contains w. T_w is strictly contained in a component of $T - v_c$. It follows that the number of nodes in T_w must be strictly smaller than $|V|/2$. Assume $v' \in R$ is the neighbor of v. Let T_v be the component of $T - (v, v')$ that contains v_c. It follows that T_v must contain $\geq |V|/2$ nodes. By Lemma 3.1 $c(R) = \sum_{v \in V}(d(v, R) + |R| - 1) + \alpha \sum_{v \in V} d(v, R)$. Let $R' = (R - \{w\}) \cup \{v\}$. Then by Lemma 3.1 $c(R') = \sum_{v \in V}(d(v, R') + |R'| - 1)$

$+\alpha \sum_{v\in V} r(v,R') = c(R) + (1+\alpha)(|V_{T_w}| - |V_{T_v}|) < c(R)$. This clearly contradicts the minimality of $c(R)$. Hence the theorem is true. □

Assume that we have a residence set $R' \subseteq V$ of a tree $T = (V,E)$ that induces a connected subgraph of T. Let $\{v_1, v_2, ..., v_l\}$ be the set of neighbors of R' in T. Denote by $T_i, i \in \{1,...,l\}$, the subgraph of T consisting of all the components of $T - v_i$ that do not contain R', respectively. If we let $R'' = R' \cup \{v_j\}$, for a certain $j \in \{1,...,l\}$, then

$$\begin{gathered} c(R'') = \textstyle\sum_{v\in V}(d_T(v,R'') + |R''| - 1) + \alpha.\sum_{v\in V} d_T(v,R'') \\ = \textstyle\sum_{v\in V-(V_{T_j}\cup\{v_j\})}(d_T(v,R') + |R'|) + \sum_{v\in V_{T_j}\cup\{v_j\}}(d_T(v,R') + |R'| - 1) + \\ \alpha.\textstyle\sum_{v\in V} d(v,R') - \alpha.(\left|V_{T_j}\right| + 1) \\ = c(R') + |V| - (\alpha+1).(\left|V_{T_j}\right| + 1). \end{gathered}$$

Define $a(R',v_j) = |V| - (\alpha+1).(\left|V_{T_j}\right| + 1)$. We conclude that if there exists a $v_j, j \in \{1,...,l\}$, such that $a(R',v_j) < 0$, then R' cannot be a minimal residence set of T.
Observe that if $a(R',v_j) < 0$, then for every v_i $(i \neq j, i \in \{1,...,l\})$ $a(R' \cup \{v_i\}, v_j) < 0$ holds, i.e., the fact that a neighbor v_i is added to the residence set does not influence the necessity of adding v_j to the residence set in order to obtain smaller costs. It follows that if R is a residence set that minimizes c, then for every node v that is a neighbor of R, $a(R,v) > 0$ must hold. Furthermore, let R' be a subset of R and $v \in R - R'$ a neighbor of R'. Assume that $a(R',v) > 0$. Let $v' \in R'$ be adjacent to v. Then it is clear that for all nodes u of the component T_v of $T - (v,v')$ that contains v, $a(R'',u) > 0$ holds, where R'' is any set of nodes that contains R' and is adjacent to u. Thus $c(R - V_{T_v}) < c(R)$, which is a contradiction. Hence $a(R',v) \leq 0$ must hold. This enables us to prove the following theorem.

Theorem 3.8 *Let $T = (V,E)$ be a tree and α a positive n-bit real (n a positive integer). There exists a linear-time algorithm that computes a residence set R of T that minimizes the cost function c as defined in Definition 2.1.*

Proof. Let $T = (V,E)$ be a tree. By Theorem 3.7 the median of T is always a subset of the residence set that minimizes the cost function c. The median C of T can be computed in linear time [G71]. Initialize the residence set R to C. Let $v_c \in C$. Orient T as a tree rooted at v_c. By doing a breadth-first search on T one can determine for every node v the number of nodes in the subtree rooted at v. This can be done in linear time. This enables us to compute $a(R,v)$ efficiently, i.e., in constant time. From the previous analysis we know that if there exists a node v such that it is a neighbor of R and $a(R,v) < 0$, then v must belong to the residence set that minimizes the cost function c. We keep adding these kind of neighbors until there are no more left, to obtain the desired residence set. This method is the basis for the following Algorithm RS.

Algorithm RS.

```
begin
  let $T = (V, E)$ be a tree and $\alpha$ a positive $n$-bit real ($n$ a positive integer) ;
  determine the median $C \subseteq V$ of $T$ ;
  $R := C$ ;
  let $v_c \in R$ and orient $T$ as a tree rooted at $v_c$ ;
  calculate for every $v \in V$ by a breadth-first search starting at $v_c$
  the number of nodes in the subtree rooted at $v$ ;
  let $V_R$ be the set of neighbors of $R$ in $T$ ;
  while there exists a node $v \in V_R$ with $a(R, v) < 0$
     do
        $R := R \cup \{v\}$ ;
        $V_R := (V_R - \{v\}) \cup$ {all neighbors of $v$ that are not an element of $R$}
     od
end {Algorithm RS}
```

It is easy to verify that the algorithm computes an optimal residence set for T using $O(|V|)$ steps. □

4 Conclusions

In this paper we showed that the problem of finding a residence set that minimizes the proposed cost function c can be solved in linear time for trees. The problem of determining the complexity of the problem restricted to other classes of graphs remains an interesting object for further research. Further, although the cost function in itself is fairly realistic, many alternative cost functions may be defined. Related research on this matter can be found in [DF82]. Many interesting problems remain open here.

References

[BHG87] P.A. Bernstein, V. Hadzilacos, N. Goodman. *Concurrency Control and Recovery in Database Systems.* Addison-Wesley Publ. Comp., Reading, Mass., 1987.

[CP84] S. Ceri, G. Pelagatti. *Distributed Databases - Principles and Systems.* Mc Graw-Hill Book Comp., New York, N.Y., 1984.

[DF82] L.W. Dowdey, D.V. Foster. *Comparative Models of the File Assignment Problem.* ACM Computing Surveys, Vol. 14, 1982, pp. 287-313.

[G71] A.J. Goldman. *Optimal Center Location in Simple Networks.* Transportation Science, Vol. 5, 1971, pp. 212-221.

[J69] C. Jordan. *Sur les Assemblages de Lignes.* J. Reine Angew. Math., Vol. 70, 1869, pp. 185-190.

[K36] D. König. *Theorie der Endlichen und Unendlichen Graphen.* Leipzig, 1936, Reprinted Chelsea, New York, 1950.

[MW87] A. Milo, O. Wolfson. *Placement of Replicated Items in Distributed Databases (Preliminary Version).* Technical Report 473, TECHNION-Israel Institute of Technology, Computer Science Department, November 1987.

[Z68] B. Zelinka. *Medians and Peripherians of Trees.* Arch. Math. Brno, 1968, pp. 87-95.