February 28, 2001 TR CS-01-03

Steiner-Optimal Data Replication in Tree Networks

Konstantinos Kalpakis[1,2], Koustuv Dasgupta[1], and
Ouri Wolfson[3]

Computer Science and Electrical Engineering Department
University of Maryland Baltimore County
1000 Hilltop Circle
Baltimore, Maryland 21250

[1]Email: {kalpakis,dasgupta}@csee.umbc.edu

[2]Supported in part by the National Science Foundation under grant number IRI–9729495, and by NASA under Cooperative Agreement NCC5–315.

[3]Department of Electrical Engineering and Computer Science, University of Illinois at Chicago, IL 60607. Email: wolfson@ouri.eecs.uic.edu. Supported in part by the National Science Foundation under grant number IRI-9712967.

Steiner-Optimal Data Replication in Tree Networks

Konstantinos Kalpakis[1,2], Koustuv Dasgupta[1], and Ouri Wolfson[3]

Abstract

We consider the problem of placing copies of objects in a tree network in order to minimize the cost of servicing read and write requests to objects when the tree nodes have limited storage and the number of copies permitted is limited. The set of nodes that have a copy of the object, called replica nodes, constitute the replica set of the object. Read requests of a node are serviced from the closest replica node. Write requests of a node are propagated to all the replicas of the object using a minimum cost Steiner tree that includes the writer and all replica nodes. The cost associated with a replica set equals the cost of servicing all the read and write requests, plus the storage cost at all the replica nodes. We are interested in finding a replica set with minimum cost, i.e. a Steiner–optimal replica set.

Given a tree with n nodes, we provide an $O(n^6p^2)$–time algorithm for finding a Steiner–optimal replica set of size p, taking into consideration the read, write, and storage costs. Our algorithm can also find a Steiner–optimal replica set for a tree with n nodes in time $O(n^8)$. We also demonstrate that the policy used to propagate write requests to all the replica nodes in the network affects the cost and configuration of the optimal replica set for the object.

Keywords: data replication, multicasting, facility location, p–medians, file allocation, tree networks, Steiner trees.

---

[1]Computer Science and Electrical Engineering Department, University of Maryland Baltimore County, 1000 Hilltop Circle, Baltimore, MD 21250. E-mail: {kalpakis,dasgupta}@csee.umbc.edu. Phone: 410-455-3143. Fax: 410-455-3969. **Please send all correspondence to Dr. K. Kalpakis.**

[2]Supported in part by the National Science Foundation under grant number IRI–9729495, and by NASA under Cooperative Agreement NCC5–315.

[3]Department of Electrical Engineering and Computer Science, University of Illinois at Chicago, IL 60607. Email: wolfson@ouri.eecs.uic.edu. Supported in part by the National Science Foundation under grant numbers IRI-9712967, CCR-9816633, and CCR-9803974, DARPA grant N66001-97-2-8901, and Army Research Labs grant DAAL01-96-2-0003.

1 Introduction

The recent growth in the World Wide Web is moving us to distributed highly interconnected information systems. In such systems, an object (a web page, an image, a video clip, or a file) is read and written from multiple locations in the Internet that are distributed geographically worldwide. Maintaining multiple copies of an object at various locations is an approach for improving system performance (time to read or write an object).

In this paper, we consider the problem of placing copies of an object at multiple locations in a distributed system, whose interconnection network is a tree, in order to minimize the total cost of servicing the read/write requests. Consider an object and a tree that connects a set of nodes V. Each node (vertex) of the tree issues a number of read requests and a number of write requests to the object. Let S be a subset of the nodes of the tree at which there exist copies (replicas) of the object. The subset of nodes S, called the replica nodes, constitute the *residence set* or *replica set* of the object. Assume that each node of the tree sends its read requests to the closest replica node, and that each write to a copy of the object must be propagated to all other replica nodes. The write propagation policy we use is as follows: each writer node sends its request along the edges of a Steiner Minimum Tree (SMT) that includes the writer node and all the replica nodes. We call this write policy the *SMT-write* policy. The cost of a read is equal to the distance of the reader node from the closest replica node, while the cost of a write is equal to the cost of the SMT that spans over the writer and all replica nodes. Further, we assume that having a replica of the object at a node has an associated storage cost that depends on the object and that node. The cost of a replica set is equal to the total read, write, and storage costs. We are interested in finding a replica set of minimum cost. This is the Steiner-optimal replica set problem with storage costs. We call the replica set that achieves the above minimum the *Steiner-optimal replica set* for a tree network.

Kalpakis et al [10] describe an $O(n^3p^2)$–time algorithm for finding an optimal replica set of size p for an object in a tree with n nodes, taking into consideration read-write-storage costs. However, in the model used in [10], each node in the tree sends its read and write requests to the closest node having a replica. Moreover, the write propagation policy used is the following: each writer node sends its request to its closest replica node; writes are broadcasted to all other replica nodes using a Minimum Spanning Tree (MST) of the *distance graph* of the replica nodes. The distance graph of the replica nodes is an edge-weighted complete undirected graph where each graph vertex corresponds to a replica node, and each graph edge (u, v) has a weight equal to the distance between the replica nodes corresponding to u and v in the tree network. We call the write policy used in [10] the Minimum Spanning Tree write policy (or *MST-write* policy), and the optimal replica set computed in [10] the MST-optimal replica set for a tree network.

The optimal residence set (file allocation) problem has been studied extensively in the literature. Dowdy and Foster [5] survey a number of mixed linear programming models for

the file allocation problem. Wolfson and Milo [15] consider the optimal residence set problem without storage costs for various interconnection networks (completely connected, tree, and ring networks). They show that the optimal residence set problem without storage costs is NP–hard for general topologies, and they provide efficient algorithms for finding optimal residence sets: an $O(n)$–time algorithm for tree networks, an $O(n^5)$–time algorithm for ring networks, and an $O(1)$–time algorithm for a completely connected network, where n is the number of nodes in the network. Our model is different from that of Wolfson and Milo [15] since we also consider storage costs. Moreover, the write policy in [15] uses the minimum spanning tree of the distance graph of the replica nodes. Fisher and Hochbaum [7] consider the problem of database location in computer networks (a problem similar to the one of this paper), and describe computational experiments based on mixed integer programming models. Our model is different from that of [7], since they assume what Wolfson and Milo [15] call a "naive–write policy" (i.e. the write cost is equal to the sum of the distances between the writer and each one of the nodes with a copy of the object).

When there are no write–costs, our problem reduces to the (uncapacitated) facility location problem which also has been studied extensively due to its applications (see [8, 9, 13]). When there are only read–costs, our problem reduces to the p–median problem [1, 9, 11], with the restriction that no more than p copies of the object are made. Hochbaum [9] describes approximation algorithms for finding p–medians on the plane (discrete and continuous) as well as on networks. Arora et al [1] describe randomized polynomial approximation schemes for finding p–medians in Euclidean spaces. Kariv and Hakimi [11] show that the p–median problem is NP–hard for general networks, and they provide an $O(n^2p^2)$–time algorithm for finding a p–median for a tree with n nodes.

The main contributions of this work deal with two critical aspects of replicated database systems i.e. design and deployment. First, we describe an $O(n^6p^2)$–time dynamic programming algorithm for finding a Steiner-optimal replica set of size p for an object on a tree with n nodes when there are read, write, and storage costs associated with storing replicas at nodes of the tree. Clearly, our algorithm can find a Steiner-optimal replica set for a tree with n nodes in $O(n^8)$ time when read, write, and storage costs are taken into consideration. In fact, to the best of our knowledge, this is the first polynomial time algorithm that finds optimal replica sets in tree networks (with storage costs) when the SMT-write policy is used. Second, we demonstrate that the cost and optimality of a replica set inherently depends upon the write propagation policy that is deployed. As an example, consider the tree network in Figure 1. Let the replica set for a data object A comprise of the nodes 4 and 5. Further, let node 1 issue a write request for A. This write needs to be propagated to nodes 4 and 5. The naive-write policy will incur a write cost of $3 + 3 = 6$, the MST-write policy incurs a write cost of $3 + 2 = 5$, while the SMT-write policy incurs a write cost of $1 + 1 + 1 + 1 = 4$. In particular, we demonstrate that an MST-optimal replica set is *not necessarily* a Steiner-optimal replica set for a tree network, and vice versa. Further, our results show that for sample network configurations, the total cost of replication using the SMT-write policy can be significantly lower than the cost incurred using the MST-write policy.

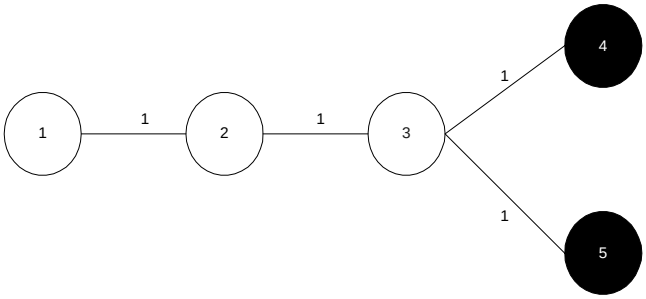

Figure 1: A sample tree network with five nodes. The shaded nodes are in the replica set of a data object A.

The rest of the paper is organized as follows. In section 2 we provide the necessary preliminaries and introduce the notion of replica assignments in trees. In section 3, we consider the restrictions of replica assignments to subtrees. In section 4, we present our algorithm for finding Steiner-optimal replica sets in tree networks. In section 5, we compare the costs of the MST-optimal and Steiner-optimal replica sets using the MST-write and SMT-write policies. Finally, in section 6, we conclude the paper.

2 Preliminaries

Consider an edge–weighted undirected tree $T = (V, E)$ with n vertices. Let $l : E \rightarrow R_+$ be the edge–weight function of T. Assume, w.l.o.g., that the children of each vertex are ordered left–to–right. Let root(T) be the root of T. For each vertex $v \in T$, let deg(v) be the number of children of v, let T_v be the subtree of T rooted at v, and let $T_v^{(i)}$ be the subtree of T that consists of v and all the subtrees rooted at each of the i leftmost children of v.

The vertices of T issue read and write requests for an object. Copies of that object can be stored at multiple vertices of T. Let $r, w : V \rightarrow N$ be two vertex–weight functions representing the number of read requests and the number of write requests issued by vertices of T, respectively. Let $s : V \rightarrow R_+$ be a vertex-weight function representing the storage cost of placing a replica of the object at vertices of T.

The distance $\delta(u, v)$ between any two vertices u and v of T is defined as the length of the shortest path from u to v in T. The distance of a vertex $v \in T$ from any subset $S \subseteq V$ is $\delta(v, S) = \min\{\delta(v, u) : \ u \in S\}$. The distance between two subsets $X, Y \subseteq V$ is $\delta(X, Y) = \min\{\delta(u, v) \ : \ u \in X, v \in Y \ \}$.

Note that the storage costs for the tree nodes and costs of the tree edges should be expressed in comparable units; for example, the storage cost for each tree node can be expressed in dollars for each copy of an object, while the cost of a tree edge can be expressed in dollars per read/write request. Assigning costs to heterogeneous resources (e.g. memory and

bandwidth) can be done using cost-benefit approaches [4], such as the opportunity cost for a resource [2, 3], and other microeconomic-based resource management approaches [6, 12, 14].

The distance graph $\Delta[V'] = (V', V' \times V')$ for $V' \subseteq V$ is an edge–weighted complete undirected graph, such that the weight of each edge (u, v) of $\Delta[V']$ is equal to the distance $\delta(u, v)$ of u from v in G. Let $\mathrm{MST}(V')$ be the cost of a minimum spanning tree of $\Delta[V']$.

Let $S \subseteq V$ be a subset of vertices of the tree T. A Steiner Minimum Tree (SMT) T' for S is a minimum cost subtree of T that spans all the vertices in S; the vertices in S are called *terminal vertices*; the vertices of T' that are not in S are called *Steiner vertices*. Let $\mathrm{SMT}(S)$ denote the cost of an SMT for S. Note that every $S \subseteq V$ has exactly one Steiner Minimum Tree. Let $d(v, S)$ denote the distance of a vertex v from the Steiner Minimum Tree for S. Observe that, for any $v \in V$ and $S \subseteq V$, the cost of a Steiner Minimum Tree for $S \cup \{v\}$ is

$$\mathrm{SMT}(S + v) = \mathrm{SMT}(S) + d(v, S). \tag{1}$$

Let $\mathrm{LCA}(u, v)$ denote the least common ancestor (in T) of any two vertices $u, v \in V$, and let $\mathrm{LCA}(S)$ denote the least common ancestor of all the vertices in a subset S of V.

2.1 Replica Assignments

A replica assignment for T is a function $\sigma : V \to V$, that assigns to each vertex in V a vertex in V such that $\forall v \in V$, $\delta(v, \sigma(v)) = \delta(v, \sigma(V))$. For brevity, we denote the range of σ with $\sigma(V)$. The replica set of σ is defined to be $\sigma(V)$. A replica assignment σ is a k–replica assignment if $|\sigma(V)| = k$. We say that u covers (is assigned to) v if $\sigma(v) = u$. We say that v is covered by (is assigned) u if $\sigma(v) = u$. A replica assignment σ is *compatible* with a set $S \subseteq V$ if $\sigma(V) = S$ and $\forall v \in V$, $\delta(v, \sigma(v)) = \delta(v, S)$. Given S it is easy to find a replica assignment σ for T that is compatible with S. We assume, w.l.o.g, that $\sigma(v) = v$ for all $v \in \sigma(V)$. The *restriction* of a replica assignment $\sigma : V \to V$ to $V' \subseteq V$ is a replica assignment $\sigma' : V' \to V$ such that $\sigma'(v) = \sigma(v)$ for all $v \in V'$. A replica assignment σ is an *extension* of σ' if the restriction of σ to the domain of σ' is σ'.

Let σ be a replica assignment for T. Each vertex v with a read request sends its request to its assigned vertex $\sigma(v)$. Each vertex v with a write request propagates its write request to all the vertices in the replica set of σ, using the edges of a Steiner Minimum Tree for the replica set of σ together with vertex v. We call this policy the *SMT–write* policy.

Let $\sigma : V' \to V$ be a replica assignment for a subtree $T' = (V', E')$ of T. The *read cost* of σ is defined as

$$C_a(\sigma) = \sum_{v \in V'} \mathrm{r}(v) \cdot \delta(v, \sigma(v)). \tag{2}$$

The read cost of σ is the total cost of sending the read requests from each vertex in the domain of σ to its assigned vertex. The *write cost* of σ is defined as

$$C_u(\sigma) = \sum_{v \in V'} \mathrm{w}(v) \cdot \mathrm{SMT}(\sigma(V') + v) = W_{\mathrm{T}} \cdot \mathrm{SMT}(\sigma(V')) + \sum_{v \in V'} \mathrm{w}(v) \cdot d(v, \sigma(V')) \quad (3)$$

where $W_{\mathrm{T}} = \sum_{v \in V} \mathrm{w}(v)$. The write cost of σ equals the cost of propagating all the writes issued by the vertices of T to the vertices in the replica set (range) of σ, plus the cost of forwarding the write requests of the vertices in V' to the Steiner tree for the replica set of σ. The *storage cost* of σ is defined as

$$C_s(\sigma) = \sum_{v \in \sigma(V')} \mathrm{s}(v). \quad (4)$$

The storage cost of σ equals the storage costs of all the vertices in the replica set of σ. The *total cost* of σ is defined as

$$C(\sigma) = C_a(\sigma) + C_u(\sigma) + C_s(\sigma). \quad (5)$$

The total cost of σ equals the sum of its read, write, and storage cost. Observe, that if σ is a replica assignment for T, i.e. it is defined for all the vertices of T, then its total cost $C(\sigma)$ is equal to the cost of servicing all the requests issued by the vertices of T plus the cost of storing the object at the vertices in the replica set of σ.

A k–replica assignment for T of minimum cost, among all k–replica assignments for T, is called a *Steiner–optimal k–replica assignment* and its replica set a *Steiner–optimal k–replica set*. Similarly, a replica assignment for T of minimum cost, among all replica assignments for T, is called a *Steiner–optimal replica assignment* for T, and its replica set a *Steiner–optimal replica set* for T.

We are interested in finding Steiner–optimal replica assignments for T, as well as in finding Steiner–optimal k–replica assignments for T, for a given integer $1 \leq k \leq n$.

Observe that for every replica assignment σ for T, there exists a replica assignment σ' with the following properties: for every vertex $u \in T$, every vertex $v \in T_u$ is assigned either a vertex in T_u or the vertex $\sigma'(u)$, $\delta(v, \sigma(v)) = \delta(v, \sigma'(v))$, and $\sigma'(V) = \sigma(V)$. In other words, for every subtree of T, every vertex in that subtree is assigned either a vertex in the subtree or the vertex assigned to the root of the subtree; without any effect on the replica set, on the distance of any vertex from its assigned vertex, or on the distance of any vertex from the Steiner minimum tree for the replica set. We call a replica assignment σ' that satisfies these properties a *normal replica assignment*. Hereafter, w.l.o.g., we assume, unless stated otherwise, normal replica assignments.

3 Restrictions of Replica Assignments to Subtrees

Consider a (normal) replica assignment σ for T.

Let v be a vertex of T. Let $T_v^{(i)}$ be the subtree of T that consists of v and the subtrees of T rooted at each one of the i leftmost children of v, $0 \leq i \leq \deg(v)$. Let v_i be the ith child of v, and let T_{v_i} be the subtree of T rooted at v_i. Note that $T_v^{(0)}$ consists of only vertex v. For brevity, $T_v^{(i)}$ and T_{v_i} will also denote the set of vertices in the subtrees $T_v^{(i)}$ and T_{v_i} respectively.

Let $\sigma_v^{(i)}$, $\sigma_v^{(i-1)}$, and σ_{v_i} be the restrictions of σ to the vertices in $T_v^{(i)}$, $T_v^{(i-1)}$, and T_{v_i} respectively. Note that $\sigma_v^{(i-1)}$ and σ_{v_i} are also the restrictions of $\sigma_v^{(i)}$ to the vertices in $T_v^{(i-1)}$, and T_{v_i} respectively. Moreover, $\sigma_v^{(i)}$, $\sigma_v^{(i-1)}$, and σ_{v_i} are all normal replica assignments.

Let S, S_1, and S_2 be the replica sets for $\sigma_v^{(i)}$, $\sigma_v^{(i-1)}$, and σ_{v_i} respectively. Note that $S_1 \subseteq S$, $S_2 \subseteq S$, and that S_1 and S_2 can have at most one vertex in common, the vertex assigned to v.

Let x_1 be the least common ancestor of the replica vertices $\sigma(V)$ that are in the subtree $T_v^{(i-1)}$; x_1 is undefined if $\sigma(V)$ does not have any vertices that are in $T_v^{(i-1)}$.

Let x_2 be the least common ancestor of the replica vertices $\sigma(V)$ that are in the subtree T_{v_i}; x_2 is undefined if $\sigma(V)$ does not have any vertices that are in T_{v_i}. Whenever x_1 or x_2 are not defined we let them be equal to $\emptyset$.

Next, in Lemmas 1–11, we derive certain relationships between the Steiner Minimum Trees for S, S_1, S_2 and the distances of a vertex in $T_v^{(i)}$ from those Steiner trees. We use these lemmas to formulate a set of recursive equations, in Theorem 1 below, between the total costs of the replica assignments $\sigma_v^{(i)}$, $\sigma_v^{(i-1)}$, and σ_{v_i}. In order to derive these equations, there are a number of cases to consider depending on whether $\sigma(v)$ belongs to any of the subtrees $T_v^{(i)}$, $T_v^{(i-1)}$, and T_{v_i}, whether $\sigma(v)$ covers v_i, and whether x_1 and x_2 are defined.

Lemmas 1–4 deal with the case where $\sigma(v)$ covers v_i and it does not belong to $T_v^{(i)}$.

Lemma 1 *Assume that $\sigma(v) = \sigma(v_i)$, $\sigma(v) \notin T_v^{(i)}$, and that both x_1 and x_2 are defined. Then, $S = S_1 \cup S_2$, $S_1 \cap S_2 = \{\sigma(v)\}$,*

$$\mathrm{SMT}(S) = \mathrm{SMT}(S_1) + \mathrm{SMT}(S_2) - \delta(v, \sigma(v)) \tag{6}$$

and for all vertices $z \in T_v^{(i)}$,

$$d(z, S) = \begin{cases} d(z, S_1), & \text{if } z \in T_v^{(i-1)} \\ d(z, S_2), & \text{otherwise.} \end{cases} \tag{7}$$

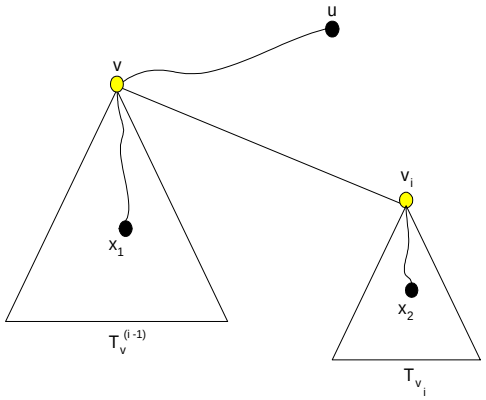

Figure 2: The subtree $T_v^{(i)}$ for Lemmas 1–4. Note that the vertex u covers both v and v_i, and u lies outside the subtree $T_v^{(i)}$.

Proof.
Note that S_1 and S_2 have exactly one vertex in common, the vertex $\sigma(v) = u$. Consider the Steiner minimum tree T_0 for S. It consists of a Steiner tree T_1 for $S_1 - u$, a Steiner tree T_2 for $S_2 - u$, and the two paths $x_1 \rightsquigarrow u$ and $x_2 \rightsquigarrow u$ in T. Moreover the tree T_1 together with the path $x_1 \rightsquigarrow u$ is the Steiner minimum tree T_1' for S_1, and the tree T_2 together with the path $x_2 \rightsquigarrow u$ is the Steiner minimum tree T_2' for S_2. Since the two SMTs T_1' and T_2' have only the edges on the path $v \rightsquigarrow u$ in common, it follows that $\mathrm{SMT}(S) = \mathrm{SMT}(S_1) + \mathrm{SMT}(S_2) - \delta(v, \sigma(v))$. Moreover, the distance of any vertex $z \in T_v^{(i)}$ from T_0 is equal to its distance from T_1' if $z \in T_v^{(i-1)}$, and is equal to its distance from T_2' otherwise. ■

Lemma 2 *Assume that $\sigma(v) = \sigma(v_i)$, $\sigma(v) \notin T_v^{(i)}$, and that x_1 is defined, while x_2 is undefined. Then, $S_2 = \{\sigma(v)\}$, $\mathrm{SMT}(S_2) = 0$, $S = S_1$, $\mathrm{SMT}(S) = \mathrm{SMT}(S_1)$, and for all vertices $z \in T_v^{(i)}$,*

$$d(z, S) = \begin{cases} d(z, S_1), & \text{if } z \in T_v^{(i-1)} \\ d(z, S_2) - \delta(v, \sigma(v)), & \text{otherwise.} \end{cases} \tag{8}$$

Proof.
Note that S_1 and S_2 have exactly one vertex in common, the vertex $\sigma(v) = u$. Since x_2 is undefined it follows that S_2 has no vertices that are in T_{v_i}. Thus, $S_2 = \{u\}$ and $S = S_1$. Thus, the Steiner minimum tree T_0 for S is also the Steiner minimum tree T_1 for S_1. Note that the Steiner minimum tree T_2 for S_2 consists of the vertex u only. Clearly, the distance of a vertex in $T_v^{(i-1)}$ from T_0 is equal to its distance from T_1. The distance of a vertex $z \in T_{v_i}$ from T_0 is equal to its distance from v. Since $u \notin T_v^{(i)}$, it follows that $d(z, S) = d(z, S_2) - \delta(v, \sigma(v))$. ■

Lemma 3 *Assume that $\sigma(v) = \sigma(v_i)$, $\sigma(v) \notin T_v^{(i)}$, and that x_1 is undefined, while x_2 is*

defined. Then, $S_1 = \{\sigma(v)\}$, $\mathrm{SMT}(S_1) = 0$, $S = S_2$, $\mathrm{SMT}(S) = \mathrm{SMT}(S_2)$, *and for all vertices* $z \in T_v^{(i)}$,

$$d(z,S) = \begin{cases} d(z,S_1) - \delta(v,\sigma(v)), & \text{if } z \in T_v^{(i-1)} \\ d(z,S_2), & \text{otherwise.} \end{cases} \tag{9}$$

Proof.
Note that S_1 and S_2 have exactly one vertex in common, the vertex $\sigma(v) = u$. Since x_1 is undefined it follows that S_1 has no vertices that are in $T_v^{(i-1)}$. Thus, $S_1 = \{u\}$ and $S = S_2$. Thus, the Steiner minimum tree T_0 for S is also the Steiner minimum tree T_2 for S_2. Note that the Steiner minimum tree T_1 for S_1 consists of the vertex u only. Clearly, the distance of a vertex in T_{v_i} from T_0 is equal to its distance from T_2. The distance of a vertex $z \in T_v^{(i-1)}$ from T_0 is equal to its distance from v. Since $u \notin T_v^{(i)}$, it follows that $d(z,S) = d(z,S_1) - \delta(v,\sigma(v))$. ■

Lemma 4 *Assume that* $\sigma(v) = \sigma(v_i)$, $\sigma(v) \notin T_v^{(i)}$, *and that both* x_1 *and* x_2 *are undefined. Then,* $S = S_1 = S_2 = \{\sigma(v)\}$, $\mathrm{SMT}(S) = 0$, *and for all vertices* $z \in T_v^{(i)}$, $d(z,S) = \delta(z,\sigma(v))$.

Proof.
Note that S_1 and S_2 have exactly one vertex in common, the vertex $\sigma(v) = u$. Since $u \notin T_v^{(i)}$ and $x_1(x_2)$ is undefined it follows that $S_1(S_2)$ have no vertices that are in $T_v^{(i-1)}(T_{v_i})$. Thus, $S = S_1 = S_2 = \{u\}$. Thus, the Steiner minimum tree T_0 for S consists of vertex u only and has cost equal to 0. Further, $d(z,S) = \delta(z,u)$. ■

Lemmas 5–8 deal with the case where $\sigma(v)$ covers v_i and it belongs to one of the subtrees $T_v^{(i-1)}$ or T_{v_i}.

Lemma 5 *Assume that* $\sigma(v) = \sigma(v_i)$, $\sigma(v) \in T_v^{(i-1)}$, *and that* x_2 *is defined. Then,* $S = S_1 \cup S_2$, $S_1 \cap S_2 = \{\sigma(v)\}$,

$$\mathrm{SMT}(S) = \mathrm{SMT}(S_1) + \mathrm{SMT}(S_2) - \delta(x_1,\sigma(v)), \tag{10}$$

x_1 *is defined, and for all vertices* $z \in T_v^{(i)}$,

$$d(z,S) = \begin{cases} d(z,S_1), & \text{if } z \in T_{x_1} \\ d(z,S_1) - \delta(x_1,\mathrm{LCA}(x_1,z)), & \text{if } z \in T_v^{(i-1)} - T_{x_1} \\ d(z,S_2), & \text{otherwise.} \end{cases} \tag{11}$$

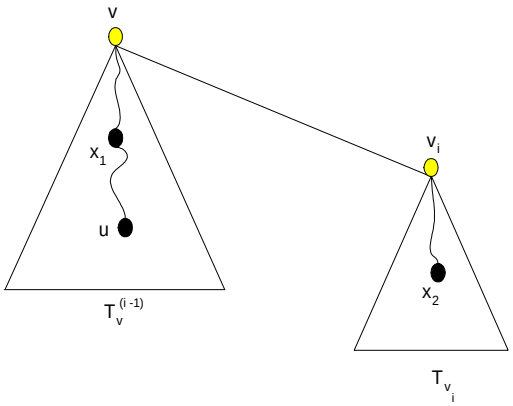

Figure 3: The subtree $T_v^{(i)}$ for Lemmas 5 and 6. Note that the vertex u covers both v and v_i, and u lies inside the subtree $T_v^{(i-1)}$.

Proof.
Note that S_1 and S_2 have exactly one vertex in common, the vertex $\sigma(v) = u$. Consider the Steiner minimum tree T_0 for S. It consists of a Steiner tree T_1 for S_1, a Steiner tree T_2 for $S_2 - u$, and the path $u \rightsquigarrow x_2$ in T. Moreover the tree T_1 is the Steiner minimum tree for S_1, and the tree T_2 together with the path $u \rightsquigarrow x_2$ is the Steiner minimum tree T_2' for S_2. Since the two SMTs T_1 and T_2' have only the edges on the path $u \rightsquigarrow x_1$ in common, it follows that $\mathrm{SMT}(S) = \mathrm{SMT}(S_1) + \mathrm{SMT}(S_2) - \delta(x_1, \sigma(v))$.

The distance of a vertex $z \in T_{v_i}$ from T_0 is equal to its distance from T_2'. The distance of any vertex $z \in T_{x_1}$ from T_0 is equal to its distance from T_1. Since the path $x_1 \rightsquigarrow x_2$ is included in T_0, the distance of any vertex $z \in T_v^{(i-1)} - T_{x_1}$ from T_0 is equal to its distance from the path $x_1 \rightsquigarrow x_2$. Thus, the distance of any vertex $z \in T_v^{(i-1)} - T_{x_1}$ from T_0 is equal to $d(z, S_1) - \delta(x_1, \mathrm{LCA}(x_1, z))$. ■

Lemma 6 *Assume that $\sigma(v) = \sigma(v_i)$, $\sigma(v) \in T_v^{(i-1)}$, and that x_2 is undefined. Then, $S_2 = \{\sigma(v)\}$, $\mathrm{SMT}(S_2) = 0$, $S = S_1$, $\mathrm{SMT}(S) = \mathrm{SMT}(S_1)$, x_1 is defined, and for all vertices $z \in T_v^{(i)}$,*

$$d(z, S) = \begin{cases} d(z, S_1), & \text{if } z \in T_v^{(i-1)} \\ d(z, S_2) - \delta(x_1, \sigma(v)), & \text{otherwise.} \end{cases} \tag{12}$$

Proof.
Note that S_1 and S_2 have exactly one vertex in common, the vertex $\sigma(v) = u$. Since x_2 is undefined, S_2 has no vertices which are in T_{v_i}. Since σ is a normal replica assignment, it follows that $S_2 = \{u\}$. Further, $S = S_1$. Consider the Steiner minimum tree T_0 for S. Tree T_0 is also the Steiner minimum tree for S_1. Note that the Steiner minimum tree T_2 for S_2 consists of the vertex u only. Clearly, $\mathrm{SMT}(S) = \mathrm{SMT}(S_1)$ and $\mathrm{SMT}(S_2) = 0$.

The distance of any vertex $z \in T_v^{(i-1)}$ from T_0 is equal to its distance from T_1, i.e. $d(z,S) = d(z,S_1)$ for all $z \in T_v^{(i-1)}$. The distance of any vertex $z \in T_{v_i}$ is equal to its distance from x_1, which is equal to its distance from u minus the distance between u and x_1. Hence, $d(z,S) = d(z,S_2) - \delta(x_1,u)$ for all $z \in T_{v_i}$. ■

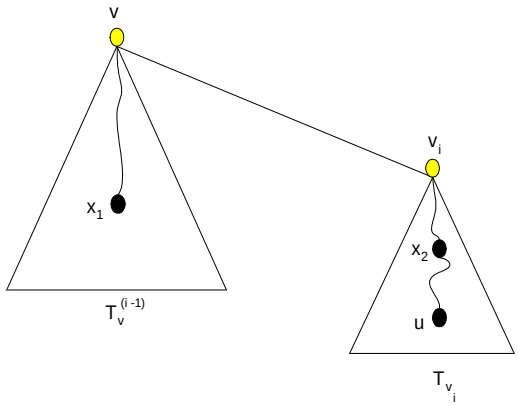

Figure 4: The subtree $T_v^{(i)}$ for Lemmas 7 and 8. Note that the vertex u covers both v and v_i, and u lies inside the subtree T_{v_i}.

Lemma 7 *Assume that $\sigma(v) = \sigma(v_i)$, $\sigma(v) \in T_{v_i}$, and that x_1 is defined. Then, $S = S_1 \cup S_2$, $S_1 \cap S_2 = \{\sigma(v)\}$, x_2 is defined,*

$$\mathrm{SMT}(S) = \mathrm{SMT}(S_1) + \mathrm{SMT}(S_2) - \delta(x_2, \sigma(v)), \tag{13}$$

and for all vertices $z \in T_v^{(i)}$,

$$d(z,S) = \begin{cases} d(z,S_1), & \text{if } z \in T_v^{(i-1)} \\ d(z,S_2), & \text{if } z \in T_{x_2} \\ d(z,S_2) - \delta(x_2, \mathrm{LCA}(x_2,z)), & \text{otherwise.} \end{cases} \tag{14}$$

Proof.
Note that S_1 and S_2 have exactly one vertex in common, the vertex $\sigma(v) = u$. Consider the Steiner minimum tree T_0 for S. It consists of a Steiner tree T_2 for S_2, a Steiner tree T_1 for $S_1 - u$, and the path $u \rightsquigarrow x_1$ in T. Moreover the tree T_2 is the Steiner minimum tree for S_2, and the tree T_1 together with the path $u \rightsquigarrow x_1$ is the Steiner minimum tree T_1' for S_1. Since the two SMTs T_1' and T_2 have only the edges on the path $u \rightsquigarrow x_2$ in common, it follows that $\mathrm{SMT}(S) = \mathrm{SMT}(S_1) + \mathrm{SMT}(S_2) - \delta(x_2, \sigma(v))$.

The distance of a vertex $z \in T_v^{(i-1)}$ from T_0 is equal to its distance from T_1'. The distance of any vertex $z \in T_{x_2}$ from T_0 is equal to its distance from T_2. Since the path $x_1 \rightsquigarrow x_2$ is

included in T_0, the distance of any vertex $z \in T_{v_i} - T_{x_2}$ from T_0 is equal to its distance from the path $x_1 \rightsquigarrow x_2$. Thus, the distance of any vertex $z \in T_{v_i} - T_{x_2}$ from T_0 is equal to $d(z, S_2) - \delta(x_2, \mathrm{LCA}(x_2, z))$. ■

Lemma 8 *Assume that $\sigma(v) = \sigma(v_i)$, $\sigma(v) \in T_{v_i}$, and that x_1 is undefined. Then, $S_1 = \{\sigma(v)\}$, $\mathrm{SMT}(S_1) = 0$, $S = S_2$, $\mathrm{SMT}(S) = \mathrm{SMT}(S_2)$, x_2 is defined, and for all vertices $z \in T_v^{(i)}$,*

$$d(z, S) = \begin{cases} d(z, S_1) - \delta(x_2, \sigma(v)), & \text{if } z \in T_v^{(i-1)} \\ d(z, S_2), & \text{otherwise.} \end{cases} \tag{15}$$

Proof.
Note that S_1 and S_2 have exactly one vertex in common, the vertex $\sigma(v) = u$. Since x_1 is not defined, S_1 has no vertices in $T_v^{(i-1)}$, and since σ is a normal replica assignment, $S_1 = \{u\}$, and $\mathrm{SMT}(S_1) = 0$. Further, $S = S_2$. Consider the Steiner minimum tree T_0 for S. Tree T_0 is also the Steiner minimum tree T_2 for S_2. Thus, $\mathrm{SMT}(S) = \mathrm{SMT}(S_2)$.

The distance of any vertex $z \in T_{v_i}$ from T_0 is equal to its distance from T_2, i.e. $d(z, S) = d(z, S_2)$. The distance of a vertex $z \in T_v^{(i-1)}$ from T_0 is equal to its distance from x_2. Thus,

$$d(z, S) = \delta(z, x_2) = \delta(z, u) - \delta(u, x_2) = d(z, S_1) - \delta(u, x_2). \tag{16}$$

■

Lemmas 9–11 deal with the case where v_i is not covered by $\sigma(v)$.

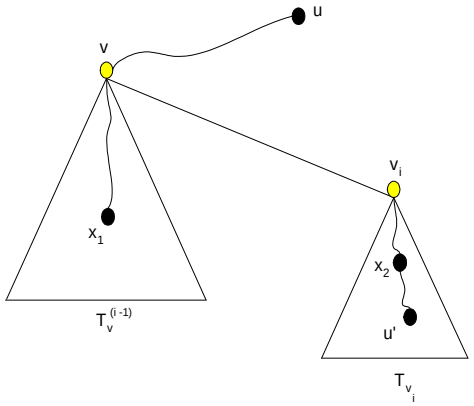

Figure 5: The subtree $T_v^{(i)}$ for Lemmas 9 and 10. Note that in this case, the vertex u covers v but not v_i, and u lies outside the subtree $T_v^{(i)}$. Let u' denote the vertex that covers v_i.

Lemma 9 *Assume that $\sigma(v) \neq \sigma(v_i)$, $\sigma(v) \notin T_v^{(i)}$, and that x_1 is defined. Then, $S = S_1 \cup S_2$, $S_1 \cap S_2 = \emptyset$, x_2 is defined,*

$$\mathrm{SMT}(S) = \mathrm{SMT}(S_1) + \mathrm{SMT}(S_2) + \delta(v, x_2), \tag{17}$$

and for all vertices $z \in T_v^{(i)}$,

$$d(z,S) = \begin{cases} d(z,S_1), & \text{if } z \in T_v^{(i-1)} \\ d(z,S_2), & \text{if } z \in T_{x_2} \\ d(z,S_2) - \delta(x_2, \mathrm{LCA}(x_2,z)), & \text{otherwise.} \end{cases} \tag{18}$$

Proof.
Since σ is a normal replica assignment, and $\sigma(v) \neq \sigma(v_i)$, and $\sigma(v) \notin T_v^{(i)}$, it follows that $S_1 \cap S_2 = \emptyset$. Clearly, $S = S_1 \cup S_2$. Consider the Steiner minimum tree T_0 for S. It consists of the Steiner tree T_1 for S_1, the Steiner tree T_2 for S_2, and the path $v \rightsquigarrow x_2$. Hence, $\mathrm{SMT}(S) = \mathrm{SMT}(S_1) + \mathrm{SMT}(S_2) + \delta(v, x_2)$.

The distance of a vertex $z \in T_v^{(i-1)}$ from T_0 is equal to its distance from T_1. The distance of any vertex $z \in T_{x_2}$ from T_0 is equal to its distance from T_2. Since the path $v \rightsquigarrow x_2$ is included in T_0, the distance of any vertex $z \in T_{v_i} - T_{x_2}$ from T_0 is equal to its distance from the path $v \rightsquigarrow x_2$. Thus, the distance of any vertex $z \in T_{v_i} - T_{x_2}$ from T_0 is equal to $d(z,S_2) - \delta(x_2, \mathrm{LCA}(x_2,z))$. ■

Lemma 10 *Assume that $\sigma(v) \neq \sigma(v_i)$, $\sigma(v) \notin T_v^{(i)}$, and that x_1 is undefined. Then, $S_1 = \{\sigma(v)\}$, $\mathrm{SMT}(S_1) = 0$, $S = S_1 \cup S_2$, $S_1 \cap S_2 = \emptyset$, x_2 is defined,*

$$\mathrm{SMT}(S) = \mathrm{SMT}(S_1) + \mathrm{SMT}(S_2) + \delta(\sigma(v), x_2), \tag{19}$$

and for all vertices $z \in T_v^{(i)}$,

$$d(z,S) = \begin{cases} d(z,S_1) - \delta(v,\sigma(v)), & \text{if } z \in T_v^{(i-1)} \\ d(z,S_2), & \text{if } z \in T_{x_2} \\ d(z,S_2) - \delta(x_2, \mathrm{LCA}(x_2,z)), & \text{otherwise.} \end{cases} \tag{20}$$

Proof.
Since σ is a normal replica assignment, and $\sigma(v) \neq \sigma(v_i)$, and $\sigma(v) \notin T_v^{(i)}$, it follows that $S_1 \cap S_2 = \emptyset$. Since x_1 is undefined, $S_1 = \{\sigma(v)\}$ and $\mathrm{SMT}(S_1) = 0$. Note that the Steiner tree T_1 for S_1 consists of the vertex $\sigma(v)$ only. Observe that $S_2 \subseteq T_{v_i}$. Clearly, $S = S_1 \cup S_2$. Consider the Steiner minimum tree T_0 for S. It consists of the Steiner tree T_1 for S_1, the

Steiner tree T_2 for S_2, and the path $\sigma(v) \rightsquigarrow x_2$. Hence, $\mathrm{SMT}(S) = \mathrm{SMT}(S_1) + \mathrm{SMT}(S_2) + \delta(\sigma(v), x_2)$.

The distance of a vertex $z \in T_v^{(i-1)}$ from T_0 is equal to its distance from the path $\sigma(v) \rightsquigarrow x_2$. The distance of any vertex $z \in T_{x_2}$ from T_0 is equal to its distance from T_2. Since the path $\sigma(v) \rightsquigarrow x_2$ is included in T_0, the distance of any vertex $z \in T_{v_i} - T_{x_2}$ from T_0 is equal to its distance from the path $\sigma(v) \rightsquigarrow x_2$. Thus, the distance of any vertex $z \in T_{v_i} - T_{x_2}$ from T_0 is equal to $d(z, S_2) - \delta(x_2, \mathrm{LCA}(x_2, z))$. ■

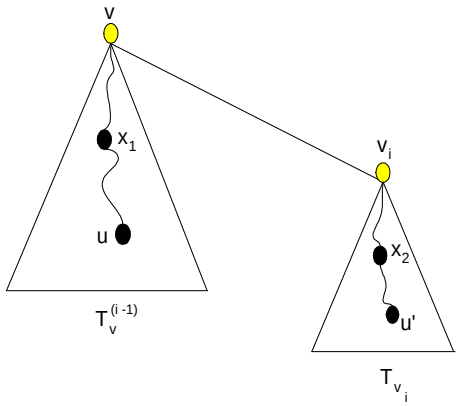

Figure 6: The subtree $T_v^{(i)}$ for Lemmas 11. Note that the vertex u covers v but not v_i, and u lies inside the subtree $T_v^{(i)}$. Let u' denote the vertex that covers v_i.

Lemma 11 *Assume that $\sigma(v) \neq \sigma(v_i)$, $\sigma(v) \in T_v^{(i)}$. Then, $\sigma(v) \in T_v^{(i-1)}$, $S = S_1 \cup S_2$, $S_1 \cap S_2 = \emptyset$, x_1 and x_2 are defined,*

$$\mathrm{SMT}(S) = \mathrm{SMT}(S_1) + \mathrm{SMT}(S_2) + \delta(x_1, x_2), \tag{21}$$

and for all vertices $z \in T_v^{(i)}$,

$$d(z, S) = \begin{cases} d(z, S_1), & \text{if } z \in T_{x_1} \\ d(z, S_1) - \delta(x_1, \mathrm{LCA}(x_1, z)), & \text{if } z \in T_v^{(i-1)} - T_{x_1} \\ d(z, S_2), & \text{if } z \in T_{x_2} \\ d(z, S_2) - \delta(x_2, \mathrm{LCA}(x_2, z)), & \text{otherwise.} \end{cases} \tag{22}$$

Proof.
Since σ is a normal replica assignment, and $\sigma(v) \neq \sigma(v_i)$, and $\sigma(v) \in T_v^{(i)}$, it follows that $\sigma(v) \in T_v^{(i-1)}$ $S_2 \subseteq T_{v_i}$, and $S_1 \cap S_2 = \emptyset$. Thus, x_1 and x_2 are both defined. Clearly, $S = S_1 \cup S_2$.

Consider the Steiner minimum tree T_0 for S. It consists of the Steiner tree T_1 for S_1, the Steiner tree T_2 for S_2, and the path $x_1 \rightsquigarrow x_2$. Hence, $\mathrm{SMT}(S) = \mathrm{SMT}(S_1) + \mathrm{SMT}(S_2) + \delta(x_1, x_2)$.

The distance of a vertex $z \in T_{x_1}$ from T_0 is equal to its distance from the path T_1. The distance of a vertex $z \in T_v^{(i-1)}$ from T_0 is equal to its distance from the path $x_1 \rightsquigarrow x_2$, which is equal to to the distance of z from T_1 minus the length of the path from the $\mathrm{LCA}(z, x_1)$ to x_1, i.e. $d(z, S) = d(z, S_1) - \delta(x_1, \mathrm{LCA}(z, x_1))$. Similarly, the distance of a vertex $z \in T_{x_2}$ from T_0 is equal to its distance from the path T_2, and the distance of a vertex $z \in T_{v_i} - T_{x_2}$ is equal to its distance from the path $x_1 \rightsquigarrow x_2$, which is equal to $d(z, S_2) - \delta(x_2, \mathrm{LCA}(z, x_2))$. ■

Finally, using Lemmas 1–11 we formulate the following set of recursive equations between the total costs of the replica assignments $\sigma_v^{(i)}$, $\sigma_v^{(i-1)}$, and σ_{v_i}.

Theorem 1 *Let σ be a normal replica assignment for a tree T. Let $\sigma_v^{(i)}$, $\sigma_v^{(i-1)}$, and σ_{v_i} be the restrictions of σ to the subtrees $T_v^{(i)}$, $T_v^{(i-1)}$, and T_{v_i} of T respectively, $1 \le i \le \deg(v)$. Let S, S_1, and S_2 be the replica sets associated with $\sigma_v^{(i)}$, $\sigma_v^{(i-1)}$, and σ_{v_i} respectively. Let x, x_1, x_2 be the least common ancestor of $S \cap T_v^{(i)}$, $S_1 \cap T_v^{(i-1)}$, and $S_2 \cap T_{v_i}$, respectively (whenever they are defined). Let k, k_1, k_2 be the size of S, S_1, S_2 respectively. There are a number of possible cases:*

1. *if $\sigma(v) = \sigma(v_i)$, $\sigma(v) \notin T_v^{(i)}$ then,*

   (a) *if $x_1 \neq \emptyset$, and $x_2 \neq \emptyset$,*

   $$C(\sigma_v^{(i)}) = C(\sigma_v^{(i-1)}) + C(\sigma_{v_i}) - \mathrm{s}(\sigma(v)) - W_{\mathrm{T}} \cdot \delta(v, \sigma(v)), \tag{23}$$

   $k = k_1 + k_2 - 1$, $k_1 \ge 2$, $k_2 \ge 2$, *and* $x = \mathrm{LCA}(x_1, x_2)$.

   (b) *if $x_1 \neq \emptyset$, and $x_2 = \emptyset$,*

   $$C(\sigma_v^{(i)}) = C(\sigma_v^{(i-1)}) + C(\sigma_{v_i}) - \mathrm{s}(\sigma(v)) - \sum_{z \in T_{v_i}} \mathrm{w}(z) \cdot \delta(v, \sigma(v)), \tag{24}$$

   $k = k_1 \ge 2$, $k_2 = 1$, *and* $x = x_1$.

   (c) *if $x_1 = \emptyset$, and $x_2 \neq \emptyset$,*

   $$C(\sigma_v^{(i)}) = C(\sigma_v^{(i-1)}) + C(\sigma_{v_i}) - \mathrm{s}(\sigma(v)) - \sum_{z \in T_v^{(i-1)}} \mathrm{w}(z) \cdot \delta(v, \sigma(v)), \tag{25}$$

   $k = k_2 \ge 2$, $k_1 = 1$, *and* $x = x_2$.

   (d) *if $x_1 = \emptyset$, and $x_2 = \emptyset$,*

   $$C(\sigma_v^{(i)}) = C(\sigma_v^{(i-1)}) + C(\sigma_{v_i}) - \mathrm{s}(\sigma(v)), \tag{26}$$

   $k = k_1 = k_2 = 1$, *and* $x = \emptyset$.

2. if $\sigma(v) = \sigma(v_i)$, $\sigma(v) \in T_v^{(i-1)}$ *then,*

(a) if $x_2 \neq \emptyset$,

$$\begin{aligned} C(\sigma_v^{(i)}) &= C(\sigma_v^{(i-1)}) + C(\sigma_{v_i}) - \mathrm{s}(\sigma(v)) - W_{\mathrm{T}} \cdot \delta(x_1, \sigma(v)) \\ &\quad - \sum_{z \in T_v^{(i-1)} - T_{x_1}} \mathrm{w}(z) \cdot \delta(x_1, \mathrm{LCA}(x_1, z)), \end{aligned} \tag{27}$$

$k = k_1 + k_2 - 1$, $k_1 \geq 1$, $k_2 \geq 2$, $x_1 \neq \emptyset$, *and* $x = \mathrm{LCA}(x_1, x_2)$.

(b) if $x_2 = \emptyset$,

$$C(\sigma_v^{(i)}) = C(\sigma_v^{(i-1)}) + C(\sigma_{v_i}) - \mathrm{s}(\sigma(v)) - \sum_{z \in T_{v_i}} \mathrm{w}(z) \cdot \delta(x_1, \sigma(v)), \tag{28}$$

$k = k_1$, $k_1 \geq 1$, $k_2 = 1$, $x_1 \neq \emptyset$, *and* $x = x_1$.

3. if $\sigma(v) = \sigma(v_i)$, $\sigma(v) \in T_{v_i}$ *then,*

(a) if $x_1 \neq \emptyset$,

$$\begin{aligned} C(\sigma_v^{(i)}) &= C(\sigma_v^{(i-1)}) + C(\sigma_{v_i}) - \mathrm{s}(\sigma(v)) - W_{\mathrm{T}} \cdot \delta(x_2, \sigma(v)) \\ &\quad - \sum_{z \in T_{v_i} - T_{x_2}} \mathrm{w}(z) \cdot \delta(x_2, \mathrm{LCA}(x_2, z)), \end{aligned} \tag{29}$$

$k = k_1 + k_2 - 1$, $k_1 \geq 2$, $k_2 \geq 1$, $x_2 \neq \emptyset$, *and* $x = \mathrm{LCA}(x_1, x_2)$.

(b) if $x_1 = \emptyset$,

$$C(\sigma_v^{(i)}) = C(\sigma_v^{(i-1)}) + C(\sigma_{v_i}) - \mathrm{s}(\sigma(v)) - \sum_{z \in T_v^{(i-1)}} \mathrm{w}(z) \cdot \delta(x_2, \sigma(v)), \tag{30}$$

$k = k_2$, $k_1 = 1$, $k_2 \geq 1$, $x_2 \neq \emptyset$, *and* $x = x_2$.

4. if $\sigma(v) \neq \sigma(v_i)$, $\sigma(v) \notin T_{v_i}$ *then,*

(a) if $x_1 \neq \emptyset$,

$$\begin{aligned} C(\sigma_v^{(i)}) &= C(\sigma_v^{(i-1)}) + C(\sigma_{v_i}) + W_{\mathrm{T}} \cdot \delta(v, x_2) \\ &\quad - \sum_{z \in T_{v_i} - T_{x_2}} \mathrm{w}(z) \cdot \delta(x_2, \mathrm{LCA}(x_2, z)), \end{aligned} \tag{31}$$

$k = k_1 + k_2$, $k_1 \geq 2$, $k_2 \geq 1$, $x_2 \neq \emptyset$, *and* $x = \mathrm{LCA}(x_1, x_2)$.

(b) if $x_1 = \emptyset$,

$$\begin{aligned} C(\sigma_v^{(i)}) &= C(\sigma_v^{(i-1)}) + C(\sigma_{v_i}) + W_{\mathrm{T}} \cdot \delta(\sigma(v), x_2) \\ &\quad - \sum_{z \in T_v^{(i-1)}} \mathrm{w}(z) \cdot \delta(v, \sigma(v)) - \sum_{z \in T_{v_i} - T_{x_2}} \mathrm{w}(z) \cdot \delta(x_2, \mathrm{LCA}(x_2, z)), \end{aligned} \tag{32}$$

$k = k_1 + k_2$, $k_1 = 1$, $k_2 \geq 1$, $x_2 \neq \emptyset$, *and* $x = x_2$.

5. *if* $\sigma(v) \neq \sigma(v_i)$, *and* $\sigma(v) \in T_{v_i}$, *then*

$$\begin{aligned} C(\sigma_v^{(i)}) &= C(\sigma_v^{(i-1)}) + C(\sigma_{v_i}) + W_{\mathrm{T}} \cdot \delta(x_1, x_2) \\ &\quad - \sum_{z \in T_v^{(i-1)} - T_{x_1}} \mathrm{w}(z) \cdot \delta(x_1, \mathrm{LCA}(x_1, z)) - \sum_{z \in T_{v_i} - T_{x_2}} \mathrm{w}(z) \cdot \delta(x_2, \mathrm{LCA}(x_2, z)), \end{aligned} \tag{33}$$

$\sigma(v) \in T_v^{(i-1)}$, $k = k_1 + k_2$, $k_1 \geq 1$, $k_2 \geq 1$, $x_1, x_2 \neq \emptyset$, *and* $x = \mathrm{LCA}(x_1, x_2)$.

Proof.
There are several cases to consider. Each case 1–11 follows from the corresponding Lemma 1–11, and the definitions of the read, write, storage cost of a replica assignment.

We provide the detailed argument for Case 1 only. Since $\sigma_v^{(i)}$, $\sigma_v^{(i-1)}$, and σ_{v_i} are restrictions of σ, it follows that the read cost of $\sigma_v^{(i)}$ is

$$C_a(\sigma_v^{(i)}) = C_a(\sigma_v^{(i-1)}) + C_a(\sigma_{v_i}). \tag{34}$$

From Lemma 1, we have that $\mathrm{SMT}(S) = \mathrm{SMT}(S_1) + \mathrm{SMT}(S_2) - \delta(v, \sigma(v))$, and that $d(z, S) = d(z, S_1)$, if $z \in T_v^{(i-1)}$, and that $d(z, S) = d(z, S_2)$, if $z \in T_{v_i}$. Hence, the write cost of $\sigma_v^{(i)}$ is

$$C_u(\sigma_v^{(i)}) = C_u(\sigma_v^{(i-1)}) + C_u(\sigma_{v_i}) - W_{\mathrm{T}} \cdot \delta(v, \sigma(v)). \tag{35}$$

From Lemma 1, we also have that $S = S_1 \cup S_2$ and that $S_1 \cap S_2 = \{\sigma(v)\}$. Thus, the storage cost of $\sigma_v^{(i)}$ is

$$C_s(\sigma_v^{(i)}) = C_s(\sigma_v^{(i-1)}) + C_s(\sigma_{v_i}) - \mathrm{s}(\sigma(v)). \tag{36}$$

Equation (23) now follows from the definition of the total cost of a replica assignment. Since $S = S_1 \cup S_2$ and that $|S_1 \cap S_2| = 1$, it follows that $k = k_1 + k_2 - 1$. Since $\sigma(v) \notin T_v^{(i)}$ and $x_1, x_2 \neq \emptyset$, S_1 and S_2 contain at least one vertex that is different than $\sigma(v)$, which implies that $k_1, k_2 \geq 2$. Further, by definition of least common ancestors, and since both x_1 and x_2 are defined, it follows that $x = \mathrm{LCA}(x_1, x_2)$.

The remaining 10 cases can be proved using similar arguments and are left to the interested reader. ■

4 Finding Steiner-Optimal Replica Sets

We describe a dynamic programming algorithm for finding Steiner-optimal replica sets for T, as well as finding Steiner-optimal k–replica sets for T, for a given integer $1 \leq k \leq n$.

Consider the subtree $T_v^{(i)}$ of T rooted at v. It consists of v and the subtrees of T rooted at each one of the i leftmost children of v, $0 \leq i \leq \deg(v)$. Let $\sigma_v^{(i)}$ be a (normal) replica assignment for $T_v^{(i)}$. We call $\sigma_v^{(i)}$ a Steiner–optimal (u, x, k)–replica assignment (or replica set) for $T_v^{(i)}$, if $\sigma_v^{(i)}$ has the *minimum* total cost among all possible replica assignments such that x is the least common ancestor of the replica vertices that are in the subtree $T_v^{(i)}$ (or *undefined* if there are no replica vertices within $T_v^{(i)}$), k is the size of the replica set associated with $\sigma_v^{(i)}$, and $\sigma(v) = u$.

We denote the cost of a Steiner-optimal (u, x, k)–replica assignment for $T_v^{(i)}$ as $R(u, k, x, v, i)$. Thus

$$R(u, k, x, v, i) = C(\sigma_v^{(i)}), \tag{37}$$

where x is the least common ancestor of the replica vertices in $T_v^{(i)}$ (or *undefined* if there are no replica vertices within $T_v^{(i)}$), k is the size of the replica set associated with $\sigma_v^{(i)}$, and $\sigma(v) = u$.

Corollary 1 *Let σ be a normal replica assignment for a tree T. Let $\sigma_v^{(i)}$, $\sigma_v^{(i-1)}$, σ_{v_i}, k, k_1, k_2 and x, x_1, x_2 be as in Theorem 1. $\sigma_v^{(i)}$ is a Steiner–optimal $(\sigma_v^{(i)}(v), k, x)$–replica assignment for $T_v^{(i)}$ if and only if $\sigma_v^{(i-1)}$ and σ_{v_i} are Steiner–optimal $(\sigma_v^{(i-i)}(v), k_1, x_1)$– and $(\sigma_{v_i}(v_i), k_2, x_2)$–replica assignments for $T_v^{(i-1)}$ and T_{v_i} respectively, where $x = LCA(x_1, x_2)$, and*

$$k = \begin{cases} k_1 + k_2 - 1, & \text{if } \sigma(v) = \sigma(v_i) \\ k_1 + k_2, & \text{if } \sigma(v) \neq \sigma(v_i) \end{cases}$$

Proof sketch. Follows immediately from Theorem 1. ■

We now show a recurrence for computing $R(u, k, x, v, i)$ in the following theorem.

Theorem 2 *Let v be any vertex of T, and v_i be the i–th child of v. Let x be $\emptyset$ or any vertex of $T_v^{(i)}$, and let u be any vertex of T, $0 \leq i \leq \deg(v)$. The cost of a Steiner-optimal (u, x, k)–replica assignment for $T_v^{(i)}$ is given by*

$$R(u, k, x, v, i) = \begin{cases} X_0, & \text{if } k = 1 \text{ and } i \geq 0 \\ \min\{X_1, X_2\} & \text{if } u \notin T_v^{(i)},\ 1 < k \leq |T_v^{(i)} \cup \{u\}| \text{ and } i > 0 \\ X_3, & \text{if } u \in T_{v_i},\ 1 < k \leq |T_v^{(i)} \cup \{u\}| \text{ and } i > 0 \\ \min\{X_4, X_5\} & \text{if } u \in T_v^{(i)},\ 1 < k \leq |T_v^{(i)} \cup \{u\}| \text{ and } i > 0 \\ \textit{undefined}, & \text{if } u \in T_v^{(i)} \text{ and } u \notin T_x \\ \textit{undefined}, & \textit{otherwise}, \end{cases} \tag{38}$$

where

$$X_0 = \sum_{z \in T_v^{(i)}} \mathrm{r}(z) \cdot \delta(z, u) + \sum_{z \in T_v^{(i)}} \mathrm{w}(z) \cdot \delta(z, u) + \mathrm{s}(u) \tag{39}$$

$$X_1 = \min\{R_1, R_2, R_3\}, \tag{40}$$

$$X_2 = \min\{R_4, R_5\}, \tag{41}$$

$$X_3 = \min\{R_6, R_7\}, \tag{42}$$

$$X_4 = \min\{R_8, R_9\}, \tag{43}$$

$$X_5 = R_{10}, \tag{44}$$

and,

- $$R_1 = \min_{k_1, x_1, x_2} \{R(u, k_1, x_1, v, i-1) + R(u, k - k_1 + 1, x_2, v_i, \deg(v_i)) - \mathrm{s}(u) - W_{\mathrm{T}} \cdot \delta(v, u)\}, \tag{45}$$

  where $1 < k_1 < k$, $x_1 \in T_v^{(i-1)}$, $x_2 \in T_{v_i}$, $x = \mathrm{LCA}(x_1, x_2)$,

- $$R_2 = R(u, k, x, v, i-1) + R(u, 1, \emptyset, v_i, \deg(v_i)) - \mathrm{s}(u) - \sum_{z \in T_{v_i}} \mathrm{w}(z) \cdot \delta(v, u), \tag{46}$$

  where $x \in T_v^{(i-1)}$,

- $$R_3 = R(u, 1, \emptyset, v, i-1) + R(u, k, x, v_i, \deg(v_i)) - \mathrm{s}(u) - \sum_{z \in T_v^{(i-1)}} \mathrm{w}(z) \cdot \delta(v, u), \tag{47}$$

  where $x \in T_{v_i}$,

- $$R_4 = \min_{k_1, x_1, x_2, u'} \{R(u, k_1, x_1, v, i-1) + R(u', k - k_1, x_2, v_i, \deg(v_i)) + W_{\mathrm{T}} \cdot \delta(v, x_2) - \sum_{z \in T_{v_i} - T_{x_2}} \mathrm{w}(z) \cdot \delta(\mathrm{LCA}(z, x_2), x_2)\}, \tag{48}$$

  where $1 < k_1 < k$, $x_1 \in T_v^{(i-1)}$, $x_2 \in T_{v_i}$, $x = \mathrm{LCA}(x_1, x_2)$, *and* $u' \in T_{x_2}$,

- $$R_5 = \min_{u'} \{R(u, 1, \emptyset, v, i-1) + R(u', k-1, x, v_i, \deg(v_i)) + W_{\mathrm{T}} \cdot \delta(u, x) - \sum_{z \in T_v^{(i-1)}} \mathrm{w}(z) \cdot \delta(v, u) - \sum_{z \in T_{v_i} - T_x} \mathrm{w}(z) \cdot \delta(\mathrm{LCA}(z, x), x)\}, \tag{49}$$

  where $x \in T_{v_i}$ *and* $u' \in T_x$,

- $$R_6 = \min_{k_1,x_1,x_2} \{R(u,k_1,x_1,v,i-1) + R(u,k-k_1+1,x_2,v_i,\deg(v_i)) - \mathrm{s}(u) - W_{\mathrm{T}} \cdot \delta(x_2,u) - \sum_{z \in T_{v_i} - T_{x_2}} \mathrm{w}(z) \cdot \delta(\mathrm{LCA}(z,x_2),x_2)\}, \quad (50)$$

  where $1 < k_1 \leq k$, $x_1 \in T_v^{(i-1)}$, $x_2 \in T_{v_i}$, $x = \mathrm{LCA}(x_1,x_2)$, *and* $u \in T_{x_2}$,

- $$R_7 = R(u,1,\emptyset,v,i-1) + R(u,k,x,v_i,\deg(v_i)) - \mathrm{s}(u) - \sum_{z \in T_v^{(i-1)}} \mathrm{w}(z) \cdot \delta(x,u), \quad (51)$$

  where $x \in T_{v_i}$,

- $$R_8 = \min_{k_1,x_1,x_2} \{R(u,k_1,x_1,v,i-1) + R(u,k-k_1+1,x_2,v_i,\deg(v_i)) - \mathrm{s}(u) - W_{\mathrm{T}} \cdot \delta(x_1,u) - \sum_{z \in T_v^{(i-1)} - T_{x_1}} \mathrm{w}(z) \cdot \delta(\mathrm{LCA}(z,x_1),x_1)\}, \quad (52)$$

  where $1 \leq k_1 < k$, $x_1 \in T_v^{(i-1)}$, $x_2 \in T_{v_i}$, $x = \mathrm{LCA}(x_1,x_2)$, *and* $u \in T_{x_1}$,

- $$R_9 = R(u,k,x,v,i-1) + R(u,1,\emptyset,v_i,\deg(v_i)) - \mathrm{s}(u) - \sum_{z \in T_{v_i}} \mathrm{w}(z) \cdot \delta(x,u), \quad (53)$$

  where $x \in T_v^{(i-1)}$,

- $$R_{10} = \min_{k_1,x_1,x_2,u'} \{R(u,k_1,x_1,v,i-1) + R(u',k-k_1,x_2,v_i,\deg(v_i)) + W_{\mathrm{T}} \cdot \delta(x_1,x_2) - \sum_{z \in T_v^{(i-1)} - T_{x_1}} \mathrm{w}(z) \cdot \delta(\mathrm{LCA}(z,x_1),x_1) - \sum_{z \in T_{v_i} - T_{x_2}} \mathrm{w}(z) \cdot \delta(\mathrm{LCA}(z,x_2),x_2)\}, \quad (54)$$

  where $1 \leq k_1 < k$, $x_1 \in T_v^{(i-1)}$, $x_2 \in T_{v_i}$, $x = \mathrm{LCA}(x_1,x_2)$, $u \in T_{x_1}$ *and* $u' \in T_{x_2}$.

Furthermore, the cost of a Steiner-optimal k–replica assignment (set) for T_v is given by

$$\min\{R(u,k,x,v,\deg(v)) \ : \ u,x \in T_v\}, \quad (55)$$

the cost of a Steiner-optimal k–replica assignment (set) for T is given by

$$\min\{R(u,k,x,\mathrm{root}(T),\deg(\mathrm{root}(T))) \ : \ u,x \in T\}, \quad (56)$$

and the cost of a Steiner-optimal replica assignment (set) for T is given by

$$\min\{R(u,k,x,\mathrm{root}(T),\deg(\mathrm{root}(T))) \ : \ u,x \in T,\ 1 \leq k \leq |T|\ \}. \quad (57)$$

Proof.
The proof of Theorem 2 is based on the cost formulations in Theorem 1 and Corollary 1.

First we show the initial conditions for $R(u, k, x, v, i)$. By definition of the cost of a Steiner–optimal (u, x, k)–replica assignment, we have that,

$$R(u, 1, x, v, i) = \sum_{z \in T_v^{(i)}} \mathrm{r}(z) \cdot \delta(z, u) + \sum_{z \in T_v^{(i)}} \mathrm{w}(z) \cdot \delta(z, u) + \mathrm{s}(u),$$

for $0 \leq i \leq \deg(v)$ and $R(u, k, x, v, 0)$ is undefined for $k \neq 1$. Furthermore, $R(u, k, x, v, i)$ is undefined when $k \leq 0$, or $k > |T_v^{(i+1)} \cup \{u\}|$.

Second, we show how to compute $R(u, k, x, v, i)$ recursively for $k > 1$.

Let $\sigma_v^{(i)}$ be a (normal) Steiner–optimal (u, x, k)–replica assignment for $T_v^{(i)}$, and $\sigma_v^{(i-1)}$, σ_{v_i} be the restrictions of $\sigma_v^{(i)}$ to the subtrees $T_v^{(i-1)}$ and T_{v_i} of T respectively, $1 \leq i \leq \deg(v)$. Let S, S_1, and S_2 be the replica sets associated with $\sigma_v^{(i)}$, $\sigma_v^{(i-1)}$, and σ_{v_i} respectively. Let x, x_1, x_2 be the least common ancestor of $S \cap T_v^{(i)}$, $S_1 \cap T_v^{(i-1)}$, and $S_2 \cap T_{v_i}$, respectively (whenever they are defined). Let k, k_1, k_2 be the size of S, S_1, S_2 respectively. The cost of the replica assignment $\sigma_v^{(i)}$ is $R(u, k, x, v, i)$. From Theorem 1, we observe that the formulation of $R(u, k, x, v, i)$ really depends upon the position of the vertex $\sigma_v^{(i)}(v) = u$ with respect to the subtree $T_v^{(i)}$, i.e. whether u lies inside or outside the subtree $T_v^{(i)}$, and whether x_1 and x_2 are defined or not. We consider the case where $u \notin T_v^{(i)}$. In this case there are two possibilities – either u covers both v and v_i, or u covers v but not v_i.

Let us assume that u covers both v and v_i and x_1, x_2 are both defined. Then the cost of the replica assignment $\sigma_v^{(i)}$ is given by Equation (23) in Theorem 1. Moreover, from Corollary 1 we know that $\sigma_v^{(i)}$ is Steiner–optimal if and only if $\sigma_v^{(i-1)}$ and σ_{v_i} are Steiner–optimal (u, k_1, x_1)– and (u, k_2, x_2)–replica assignments for $T_v^{(i-1)}$ and T_{v_i} respectively. Finally since $k = k_1 + k_2 - 1$, it is easy to see that the formulation of $R(u, k, x, v, i)$ can be given by the minimization in Equation (44).

When $x_1 \neq \emptyset$ and x_2 is undefined, the cost of the replica assignment $\sigma_v^{(i)}$ is given by Equation (24) in Theorem 1. In this case, the replica set S_2 consists only of the vertex u, $x = x_1$ and $R(u, k, x, v, i)$ is given by Equation (45). Similarly, when $x_1 = \emptyset$ and x_2 is defined, the cost of $\sigma_v^{(i)}$ is given by Equation (46).

Let us now consider the case when u covers v but not v_i. We assume that u' is the vertex that covers v_i i.e. $\sigma(v_i) = u'$. If x_1 and x_2 are both defined, the cost of replica assignment for $\sigma_v^{(i)}$ is given by Equation (31) in Theorem 1. From Corollary 1 again, $\sigma_v^{(i)}$ is Steiner–optimal if and only if $\sigma_v^{(i-1)}$ and σ_{v_i} are Steiner–optimal (u, k_1, x_1)– and (u', k_2, x_2)–replica assignments for $T_v^{(i-1)}$ and T_{v_i} respectively. Note that in this case $k = k_1 + k_2$. Hence, the formulation of $R(u, k, x, v, i)$ in Equation (47) follows. Further when x_1 is undefined, the cost of the replica assignment $\sigma_v^{(i)}$ is given by Equation (32) in Theorem 1. In this case, the replica set S_1 consists only of the vertex u, $x = x_2$ and $R(u, k, x, v, i)$ is given by Equation (48).

Similar arguments hold for the cases when $u \in T_v^{(i-1)}$ and $u \in T_{v_i}$, and the details are left to the interested readers. ■

Finally, we analyze the complexity of the algorithm for computing Steiner-optimal replica sets for a tree with storage costs in the following theorem.

Theorem 3 *Let T be a tree with n vertices and k an integer constant $1 \leq k \leq n$. We can find a Steiner-optimal k–replica set for T in $O(n^6k^2)$ time, and a Steiner–optimal replica set for T in $O(n^8)$ time.*

Proof.
Clearly, equation (38) in Theorem 2 can be evaluated by a straightforward dynamic programming procedure, either using memoization, or bottom–up. The running–time of such a dynamic programming algorithm for computing a Steiner–optimal replica set of size k for subtrees $T_v^{(i)}, v \in T, 0 \leq i \leq \deg(v)$ via equation (38) is $O(n^6k^2)$. Note that $R(u, k, x, v, i)$ is defined for at most n^3k entries, and each entry requires $O(n^3k)$ time. Hence, we can compute a Steiner-optimal k–replica set for T in $O(n^6k^2)$ time, and a Steiner-optimal replica set for T in $O(n^8)$ time. ■

5 Relationship of write policy to replicated data placement

Consider the following natural questions. Is an optimal replica set when the MST–write policy is used, also optimal when the SMT–write policy is used? Is an optimal replica set when the SMT–write policy is used, also optimal when the MST–policy is used? Answers to these questions shed light on the role of the write policy on the optimality of replica sets.

As discussed in previous sections, the model in this paper assumes the *SMT-write* policy and the replica sets computed are the Steiner-optimal replica sets. This is different from the *MST-write* policy used by Kalpakis et al [10]; the replica sets computed in [10] are MST-optimal. We provide evidence which shows that the optimality of a replica set is inherently dependent on the write policy, i.e. an MST-optimal replica set is not necessarily a Steiner-optimal replica set and similarly a Steiner-optimal replica set is not always an MST-optimal replica set. We further show that the use of the SMT-write policy can decrease the total cost of a replica set by as much as 33%, when compared to the total cost of the same replica set using the MST-write policy.

For each of the following three examples, we consider the tree communication network shown in Figure 7. For each example, we compute (i) an MST-optimal replica set using a dynamic programming algorithm described in [10] and (ii) a Steiner-optimal replica set using Theorem 2. Finally, we calculate the costs of both the replica sets for the MST-write policy as well as the SMT-write policy.

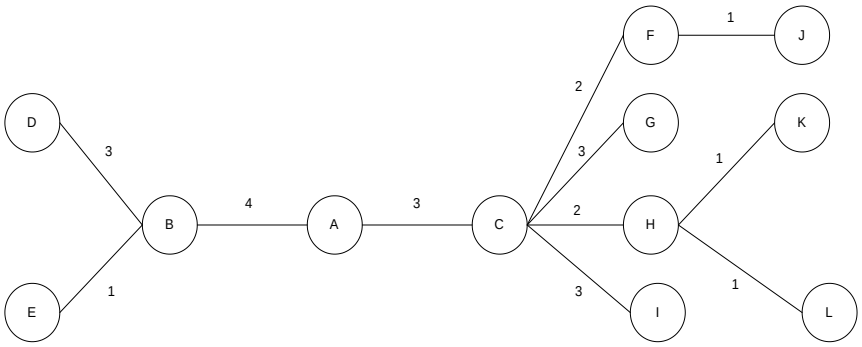

Figure 7: Tree network used in Examples 1-3. The edge costs are shown next to each edge.

5.1 Example 1.

For this example, the read-write requests issued by the various nodes and the corresponding storage costs are given in Table 1. We compute an MST-optimal replica set and a Steiner-optimal replica set. Let S_{MST} and S_{SMT} denote such an MST-optimal and a Steiner-optimal replica set, respectively. Interestingly, for the given read-write-storage costs, both the replica sets are exactly identical and consist of the nodes $\{A, G, I, J, L\}$. The costs of the two replica sets using the MST-write and SMT-write policies are given in Table 2. Note that the total costs of both the replica sets, when using the SMT-write policy, are about 33% lower than the corresponding costs when the MST-write policy is used.

Node	Number of Reads	Number of Writes	Storage Cost
A	1000	6	1
B	12	100	1000
C	2	1	2575
D	3	6	1
E	13	62	5
F	12	100	2000
G	2000	0	1
H	1	5	1000
I	405	1	2
J	2000	1	1
K	100	1	200
L	1222	1	1

Table 1: Number of read and write requests, and storage costs used in Example 1.

	MST-Write Policy	SMT-Write Policy
S_{MST}	8037 (1.000)	5372 (0.668)
S_{SMT}	8037 (1.000)	5372 (0.668)

Table 2: Total costs of the S_{MST} and S_{SMT} replica sets using the MST-write and the SMT-write policies.[4]

5.2 Example 2.

The read, write requests and the storage costs for this example are given by Table 3. In this case, an MST-optimal replica set is $S_{MST} = \{A, C, G, I, J, K\}$, while a Steiner-optimal replica set is $S_{SMT} = \{A, G, I, J, K\}$. Note that S_{SMT} is a subset of S_{MST}. Table 4 gives the total costs of the two replica sets using either the MST-write or the SMT-write policy. We observe that, if the SMT-write policy is used, the replica set S_{SMT} incurs a total cost that is about 32% lower than that of the replica set S_{MST}. However, if the MST-write policy is used, the replica set S_{MST} gives lower cost than the replica set S_{SMT}. Further, the total cost of the replica set S_{SMT}, using an SMT-write policy, is about 33% lower than its total cost using an MST-write policy.

Node	Number of Reads	Number of Writes	Storage Cost
A	1000	6	1
B	12	200	5
C	2	10	3139
D	3	68	1
E	1	6	5
F	1	1	1000
G	1000	0	1
H	1	5	2500
I	1000	1	100
J	1000	0	1
K	1000	1	0
L	1	1	1

Table 3: Number of read and write requests, and storage costs used in Example 2.

	MST-Write Policy	SMT-Write Policy
S_{MST}	9929 (1.000)	9922 (0.993)
S_{SMT}	10003 (1.007)	6789 (0.683)

Table 4: Total costs of the S_{MST} and S_{SMT} replica sets using the MST-write and the SMT-write policies.[4]

[4] The value in parentheses gives the ratio of the total cost of the replica set using the corresponding write policy, to the total cost of the S_{MST} replica set using the MST-write policy.

5.3 Example 3.

For this example, the read/write requests and the storage costs are given in Table 5. In this case, $S_{MST} = \{A, C\}$ and $S_{SMT} = \{A, G, I, J, K\}$. The total costs of the two replica sets using either the MST-write or the SMT-write policy are given in Table 6. Note that if the SMT-write policy is used, S_{SMT} incurs a total cost that is about 30% lower than that of S_{MST}. However, if the MST-write policy is used, the replica set S_{MST} gives lower cost than the replica set S_{SMT}. Finally, the total cost of the replica set S_{SMT}, using the SMT-write policy, is about 30% lower than its total cost using the MST-write policy.

Node	Number of Reads	Number of Writes	Storage Cost
A	1000	6	1
B	12	1050	5
C	2	1	10400
D	3	68	1
E	1	6	5
F	1	100	4000
G	1190	0	1
H	1	5	5000
I	1222	1	100
J	1100	1	1
K	1100	1	0
L	1	1	1

Table 5: Number of read and write requests, and storage costs used in Example 3.

	MST-Write Policy	SMT-Write Policy
S_{MST}	32966 (1.000)	32966 (1.000)
S_{SMT}	34763 (1.055)	23494 (0.712)

Table 6: Total costs of the S_{MST} and S_{SMT} replica sets using the MST-write and the SMT-write policies.[4]

5.4 Example 4.

In this example, we consider the star network of Figure 8 with read, write requests and storage costs given in Table 7. In this case, $S_{MST} = \{H\}$ and $S_{SMT} = \{B, C, D, E, F, H, I\}$. The total costs of the two replica sets for the MST-write and SMT-write policies are given in Table 8. Note that the total cost of the replica set S_{MST} remains the same when either one of the two write policies is used. However, the total cost of the replica set S_{SMT} is about 29% lower when using the SMT-write policy as opposed to the MST-write policy. Further, if the SMT-write policy is used, S_{SMT} incurs a total cost that is about 32% lower than that

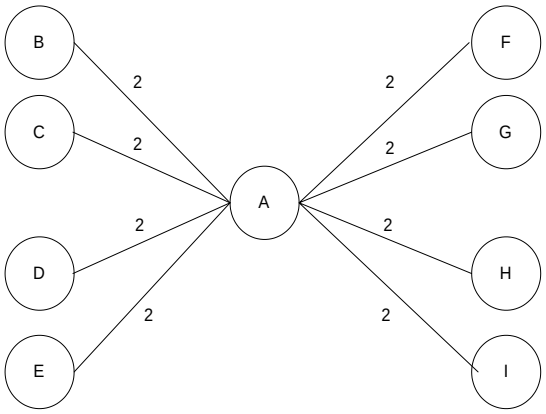

Figure 8: Star network used in Example 4. The edge costs are shown next to each edge.

Node	Number of Reads	Number of Writes	Storage Cost
A	1	1567	21222
B	1350	0	111
C	1640	43	21
D	1740	0	111
E	1510	34	21
F	1528	23	2
G	1803	3	8000
H	1991	0	100
I	1039	43	20

Table 7: Number of read and write requests, and storage costs used in Example 4.

	MST-Write Policy	Steiner-Write Policy
S_{MST}	46260 (1.000)	46260 (1.000)
S_{SMT}	51858 (1.121)	31588 (0.682)

Table 8: Total costs of the S_{MST} and S_{SMT} replica sets using the MST-write and the SMT-write policies.[4]

of S_{MST}. However, if the MST-write policy is used, S_{MST} has a total cost that is about 12% lower than the total cost of S_{SMT}.

6 Conclusion

We described a $O(n^6p^2)$–time algorithm for finding a Steiner-optimal replica set of size p for an object on a tree with n nodes, taking into consideration not only the read and write costs, but also the associated storage costs. Furthermore, our algorithm can find a Steiner-optimal replica set for a tree with n nodes in $O(n^8)$–time. The algorithm, though static in nature, provides the basis for comparison of different (dynamic) heuristics for data replication on tree networks. We also demonstrated the relationship of a write policy to replicated data placement. The sample results showed that a replica set that is optimal for the MST-write policy is not always optimal for the SMT-write policy, and vice versa. Moreover, the examples demonstrated that the use of the SMT-write policy can decrease the cost of replication by as much as 33%. From the perspective of database engineering, this provides critical insights on the design considerations of replicated data systems.

Note that a related replication problem is that of finding Steiner-optimal replica sets for a tree network with read-write-storage costs, which additionally take into account the loads imposed by the nodes in the tree on the replica nodes and the capacity constraints of the replica nodes. Taking into account capacity constraints is an important consideration. The average number of requests serviced by each node u of a replica set directly affects the average response that the nodes covered by u observe (due to queuing delays and network congestion). Further, in certain types of applications, such as image, video, and map servers, it is necessary to place with each copy of an object a copy of an appropriate software system, for example a DBMS or a GIS system, for servicing read and write requests. In many cases however, such systems impose restrictions on the number of concurrent users. This kind of situation can be easily modeled with the use of imposed loads and capacity constraints. Kalpakis et al [10] describe a $O(n^3p^2\Lambda_{\max}^2)$–time algorithm for finding minimum total read–write–storage cost (MST-optimal) replica sets for trees, taking into account the capacity constraints of the nodes in a replica set, where $\Lambda_{\max}$ is an upper bound on the capacity of each tree node. We can use a similar algorithm for finding Steiner-optimal replica sets that take into account read-write-storage costs and capacity constraints of the nodes.

There are a number of additional important issues to be resolved regarding the problem of finding optimal replica sets. Some of those issues are: (a) designing faster algorithms for computing Steiner-optimal replica sets in tree networks, (b) finding optimal or near–optimal replica sets for multiple objects, (c) finding optimal replica sets that have additional properties, such as high availability of the replicated object, (d) finding optimal or near–optimal replica sets, that consider read,write, and storage costs as well as capacity constraints, in a distributed manner, (e) finding optimal or near–optimal replica sets for other network topologies.

References

[1] S. Arora, P. Raghavan, and S. Rao. Approximation schemes for euclidean k–medians and related problems. In *Proceedings of the 30th ACM STOC*, pages 106–113, 1998.

[2] B. Awerbuch and Y. Azar. Blind bidding and competitive online pricing. In *unpublished manuscript*, 1996.

[3] B. Awerbuch, Y. Azar, and S. Plotkin. Online admission control for virtual circuits. In *In IEEE FOCS Proceedings*, 1993.

[4] B. Awerbuch, K. Kalpakis, and Y. Yesha. Towards free information markets. In *In 11th International Conference on Mathematical and Computer Modelling and Scientific Computing*, March, 1997.

[5] L. W. Dowdy and D. V. Foster. Comparative models of the file assignment problem. *ACM Computing Surveys*, 14(2):287–313, 1982.

[6] D. Ferguson, Y. Yemini, and C. Nikolaou. Microeconomic algorithms for load balancing in distributed computer systems. In *In Proceedings of the 8th International Conference on Distributed Computing Systems*, 1988.

[7] M. L. Fisher and D. S. Hochbaum. Database location in computer networks. *Journal of the ACM*, 27(4):718–735, 1982.

[8] S. Guha and S. Khuller. Greedy strikes back: Improved facility location algorithms. In *Proceedings of the 9th ACM–SIAM Symposium on Discrete Algorithms*, 1998.

[9] D. S. Hochbaum. Heuristics for the fixed cost median problem. *Mathematical Programming*, 22:148–162, 1982.

[10] K. Kalpakis, K. Dasgupta, and O. Wolfson. Optimal placement of replicas in trees with read-write-storage costs. *To appear in IEEE Transactions on Parallel and Distributed Systems*, 2001.

[11] O. Kariv and S. L. Hakimi. An algorithmic approach to location problems. ii: The p–medians. *SIAM Journal of Applied Mathematics*, 37(3):539–560, 1979.

[12] J. Kurose and R. Simha. A microeconomic approach to optimal resource allocation in distributed computer systems. *IEEE Transactions on Comuters*, 38(5):705–717, 1989.

[13] D. B. Shmoys, E. Tardos, and K. Aardal. Approximation algorithms for facility location problems (extended abstract). In *Proceedings of the 29th ACM STOC*, pages 265–274, 1997.

[14] M. Stonebraker, P. Aoki, R. Devine, M. Litwin, and W. Olson. Mariposa: A new architecture for distributed data. In *In Proceedings of IEEE 10th International Conference on Data Engineering*, February, 1994.

[15] O. Wolfson and A. Milo. The multicast policy and its relationship to replicated data placement. *ACM Transactions on Database Systems*, 16(1):181–205, 1991.