Minimizing the Continuous Diameter when Augmenting a Geometric Tree with a Shortcut[☆,☆☆]

Jean-Lou De Carufel[a], Carsten Grimm[b,c,*], Anil Maheshwari[c], Stefan Schirra[b], Michiel Smid[c]

[a] *School of Electrical Engineering and Computer Science, University of Ottawa*
800 King Edward Avenue, Ottawa, Ontario, K1N 6N5, Canada
[b] *Institut für Simulation und Graphik, Otto-von-Guericke-Universität Magdeburg*
Universitätsplatz 2, D-39106 Magdeburg, Germany
[c] *School of Computer Science, Carleton University*
1125 Colonel By Drive, Ottawa, Ontario, K1S 5B6, Canada

Abstract

We augment a tree T with a shortcut pq to minimize the largest distance between any two points along the resulting augmented tree $T + pq$. We study this problem in a continuous and geometric setting where T is a geometric tree in the Euclidean plane, a shortcut is a line segment connecting any two points along the edges of T, and we consider all points on $T + pq$ (i.e., vertices and points along edges) when determining the largest distance along $T + pq$. The *continuous diameter* is the largest distance between any two points along edges. We establish that a single shortcut is sufficient to reduce the continuous diameter of a geometric tree T if and only if the intersection of all diametral paths of T is neither a line segment nor a point. We determine an optimal shortcut for a geometric tree with n straight-line edges in $O(n \log n)$ time.

Keywords: Geometric Network, Augmentation, Continuous Diameter, Shortcut, Tree
2010 MSC: 68U05, 05C85

1. Introduction

A *network* is a connected, undirected graph with positive edge weights. A curve is rectifiable if it has a finite length. A *geometric network* is a network that is embedded in the Euclidean plane whose edges are rectifiable curves weighted with their length. We describe our algorithmic results for straight-line edges, even though they extend to more general edges. We define points along the edges of a network as follows. Let $uv \in E$ be an edge in G with weight w_{uv} that connects the vertices $u, v \in V$. For every value $\lambda \in [0, 1]$, we define a point p on edge uv that subdivides uv into two sub-edges up and

☆This work was partially supported by NSERC and FQRNT.
☆☆A preliminary version of this work was presented at the 15th International Symposium on Algorithms and Data Structures (WADS 2017), July 31 to August 2, 2017, St. John's, NL, Canada.
*Corresponding author

pv of weights $w_{up} = \lambda w_{uv}$ and $w_{pv} = (1 - \lambda)w_{uv}$, respectively. We write $p \in uv$ to indicate that p is a point along the edge uv, for some $\lambda \in [0, 1]$, and we write $p \in G$ to denote that p is a point along some edge of the network G. There is no ambiguity if two edges cross: there are two points along the network that correspond to the crossing in the plane, since points along edges are specified by their relative position to the endpoints of their containing edge, expressed by λ, and not by coordinates in the plane.

The *network distance* between any two points p and q on a geometric network G is the length of a shortest weighted path from p to q in G and it is denoted by $d_G(p, q)$. The *continuous diameter* of G is the largest network distance between any two points on G, and it is denoted by $\mathrm{diam}(G)$, i.e., $\mathrm{diam}(G) = \max_{p,q \in G} d_G(p, q)$. In contrast, for a network with vertex set V, the *discrete diameter* is the largest distance between any two vertices, i.e., $\max_{u,v \in V} d_G(u, v)$. A pair $p, q \in G$ is *diametral* when their distance is the continuous diameter, i.e., $\mathrm{diam}(G) = d_G(p, q)$. A point $p \in G$ is a *diametral partner* in G if there exists some point $q \in G$ such that p, q is a diametral pair of G. A *diametral path* in G is a shortest weighted path in G that connects a diametral pair of G.

We denote the Euclidean distance between two points p and q by $|pq|$. A line segment pq with endpoints $p, q \in G$ is a *shortcut* for G. We *augment* a geometric network G with a shortcut pq: If they do not exist already, we introduce vertices at p and at q, thereby subdividing the edges containing p and q. We add the line segment pq as an edge of length $|pq|$ to G without introducing vertices at crossings between pq and other edges. We denote the resulting network by $G + pq$. In this work, we move a shortcut along a network and some of the intermediate shortcuts may coincide with edges or parts of edges. To simplify this discussion, we allow shortcuts pq for a network G with $d_G(p, q) = |pq|$. Instead a local constraint on the definition of a shortcut, we consider a shortcut to be *useful* for G when its addition to G reduces the continuous diameter, i.e., $\mathrm{diam}(G + pq) < \mathrm{diam}(G)$.

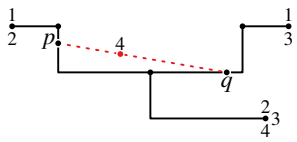


Figure 1: An optimal shortcut pq for a geometric tree T with the diametral pairs of $T + pq$ marked by matching numbers.

Our goal is to locate a shortcut pq for a geometric tree T that minimizes the continuous diameter of the augmented tree $T + pq$, as illustrated in Figure 1. This means we seek two points $p, q \in T$ with $\mathrm{diam}(T + pq) = \min_{r,s \in T} \mathrm{diam}(T + rs)$. We call a shortcut that minimizes the continuous diameter an *optimal shortcut*.

Let $C(p, q)$ be the simple cycle in $T + pq$. The *backbone* of T, denoted by $\mathcal{B}$, is the intersection of all diametral paths of T. The *absolute center* of T is the unique point $c \in T$ that minimizes the largest network distance from c, i.e., $\max_{q \in T} d_T(c, q) = \min_{p \in T} \max_{q \in T} d_T(p, q)$. Note that $c \in \mathcal{B}$. Determining the absolute center—and, thus, the backbone—of a geometric tree with n vertices takes $O(n)$ time [1, 2].

Our Contributions. We obtain the following structural and algorithmic results:

1. A geometric tree T admits a useful shortcut if and only if its backbone $\mathcal{B}$ is neither a straight-line segment nor a point (i.e., a degenerate line segment).
2. Every geometric tree T has an optimal shortcut pq with both endpoints on the backbone, i.e., $p, q \in \mathcal{B}$, and the absolute center c on the path from p to q in T.
3. We develop an algorithm that produces an optimal shortcut for a geometric tree T with n vertices whose edges are straight-line segments in $O(n \log n)$ time.

1.1. Related Work

We summarize related work on minimum-diameter network augmentation.

In the *abstract and discrete setting*, the goal is to minimize the discrete diameter of an abstract graph $G = (V, E)$ with positive weights for the edges of G and the edges of its complement graph $\bar{G} = (V, \binom{V}{2} - E)$ by inserting edges of $\bar{G}$ as shortcuts to G. If the edges of G and $\bar{G}$ have unit weight, then it is NP-hard to decide whether the diameter can be reduced below some target value $D \geq 2$ by adding at most k shortcuts [3–5]. This problem remains NP-hard when the number of shortcuts is variable, even for trees [3]. Its weighted version falls into the parameterized complexity class W[2]-hard [6, 7]. On the other hand, Oh and Ahn [8] determine an optimal vertex-to-vertex shortcut that minimizes the discrete diameter of an abstract n-vertex tree with positive edge weights in $O(n^2 \log^3 n)$ time. Minimum-diameter augmentation has also been studied as a bicriteria optimization in which both the diameter and the number (or cost) of the additional edges are minimized. For instance, Frati et al. [6] summarize the literature regarding the research on bicriteria optimization in minimum-diameter network augmentation.

Große et al. [9] introduce the *geometric and discrete setting* in which the problem is to minimize the discrete diameter of a geometric network by connecting vertices with line segments. Große et al. [9] determine an optimal shortcut for a polygonal path with n vertices in $O(n \log^3 n)$ time using a parametric search technique. Recently, Wang [10] improved this algorithm to $O(n \log n)$ time. Apart from the discrete diameter, the stretch factor, i.e., the largest ratio of the network distance between any two vertices and their Euclidean distance, has also been considered as a target function [11, 12].

In the *geometric and continuous setting* [13], the task is to minimize the continuous diameter of a geometric network by inserting line segments that may connect any two points along the edges. For a polygonal path of length n, one can determine an optimal shortcut in $O(n)$ time. For a cycle, one shortcut can never decrease the continuous diameter while two always suffice. For convex cycles with n vertices, one can determine an optimal pair of shortcuts in $O(n)$ time. In the model studied in this work, a crossing of a shortcut with an edge or another shortcut is not a vertex: a path may only enter edges at their endpoints. In the *planar model* [14, 15], every crossing is a vertex of the resulting network, which leads to a different graph structure and, thus, continuous diameter. In the planar model, Yang [15] characterizes optimal shortcuts for a polygonal path. Cáceres et al. [14] determine in polynomial time whether the continuous diameter of a plane geometric network can be reduced with a single shortcut.

1.2. Structure and Results

This work is structured as follows. In Section 2, we establish our structural results: We observe that—unlike in the discrete version of this problem—adding a shortcut to a tree might increase the continuous diameter. We characterize the trees that have a useful shortcut, i.e., a shortcut that reduces the continuous diameter, as precisely those trees where the intersection of all diametral paths is neither a straight-line segment nor a point (i.e., a degenerate line segment). The intersection of all diametral paths, called the backbone $\mathcal{B}$ of T plays a key role when locating an optimal shortcut. In the discrete setting, Große et al. [16] show that there exists an optimal shortcut for a tree with both endpoints along the backbone. We prove that this result carries over to the continuous

setting and strengthen it: we show that every geometric tree has an optimal shortcut pq with both endpoints along the backbone such that the absolute center c of T lies on the path from p to q in T. This yields a restriction of the search space that allows us to find an optimal shortcut for a geometric tree with n straight-line edges in $O(n \log n)$ time. We develop this algorithm in three steps: In Section 4, we examine how the diametral paths in the augmented tree rule out certain directions for the search. In Section 5, we develop a set of rules that inform us how to continuously slide a shortcut along the backbone until we eventually reach an optimal shortcut. In Section 6, we simulate this conceptual continuous algorithm with a discretization that achieves the desired running time.

The structural results hold for a geometric tree whose edges are rectifiable curves, i.e., curves that have a well defined length. We describe the algorithmic result for trees with straight-line edges; the techniques carry over to more general types of edges, e.g., algebraic curves of constant degree. Our model of computation is the real RAM.

2. Usefulness

We say a shortcut pq is *useful* for T when $\text{diam}(T + pq) < \text{diam}(T)$, we say pq is *indifferent* for T when $\text{diam}(T + pq) = \text{diam}(T)$, and we say pq is *useless* for T when $\text{diam}(T + pq) > \text{diam}(T)$. In the discrete setting, every shortcut is useful or indifferent, as the discrete diameter only considers vertices of T. In the continuous setting, a shortcut may be useless for T, since the points on the shortcut pq matter as well, as in Figure 2.

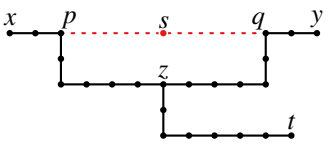


Figure 2: A geometric tree T with a shortcut pq that is useless for T. Each edge has a unit weight and $|pq| = 8$. The continuous diameter increases when we augment T with pq, since $\text{diam}(T) = d_T(x, y) = 16$ and $\text{diam}(T + pq) = d_{T+pq}(s, t) = 17$.

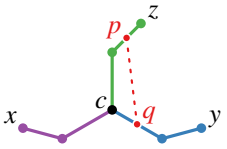


Figure 3: A geometric tree T whose continuous diameter cannot be reduced with a single shortcut. Each pair of the three leaves x, y, and z is diametral, as $d_T(x, c) = d_T(y, c) = d_T(z, c)$. The shortcut pq is not useful for T, as it is indifferent for $\{x, y\}$.

For two points $u, v \in T$, a shortcut pq is *useful* for the unordered pair $\{u, v\}$ when $d_{T+pq}(u, v) < d_T(u, v)$, and pq is *indifferent* for $\{u, v\}$ when $d_{T+pq}(u, v) = d_T(u, v)$.

We say a shortcut pq is *useful* for the ordered pair (u, v) when $d_T(u, p) + |pq| + d_T(q, v) < d_T(u, v)$, we say pq is *indifferent* for (u, v) when $d_T(u, p) + |pq| + d_T(q, v) = d_T(u, v)$, and we say that pq is *useless* for (u, v) when $d_T(u, p) + |pq| + d_T(q, v) > d_T(u, v)$. If pq is useful for (u, v) then the shortest path from u to v in $T + pq$ travels from u to p in T, then along the shortcut to q, and then from q to v in T. The order matters for ordered pairs: if pq is useful for the ordered pair (u, v), then pq is useless for the ordered pair (v, u), and qp is useless for the ordered pair (u, v). We only distinguish pq and qp for ordered pairs; there is no difference between pq and qp and, thus, $T + pq = T + qp$.

For some trees, we cannot reduce the continuous diameter with a single shortcut. For instance, every shortcut pq for the tree T in Figure 3 is either useless or indifferent for T, since pq would be indifferent for at least one of the diametral pairs $\{x, y\}$, $\{x, z\}$,

or $\{y, z\}$ of T. In Figure 3, the intersection of the diametral paths of T consists of a single vertex. We argue that a tree with this property cannot have a useful shortcut.

Lemma 1 *If the backbone of a geometric tree T consists only of the absolute center of T, then there does not exists a useful shortcut for T.*

Proof. Suppose T is a geometric tree whose backbone $\mathcal{B}$ consists only of the absolute center c of T. Then, c must be a vertex of T, since $\mathcal{B}$ would contain the entire edge containing c, otherwise. Let x, y be a diametral pair of T. Then we have $\frac{1}{2}\operatorname{diam}(T) = d_T(x, c) = d_T(y, c)$, since c is the midpoint of the path from x to y in T. Since the backbone only consists of c, there must be at least one other leaf z with $\frac{1}{2}\operatorname{diam}(T) = d_T(z, c)$ such that the paths from z to x and from z to y pass through c. This means that x, y, and z lie in three different sub-trees T_X, T_Y, and T_Z attached to c.

Assume, for the sake of a contradiction, that there exists a shortcut pq that is useful for T. Then $p \neq q$, since pq would be indifferent for T, otherwise. Hence, we have $p \neq c$ or $q \neq c$, and one of the sub-trees T_X, T_Y, or T_Z contains neither p nor q. Without loss of generality, suppose $p, q \notin T_X$. By our assumption, pq must be useful for every diametral pair of T including $\{x, y\}$. Without loss of generality, let pq be useful for (x, y), i.e., $d_{T+pq}(x, y) = d_T(x, p) + |pq| + d_T(q, y) < d_T(x, y)$. Otherwise, we swap p and q. This implies that q lies in T_Y, since otherwise the shortest path in $T + pq$ from x to y would contain c twice: once along the sub-path from x to p and once along the sub-path from q to y. Since pq is useful for T by assumption, pq must be useful for the unordered pair $\{x, z\}$ and, thus, pq must be useful for either (x, z) or (z, x). The shortcut pq cannot be useful for the ordered pair (x, z), since the path from q to z contains c. Therefore, pq is useful for (z, x) and, thus, p lies in T_Z, since the path from x to q in $T + pq$ contains c.

In summary, the shortcut pq is useful for (x, y) and for (z, x) with $p \in T_Z$ and $q \in T_Y$. This is impossible: Along the shortest path from x to y in $T + pq$ the sub-path from c to p *does not* contain pq, i.e., $d_{T+pq}(c, p) = d_T(c, p)$. On the other hand, along the shortest path from x to z in $T + pq$, the sub-path from c to p *does* contain pq, i.e., $d_{T+pq}(c, p) < d_T(c, p)$. Therefore, there does not exist a single useful shortcut for T. □

Let T be a geometric tree whose backbone $\mathcal{B}$ does not consist of a single vertex, as illustrated in Figure 4. In this case, the backbone $\mathcal{B}$ is a path with endpoints a and b such that $a \neq b$. We call the sub-trees that are attached to $\mathcal{B}$ the *$\mathcal{B}$-sub-trees* of T. The *root* of a $\mathcal{B}$-sub-tree S is the vertex r connecting S and $\mathcal{B}$. Let X be the $\mathcal{B}$-sub-tree with root a, and let Y be the $\mathcal{B}$-sub-tree with root b. We refer to X and Y as the *primary $\mathcal{B}$-sub-trees*, because every diametral pair of T consists of a leaf x in X and of a leaf y in Y. The other $\mathcal{B}$-sub-trees $S_1, S_2, \ldots, S_k$ with roots r_1, $r_2, \ldots, r_k$, respectively, are the *secondary $\mathcal{B}$-sub-trees* of T. Each primary $\mathcal{B}$-sub-tree may consist only of an endpoint of the backbone, as in Figure 4.

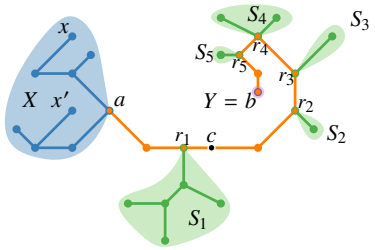


Figure 4: A geometric tree T with its backbone $\mathcal{B}$ (orange), primary $\mathcal{B}$-sub-trees, X (blue) and Y (purple), and secondary $\mathcal{B}$-sub-trees (green). The primary $\mathcal{B}$-sub-tree Y coincides with the endpoint b of $\mathcal{B}$.

Theorem 2 *For geometric tree T with backbone $\mathcal{B}$, there exist two points $p, q \in T$ such that pq is a useful shortcut for T if and only if $\mathcal{B}$ is not a line segment (or a point).*

Proof. Let T be a geometric tree and let pq be a useful shortcut for T. We show that the backbone $\mathcal{B}$ of T is neither a line segment nor a point (i.e., a degenerate line segment). Since T possesses a useful shortcut, the backbone $\mathcal{B}$ of T cannot be a point, due to Lemma 1. Let a and b be the endpoints of the path $\mathcal{B}$. Let X and Y be the primary $\mathcal{B}$-sub-trees attached to a and b, respectively. Let x, y be a diametral pair of T with $x \in X$ and $y \in Y$. Since pq is useful for T, the shortcut pq must be useful for $\{x, y\}$. Without loss of generality, pq is useful for (x, y). Otherwise, we swap p and q.

We argue that the backbone $\mathcal{B}$ differs from the line segment ab that connects its endpoints by showing that pq is useful for (a, b), since this implies $|ab| < d_T(a, b)$. We locate a diametral pair $\hat{x}, \hat{y}$ of T with $\hat{x} \in X$ and $\hat{y} \in Y$ such that a lies on the path from $\hat{x}$ to p, and b lies on the path from q to $\hat{y}$, and pq is useful for $(\hat{x}, \hat{y})$. The existence of a diametral pair $\hat{x}, \hat{y}$ of T with these properties implies that pq is useful for (a, b), i.e., $d_T(a, p) + |pq| + d_T(q, b) < d_T(a, b)$, because then

$$\begin{aligned} &d_T(\hat{x}, a) + d_T(a, p) + |pq| + d_T(q, b) + d_T(b, \hat{y}) \\ &= d_T(\hat{x}, p) + |pq| + d_T(q, \hat{y}) < d_T(\hat{x}, \hat{y}) = d_T(\hat{x}, a) + d_T(a, b) + d_T(b, \hat{y}) \ . \end{aligned}$$

We start with x, y as a candidate for $\hat{x}, \hat{y}$. If a lies on the path from x to p, then we let $\hat{x} = x$. If a does not lie on the path from x to p, then $p \in X$ with $p \neq a$. Since a is an endpoint of the intersection of the diametral paths in T, there is some leaf x' of X such that x', y is diametral in T and a lies on the path from x' to p. Let $\hat{x} = x'$. We argue that pq is useful for (x', y). Since pq is useful for T, it must be useful for $\{x', y\}$. The shortcut pq is useful for (x, y) and, therefore, useful for (p, y). Thus, the shortest path connecting p and y in $T + pq$ contains q. If pq was useful for (y, x'), then the shortest path connecting x' and y in $T + pq$ would travel from x' to q, then via the shortcut qp to p, and then backwards via q to y. This is impossible and pq must be useful for (x', y). Likewise, we find $\hat{y}$ where pq is useful for $(\hat{x}, \hat{y})$ and b lies on the path from q to $\hat{y}$.

Therefore, pq is useful for (a, b). By the triangle inequality, $|ab| \leq d_T(a, p) + |pq| + d_T(q, b)$ and, thus, $|ab| < d_T(a, b)$. The backbone is strictly longer than the line segment connecting its endpoints and, thus, the backbone cannot be a line segment itself.

Conversely, suppose T is a geometric tree whose backbone $\mathcal{B}$ is neither a line segment nor a single point, as in Figure 5. Thus, the backbone $\mathcal{B}$ of T is a path with endpoints a and b such that $|ab| < d_T(a, b)$. Let X and Y be the primary $\mathcal{B}$-sub-trees of T attached to a and to b, respectively, and let $S_1, S_2, \ldots, S_k$ be the secondary $\mathcal{B}$-sub-trees of T.

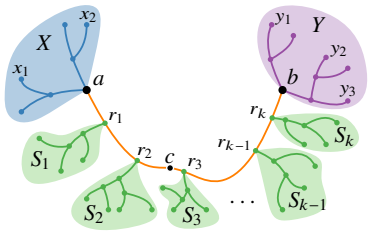


Figure 5: A geometric tree whose backbone is a path from a to b that is not a line segment.

Let s, t be a diametral pair of $T + ab$. We distinguish three cases: we have $s, t \in T$ and s, t is a diametral in T, or we have $s, t \in T$ and s, t is not diametral in T, or we have $s \notin T$ or $t \notin T$. In the first two cases, we show that ab is useful for T, and in the third case we construct a useful shortcut pq for T with $p, q \in \mathcal{B}$ if ab is not useful for T.

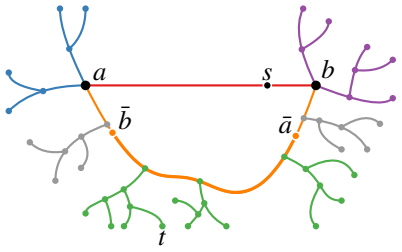


Figure 6: A sketch of a geometric tree T whose backbone is not a line segment connecting its endpoints a and b. A diametral pair s, t of $T + ab$ with $s \notin T$. The diametral partner t lies in a $\mathcal{B}$-sub-tree attached to the path from $\bar{a}$ to $\bar{b}$ in T, where $\bar{a}$ and $\bar{b}$ are the respective farthest points from a and from b along the simple cycle $C(a, b)$ in $T + ab$.

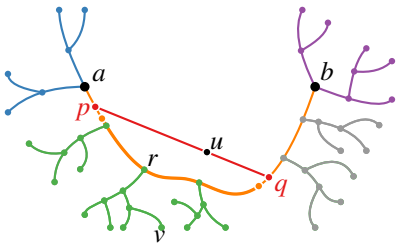


Figure 7: A sketch of a geometric tree whose backbone is not a line segment connecting its endpoints a and b. We moved the shortcut to a position pq, since the shortcut ab increased the diameter. The points p and q are chosen such that the simple cycle $C(p, q)$ in $T + pq$ is too small to contain the depicted diametral pair u, v with $u \notin T$.

1. Suppose $s, t \in T$ such that s, t is diametral in T, i.e., $d_T(s, t) = \operatorname{diam}(T)$.
   Then s and t lie in different primary $\mathcal{B}$-sub-trees of T. Without loss of generality, we assume that $s \in X$ and $t \in Y$. Otherwise, we swap s and t.
   The shortcut ab is useful for (s, t), i.e., $d_T(s, a) + |ab| + d_T(b, y) < d_T(s, t)$, since
   $$d_T(s, a) + |ab| + d_T(b, y) < d_T(s, a) + d_T(a, b) + d_T(b, y) = d_T(s, t) \ .$$
   Therefore, $d_{T+ab}(s, t) < d_T(s, t)$ and, thus, the shortcut ab is useful for T, since
   $$\operatorname{diam}(T + ab) = d_{T+ab}(s, t) < d_T(s, t) = \operatorname{diam}(T) \ .$$
2. Suppose $s, t \in T$ such that s, t is not diametral in T, i.e., $d_T(s, t) < \operatorname{diam}(T)$.
   Then the shortcut ab is useful for T, since
   $$\operatorname{diam}(T + ab) = d_{T+ab}(s, t) \leq d_T(s, t) < \operatorname{diam}(T) \ .$$
3. Suppose $s \notin T$ or $t \notin T$. Without loss of generality, let $s \notin T$.
   Then $s \in ab$ with $a \neq s \neq b$. Let $C(a, b)$ be the simple cycle in $T + ab$. We distinguish two cases depending on whether t lies on $C(a, b)$ or not.
   (a) Suppose $t \in C(a, b)$. Then we have $d_{T+ab}(s, t) < d_T(a, b)$, because
   $$d_{T+ab}(s, t) \leq \frac{|ab| + d_T(a, b)}{2} < \frac{d_T(a, b) + d_T(a, b)}{2} = d_T(a, b) \ .$$
   Thus, ab is useful for T, as $\operatorname{diam}(T+ab) = d_{T+ab}(s, t) < d_T(a, b) \leq \operatorname{diam}(T)$.
   (b) Suppose $t \notin C(a, b)$, as illustrated in Figure 6.
   We argue that t is a leaf of a secondary $\mathcal{B}$-sub-tree. Let $\bar{a}$ be the farthest point on $C(a, b)$ from a with respect to $T + ab$. The point $\bar{a}$ lies in T, because
   $$|ab| = \frac{|ab| + |ab|}{2} < \frac{|ab| + d_T(a, b)}{2} = d_{T+ab}(a, \bar{a}) \ .$$
   Therefore, t cannot lie in X, because $\bar{a}$ is the farthest point on $C(a, b)$ from any point in X, and $s \neq \bar{a}$, since $s \notin T$ and $\bar{a} \in T$. Likewise, $t \notin Y$. In summary, s lies on the shortcut ab and t is a leaf of a secondary $\mathcal{B}$-sub-tree.

Let δ be the largest diameter of the secondary $\mathcal{B}$-sub-trees $S_1, \ldots, S_k$, i.e.,

$$\delta := \max_{i=1}^{k} \operatorname{diam}(S_i) \ ,$$

and let $\epsilon = \operatorname{diam}(T) - \delta$. We have $\epsilon > 0$, as none of the secondary $\mathcal{B}$-sub-trees contains diametral pairs of T, i.e., $\operatorname{diam}(S_i) < \operatorname{diam}(T)$ for all $i = 1, 2, \ldots, k$. Since $|ab| < d_T(a, b)$, there exist $p, q \in \mathcal{B}$ with $|pq| < d_T(p, q)$ and $|pq| + d_T(p, q) < 2\epsilon$. We argue that pq is useful for T. Let u, v be a diametral pair of $T + pq$ and let $C(p, q)$ be the simple cycle in $T + pq$. Analogously to diametral pairs of $T + ab$, we argue that pq is useful for T when $u, v \in T$ (Case 1 and 2) and when $u \in pq$ with $p \neq u \neq q$ and $v \in C(p, q)$ (Case 3a). It remains to show that pq is useful for T when $u \in pq$ with $p \neq u \neq q$ and when v is a leaf of a secondary $\mathcal{B}$-sub-tree that is attached to an interior vertex r of the path from p to q in T (Case 3b), as illustrated in Figure 7. By definition, $d_T(v, r) \leq \delta$. By choice of p and q,

$$d_{T+pq}(r, u) \leq \frac{|C(p, q)|}{2} = \frac{|pq| + d_T(p, q)}{2} < \frac{2 \cdot \epsilon}{2} = \epsilon \ .$$

Therefore, the shortcut pq is useful for T, since

$$\begin{aligned}
\operatorname{diam}(T + pq) &= d_{T+pq}(u, v) && (u, v \text{ is diametral in } T + pq) \\
&= d_T(v, r) + d_{T+pq}(r, u) \\
&&& (r \text{ is on any path from } v \text{ to } u \text{ in } T + pq) \\
&< d_T(v, r) + \epsilon && (d_{T+pq}(r, u) < \epsilon) \\
&\leq \delta + \epsilon && (d_T(v, r) \leq \delta) \\
&= \delta + \operatorname{diam}(T) - \delta && (\epsilon = \operatorname{diam}(T) - \delta) \\
&= \operatorname{diam}(T) \ .
\end{aligned}$$

Every diametral pair s, t of $T + ab$ falls into one of the above cases and, in each case, we either argued that ab is useful for T or found a useful shortcut for T when ab was not useful for T. Therefore, T possesses a useful shortcut when $\mathcal{B}$ is not a line segment. □

3. Optimal Shortcuts

Consider a geometric tree T whose backbone is a path from a to b. This path contains the absolute center c. We prove that there is an optimal shortcut pq for T such that p lies on the path from a to c and q lies on the path from c to b. This holds when $\mathcal{B}$ consists only of c, since then T has no useful shortcuts and the degenerate shortcut cc is optimal. For the other cases, we establish our claim by proving the following statements.

1. If pq is a useful shortcut for a geometric tree T, then every $\mathcal{B}$-sub-tree S of T contains at most one endpoint of pq, i.e., we have $p \notin S$ or $q \notin S$.
2. If an endpoint p of a useful, optimal shortcut pq for T lies in a $\mathcal{B}$-sub-tree S with root r, then the shortcut rq is also optimal for T—regardless of the position of q.
3. There exists an optimal shortcut pq for T with $p, q \in \mathcal{B}$.

4. If pq is an optimal shortcut for T with $p, q \in \mathcal{B}$ and the path from p to q along $\mathcal{B}$ does not contain c, then at least one of pc or cq is an optimal shortcut for T.

The last statement implies our claim, since c lies on the path from c to b and the path from c to a. We prove the four statements above in Lemmas 4 to 6 and Theorem 7.

The idea for restricting the search for an optimal shortcut along the backbone stems from Große et al. [16] who establish the following for the discrete setting [16, Lemma 6]. We generalize their result to the continuous setting and incorporate the absolute center.

Theorem 3 (Discrete Backbone Theorem by Große et al. [16]) *Let T be a geometric tree with vertex set V and backbone $\mathcal{B}$. There exists a pair of vertices $p, q \in \mathcal{B} \cap V$ such that pq minimizes the discrete diameter of the augmented tree $T + pq$ among all possible discrete shortcuts for T, i.e.,*

$$\max_{u,v \in V} d_{T+pq}(u, v) = \min_{r,s \in V} \max_{u,v \in V} d_{T+rs}(u, v) \ .$$

Lemma 4 *Let T be a geometric tree, let pq be a shortcut for T, and let S be a $\mathcal{B}$-sub-tree of T. If the shortcut pq is useful for T, then $p \notin S$ or $q \notin S$.*

Proof. Assume, for a contradiction, that there is a geometric tree T, with a $\mathcal{B}$-sub-tree S with root r, such that there is a shortcut pq for T with $p, q \in S$ that is useful for T.

Suppose the backbone $\mathcal{B}$ of T is a path that connects the vertices a and b. Let X and Y be the primary $\mathcal{B}$-sub-trees with roots a and b, respectively. Furthermore, let x, y be a diametral pair of T with $x \in X$ and $y \in Y$. Since pq is useful for T, we have $p \neq q$ and pq is useful for (x, y) or for (y, x). Without loss of generality, let pq be useful for (x, y). Otherwise, we swap x, y and X, Y. We distinguish whether the root r of S lies on the path from x to p in T or not and we derive a contradiction in both of these cases.

Suppose r *does not* lie on the path from x to p. Then x, p, and q lie in the same $\mathcal{B}$-sub-tree, i.e., $S = X$ and $r = a$, since $x \in X$, as in Figure 8. Since y lies in the other primary $\mathcal{B}$-sub-tree, we have $y \notin S$ and, thus, r lies on the path from q to y.

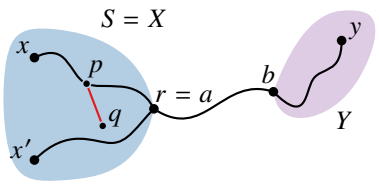


Figure 8: A shortcut pq for T that has both endpoints in the same $\mathcal{B}$-sub-tree S with root r and where r does not lie on the path from x to p, which implies $S = X$.

Since a is an endpoint of the backbone, i.e., the intersection of all diametral paths in T, there is a leaf x' of X such that x', y is diametral in T and r is on the path from x' to p. The shortcut pq cannot be useful for (x', y), since r lies on the paths from x' to p and from q to y. Thus, for pq to be useful for T, the shortcut pq must be useful for (y, x'). The shortest path from y to x' in $T + pq$ contains the path from r to p in T, i.e., $d_T(r, p) < d_T(r, q) + |qp|$. On the other hand, the shortest path from y to x in $T + pq$ travels from r via the shortcut to p and, thus, $d_T(r, q) + |qp| < d_T(r, p)$. This is impossible.

Suppose r lies on the path from x to p. Then r *does not* lie on the path from y to q and, thus, y, q, and p lie in the same $\mathcal{B}$-sub-tree, meaning $S = Y$ and $r = b$ and $x \notin S$. This situation is symmetric to the previous case and, therefore, impossible as well.

Therefore, if the shortcut pq is useful for T, then we have $p \notin S$ or $q \notin S$. □

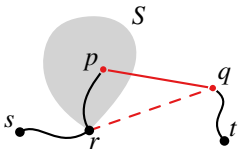
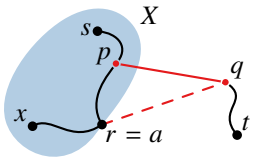
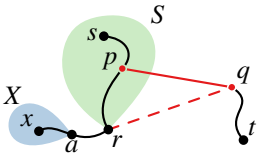


Figure 9: A sketch of the shortest path in $T + pq$ from s via pq to t, where p lies in some $\mathcal{B}$-sub-tree S whose root r lies on the path from s to p in T. We have $q \notin S$ by Lemma 4, and, thus, $t \notin S$.



(a) The case $S = X$.



(b) The case $S \neq X$.

Figure 10: A sketch of the shortest path in $T + pq$ connecting s and t via pq where p lies in some $\mathcal{B}$-sub-tree S and p lies on the path from s to the root r of S. There is a diametral partner x in T such that the path from x to p passes through r. This holds when S is (a) primary or (b) secondary.

Lemma 5 *Let T be a geometric tree and let S be a $\mathcal{B}$-sub-tree of T with root r. If pq is a useful shortcut for T with $p \in S$, then* $\operatorname{diam}(T + rq) \leq \operatorname{diam}(T + pq)$.

The proof of Lemma 5 follows the proof of the discrete backbone theorem [16, Lemma 6]. The first two cases are due to Große et al. [16]; they are provided for self-containment. We add the third case to generalize the result to the continuous setting.

Proof (Proof of Lemma 5). Let T be a geometric tree, let S be a $\mathcal{B}$-sub-tree of T with root r, let pq be a useful shortcut for T with $p \in S$, and let s, t be a diametral pair of $T + rq$. Either we have $s, t \in T$ and pq is indifferent for $\{s, t\}$, or we have $s, t \in T$ and pq is useful for $\{s, t\}$, or we have $s \notin T$ or $t \notin T$. In each case, we argue that $d_{T+rq}(s, t) \leq d_{T+pq}(s', t')$ for some $s', t' \in T + pq$ and, thus, $\operatorname{diam}(T + rq) \leq \operatorname{diam}(T + pq)$.

1. Suppose $s, t \in T$ and pq is indifferent for $\{s, t\}$, i.e., $d_T(s, t) = d_{T+pq}(s, t)$. Then we have $\operatorname{diam}(T + rq) \leq \operatorname{diam}(T + pq)$, because

$$\operatorname{diam}(T + rq) = d_{T+rq}(s, t) \leq d_T(s, t) = d_{T+pq}(s, t) \leq \operatorname{diam}(T + pq) \ .$$

2. Suppose $s, t \in T$ and pq is useful for $\{s, t\}$, i.e., $d_T(s, t) > d_{T+pq}(s, t)$. Then pq is useful for (s, t) or (t, s). Without loss of generality, pq is useful for (s, t), i.e., $d_T(s, p) + |pq| + d_T(q, t) = d_{T+pq}(s, t) < d_T(s, t)$. Otherwise, we swap s and t.
   We distinguish whether the path from s to p in T contains r or not.
   (a) Suppose the path from s to p in T contains r, as shown in Figure 9. Then we have $\operatorname{diam}(T + rq) \leq \operatorname{diam}(T + pq)$, because

$$\begin{aligned}
\operatorname{diam}(T + rq) &= d_{T+rq}(s, t) && (s, t \text{ is diametral in } T + rq) \\
&\leq d_T(s, r) + |rq| + d_T(q, t) && (\text{triangle inequality for } d_{T+rq}) \\
&\leq d_T(s, r) + d_T(r, p) + |pq| + d_T(q, t) && (\text{triangle inequality } |\cdot|) \\
&= d_T(s, p) + |pq| + d_T(q, t) && (r \text{ is on the } s\text{-}t\text{-path in } T) \\
&= d_{T+pq}(s, t) && (pq \text{ is useful for } (s, t)) \\
&\leq \operatorname{diam}(T + pq) \ .
\end{aligned}$$

(b) Suppose the path from s to p in T *does not* contain r, as shown in Figure 10. Let a and b be the endpoints of the backbone $\mathcal{B}$ of T, and let X and Y be the primary $\mathcal{B}$-sub-trees with roots a and b, respectively. We find a diametral pair x, y of T such that $x \in X$, $y \in Y$, and r lies on the path from x to p.
Every path in $T + pq$ that connects a diametral pair of T contains pq, since pq is a useful shortcut for T. If $S = X$, then there is a diametral pair x, y of T with $x \in X$ and $y \in Y$ such that the path from x to p contains $r = a$, as in Figure 10a. Otherwise, a would not be the endpoint of the backbone. If $S \neq X$, then r lies on the path from x to p for every $x \in X$, since $p \in S$, as in Figure 10b. The shortcut pq is useful for (x, y) and, thus, for (r, q), since r lies on the path from x to p. Therefore, no shortest path in $T + pq$ may contain the path connecting r and q. This includes the path from s to t. Hence, $t \notin S$ and $d_{T+rq}(s,t) = d_T(s,r) + d_{T+rq}(r,t)$. By Lemma 4, $p \in S$ implies $q \notin S$. Since the path from x to y in $T + pq$ cannot contain r two times, we have $y \notin S$ and the shortest path in $T + rq$ from x to t contains r, i.e., $d_{T+rq}(x,t) = d_T(x,r) + d_{T+rq}(r,t)$. We have $d_T(s,r) \leq d_T(x,r)$, as x, y is a diametral path of T and r lies on the path from x to y and on the path from s to y. This implies that x, t is a diametral pair of $T + rq$, since

$$\begin{aligned}
\operatorname{diam}(T + rq) &= d_{T+rq}(s,t) && (s,t \text{ is diametral in } T + rq) \\
&= d_T(s,r) + d_{T+rq}(r,t) && (s \in S \text{ and } t \notin S) \\
&\leq d_T(x,r) + d_{T+rq}(r,t) && (d_T(s,r) \leq d_T(x,r)) \\
&= d_{T+rq}(x,t) && (r \text{ is on any } x\text{-}t\text{-path in } T + rq) \\
&\leq \operatorname{diam}(T + rq) \ .
\end{aligned}$$

More precisely, x, t is a diametral pair of $T + rq$ such that the path from x to p contains r. With the argument from the previous case, we obtain

$$\operatorname{diam}(T + rq) = d_{T+rq}(s,t) = d_{T+rq}(x,t) \leq \operatorname{diam}(T + pq) \ .$$

3. Suppose $s \notin T$ or $t \notin T$. Without loss of generality, let $s \notin T$. Otherwise, we swap s and t. Then we have $s \in rq$ with $r \neq s \neq q$, as illustrated in Figure 11.
Let $C(r,q)$ be the simple cycle in $T + rq$. The path from s to t leaves $C(r,q)$ at the point $\bar{s} \in C(r,q)$ with $d_{T+rq}(s,\bar{s}) = \frac{1}{2}(d_T(r,q) + |rq|)$. Note that $r \neq \bar{s} \neq q$, since $r \neq s \neq q$ and $|rq| \leq d_T(r,q)$. By Lemma 4, $p \in S$ implies $q \notin S$.

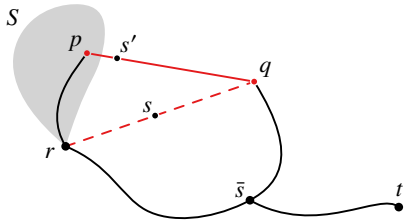


Figure 11: A diametral pair s, t of $T + rq$ with $s \in rq$. We move $r \in \mathcal{B}$ to a point p in a $\mathcal{B}$-sub-tree S such that pq is useful for T. Then $q, t \notin S$ and the diameter increases.

This means the path connecting r and q in T lies outside of S and, thus, $\bar{s} \notin S$ and $t \notin S$. The cycle $C(p,q)$ is formed by pq and the path from p to q in T. Since r lies on the path from p to q, we know that $\bar{s} \in C(p,q)$. Let s' be the farthest point from $\bar{s}$ on $C(p,q)$. Then we have $\operatorname{diam}(T + rq) \leq \operatorname{diam}(T + pq)$, because

$$\begin{aligned}
\operatorname{diam}(T + rq) &= d_{T+rq}(s,t) && (s,t \text{ is diametral in } T + rq)\\
&= d_{T+rq}(s,\bar{s}) + d_T(\bar{s},t) && (\bar{s} \text{ is on any } t\text{-}s\text{-path in } T + rq)\\
&= \frac{d_T(r,q) + |rq|}{2} + d_T(\bar{s},t) && (s \text{ and } \bar{s} \text{ are antipodals on } C(r,q))\\
&\leq \frac{d_T(r,q) + d_T(r,p) + |pq|}{2} + d_T(\bar{s},t) && (\text{triangle inequality})\\
&= \frac{d_T(p,q) + |pq|}{2} + d_T(\bar{s},t) && (r \text{ is on the } p\text{-}q\text{-path in } T)\\
&= d_{T+pq}(s',\bar{s}) + d_T(\bar{s},t) && (s \text{ and } \bar{s}' \text{ are antipodals on } C(p,q))\\
&= d_{T+pq}(s',t) && (\bar{s} \text{ is on any } t\text{-}s'\text{-path in } T + pq)\\
&\leq \operatorname{diam}(T + pq)\ .
\end{aligned}$$

Thus, if pq is a useful shortcut for T with $p \in S$, then $\operatorname{diam}(T + rq) \leq \operatorname{diam}(T + pq)$. □

Lemma 6 (Continuous Backbone Lemma) *Let T be a geometric tree T with backbone $\mathcal{B}$. There exist $p, q \in \mathcal{B}$ such that pq is an optimal shortcut for T.*

Proof. Let pq be an optimal shortcut for a geometric tree T. We assume that pq is useful for T, since otherwise the shortcut cc is indifferent for T with $c \in \mathcal{B}$.

If $p, q \in \mathcal{B}$, we are done. Suppose $p \notin \mathcal{B}$, i.e., p lies in some $\mathcal{B}$-sub-tree S with root $r \in \mathcal{B}$. According to Lemma 5 this implies $\operatorname{diam}(T + rq) \leq \operatorname{diam}(T + pq)$. This means that rq is also an optimal shortcut for T. If $q \in \mathcal{B}$, then the claim follows. Otherwise, $q \notin \mathcal{B}$, i.e., q lies in some $\mathcal{B}$-sub-tree S' with root $r' \in \mathcal{B}$. Since pq is useful for T and $p \in S$, Lemma 4 implies that $q \notin S$ and, thus, $S \neq S'$ and $r \neq r'$. By Lemma 5, this implies $\operatorname{diam}(T + rr') \leq \operatorname{diam}(T + rq) \leq \operatorname{diam}(T + pq)$, and, thus, rr' is an optimal shortcut for T with both endpoints along the backbone of T. □

Theorem 7 *Let T be a geometric tree with backbone $\mathcal{B}$ and absolute center c. There exist an optimal shortcut pq for T with $p, q \in \mathcal{B}$ and c on the path from p to q in T.*

Proof. Let T be a geometric tree with backbone $\mathcal{B}$ and absolute center c. By Lemma 6, there exists an optimal shortcut pq for T with $p, q \in \mathcal{B}$. If pq is indifferent for T, then the degenerate shortcut cc satisfies the claim. Thus, we assume that pq is useful for T.

Since there exists a useful shortcut for T, the backbone $\mathcal{B}$ of T is a path with endpoints a and b that contains c with $a \neq b$. Suppose c does not lie on the path from p to q along $\mathcal{B}$. Without loss of generality, we assume that p and q lie on the path from a to c with $p \neq c \neq q$. Otherwise, we swap a and b. Without loss of generality, the path from p to c along $\mathcal{B}$ contains q, i.e., $d_T(q,c) \leq d_T(p,c)$. Otherwise, we swap p and q.

We argue that pc is at least as good as pq, i.e., $\operatorname{diam}(T + pc) \leq \operatorname{diam}(T + pq)$. Let s, t be a diametral pair of $T + pc$. Either we have $s, t \in T$ and pq is indifferent for $\{s,t\}$, or we have $s, t \in T$ and pq is useful for $\{s,t\}$, or we have $s \notin T$ or $t \notin T$. We show $\operatorname{diam}(T + pc) \leq \operatorname{diam}(T + pq)$, for the first two cases and we rule out the third case.

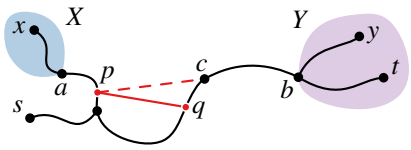


Figure 12: A shortcut pq for T with both endpoints on one side of the absolute center c with a diametral pair (s,t) of $T+pc$ for which pq is useful.

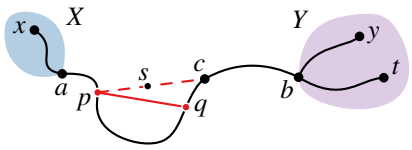


Figure 13: A shortcut pq for T with both endpoints on one side of the absolute center c with an impossible diametral pair s,t of $T+pc$ with $s \notin T$.

1. Suppose $s,t \in T$ and pq is indifferent for $\{s,t\}$, i.e., $d_T(s,t) = d_{T+pq}(s,t)$. Then we have $\operatorname{diam}(T+pc) \leq \operatorname{diam}(T+pq)$, because

$$\operatorname{diam}(T+pc) = d_{T+pc}(s,t) \leq d_T(s,t) = d_{T+pq}(s,t) \leq \operatorname{diam}(T+pq) \ .$$

2. Suppose $s,t \in T$ and pq is useful for $\{s,t\}$, i.e., $d_{T+pq}(s,t) < d_T(s,t)$. Then pq is useful for (s,t) or (t,s). Without loss of generality, let pq be useful for (s,t). Otherwise, we swap s and t. Figure 12 illustrates the following arguments. Let X and Y be the primary $\mathcal{B}$-sub-trees with roots a and b, respectively, and let x,y be a diametral pair of T with $x \in X$ and $y \in Y$. The shortcut pq is useful for T and, thus, pq is useful for $\{x,y\}$. Hence, pq is useful for (x,y), since p lies on the path from x to q in T. The path from s to p cannot contain q, since pq is useful for (s,t). By cutting the tree T at q, we obtain several sub-trees. Consider the sub-tree that contains x. Then s is contained in this sub-tree. Moreover, the path in T from s to c contains q. In the following, we argue $t \in Y$, which implies that c lies on the path from s to t in $T+pc$. As pq is useful for (s,t), the shortcut pq is useful for (s,q), i.e., $d_T(s,p) + |pq| < d_T(s,q)$, since

$$d_T(s,p) + |pq| + d_T(q,t) = d_{T+pq}(s,t) < d_T(s,t) \leq d_T(s,q) + d_T(q,t) \ .$$

This means pc is also useful for (s,c), i.e., $d_T(s,p) + |pc| < d_T(s,c)$, since

$$d_T(s,p) + |pc| \leq d_T(s,p) + |pq| + d_T(q,c) < d_T(s,q) + d_T(q,c) = d_T(s,c) \ .$$

The point t is a farthest leaf from c in T, i.e., $d_T(c,t) = d_T(c,y)$, because

$$\begin{aligned} d_T(s,p) + |pc| + d_T(c,t) &\leq d_T(s,p) + |pc| + d_T(c,y) \\ &= d_{T+pc}(s,c) + d_T(c,y) && (pc \text{ is useful for } (s,c)) \\ &= d_{T+pc}(s,y) && (c \text{ lies on the path from } s \text{ to } y) \\ &\leq d_{T+pc}(s,t) && (s,t \text{ is diametral in } T+pc) \\ &\leq d_T(s,p) + |pc| + d_T(c,t) \ . \end{aligned}$$

Since pq was useful for (s,t), the path from q to t cannot contain p. On the other hand, p lies on the backbone and blocks the path from q to any farthest leaf from c in X. Since t is a farthest leaf from c in T, this means that $t \in Y$ and, thus,

$$\operatorname{diam}(T+pc) = d_{T+pc}(s,t) \qquad (s,t \text{ diametral in } T+pc)$$

$$
\begin{aligned}
&\leq d_T(s,p) + |pc| + d_T(c,t) \\
&\leq d_T(s,p) + |pq| + d_T(q,c) + d_T(c,t) && \text{(triangle inequality)} \\
&= d_T(s,p) + |pq| + d_T(q,t) && (c \text{ is on the path from } q \text{ to } t) \\
&= d_{T+pq}(s,t) && (pq \text{ is useful for } (s,t)) \\
&\leq \operatorname{diam}(T+pq) \ .
\end{aligned}
$$

3. Suppose $s \notin T$ or $t \notin T$. We assume, without loss of generality, that $s \notin T$, i.e., $s \in pc$ with $p \neq s \neq c$. Otherwise, we swap s and t.
   Let X and Y be the primary $\mathcal{B}$-sub-trees with roots a and b, respectively. As illustrated in Figure 13, the point s lies on the simple cycle $C(p,c)$ in $T + pc$. The largest tree attached to $C(p,c)$ in $T + pc$ is the one containing Y, since c is the absolute center of T and both p and q lie on the path from a to c. Therefore, the point t lies in Y and s is the farthest point from t on $C(p,c)$. The path from t to s in $T + pc$ enters $C(p,c)$ at c. Therefore, s is the farthest point from c on $C(p,c)$. However, this implies that s lies on the path from p to c in T, since $|pc| \leq d_T(p,c)$ contradicting $s \notin T$. This means that this case is impossible.

Thus, if pq is an optimal shortcut for T such that p and q lie on the path from a to c in T with $d_T(q,c) \leq d_T(p,c)$ then pc is also an optimal shortcut for T. Therefore, there exist an optimal shortcut pq for T with $p, q \in \mathcal{B}$ and c on the path from p to q in T. □

4. Preparations for the Algorithm

Our search for an optimal shortcut pq for T proceeds as follows. Initially, we place the endpoints of the shortcut, p and q, at the absolute center c of T. Conceptually, we move p and q continuously along the backbone $\mathcal{B}$ balancing the diametral paths in $T + pq$. Throughout this continuous movement p remains along the path from a to c and q remains on the path from c to b, where a and b are the endpoints of $\mathcal{B}$. The diametral pairs in $T + pq$ guide our search: each diametral pair in $T + pq$ rules out some direction in which we could search for a better shortcut. We have found an optimal shortcut when p and q reach a position where the diametral pairs block all directions of movement, except perhaps going back the way we came. In order to implement this search, we discretize the continuous algorithm in Section 6. We proceed along the following steps.

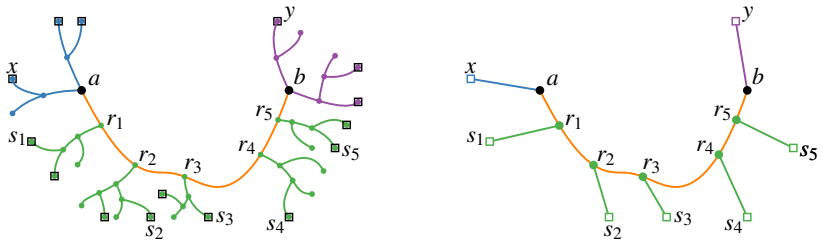
1. We simplify the geometric tree T by compressing the $\mathcal{B}$-sub-trees, thereby simplifying the discussion about diametral pairs and paths in $T + pq$.
2. We define algorithm states in terms of the diametral paths and diametral pairs that are present in the augmented tree, and we distinguish four types of movements for the shortcut, called *in-shift*, *out-shift*, *x-shift*, and *y-shift*.
3. We show that each type of diametral pair rules out a better shortcut in some direction, and that some combinations of pair types imply optimality.
4. We describe a continuous and conceptual movement of the shortcut that is guided by the set of types of diametral pairs of $T + pq$. We identify the invariants that are upheld by this movement and that guarantee that we find an optimal shortcut.

5. We specify the speeds at which the endpoints of the shortcut would move in the continuous algorithm. These speeds depend on the set of types of diametral paths in $T + pq$; the changes in this set constitute the events for the discretization.
6. We bound the number of events of the discrete algorithm by $O(n)$. This involves ruling out some transitions between the algorithm states as well as identifying situations where we can safely ignore events without compromising optimality.
7. Finally, we explain how we can process each of the $O(n)$ events in $O(\log n)$ amortized time and, thus, bound the running time of our algorithm by $O(n \log n)$.

In this section, we discuss Steps 1, 2, and 3, i.e., the preparations for the algorithm. In Section 5, we describe Step 4, i.e., the continuous algorithm and its correctness. In Section 6, we discuss Steps 5 through 7, i.e., the discretization of the algorithm.

4.1. Simplifying the Tree

Let T be a geometric tree whose backbone B consists of more than its absolute center c. Let a and b be the endpoints of $\mathcal{B}$, and let X and Y be the primary $\mathcal{B}$-sub-trees of T with roots a and b, respectively, and let $S_1, S_2, \ldots, S_k$ be the secondary $\mathcal{B}$-sub-trees of T that are attached to $\mathcal{B}$ at their roots $r_1, r_2, \ldots, r_k$, respectively. We assume, without loss of generality, that the roots $r_1, r_2, \ldots, r_k$ appear in this order from a to b along $\mathcal{B}$. Let x be a farthest leaf from a in X, let y be a farthest leaf from b in Y and, for every $i = 1, 2, \ldots, k$, let s_i be a farthest leaf from r_i in S_i, as illustrated in Figure 14a.



(a) A geometric tree T.

(b) The perspective from $\mathcal{B}$ on T.

Figure 14: A geometric tree a together with the perspective from its backbone b. We represent each $\mathcal{B}$-sub-tree S with an edge whose length is the length of a path from the root of S to a farthest leaf (squares) of S.

We simplify the discussion about diametral pairs in $T + pq$. First, there is no need to distinguish diametral pairs with partners in the same $\mathcal{B}$-sub-trees: for any $i, j = 1, 2, \ldots, k$ with $i \neq j$, the pair s_i, s_j is diametral in $T + pq$ if and only if every farthest leaf from r_i in S_i forms a diametral pair with every farthest leaf from r_j in S_j. Second, there is no need to consider diametral pairs with both endpoints in the same $\mathcal{B}$-sub-tree, as we argue in Lemma 8 below. Therefore, we simplify T by replacing each $\mathcal{B}$-sub-tree S_i with an edge from r_i to a vertex representing s_i of length $d_T(r_i, s_i)$. Likewise, we replace X and Y with edges of appropriate length, as illustrated in Figure 14. We refer to the resulting caterpillar network as the *perspective* from $\mathcal{B}$ on T.

Lemma 8 *Let T be a geometric tree with backbone $\mathcal{B}$, let $p, q \in \mathcal{B}$, and let S be a $\mathcal{B}$-sub-tree of T. If there exist $u, v \in S$ such that u, v is diametral in $T + pq$, then pq is an optimal shortcut for T.*

Proof. Let T be a geometric tree with backbone $\mathcal{B}$, let $p, q \in \mathcal{B}$, and let S be a $\mathcal{B}$-sub-tree of T. Suppose there exist $u, v \in S$ such that u, v is a diametral pair of $T + pq$.

Since S is a tree that is attached to the remainder of $T + pq$, we have $\operatorname{diam}(S) \leq \operatorname{diam}(T + pq)$. Every path from u to v via pq contains r twice, hence the shortest path from u to v in $T + pq$ remains in S. This implies $\operatorname{diam}(T + pq) = \operatorname{diam}(S) = d_S(u, v)$, since $\operatorname{diam}(T + pq) = d_{T+pq}(u, v) = d_S(u, v) \leq \operatorname{diam}(S) \leq \operatorname{diam}(T + pq)$. By Lemma 6, there is an optimal shortcut p^*q^* for T with $p^*, q^* \in \mathcal{B}$. By repeating the above, we obtain $d_S(u, v) = d_{T+p^*q^*}(u, v)$ and, thus, $\operatorname{diam}(T + pq) = \operatorname{diam}(T + p^*q^*)$, since

$$\begin{aligned}\operatorname{diam}(T + p^*q^*) &\leq \operatorname{diam}(T + pq) = \operatorname{diam}(S) \\ &= d_S(u, v) = d_{T+p^*q^*}(u, v) \leq \operatorname{diam}(T + p^*q^*) \ .\end{aligned}$$

Therefore, pq is an optimal shortcut for T and $\operatorname{diam}(T + pq) = \operatorname{diam}(S)$. □

Corollary 9 *Let T be a geometric tree, and let δ be the largest diameter of any $\mathcal{B}$-sub-tree of T. For every shortcut pq for T, we have $\delta \leq \operatorname{diam}(T + pq)$.* □

We compute the largest diameter δ of any $\mathcal{B}$-sub-tree of T as part of our preprocessing and we halt our search for an optimal shortcut if the current diameter reaches δ. This is not strictly necessary: if we ignore diametral pairs in the same $\mathcal{B}$-sub-tree, we still obtain an optimal shortcut even though we might not know the true optimal diameter. We may safely exclude diametral pairs in the same $\mathcal{B}$-sub-tree from consideration.

4.2. States and Operations

We group the diametral pairs and paths of the augmented tree $T + pq$ into types. The types of diametral pairs and paths in $T + pq$ define the states of our algorithm. We specify four types of movements for the shortcut as the operations for the algorithm.

4.2.1. Pair States

We define the following symbols to make the discussion about the candidates for diametral pairs of the augmented tree $T + pq$ more accessible.

- $\boldsymbol{x}$: A point in the primary $\mathcal{B}$-sub-tree X.
- $\boldsymbol{y}$: A point in the primary $\mathcal{B}$-sub-tree Y.
- ■: A point on $T \setminus (X \cup Y)$.
- ▲: A point in a secondary $\mathcal{B}$-sub-tree S_i for some $i = 1, 2, \ldots, k$.
- •: A point on the simple cycle $C(p, q)$ in $T + pq$ that lies on T.
- ∘: A point on the simple cycle $C(p, q)$ in $T + pq$ that lies on pq.

Every point on $T + pq$ lies (x) in the primary $\mathcal{B}$-sub-tree X, or (y) in the primary $\mathcal{B}$-sub-tree Y, or (■) somewhere along $T \setminus (X \cup Y)$, or (∘) on the shortcut pq.

If u, v is a diametral pair of $T + pq$, then u is (x, y) a leaf of a primary $\mathcal{B}$-sub-tree, or (▲) a leaf of a secondary $\mathcal{B}$-sub-tree, or (•, ∘) u is a point along the simple cycle $C(p, q)$ in $T + pq$. If u lies on the cycle $C(p, q)$, then u lies (•) on T or (∘) on the shortcut pq. This distinction is complete: the only remaining location for u would be along a part of the backbone that does not belong to the cycle $C(p, q)$, i.e., on the path from a to p or

the path from q to b. In both cases, we could extend the supposedly diametral path from v to u in $T + pq$ to a strictly longer path from v to a leaf of X or a leaf of Y.

This leads to the following types of diametral pairs in $T + pq$. The choice for this particular distinction will become clear when we discuss the details of the algorithm.

x-y: Diametral pairs x, y of $T + pq$ with $x \in X$ and $y \in Y$.

x-■: Diametral pairs x, v of $T + pq$ with $x \in X$ and $v \in T \setminus (X \cup Y)$.

Diametral pairs of this type manifest as one of the following two sub-types.

- **x-▲**: Diametral pairs x, s_j of $T + pq$ with $x \in X$ and $s_j \in S_j$ for $j = 1, 2, \ldots, k$.
- **x-●**: Diametral pairs $x, \bar{x}$ of $T + pq$, where $x \in X$ and where $\bar{x}$ is the farthest point from x on $C(p, q)$. Since $|pq| \leq d_T(p, q)$, we always have $\bar{x} \in T$.

■-y: Diametral pairs u, y of $T + pq$ with $u \in T \setminus (X \cup Y)$ and $y \in Y$.

Diametral pairs of this type manifest as one of the following two sub-types.

- **▲-y**: Diametral pairs s_i, y of $T + pq$ with $s_i \in S_i$ and $y \in Y$ for some $i = 1, 2, \ldots, k$.
- **●-y**: Diametral pairs $y, \bar{y}$ of $T + pq$, where $y \in Y$ and where $\bar{y}$ is the farthest point from y on $C(p, q)$. Since $|pq| \leq d_T(p, q)$, we always have $\bar{y} \in T$.

■-■: Diametral pairs u, v of $T + pq$ with $u, v \in T \setminus (X \cup Y)$.

Diametral pairs of this type manifest as one of the following two sub-types.

- **▲-▲**: Diametral pairs s_i, s_j of $T + pq$ with $s_i \in S_i$ and $s_j \in S_j$ for $i, j = 1, 2, \ldots, k$.
- **▲-●**: Diametral pairs $s_i, \bar{s}_i$ of $T + pq$ with $s_i, \bar{s}_i \in T$ where $s_i \in S_i$ for some $i = 1, 2, \ldots, k$ and where $\bar{s}_i \in T$ is the farthest point from s_i on $C(p, q)$.

■-○: Diametral pairs u, v of $T + pq$ with $v \notin T$, i.e., $v \in pq$ with $p \neq v \neq q$.

Since $|pq| \leq d_T(p, q)$, we have $u \in T$. These diametral pairs only manifest as sub-type ▲-○, i.e., in the form $s_i, \bar{s}_i$ with $s_i \in S_i$, for $i = 1, 2, \ldots, k$, and where $\bar{s}_i$ is the farthest point from s_i on $C(p, q)$ that lies in the interior of the shortcut pq.

The order of the types of endpoints in the above notation is entirely arbitrary, e.g., x-■ and ■-x describe the same type of pair. There is no need to consider diametral pairs of sub-type ●-● or ●-○: the distance from x or from y to their farthest points on $C(p, q)$ is always larger than the distance between any two points on $C(p, q)$—unless T is a path with endpoints p and q, i.e., $T + pq = C(p, q)$. Since $|pq| \leq d_T(p, q)$, the farthest points $\bar{x}$ and $\bar{y}$ from x and from y on $C(p, q)$ lie on T. If x, v is a diametral pair of type x-■ in $T + pq$, then $v \in T$, and if u, y is of type ■-y in $T + pq$, then $u \in T$. Therefore, there do not exist any diametral pairs of sub-type x-○ or ○-y.

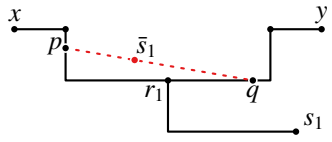


Figure 15: An optimal shortcut pq for a geometric tree T. The diametral pairs in the augmented tree $T + pq$ are (x, y), (x, s_1), (s_1, y), and $(\bar{s}_1, s_1)$, i.e., $T + pq$ is in pair state $\{x\text{-}y, x\text{-}■, ■\text{-}y, ■\text{-}○\}$ and in pair sub-state $\{x\text{-}y, x\text{-}▲, ▲\text{-}y, ▲\text{-}○\}$. Since pq is useful for x, y, but useless for (x, s_1) and (s_1, y), the path state of $T + pq$ is $\{x\text{-}pq\text{-}y, x\text{-}T\text{-}▲, ▲\text{-}T\text{-}y, ▲\text{-}p\text{-}○, ▲\text{-}q\text{-}○\}$.

The *pair state* is the set of types of diametral pairs in $T + pq$ and the *pair sub-state* is the set of sub-types of diametral pairs that are present in $T + pq$. For instance, if $T + pq$ has the diametral pairs x, y; x, s_3; x, s_5; and $x, \bar{x}$ then $T + pq$ is in pair state $\{x\text{-}y, x\text{-}\blacksquare\}$ and in the pair sub-state $\{x\text{-}y, x\text{-}\blacktriangle, x\text{-}\bullet\}$. Figure 15 illustrates the pair state and pair sub-state of an augmented tree together with its path state that we introduce next.

4.2.2. Path States

Let u, v be a diametral pair in $T + pq$. If $u, v \in T$ then every diametral path in $T + pq$ that connects u and v either contains the shortcut ($*\text{-}pq\text{-}*$) or not ($*\text{-}T\text{-}*$). If $u \in T$ and $v \notin T$, then every diametral path in $T + pq$ that connects u and v contains either p ($*\text{-}p\text{-}*$) or q ($*\text{-}q\text{-}*$). There are no diametral pairs with $u, v \notin T$. This leads to the following.

- ($*\text{-}\boldsymbol{pq}\text{-}*$): A diametral path that *does* contain pq and connects $u \in T$ with $v \in T$.
- ($*\text{-}\boldsymbol{T}\text{-}*$): A diametral path that *does not* contain pq and connects $u \in T$ with $v \in T$.
- ($*\text{-}\boldsymbol{p}\text{-}*$): A diametral path that contains p and connects $u \in T$ with $v \notin T$.
- ($*\text{-}\boldsymbol{q}\text{-}*$): A diametral path that contains q and connects $u \in T$ with $v \notin T$.

Any type of diametral partner (e.g., $x, y, \blacksquare, \blacktriangle, \bullet, \circ$) may appear in place of $*$. For instance, we denote a diametral path from x to $\bar{x}$ via the shortcut by $x\text{-}pq\text{-}\bullet$. The *path state* is the set of types of diametral paths that are present in $T + pq$. If $T + pq$ has the diametral pairs x, y; x, s_3; x, s_5; and $x, \bar{x}$ such that pq is useful for (x, y) and (x, s_3) but useless for (x, s_5) then $T + pq$ is in path state $\{x\text{-}pq\text{-}y, x\text{-}pq\text{-}\blacktriangle, x\text{-}T\text{-}\blacktriangle, x\text{-}pq\text{-}\bullet, x\text{-}T\text{-}\bullet\}$.

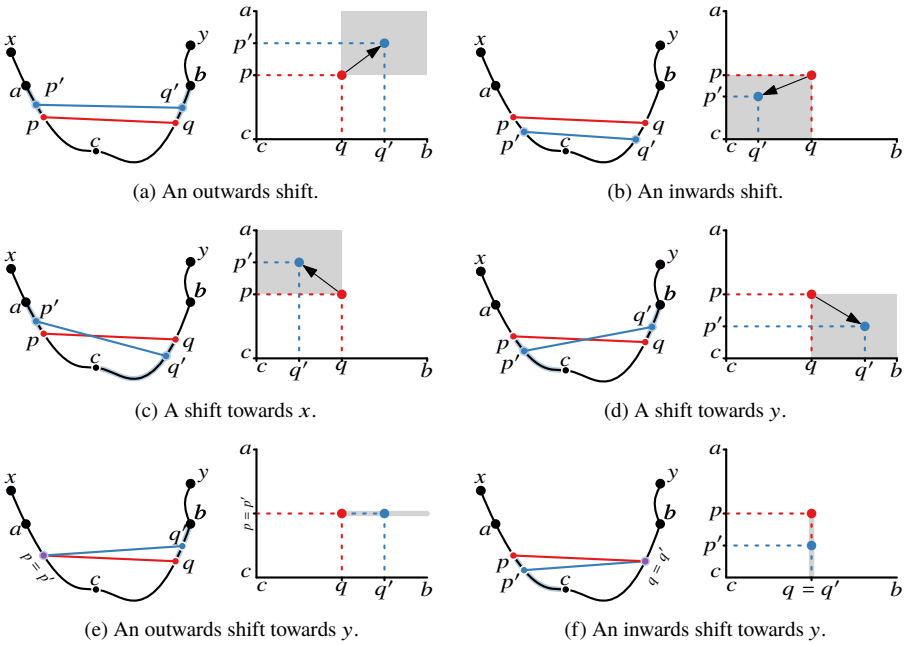
4.2.3. Operations

We distinguish the four types of movements for the shortcut depicted in Figure 16. Suppose we move a shortcut pq with $p, q \in \mathcal{B}$ such that $d_T(a, p) \leq d_T(a, q)$ to a position $p'q'$ with $p', q' \in \mathcal{B}$ such that $d_T(a, p') \leq d_T(a, q')$. The movement from pq to $p'q'$ is

- an *out-shift* if p approaches a and q approaches b, i.e., $d_T(a, p') \leq d_T(a, p)$ and $d_T(b, q') \leq d_T(b, q)$ and $d_T(a, p') \leq d_T(a, p) \leq d_T(a, q) \leq d_T(a, q')$,
- an *in-shift* if p recedes from a and q recedes from b, i.e., $d_T(a, p) \leq d_T(a, p')$ and $d_T(b, q) \leq d_T(b, q')$ and $d_T(a, p) \leq d_T(a, p') \leq d_T(a, q') \leq d_T(a, q)$,
- an *x-shift* if p approaches a and q recedes from b, i.e., $d_T(a, p') \leq d_T(a, p)$ and $d_T(b, q) \leq d_T(b, q')$ and $d_T(a, p') \leq d_T(a, p) \leq d_T(a, q') \leq d_T(a, q)$, or
- a *y-shift* if p recedes from a and q approaches b, i.e., $d_T(a, p) \leq d_T(a, p')$ and $d_T(b, q') \leq d_T(b, q)$ and $d_T(a, p) \leq d_T(a, p') \leq d_T(a, q) \leq d_T(a, q')$.

These types of movements intentionally overlap when one of the endpoints remains stationary, e.g., when $p = p'$ every y-shift is also an out-shift, as illustrated in Figure 16e. An out-shift is short for an outward shift, an in-shift is short for an inward shift, an x-shift is short for a shift towards x, and a y-shift is short for a shift towards y.

The operations, as defined above, allow us to move p or q through the absolute center c. However, the algorithm automatically maintains that p stays on the path from a to c and q stays on the path from c to b—without taking any special care to ensure this.



(a) An outwards shift.

(b) An inwards shift.

(c) A shift towards x.

(d) A shift towards y.

(e) An outwards shift towards y.

(f) An inwards shift towards y.

Figure 16: The four ways to move a shortcut pq to a new position $p'q'$. They are (a) the in-shift, (b) the out-shift, (c) the x-shift, and (d) the y-shift. These cases intentionally overlap when one endpoint of the shortcut remains stationary, i.e., when $p = p'$, as in (e), or when $q = q'$, as in (f). We sketch each operation on the left and plot of the positions of the shortcuts along the path from c to a and the path from c to b on the right. In each plot, the shaded region indicates all the shortcuts that can be reached with the same movement.

4.3. Blocking

Each type of diametral pair certifies that some type of movement cannot lead to a better shortcut, i.e., cannot reduce the continuous diameter. For instance, if an augmented tree $T + pq$ has a diametral pair of type x-y, then the distance between x and y will increase or remain the same when we in-shift pq. In this sense, x-y *blocks* any in-shift, x-■ *blocks* any y-shift, ■-y *blocks* any x-shift, and ■-■ and ■-○ *block* any out-shift.

Lemma 10 (Blocking) *Let pq be a shortcut for a geometric tree T with $p, q \in \mathcal{B}$.*

(1) *If $T + pq$ has a diametral pair of type x-y, then* $\operatorname{diam}(T + pq) \leq \operatorname{diam}(T + p'q')$ *for every shortcut $p'q'$ such that the movement from pq to $p'q'$ is an inward shift.*

(2) *If $T + pq$ has a diametral pair of type x-■, then* $\operatorname{diam}(T + pq) \leq \operatorname{diam}(T + p'q')$ *for every shortcut $p'q'$ such that the movement from pq to $p'q'$ is a shift towards y.*

(3) *If $T + pq$ has a diametral pair of type ■-y, then* $\operatorname{diam}(T + pq) \leq \operatorname{diam}(T + p'q')$ *for every shortcut $p'q'$ such that the movement from pq to $p'q'$ is a shift towards x.*

(4) *If $T + pq$ has a diametral pair of type ■-■ or ■-○, then* $\operatorname{diam}(T + pq) \leq \operatorname{diam}(T + p'q')$ *for every shortcut $p'q'$ such that the movement from pq to $p'q'$ is an outward shift.*

Proof. Let T be a geometric tree with backbone $\mathcal{B}$, and let $p, q, p', q' \in \mathcal{B}$.

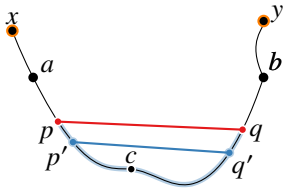
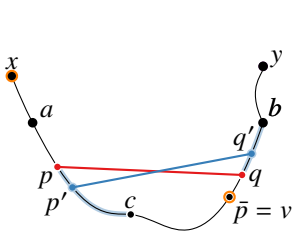
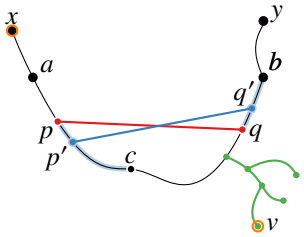


Figure 17: An in-shift from pq to $p'q'$ that increases the distance between a diametral pair x, y of $T + pq$.



(a) x-■ as x-●.



(b) x-■ as x-▲.

Figure 18: A y-shift from pq to $p'q'$ that increases the distance between a diametral pair x, v of type x-■ manifesting as (a) x-● and (b) x-▲.

If there exists a diametral pair u, v of $T + pq$ such that $u, v \in T$, and $p'q'$ is indifferent for $\{u, v\}$, i.e., $d_T(u, v) = d_{T+p'q'}(u, v)$, then $\mathrm{diam}(T + pq) \leq \mathrm{diam}(T + p'q')$, since

$$\mathrm{diam}(T + pq) = d_{T+pq}(u, v) \leq d_T(u, v) = d_{T+p'q'}(u, v) \leq \mathrm{diam}(T + p'q') \ .$$

Thus, we may assume $p'q'$ is useful for any diametral pair (u, v) of $T + pq$ with $u, v \in T$.

We have $\mathrm{diam}(T + pq) \leq \mathrm{diam}(T + p'q')$ if there exists a diametral pair u, v of $T + pq$ with $u, v \in T$ such that $p'q'$ is useful for (u, v), the path from u to p' in T contains p, i.e., $d_T(u, p') = d_T(u, p) + d_T(p, p')$, and the path from v to q' contains q, i.e., $d_T(q', v) = d_T(q', q) + d_T(q, v)$, since

$$\begin{aligned}
\mathrm{diam}(T + pq) &= d_{T+pq}(u, v) && (u, v \text{ is diametral in } T + pq) \\
&\leq d_T(u, p) + |pq| + d_T(q, v) \\
&\leq d_T(u, p) + d_T(p, p') + |p'q'| + d_T(q', q) + d_T(q, v) \\
&= d_T(u, p') + |p'q'| + d_T(q', v) && (p' \text{ lies on the path from } u \text{ to } p) \\
&= d_{T+p'q'}(u, v) && (p'q' \text{ is useful for } (u, v)) \\
&\leq \mathrm{diam}(T + p'q') \ .
\end{aligned}$$

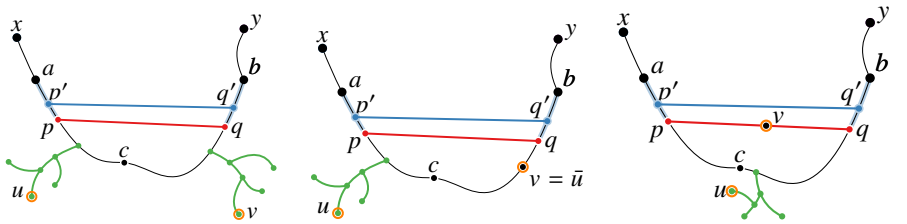
We use this observation to prove that each diametral pair blocks the stated operation.

(1) Suppose $T + pq$ has a diametral pair of type x-y. Let $p'q'$ be a shortcut for T such that the movement from pq to $p'q'$ is an inward shift, as illustrated in Figure 17. Then $\mathrm{diam}(T + pq) \leq \mathrm{diam}(T + p'q')$, since x, y is a diametral pair of $T + pq$ with $x, y \in T$ such that p lies on the path from x to p' and q lies on the path from q' to y.

(2) Suppose $T + pq$ has a diametral pair of type x-■. Let $p'q'$ be a shortcut for T such that the movement from pq to $p'q'$ is a shift towards y, as illustrated in Figure 18. Let x, v be a diametral pair of type x-■ in $T + pq$. Then $v \in T$, since $|pq| \leq d_T(p, q)$. The path from x to p' contains p, since the movement from pq to $p'q'$ is a shift towards y. The path from q' to v in T contains q, since otherwise y would be farther away from x than v in $T + pq$. Therefore, we have $\mathrm{diam}(T + pq) \leq \mathrm{diam}(T = p'q')$.

(3) Suppose $T + pq$ has a diametral pair of type ■-y. This is symmetric to Case (2).

(4) Suppose $T + pq$ has a diametral pair of type ■-■ or ■-○. Let $p'q'$ be a shortcut for T such that the movement from pq to $p'q'$ is an out-shift, as illustrated in Figure 19. Every diametral pair of type ■-■ or ■-○ manifests as subtype ▲-▲, ▲-● or ▲-○.



(a) A pair of subtype ▲-▲. (b) A pair of subtype ▲-●. (c) A pair of subtype ▲-○.

Figure 19: An illustration of an outward shift from pq to $p'q'$ with a diametral pair u, v in $T + pq$ manifesting as (a) subtype ▲-▲, (b) subtype ▲-●, and (c) subtype ▲-○.

(a) Suppose there is some diametral pair u, v of $T + pq$ of subtype ▲-▲. Then u and v are leaves of two secondary $\mathcal{B}$-sub-trees that are attached to $\mathcal{B}$ along the path from p to q in T. Otherwise, x or y would be strictly farther from v than u. Then we have $\operatorname{diam}(T + pq) \leq \operatorname{diam}(T + p'q')$, since u, v is a diametral pair of $T + pq$ with $u, v \in T$ where p lies on the path from u to p' and q lies on the path from q' to v.

(b) Suppose there is some diametral pair u, v in $T + pq$ of subtype ▲-● or ▲-○. Then one of u or v is a leaf of a secondary $\mathcal{B}$-sub-tree and the other is the farthest point on $C(p, q)$ from said leaf. Without loss of generality, suppose u lies in a secondary $\mathcal{B}$-sub-tree S. The root r of S must lie along the path from p to q in T. Otherwise, x or y would be strictly farther from u than v. Thus,

$$d_{T+pq}(u, v) = d_T(u, r) + d_{T+pq}(r, v) = d_T(u, r) + \frac{|pq| + d_T(p, q)}{2} \quad .$$

Let v' be the farthest point from u on the simple cycle $C(p'q')$ in $T + p'q'$. The vertex r lies on the path from p' to q' in T, since the movement from pq to $p'q'$ is an outward shift, and since r lies on the path from p to q in T. Thus,

$$d_{T+p'q'}(u, v') = d_T(u, r) + d_{T+p'q'}(r, v') = d_T(u, r) + \frac{|p'q'| + d_T(p', q')}{2} \quad .$$

The simple cycle in the augmented tree is growing as pq shifts outwards to $p'q'$, i.e., $|pq| + d_T(p, q) \leq |p'q'| + d_T(p'q')$. This implies $d_{T+pq}(u, v) \leq d_{T+p'q'}(u, v')$ and, therefore, $\operatorname{diam}(T + pq) = d_{T+pq}(u, v) \leq d_{T+p'q'}(u, v') \leq \operatorname{diam}(T + p'q')$.

Therefore, each type of diametral pair blocks one type of movement, as claimed. □

4.4. Sudden Optimality

As a consequence of Lemma 10, a shortcut pq is optimal for a geometric tree T when each of the four types of movements is blocked by some diametral pair in $T + pq$.

Corollary 11 *If an augmented tree $T + pq$ has diametral pairs of type x-y, x-■, and ■-y, as well as a diametral pair of type ■-■ or ■-○, then pq is an optimal shortcut for T.* □

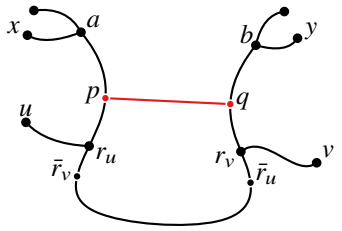


Figure 20: An augmented tree $T + pq$ with diametral paths of type x-pq-y and ■-pq-■. The diametral path of type ■-pq-■ connects u with v via pq. The points $\bar{r}_u$ and $\bar{r}_v$ mark the farthest points from r_u and r_v, respectively on the simple cycle in $T + pq$. Their position indicates that pq is useful for (u, v).

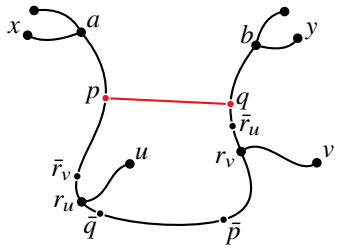


Figure 21: An augmented tree $T + pq$ with diametral paths of type x-pq-y and ■-T-■. The diametral path of type ■-T-■ connects u with v via the tree only. The points $\bar{r}_u$ and $\bar{r}_v$ mark the farthest points from r_u and r_v, respectively on the simple cycle in $T + pq$. Their position indicates that pq is useless for (u, v).

We argue that we may ignore diametral pairs of type ■-■ when a diametral pair of x-y is present, provided that we keep track of diametral pairs of type ■-y and x-■. This is important for the running time, as there may be $\Omega(n^2)$ candidates for pairs of type ■-■.

Theorem 12 *If an augmented tree $T + pq$ has diametral pairs of types x-y and ■-■, then $T + pq$ has diametral pairs of type x-■ and ■-y and, thus, pq is optimal for T.*

Proof. Suppose an augmented tree $T + pq$ has diametral pairs of types x-y and ■-■. Then pq is useful for (x, y), since $T + pq$ has diametral pairs besides those of type x-y. Therefore, the diametral pairs of type x-y in $T + pq$ are connected by diametral paths of type x-pq-y. We distinguish whether the diametral pairs of type ■-■ in $T + pq$ are connected by diametral paths of type ■-pq-■ or ■-T-■. In each case, we argue that there are diametral pairs of types x-■ and of type ■-y, which implies that pq is optimal for T.

1. Suppose $T + pq$ has diametral paths of type x-pq-y and ■-pq-■, as in Figure 20. Then $T + pq$ has a diametral pair $u, v \in T \setminus (X \cup Y)$ such that a shortest path from u to v contains pq. We show that the pairs u, y and x, v are diametral in $T + pq$. Since pq is useful for (u, v), the shortcut pq is also useful for (u, q) and, thus, (u, y). By comparing the paths x-pq-y and u-pq-y, we obtain $d_T(u, p) \leq d_T(x, p)$, since

$$\begin{aligned} d_T(u,p) + |pq| + d_T(q,y) &= d_{T+pq}(u,y) && (pq \text{ is useful for } (u,y)) \\ &\leq d_{T+pq}(x,y) && (x,y \text{ is diametral in } T+pq) \\ &= d_T(x,p) + |pq| + d_T(q,y)\ . && (pq \text{ is useful for } (x,y)) \end{aligned}$$

   Analoguously, we obtain $d_T(q, v) \leq d_T(q, y)$ by comparing the paths x-pq-y and x-pq-v. Comparing the diametral paths x-pq-y and u-pq-v yields $d_T(x, p) + d_T(q, y) = d_T(u, p) + d_T(q, v)$. This equation cannot be satisfied when $d_T(u, p) < d_T(x, p)$ or $d_T(q, v) < d_T(q, y)$, since $d_T(u, p) \leq d_T(x, p)$ and $d_T(q, v) \leq d_T(q, y)$. Therefore, we have $d_T(u, p) = d_T(x, p)$ and $d_T(q, v) = d_T(q, y)$. This implies that u, y and x, v are diametral pairs in $T + pq$, since

$$d_{T+pq}(u, y) = d_T(u, p) + |pq| + d_T(q, y)$$

$$
\begin{aligned}
&= d_T(x,p) + |pq| + d_T(q,y) = \operatorname{diam}(T+pq)\ , \\
\text{and}\ \ d_{T+pq}(x,v) &= d_T(x,p) + |pq| + d_T(q,v) \\
&= d_T(x,p) + |pq| + d_T(q,y) = \operatorname{diam}(T+pq)\ .
\end{aligned}
$$

Hence, $T+pq$ is in the pair state $\{x\text{-}y, \blacksquare\text{-}\blacksquare, x\text{-}\blacksquare, \blacksquare\text{-}y\}$ and, thus, pq is optimal for T.

2. Suppose $T+pq$ has diametral paths of type x-pq-y and $\blacksquare$-T-$\blacksquare$, as in Figure 21. Then there exists a diametral pair u, v of $T+pq$ with $u, v \in T$ such that there is a shortest path from u to v in $T+pq$ that does not contain pq. If u lies in a secondary $\mathcal{B}$-sub-tree S_u, then let r_u be the root of S_u. Otherwise, let $r_u = u$. Likewise, let r_v be the root of the $\mathcal{B}$-sub-tree containing v or let $r_v = v$ when $v \in \mathcal{B}$. Without loss of generality, r_u lies on the path in T from a to r_v. Otherwise, we swap u and v. We show that u, y and x, v are diametral in $T+pq$. Since u-T-v is diametral, pq cannot be useful for (u,v) and, thus, pq cannot be useful for (r_u, r_v), i.e., $d_T(r_u, r_v) \le d_T(r_u, p) + |pq| + d_T(q, r_v)$. On the other hand, pq must be useful for (x,v), i.e., $d_{T+pq}(x,v) < d_T(x,v)$, since

$$
\begin{aligned}
d_{T+pq}(x,v) &\le \operatorname{diam}(T+pq) \\
&= d_{T+pq}(u,v) && (u,v \text{ is diametral in } T+pq) \\
&= d_T(u,v) && (pq \text{ is not useful for } (u,v)) \\
&= d_T(u,r_u) + d_T(r_u,v) && (r_u \text{ lies on the path from } u \text{ to } v) \\
&< d_T(x,r_u) + d_T(r_u,v) && (u \in T \setminus X) \\
&= d_T(x,v)\ . && (r_u \text{ lies on the path from } x \text{ to } v)
\end{aligned}
$$

Likewise, pq must be useful for (u,y). We have $d_T(q,v) \le d_T(q,y)$, as

$$
\begin{aligned}
d_T(x,p) + |pq| + d_T(q,v) &= d_{T+pq}(x,v) && (pq \text{ is useful for } (x,v)) \\
&\le d_{T+pq}(x,y) && (x,y \text{ is diametral in } T+pq) \\
&= d_T(x,p) + |pq| + d_T(q,y) && (pq \text{ is useful for } (x,y))
\end{aligned}
$$

The pair u, y is diametral in $T+pq$, i.e., $d_{T+pq}(u,y) = \operatorname{diam}(T+pq)$, since

$$
\begin{aligned}
\operatorname{diam}(T+pq) &= d_{T+pq}(u,v) && (u,v \text{ is diametral in } T+pq) \\
&= d_T(u,v) && (pq \text{ is not useful for } (u,v)) \\
&= d_T(u,r_u) + d_T(r_u,r_v) + d_T(r_v,v) \\
&\le d_T(u,r_u) + d_T(r_u,p) + |pq| + d_T(q,r_v) + d_T(r_v,v) \\
& && (pq \text{ not useful for } (r_u,r_v)) \\
&= d_T(u,p) + |pq| + d_T(q,v) \\
&\le d_T(u,p) + |pq| + d_T(q,y) && (d_T(q,v) \le d_T(q,y)) \\
&= d_{T+pq}(u,y) \le \operatorname{diam}(T+pq)\ . && (pq \text{ is useful for } (u,y))
\end{aligned}
$$

Likewise, x, v is diametral in $T+pq$ and $T+pq$ is in pair state $\{x\text{-}y, x\text{-}\blacksquare, \blacksquare\text{-}y, \blacksquare\text{-}\blacksquare\}$.

Hence, if $T+pq$ has diametral pairs of type x-y and $\blacksquare$-$\blacksquare$, then pq is optimal for T. □

5. Continuous Algorithm

Inspired by the plane-sweep paradigm, we—conceptually—move the shortcut continuously while changing its speed and direction at certain events where the pair state or path state changes. To implement this approach, we discretize this movement such that the shortcut jumps from one event to the next.

5.1. The Algorithm from the Pair State Perspective

Figure 22 describes the continuous algorithm in terms of the pair states and operations. Initially, we place the shortcut with both endpoints on the absolute center c of the geometric tree T. This ensures that we start in pair state $\{x\text{-}y\}$. The algorithm consists of at most three phases: an outwards shift, possibly followed by a shift towards x or a shift towards y, possibly followed by another outwards shift. Some pair states are marked as final states with a double border. If we reach a final state, we terminate our search and report the best shortcut that we have found. For the other states, we specify the direction in which we move the shortcut.

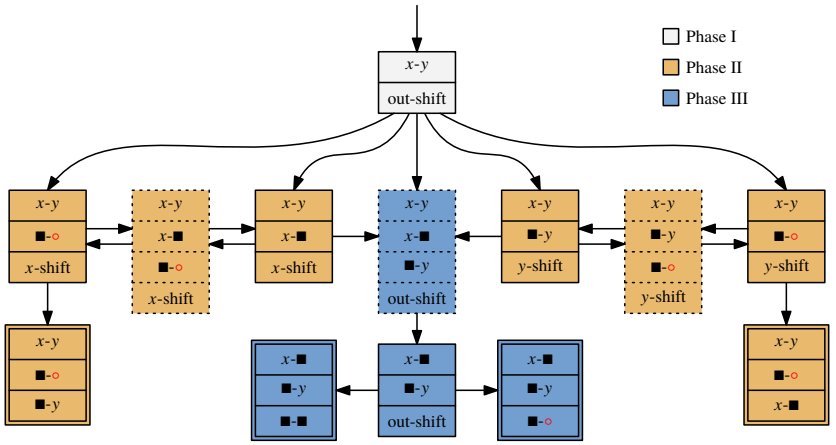


Figure 22: The pair states encountered during our search for an optimal shortcut for a tree. There are three types of states: First, regular states (single boundary) indicate the pair state and the operation applied (out-shift, x-shift, or y-shift). Second, transition states (dotted boundary) are visited only momentarily while transit from one regular state to another regular state. Third, final states (double boundary) where we terminate our search and report the best shortcut encountered. Under certain conditions, the search may also terminate early in non-final states. We always start in state $\{x\text{-}y\}$ with an outward shift. When we reach the pair state $\{x\text{-}y, \blacksquare\text{-}\circ\}$ then we perform both a shift towards x *and* separately a shift towards y.

For the sake of simplicity, we omit some pair states and transitions from Figure 22. First, we omit all pair states containing $x\text{-}y$ and $\blacksquare\text{-}\blacksquare$, due to Theorem 12. Second, we omit transitions that are implied by transitivity, e.g., we model a transition from $\{x\text{-}y\}$ to $\{x\text{-}y, x\text{-}\blacksquare, \blacksquare\text{-}y\}$ by transitioning from $\{x\text{-}y\}$ to $\{x\text{-}y, x\text{-}\blacksquare\}$ and then from $\{x\text{-}y, x\text{-}\blacksquare\}$ to $\{x\text{-}y, x\text{-}\blacksquare, \blacksquare\text{-}y\}$. Third, we omit pair states that are supersets of any final states.

Phase I: Shifting Outwards. In Phase I, we continuously shorten all diametral paths of type x-y with an out-shift: we move p from c towards a and we move q from c towards b. If p reaches a before q reaches b, then p remains at a and q continues to move towards b. Likewise, p continues to move towards a if q reaches b. In Phase I, the current shortcut is the best shortcut encountered so far. Phase I ends when the shortcut reaches the end of the backbone, i.e., $pq = ab$, or when a second type of diametral pair appears.

We might switch directly from Phase I to Phase III and skip Phase II when diametral pairs of type x-■ and ■-y appear simultaneously. For instance, suppose that T is a path and that we move both endpoints of the shortcut with unit speed during Phase I. Then, $x, \bar{x}$ and $\bar{y}, y$ will become diametral pairs of types x-● and ●-y at the same time, where $\bar{x}$ and $\bar{y}$ are the farthest points from x and y along the simple cycle in $T + pq$, respectively.

Phase II: Shifting Sideways. The second phase begins when we transit from pair state $\{x\text{-}y\}$ to a pair state containing x-y. If we transit from $\{x\text{-}y\}$ to $\{x\text{-}y, x\text{-}■\}$, then we shift towards x. If we transit from $\{x\text{-}y\}$ to $\{x\text{-}y, ■\text{-}y\}$, then we shift towards y. If we transit from $\{x\text{-}y\}$ to $\{x\text{-}y, ■\text{-}○\}$, then we branch the search into a shift towards x and a shift towards y. In the following, we discuss the x-shift in Phase II for the pair states $\{x\text{-}y, x\text{-}■\}$, $\{x\text{-}y, x\text{-}■, ■\text{-}○\}$, and $\{x\text{-}y, ■\text{-}○\}$. The y-shift in Phase II for the pair states $\{x\text{-}y, ■\text{-}y\}$, $\{x\text{-}y, ■\text{-}y, ■\text{-}○\}$, and $\{x\text{-}y, ■\text{-}○\}$ is symmetric.

Suppose we reach the pair state $\{x\text{-}y, x\text{-}■\}$ from $\{x\text{-}y\}$. All diametral paths in $T + pq$ contain the path from a to p. We move p closer to a, thereby shrinking the diameter. At the same time, we move q with a speed towards a that keeps all diametral paths in balance. Thus, we remain in the current pair state until another diametral pair appears. In state $\{x\text{-}y, x\text{-}■\}$, the current shortcut is the best shortcut encountered so far.

If we reach the pair state $\{x\text{-}y, ■\text{-}○\}$, then we move p towards a and adjust the position of q to balance the diametral paths of type x-pq-y with those of type ■-p-○ and ■-q-○. In this state, the diameter shrinks and grows with the length of the shortcut and the best shortcut so far is the shortest shortcut encountered since we entered this state.

Balancing the diametral paths when moving pq ensures that we remain in the current pair state until another diametral pair appears. It also restricts our search considerably: we are performing a linear search, since the speed of q is determined by the speed of p, the path state, and the change in the length of the shortcut, as shown in Section 6.

Phase II ends when p reaches a, when we transit to a pair state containing x-y and ■-■, when we transit from $\{x\text{-}y, ■\text{-}○\}$ to the final pair state $\{x\text{-}y, ■\text{-}○, ■\text{-}y\}$, or when we transit from $\{x\text{-}y, x\text{-}■\}$ to the pair state $\{x\text{-}y, x\text{-}■, ■\text{-}y\}$ where Phase III begins.

Phase III: Shifting Outwards. Phase III starts when we reach $\{x\text{-}y,\ x\text{-}■,\ ■\text{-}y\}$ from $\{x\text{-}y, x\text{-}■\}$ or from $\{x\text{-}y, ■\text{-}y\}$. Since x-y, x-■, and ■-y block all other movements, we shift outwards balancing x-■ and ■-y. We immediately transit to $\{x\text{-}■, ■\text{-}y\}$, since the path from x to y via the shortcut shrinks faster than the diametral paths connecting x-■ and ■-y. If we reach Phase III, then the shortest shortcut encountered during Phase III is optimal. Phase III ends when p meets a, when q meets b, or when ■-■ or ■-○ appears.

5.2. Optimality

We argue that the shortcut produced by the above algorithm is indeed optimal, using invariants for each pair state that follow from the blocking lemma (Lemma 10).

Moreover, we show that one endpoint of the shortcut remains on the path from a to c while the other endpoint remains on the path from c to b along the backbone.

While the algorithm is in Phase I, there is an optimal shortcut p^*q^* that we reach from the current shortcut pq by shifting upwards, by shifting towards x, by shifting towards y, or by remaining stationary. This invariant holds because the diametral pairs of type x-y block any inward shift, i.e., we have $\mathrm{diam}(T + pq) \leq \mathrm{diam}(T + p'q')$ for any shortcut $p'q'$ that we reach with an inward shift from pq, due to Lemma 10. Therefore, if Phase I ends with $pq = ab$, then ab is an optimal shortcut for T.

While the algorithm is in pair state $\{x\text{-}y, x\text{-}\blacksquare\}$ of Phase II, there is an optimal shortcut p^*q^* that we reach from the current shortcut pq by shifting towards x, by shifting outwards, or by remaining stationary. This invariant holds because the diametral pairs of type x-y and x-$\blacksquare$ in $T + pq$ block any inward shift and any shift towards y, i.e., we have $\mathrm{diam}(T + pq) \leq \mathrm{diam}(T + p'q')$ for any shortcut $p'q'$ that we reach with an inward shift or a shift towards y from pq, due to Lemma 10. Therefore, if Phase II concludes with $p = a$ in pair state $\{x\text{-}y, x\text{-}\blacksquare\}$, then the current shortcut is optimal, because when $p = a$, every shift towards x is also an inwards shift and, thus, blocked by x-y, and every outwards shift is also a shift towards y and, thus, blocked by x-$\blacksquare$.

While the algorithm is in pair state $\{x\text{-}y, \blacksquare\text{-}\circ\}$ of Phase II, there is an optimal shortcut p^*q^* that we reach from the current shortcut pq by shifting towards x, by shifting towards y, or by remaining stationary. This invariant holds because the diametral pairs of type x-y and $\blacksquare$-$\circ$ in $T + pq$ block any in-shift and any out-shift, i.e., we have $\mathrm{diam}(T + pq) \leq \mathrm{diam}(T + p'q')$ for any shortcut $p'q'$ that we reach with an inward or outward shift from pq, due to Lemma 10. We cannot miss an optimal shortcut:

Invariant 1 *Suppose there is an optimal shortcut p^*q^* for T in the direction of an x-shift when the algorithm transits to the pair state $\{x\text{-}y, \blacksquare\text{-}\circ\}$ for the first time. While performing an x-shift in pair state $\{x\text{-}y, \blacksquare\text{-}\circ\}$ of Phase II, we have already encountered an optimal shortcut or we can reach p^*q^* from the current shortcut pq with an x-shift.*

Proof. Suppose we perform an x-shift from pq while balancing the diametral paths x-y and $\blacksquare$-$\circ$ until the pair state changes at some position $p'q'$. If the movement from $p'q'$ to p^*q^* is an x-shift, then so is the movement from pq to p^*q^*, by transitivity.

Suppose pq to p^*q^* is an x-shift while $p'q'$ to p^*q^* is not an x-shift. We argue that we encounter an optimal shortcut while shifting from pq to $p'q'$. Let $p''q''$ be the last position during the shift from pq to $p'q'$ where the movement from $p''q''$ to p^*q^* is an x-shift. Then $T + p''q''$ is in pair state $\{x\text{-}y, \blacksquare\text{-}\circ\}$ and $p'' = p^*$ or $q'' = q^*$. If $p'' = p^*$, then $p''q''$ to p^*q^* is an inward shift towards x. Since x-y blocks any in-shift, $\mathrm{diam}(T+p''q'') \leq \mathrm{diam}(T+p^*q^*)$. Likewise, if $q'' = q^*$, then $p''q''$ to p^*q^* is an outward shift towards x. Since $\blacksquare$-$\circ$ blocks any out-shift, $\mathrm{diam}(T + p''q'') \leq \mathrm{diam}(T + p^*q^*)$. In both cases, $p''q''$ is optimal, as p^*q^* is optimal. Therefore, we have encountered an optimal shortcut when the movement from pq to p^*q^* is no longer an x-shift. □

If Phase II ends in $\{x\text{-}y, \blacksquare\text{-}\circ\}$ with $p = a$ or in the final state $\{x\text{-}y, \blacksquare\text{-}\circ, \blacksquare\text{-}y\}$, then all directions are blocked and, by the invariant, we have encountered an optimal shortcut.

When the algorithm enters Phase III with a transition to $\{x\text{-}y, x\text{-}\blacksquare, \blacksquare\text{-}y\}$, then there is an optimal shortcut p^*q^* that we reach from the current shortcut pq by shifting outwards or by remaining stationary. This is because all movements, except for the out-shift, are

blocked by the diametral pairs of types x-y, x-■, and ■-y. As we perform an out-shift in the pair state $\{x\text{-}■, ■\text{-}y\}$ of Phase III, we have already encountered an optimal shortcut or the optimal shortcut p^*q^* can still be reached with an out-shift from the current shortcut.

Therefore, we have encountered an optimal shortcut when Phase III ends in the final state $\{x\text{-}■, ■\text{-}y, ■\text{-}■\}$, or in $\{x\text{-}■, ■\text{-}y, ■\text{-}○\}$, or when $p = a$ or $q = b$ in the state $\{x\text{-}■, ■\text{-}y\}$.

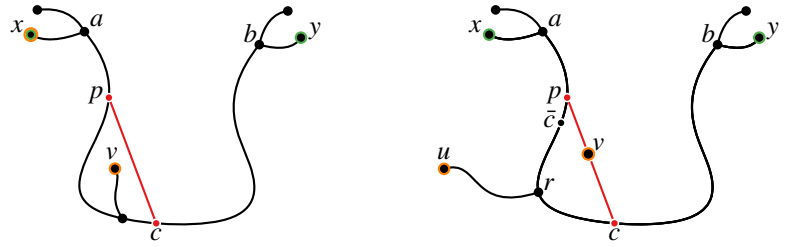
Finally, we argue that the endpoints of the shortcut remain on their respective sub-paths of the backbone, i.e., the absolute center c remains on the path from p to q.

Invariant 2 *At any moment during the course of the continuous algorithm, the point p lies on the path from a to c in T and the point q lies on the path from c to b in T.*

Proof. The invariant holds at the start of Phase I, since we begin by setting $pq = cc$. Throughout Phase I, we perform an out-shift that upholds the invariant: both p and q move away from c and neither does p move past a nor does q move past b.

For Phase II, we prove that q cannot pass through c during an x-shift and, symmetrically, that p cannot pass through c during y-shift. Hence, the invariant holds throughout Phase II, since the algorithm terminates when p reaches a or when q reaches b.

Assume, for a contradiction, that q reaches c when performing an x-shift in Phase II. Without loss of generality, let this be the first time q reaches c during Phase II. Hence, the invariant holds until now and, thus, p lies on the path from a to c. Since we are in Phase II, there are diametral pairs of type x-y in $T + pq$ as well as diametral pairs of type x-■ or ■-○. We distinguish two cases depending on the pair state of the augmented tree $T + pc$. In each case, we derive a contradiction, i.e., q could never have reached c.



(a) Path state $\{x\text{-}y, x\text{-}■\}$ for $T + pc$.

(b) Path state $\{x\text{-}y, ■\text{-}○\}$ for $T + pc$.

Figure 23: Two impossible cases when $T + pc$ has diametral pairs of types (a) x-y and x-■ or (b) x-y and ■-○.

1. Suppose $T + pc$ has diametral pairs of type x-y and x-■, as illustrated in Figure 23a. Let x, v be a diametral pair of type x-■ in $T + pc$. Since v does not lie in a primary $\mathcal{B}$-sub-tree of T, we have $d_T(c, v) < d_T(c, y)$. Since $T + pc$ has other diametral pairs besides those of type x-y, the shortcut pc must be useful for (x, y), i.e., $d_T(x, p) + |pc| + d_T(c, y) = d_{T+pc}(x, y)$. This leads to the contradiction $\operatorname{diam}(T + pc) < \operatorname{diam}(T + pc)$, since

$$\begin{aligned} \operatorname{diam}(T + pc) &= d_{T+pc}(x, v) && (x, v \text{ is diametral in } T + pc) \\ &\leq d_T(x, p) + |pc| + d_T(c, v) \end{aligned}$$

$$
\begin{aligned}
&< d_T(x,p) + |pc| + d_T(c,y) && (d_T(c,v) < d_T(c,y)) \\
&= d_{T+pc}(x,y) && (pc \text{ is useful for } (x,y)) \\
&= \operatorname{diam}(T+pc)\ . && (x,y \text{ is diametral in } T+pc)
\end{aligned}
$$

2. Suppose $T+pc$ has diametral pairs of type x-y and ■-○, as illustrated in Figure 23b. Let u, v be a diametral pair of type ■-○ in $T+pc$ with $v \notin T$. Then $v \in pc$ with $p \neq v \neq c$ and u lies in a secondary $\mathcal{B}$-sub-tree with root r attached to the path from p to c along $\mathcal{B}$. Let $\bar{c}$ be the farthest point from c on the simple cycle $C(p,c)$ in $T+pc$. This leads to the contradiction $\operatorname{diam}(T+pc) < \operatorname{diam}(T+pc)$, since

$$
\begin{aligned}
\operatorname{diam}(T+pc) &= d_{T+pc}(u,v) && (u,v \text{ is diametral in } T+pc) \\
&= d_T(u,r) + d_{T+pc}(r,v) && (u,v \text{ is of type ■-○}) \\
&\leq d_T(u,c) + d_{T+pc}(r,v) && (r \text{ is on the path from } p \text{ to } c) \\
&< d_T(y,c) + d_{T+pc}(r,v) && (u \text{ is in a secondary } \mathcal{B}\text{-sub-tree}) \\
&= d_T(y,c) + (|pc| + d_T(p,c))/2 \\
& && (r \text{ and } v \text{ are antipodal along } C(p,c)) \\
&= d_T(y,c) + d_{T+pc}(c,\bar{c}) && (c \text{ and } \bar{c} \text{ are antipodal along } C(p,c)) \\
&= d_{T+pc}(y,\bar{c}) && (c \text{ is on any path from } y \text{ to } \bar{c} \text{ in } T+pc) \\
&\leq \operatorname{diam}(T+pc)\ .
\end{aligned}
$$

Therefore, during Phase II, q remains on the path from c to b and, p remains on the path from a to c. This invariant holds for the entire algorithm, since the out-shift of Phase III cannot move p or q through c and it ends when p reaches a or when q reaches b. □

If we could implement and run the continuous algorithm, then it would produce an optimal shortcut pq for T where p lies along the path from a to c and q lies along the path from c to b. We simulate the continuous algorithm with a discretization.

6. Discretization

We view the trajectory of the shortcut throughout the continuous algorithm as a function of time. This function changes its shape at certain events, e.g., when one endpoint of the shortcut passes through a vertex or when the path state changes. We trace this trajectory from one event to the next in order to obtain a discrete simulation of the continuous algorithm. We describe this discretization for geometric trees whose edges are straight-line segments, where the trajectory is sufficiently simple between two subsequent events. Every event is a candidate for a change in the shape of the trajectory, but the shape of the trajectory may not change at every event. We compute some of the events in advance while other events emerge only during the simulation.

This section is structured as follows. In Section 6.1, we describe the preparations for the discrete algorithm, e.g., we decompose the tree into its backbone and its $\mathcal{B}$-sub-trees, we precompute the distances of candidates for diametral paths, and we place additional vertices at locations where events might occur during the simulation. This preprocessing takes $O(n \log n)$ time for a geometric tree with n vertices and straight-line edges.

In Sections 6.2 to 6.4, we describe how the discrete algorithm simulates Phases I, II, and III of the continuous algorithm. For Phase I and II, this simulation follows the trajectory of the continuous algorithm, precisely; these phases involve $O(n)$ events that we handle in $O(\log n)$ time per event. For Phase III, we need to resolve two issues:

1. We demonstrate that the continuous algorithm may encounter $\Omega(n^2)$ path state changes in Phase III. To rectify this, we modify Phase III such that the number of events that we have to handle reduces to $O(n)$ while preserving optimality.
2. In Phase III, it is too expensive to detect diametral paths of type ▲-pq-▲ that mark the end of Phase III. Thus, we first simulate an unconstrained version of the (modified) Phase III that ignores diametral pairs of type ▲-pq-▲ and then determine where this simulation should have terminated in a post-processing step.

The unconstrained modified Phase III consists of $O(n)$ events that we handle in $O(\log n)$ time each and the post-processing step takes $O(n \log n)$ time. Therefore, the discrete algorithm produces an optimal shortcut for a geometric tree in $O(n \log n)$ total time.

6.1. Preprocessing and Preparations

We are given a geometric tree T with n vertices and straight-line edges. The first step of our preprocessing is to determine the continuous diameter of T and to locate the absolute center c of T, which takes $O(n)$ time [2]. If c is not a vertex already, then we introduce a new vertex at c. We determine the network distance from c to every vertex of T and the set L of the farthest leaves from c using a breadth-first-search from c. Each leaf ℓ in L is a diametral partner in T and $d_T(c, \ell) = \operatorname{diam}(T)/2$. With this preparation, we determine the backbone $\mathcal{B}$ of T in $O(n)$ time using the following algorithm.

For each leaf $\ell \in L$, we walk from ℓ towards c in T and mark every visited vertex; the distances from the vertices to the absolute center guide our way. We stop traversing the path from ℓ to c when we reach c or when we encounter a vertex that is already marked as visited. This takes $O(n)$ time, since we mark one vertex in each step and each vertex is marked at most once. At the end, the marked vertices are precisely the vertices that lie along some diametral path of T. If we marked three or more neighbouring vertices of c as visited, then there are three diametral paths whose intersection is c. In this case, the backbone consists only of c itself. Otherwise, we marked exactly two neighbouring vertices of c as visited and the backbone is a path containing these vertices. We follow the trail of marked vertices from c in one direction until we reach a vertex a that is either a leaf of T or a vertex with more than two marked neighbours. Either all diametral paths of T end at a or some diametral paths of T diverge at a. Thus, a is one endpoint of the backbone. We locate the other endpoint b of $\mathcal{B}$ by following the trail of marked vertices from c in the other direction. Hence, we obtain the backbone $\mathcal{B}$ of T in $O(n)$ time.

Consider a $\mathcal{B}$-sub-tree S of T with root r. Since we already know the distance from every vertex of T to c, we can compute the distance from r to a farthest point from r in S using $\max_{s \in S} d_T(r, s) = \left[\max_{\ell \in L(S)} d_T(\ell, c)\right] - d_T(r, c)$, where $L(S)$ is the set of leaves of S. With this approach, we obtain the heights $d_T(x, a)$, $d_T(r_1, s_1)$, $d_T(r_2, s_2)$, $\ldots$, $d_T(r_k, s_k)$, and $d_T(b, y)$ of all $\mathcal{B}$-sub-trees of T as well as the height of the tallest secondary $\mathcal{B}$-sub-tree $\hat{h} = \max\{d_T(r_1, s_1), d_T(r_2, s_2), \ldots, d_T(r_k, s_k)\}$ in $O(n)$ time.

For two points $s, t \in \mathcal{B}$, we have $d_T(s, t) = d_T(s, c) + d_T(c, t)$ when c lies on the path from s to c and $d_T(s, t) = |d_T(s, c) - d_T(t, c)|$, otherwise. We label the vertices along

$\mathcal{B}$ to indicate whether they lie along the path from a to c or along the path from c to b. With this information, we precompute the lengths of the candidates for diametral paths of type x-T-▲ using $d_T(x, s_i) = d_T(x, a) + d_T(a, r_i) + d_T(r_i, s_i)$ for each $i = 1, 2, \ldots, k$. In addition, we sort the distances $d_T(x, s_1), d_T(x, s_2), \ldots, d_T(x, s_k)$ in descending order. This takes $O(n + k \log k)$ time and will help us to detect events where paths of type x-T-▲ become diametral. Likewise, we obtain the lengths $d_T(s_1, y), d_T(s_2, y), \ldots, d_T(s_k, y)$ of the candidates for diametral paths of type ▲-T-y and sort them descendingly.

We place new vertices to detect events in the simulation of Phase I and II where diametral paths of type x-pq-▲ appear while diametral paths of type x-pq-y are present. Suppose x-pq-y and x-pq-s_j are diametral paths in $T + pq$, for some $j = 1, 2, \ldots, k$. Then, the path from q to y in T has the same length as the path from q to s_j in T, i.e., $d_T(y, q) = d_T(q, s_j)$. Thus, q lies at the midpoint q_j of the path from y to s_j in T. We traverse the path from b to c and place the points among $q_1, q_2, \ldots, q_k$ that fall on this path at the appropriate distance from y. This takes only $O(n)$ additional time, because we know the order of the points $q_1, q_2, \ldots, q_k$ sorted by their distance from y, since $d_T(y, q_j) = d_T(y, s_j)/2 = d_T(q_j, s_j)$ and, since we have already sorted the distances $d_T(y, s_1), d_T(y, s_2), \ldots, d_T(y, s_k)$ in the previous step. Symmetrically, we place vertices at the points p_i with $d_T(x, p_i) = d_T(p_i, s_i)$, for $i = 1, 2, \ldots, k$ that fall on the path from a to c in order to detect path state changes involving x-pq-y and ▲-pq-y.

The following lemma will aid us in detecting events where the path state changes for certain types of diametral paths. We explain event handling below the lemma.

Lemma 13 *Let T be a geometric tree with straight-line edges and n vertices. There is a data structure with construction time $O(n)$ that supports $O(\log n)$-time queries for the longest paths of types x-pq-▲, x-T-▲, x-pq-●, x-T-●, ▲-pq-y, ▲-T-y, ●-pq-y, ●-T-y, ▲-p-○, and ▲-q-○ in $T + pq$ where a query consists of the position of a shortcut pq.*

Proof. For two points u, v along the backbone $\mathcal{B}$ of T, let $\mathcal{B}(u, v)$ be the path from u to v along $\mathcal{B}$. We refer to $\mathcal{B}(u, v)$ as the *range from u to v*. We build three data structures for range maximum queries [17] that report the following for a query consisting of $u, v \in \mathcal{B}$.

- $\arg\max\{d_T(x, s_i) \mid r_i \in \mathcal{B}(u, v),\ i = 1, 2, \ldots, k\}$,
- $\arg\max\{d_T(s_i, y) \mid r_i \in \mathcal{B}(u, v),\ i = 1, 2, \ldots, k\}$, and
- $\arg\max\{d_T(r_i, s_i) \mid r_i \in \mathcal{B}(u, v),\ i = 1, 2, \ldots, k\}$.

Constructing these data structures takes $O(n)$ time; each query takes $O(1)$ time assuming the we know the edges containing u and v. Using a binary search, we can find the edges along the backbone containing the points $\bar{p}$ and $\bar{q}$ in $O(\log n)$ time.

x*-*pq*-▲, ▲-*pq*-*y The longest path of type x-pq-▲ has length $d_T(x, p) + |pq| + d_T(q, s_i)$, where s_i is a farthest leaf from q among $s_1, s_2, \ldots, s_k$ such that pq is useful for (x, s_i), i.e., such that the root r_i lies on the path from $\bar{p}$ to q. We find the index i using a range maximum query for the range $\mathcal{B}(\bar{p}, q)$, since s_i is the farthest leaf from y among $s_1, s_2, \ldots, s_k$ such that the root r_i lies on the path from $\bar{p}$ to q. Therefore, we can determine the longest path of type x-pq-▲ in $O(\log n)$ time, if we need to locate the edge containing $\bar{p}$. Otherwise, this takes only $O(1)$ time. Symmetrically, we calculate the lengths of the longest paths of type ▲-pq-y.

x*-*T*-▲, ▲-*T*-*y The longest path of type x-T-▲ has length $d_T(x,p) + d_T(p,s_i)$, where s_i is a farthest leaf from p among $s_1, s_2, \ldots, s_k$ such that pq is not useful for (x, s_i), i.e., such that the root r_i lies on the path from p to $\bar{p}$. We find the index i using a range maximum query for the range $\mathcal{B}(p,\bar{p})$, since s_i is the farthest leaf from x among $s_1, s_2, \ldots, s_k$ such that the root r_i lies on the path from p to $\bar{p}$. Therefore, we can determine the longest path of type x-T-▲ in $O(\log n)$ time, if we need to locate the edge containing $\bar{p}$. Otherwise, this takes only $O(1)$ time. Symmetrically, we can calculate the lengths of the longest paths of type ▲-T-y.

x*-*pq*-•, *x*-*T*-•, •-*pq*-*y*, •-*T*-*y The paths x-pq-• and x-T-• have length $d_T(x,p) + (|pq| + d_T(p,q))/2$ and we can compute this value in constant time. Symmetrically, we can calculate the lengths of •-pq-y and •-T-y in constant time.

▲-○, ▲-• The longest path of type ▲-p-○, ▲-q-○, ▲-pq-•, or ▲-T-• has length $d_T(s_i,r_i) + (|pq| + d_T(p,q))/2$, where S_i is a tree with the maximal distance from its root to any of its leaves among the $\mathcal{B}$-sub-trees that have their root along the path from p to q. We find the index i using a range maximum query for the range $\mathcal{B}(p,q)$, since $d_T(r_i,s_i) \geq d_T(r_j,s_j)$ for all $j = 1,2,\ldots,k$ with $r_j \in \mathcal{B}(p,q)$. Thus, it takes $O(1)$ time to determine the longest path of type ▲-p-○, ▲-q-○, ▲-pq-•, or ▲-T-•.

In summary, we can locate the longest paths of all stated types in $O(\log n)$ time. □

Throughout the simulation of the continuous algorithm, the shortcut jumps from one event where the shape of the movement of the shortcut changes—or might change—to the next. In between two events we describe the position of the shortcut as a function of time, where the endpoints of the shortcut move at speeds depending on the path state. Our strategy to locate the next event is as follows. Suppose the last event was reached at time τ_1. We make the tentative assumption that the path state remains unchanged until the next vertex event, i.e., until either p or q meets a vertex. Based on this assumption, we predict the movement of the shortcut and determine the time τ_2 at which the next vertex event would occur. We consider the function f_A that maps $\tau \in [\tau_1, \tau_2]$ to the projected continuous diameter at time τ, i.e., the lengths of the paths that we assumed to remain diametral until τ_2. In addition, we consider the function f_B that maps $\tau \in [\tau_1, \tau_2]$ to the length of the longest path at time τ of any type that is assumed not to be diametral until τ_2. If the graphs of the functions f_A and f_B have no intersection in the interval $[\tau_1, \tau_2]$, then our tentative assumption was justified and the discrete algorithm continues at τ_2 in the same path state. Otherwise, the tentative assumption was wrong and the first intersection of f_A and f_B in the interval $[\tau_1, \tau_2]$ marks the true next event.

To apply the above, we maintain the longest path of each type that is listed in Lemma 13—the types that are not listed will be handled separately. We detect when the longest path of each stated type changes between vertex events using the following.

Corollary 14 *Let T be a geometric tree with straight-line edges and n vertices. There is a data structure with construction time $O(n)$ that supports $O(\log n)$-time queries for the next change in the longest paths of types x-pq-▲, x-T-▲, x-pq-•, x-T-•, ▲-pq-y, ▲-T-y, •-pq-y, •-T-y, ▲-p-○, and ▲-q-○ in $T + pq$ between subsequent vertex events.*

Proof. We tentatively assume that the path state remains unchanged until the longest path of some type changes. During the simulation this assumption may turn out to be wrong when another event occurs before the change in the longest path of this type.

x-pq-▲, ▲-pq-y The longest path of type x-pq-▲ changes when the answer to the corresponding range maximum query from Lemma 13 changes, i.e., when the farthest leaf s_i from y among $s_1, s_2, \ldots, s_k$ such that the root r_i lies on the path from $\bar{p}$ to q changes. Since q remains on its edges, the answer to this query can only change when $\bar{p}$ moves through a root of a secondary $\mathcal{B}$-sub-tree. We perform a query for the farthest leaf s_j from y with r_j in the range $\mathcal{B}(r_{i+1}, q)$, and we perform a binary search for the largest index $l = 1, 2, \ldots, i-1$ such that r_l lies on the path from p to r_i and i is not the answer to the range query for the range $\mathcal{B}(r_l, r_i)$. The longest path of type x-pq-▲ changes when $\bar{p}$ passes through r_j or r_l, i.e., when $d_T(p, r_j) = |pq| + d_T(q, r_j)$ or $d_T(p, r_l) = |pq| + d_T(q, r_l)$, respectively. Therefore, we can detect the next change in the longest path of type x-pq-▲ in $O(\log n)$ time. Symmetrically, we proceed for the longest path of type ▲-pq-y.

x-T-▲, ▲-T-y This is analogous to the previous case.

x-pq-•, x-T-•, •-pq-y, •-T-y There is only one path of each of these types.

▲-○, ▲-• The longest path of type ▲-p-○, ▲-q-○, ▲-pq-•, and ▲-T-• may only change when p or q enter a new edge. Hence, we detect these changes at vertex events.

In summary, we can locate the next change in the longest paths of all stated types between one vertex event and the next in $O(\log n)$ time by using Lemma 13. □

At certain events throughout the discrete algorithm, we need to decide whether the shortcut pq is useful, indifferent, or useless for some pair (x, s_i) with $i = 1, 2, \ldots, k$. The relative position of $\bar{p}$ to r_i determines the usefulness of pq for (x, s_i): If $\bar{p}$ lies on the path from p to r_i with $\bar{p} \neq r_i$, then pq is useful for (x, s_i). If $\bar{p}$ lies at r_i, then pq is indifferent for (x, s_i). If $\bar{p}$ lies on the path from r_i to q with $r_i \neq \bar{p}$, then pq is useless for (x, s_i). In order to decide which of these cases applies, we compare the distances $d_T(p, r_i) = d_T(x, s_i) - d_T(x, p) - d_T(r_i, s_i)$ and $d_T(p, \bar{p}) = |C(p, q)|/2$. Therefore, we can decide whether pq is useful, indifferent, or useless for any pair (x, s_i) and predict in constant time where this is going to change. Likewise, we are able to predict when pq becomes useful, indifferent, or useless for any pair (s_j, y) with $j = 1, 2, \ldots, k$.

6.2. Simulating Phase I

In Phase I of the continuous algorithm, we move p and q with unit speed towards a and b, respectively. In the discrete algorithm, we process a vertex event whenever p or q would meet a vertex during the continuous movement. We locate the next vertex event by comparing the distance from p and from q to the next vertex along their respective paths towards a and towards b. Phase I begins in path state $\{x\text{-}pq\text{-}y, x\text{-}T\text{-}y\}$ with $pq = cc$; the diametral paths of type x-T-y disappear when the shortcut becomes useful for T.

During Phase I, the continuous diameter decreases or remains constant. Therefore, it is sufficient for the discrete algorithm to determine where Phase I of the continuous algorithm ends. At the end of Phase I, we have either reached the end of the backbone,

i.e., $pq = ab$, or a diametral pair of type x-■, ■-y, or ■-○ has appeared alongside the diametral pairs of type x-y. We may ignore diametral pairs of type ■-■ as they appear together with x-■ and ■-y when x-y is diametral, as shown in Theorem 12.

We detect changes in the path state by monitoring the candidates for each type of diametral path—except for those connecting diametral pairs of type ■-■—as follows.

x*-*pq*-*y The path x-pq-y has length $d_{T+pq}(x, y) = d_T(x, p) + |pq| + d_T(q, y)$. Between two subsequent events, p and q remain on their respective containing edge. If the edges of T are straight-line segments, then we can express the positions of p and q as an algebraic function of time with constant degree. Therefore, we can also express $d_{T+pq}(x, y)$ as an algebraic function of constant degree. During Phase I and II, x-pq-y is diametral and, thus, $\operatorname{diam}(T + pq) = d_{T+pq}(x, y)$.

x*-*pq*-▲, ▲-*pq*-*y If the path x-pq-s_i, for $i = 1, 2, \ldots, k$ becomes a diametral path of type x-pq-▲, then pq is useful or indifferent for (x, s_i), i.e., the leaf s_i belongs to a secondary $\mathcal{B}$-sub-tree S_i that is attached to the path from $\bar{p}$ to b in T. If x-pq-s_i becomes diametral in Phase I or II, then x-pq-y is also diametral and S_i is attached to the path from q to $\bar{p}$, as otherwise y would be farther from x than s_i in $T + pq$. Therefore, for any $i = 1, 2, \ldots, k$, the paths x-pq-y and x-pq-s_i are diametral in $T + pq$ if and only if pq is useful or indifferent for (x, s_i) and $d_T(q, y) = d_T(q, s_i)$, i.e., q lies at the precomputed midpoint q_i of the path from s_i to y in T.

When q reaches the vertex q_i, for $i = 1, 2, \ldots, k$, during Phase I we test whether pq is useful or indifferent for (x, s_i), which takes constant time. If pq is useful or indifferent for (x, s_i), then x-pq-s_i becomes diametral and Phase I ends. Otherwise, pq is useless for (x, s_i) and x-pq-s_i cannot become diametral during Phase I. Symmetrically, we proceed when p reaches the vertex p_j for $j = 1, 2, \ldots, k$.

Since neither the endpoints of the shortcut changes direction during Phase I, we encounter each point among $q_1, q_2, \ldots, q_k$ and $p_1, p_2, \ldots, p_k$ at most once. This leads to $O(k)$ events at vertices that we can process in constant time each.

x*-*T*-▲, ▲-*T*-*y The length of the path x-T-s_i for some $i = 1, 2, \ldots k$ does not change with pq; what does change is whether x-T-s_i is a *shortest* path in $T + pq$ or not.

During preprocessing, we obtained the lengths $d_T(x, s_1)$, $d_T(x, s_2)$, $\ldots$, $d_T(x, s_k)$ and sorted them in decreasing order. When $\operatorname{diam}(T + pq)$ decreases to $d_T(x, s_i)$, then we check whether pq is useful for (x, s_i) at that moment. If pq is useful for (x, s_i), then x-T-s_i does not become diametral during Phase I, since x-pq-s_i would have to become diametral first. If pq is useless or indifferent for (x, s_i), then x-T-s_i becomes diametral and Phase I ends. Symmetrically, we detect when a diametral path of type ▲-T-y emerges by introducing events when $\operatorname{diam}(T + pq)$ decreases to any of the presorted values $d_T(s_1, y), d_T(s_2, y), \ldots, d_T(s_k, y)$.

Since the continuous diameter only decreases during Phase I, this leads to $O(k)$ events and we perform one constant-time usefulness check for each event.

***x*-*pq*-•, •-*pq*-*y*, *x*-*T*-•, •-*T*-*y*, ▲-*p*-○, ▲-*q*-○, ▲-*pq*-•, ▲-*T*-•** We turn to diametral paths in $T + pq$ that connect a leaf l of T with the farthest point from l along $C(p, q)$. Any candidate for a diametral path of this kind has length $h + \frac{1}{2}(d_T(p, q) + |pq|)$, where h is the height of a tallest sub-tree attached to the cycle $C(p, q)$ in $T + pq$.

At the beginning of Phase I, the sub-trees containing x and y are the tallest sub-trees attached to $C(p,q)$, i.e., $h = d_T(x,p) = d_T(q,y)$. As Phase I progresses, these sub-trees shrink as p moves to a and q moves to b. For any secondary $\mathcal{B}$-sub-tree S_i attached to the path from a to p, we have $d_T(r_i,s_i) \leq d_T(p,s_i) < d_T(p,x)$, and for every secondary $\mathcal{B}$-sub-tree S_j attached to the path from q to b, we have $d_T(r_j,s_j) \leq d_T(q,s_j) < d_T(q,y)$. Thus, a secondary $\mathcal{B}$-sub-tree S_i may only become a tallest sub-tree if its root r_i lies on the path from p to q and $h = \max\{d_T(x,p), d_T(r_1,s_1), d_T(r_2,s_2), \ldots, d_T(r_k,s_k), d_T(q,y)\}$.

We have computed $\hat{h} = \max\{d_T(r_1,s_1), d_T(r_2,s_2), \ldots, d_T(r_k,s_k)\}$ as part of our preprocessing. We detect diametral paths of type x-pq-•, •-pq-y, x-T-•, •-T-y, ▲-p-○, ▲-q-○, ▲-pq-•, and ▲-T-• as follows. While we are in Phase I and II, we have $\operatorname{diam}(T + pq) = d_{T+pq}(x,y)$. We consider two subsequent vertex events at the times τ_1 and τ_2. We tentatively assume that no path state change occurs between τ_1 and τ_2, i.e., for $\tau \in [\tau_1,\tau_2]$, the point $p(\tau)$ moves at unit speed towards a and $q(\tau)$ moves with unit speed towards b. For this scenario, we compare the function $\tau \mapsto d_{T+p(\tau)q(\tau)}(x,y)$ that describes the lengths of the supposedly diametral paths x-pq-y in $T + p(\tau)q(\tau)$ for $\tau \in [\tau_1,\tau_2]$ with the function

$$\tau \mapsto \max\{d_T(x,p(\tau)), \hat{h}, d_T(q(\tau),y)\} + \frac{d_T(p(\tau),q(\tau)) + |p(\tau)q(\tau)|}{2}$$

that describes the farthest distance from $s \in C(p(\tau),q(\tau))$ to any leaf $l \in T$ for $\tau \in [\tau_1,\tau_2]$. If these two functions intersect within $[\tau_1,\tau_2]$, then our tentative assumption was wrong and the first intersection of the functions marks the position of the shortcut where a diametral path of type x-pq-•, •-pq-y, x-T-•, •-T-y, ▲-p-○, ▲-q-○, ▲-pq-•, or ▲-T-• emerges and Phase I ends. If there is no such intersection, then the tentative assumption was correct and Phase I continues.

During Phase I, we encounter each vertex at most once at a vertex event. Hence, there are $O(n)$ time intervals between subsequent vertex events that we check for path state changes involving diametral paths of type x-pq-•, •-pq-y, x-T-•, •-T-y, ▲-p-○, ▲-q-○, ▲-pq-•, and ▲-T-•. Since the functions involved in this check are algebraic functions of sufficiently small degree, the intersection test in each interval takes constant time that we count as part of processing a vertex event.

In summary, to simulate Phase I, we process $O(n)$ vertex events and $O(k)$ other events related to paths of type x-T-▲ and ▲-T-y. Thanks to our preparations, we can process each event in $O(1)$ time. Therefore, simulating Phase I takes $O(n)$ time.

6.3. Simulating Phase II

We describe the discretization of Phase II for a shift towards x, where the continuous algorithm balances the diametral path x-pq-y and a diametral path with endpoints of type x-■ or ■-○. Suppose p moves with unit speed towards a. The following lemma specifies the speed at which q should move towards c to balance the diametral paths.

Lemma 15 *Let pq and $p'q'$ be two shortcuts for a geometric tree T where we reach $p'q'$ from pq with an x-shift. If $T + pq$ and $T + p'q'$ are in the same path state, then we can express the distance from q to q' and the change in diameter as stated in Table 1.*

Path State	$d_T(q,q')$	$\operatorname{diam}(T+pq) - \operatorname{diam}(T+p'q')$
$\{x\text{-}pq\text{-}y, x\text{-}pq\text{-}\blacktriangle\}$	0	$d_T(p',p) + \lvert pq\rvert - \lvert p'q'\rvert$
$\{x\text{-}pq\text{-}y, x\text{-}pq\text{-}\bullet, x\text{-}T\text{-}\bullet\}$	$\frac{1}{3}\left(d_T(p,p') + \lvert pq\rvert - \lvert p'q'\rvert\right)$	$\frac{2}{3}\left(d_T(p,p') + \lvert pq\rvert - \lvert p'q'\rvert\right)$
$\{x\text{-}pq\text{-}y, x\text{-}T\text{-}\blacktriangle\}$	$d_T(p,p') + \lvert pq\rvert - \lvert p'q'\rvert$	0
$\{x\text{-}pq\text{-}y, \blacktriangle\text{-}p\text{-}\circ, \blacktriangle\text{-}q\text{-}\circ\}$	$d_T(p,p') + \frac{1}{3}\left(\lvert pq\rvert - \lvert p'q'\rvert\right)$	$\frac{2}{3}\left(\lvert pq\rvert - \lvert p'q'\rvert\right)$

Table 1: The distance between q and q' and the change in diameter when shifting the shortcut towards x from pq to $p'q'$ while maintaining the diametral paths in balance.

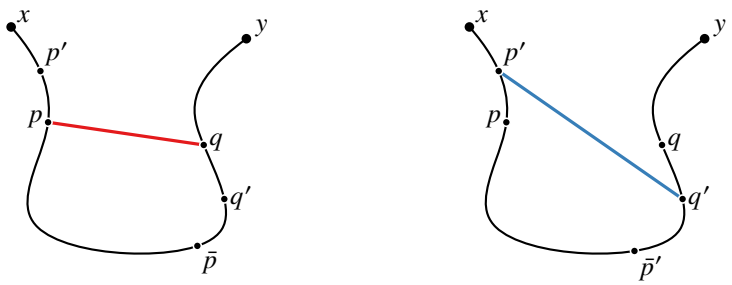


Figure 24: The situation at the start (left) and at the end (right) of an x-shift from pq to $p'q'$.

Proof. We show the result for the path state $\{x\text{-}pq\text{-}y, x\text{-}pq\text{-}\bullet, x\text{-}T\text{-}\bullet\}$. During an x-shift from pq to $p'q'$, as in Figure 24, the length of the paths of type $x\text{-}pq\text{-}y$ changes by

$$\begin{aligned} d_{T+pq}(x,y) - d_{T+p'q'}(x,y) &= \left[d_T(x,p') + d_T(p',p) + \lvert pq\rvert + d_T(q,y)\right] \\ &\quad - \left[d_T(x,p') + \lvert p'q'\rvert + d_T(q',q) + d_T(q,y)\right] \\ &= d_T(p',p) + \lvert pq\rvert - \lvert p'q'\rvert - d_T(q',q) \ . \end{aligned}$$

Let $\bar{p}$ and $\bar{p}'$ be the farthest points from p along $C(p,q)$ and along $C(p',q')$, respectively. Each diametral path of type $x\text{-}pq\text{-}\bullet$ and $x\text{-}T\text{-}\bullet$ connects x with $\bar{p}$ in $T+pq$. Since p and $\bar{p}$ are antipodal along $C(p,q)$, we have $d_{T+pq}(p,\bar{p}) = \lvert C(p,q)\rvert/2 = \left(\lvert pq\rvert + d_T(p,q)\right)/2$. Therefore, the length of the paths of type $x\text{-}pq\text{-}\bullet$ and $x\text{-}T\text{-}\bullet$ in $T+pq$ is

$$d_{T+pq}(x,\bar{p}) = d_T(x,p) + d_{T+pq}(p,\bar{p}) = d_T(x,p) + \frac{\lvert pq\rvert + d_T(p,q)}{2} \ ,$$

Likewise, the lengths of the paths of type $x\text{-}pq\text{-}\bullet$ and $x\text{-}T\text{-}\bullet$ in $T+p'q'$ is $d_{T+p'q'}(x,\bar{p}') = d_T(x,p') + \left(\lvert p'q'\rvert + d_T(p',q')\right)/2$. During an x-shift from pq to $p'q'$, the length of the paths of type $x\text{-}pq\text{-}\bullet$ and $x\text{-}T\text{-}\bullet$ changes by

$$\begin{aligned} d_{T+pq}(x,\bar{p}) - d_{T+p'q'}(x,\bar{p}') &= \left[d_T(x,p') + d_T(p',p) + \frac{\lvert pq\rvert + d_T(q,q') + d_T(q',p)}{2}\right] \\ &\quad - \left[d_T(x,p') + \frac{\lvert p'q'\rvert + d_T(q',p) + d_T(p,p')}{2}\right] \\ &= \frac{d_T(p,p') + \lvert pq\rvert - \lvert p'q'\rvert + d_T(q,q')}{2} \ . \end{aligned}$$

Since x-pq-y, x-pq-$\bullet$, and x-T-$\bullet$ remain diametral during the x-shift from pq to $p'q'$, their lengths change by the same amount, i.e., $d_{T+pq}(x,y) - d_{T+p'q'}(x,y) = d_{T+pq}(x,\bar{p}) - d_{T+p'q'}(x,\bar{p}')$ and, thus, $d_T(q,q') = \frac{1}{3}\left(d_T(p,p') + |pq| - |p'q'|\right)$, since

$$\begin{aligned}
3d_T(q,q') &= 3d_T(q,q') + 2 \cdot 0 \\
&= 3d_T(q,q') + 2\left[d_{T+pq}(x,y) - d_{T+p'q'}(x,y)\right] \\
&\qquad - 2\left[d_{T+pq}(x,\bar{p}) - d_{T+p'q'}(x,\bar{p}')\right] \\
&= 3d_T(q,q') + [2d_T(p',p) + 2|pq| - 2|p'q'| - 2d_T(q',q)] \\
&\qquad - [d_T(p,p') + |pq| - |p'q'| + d_T(q,q')] \\
&= d_T(p',p) + |pq| - |p'q'| \quad .
\end{aligned}$$

In this case, the diameter changes by $\frac{2}{3}\left(d_T(p,p') + |pq| - |p'q'|\right)$, since

$$\begin{aligned}
\operatorname{diam}(T + pq) - \operatorname{diam}(T + p'q') &= d_{T+pq}(x,y) - d_{T+p'q'}(x,y) \\
&= d_T(p',p) + |pq| - |p'q'| - d_T(q',q) \\
&= d_T(p',p) + |pq| - |p'q'| - \frac{d_T(p,p') + |pq| - |p'q'|}{3} \\
&= \frac{2}{3}\left[d_T(p,p') + |pq| - |p'q'|\right] \quad .
\end{aligned}$$

Analogously, we establish the results for the other path states listed in Table 1. □

In Table 1, $d_T(q,q')$ is expressed in terms of $d_T(p,p')$, $|pq|$, and $|p'q'|$. For straight-line edges, we can remove the dependency on $|p'q'|$ by expressing $|p'q'|$ as a function of $d_T(p,p')$ and $d_T(q,q')$ and solving for $d_T(q,q')$. We obtain $d_T(q,q')$ in terms of $d_T(p,p')$, $|pq|$ and certain geometric constants that do not depend on the position of q', such as the angle at which the lines through the edges containing p and q intersect.

When p and q traverse a fixed pair of straight-line edges while shifting towards x, Lemma 15 allows us to compute the next vertex event and how the diameter changes between subsequent vertex events—assuming that the path-state does not change. While simulating Phase II, we encounter $O(n)$ events where the shortcut meets a vertex or where the shortcut begins to shrink or grow, due to the following. The shortcut enters each edge at most once, since p and q never change direction (we have $d_T(p,p') > 0$ and $d_T(q,q') \geq 0$ from Lemma 15). Thus, we encounter $O(n)$ pairs of edges. Moreover, the shortcut changes a constant number of times between growing and shrinking when both endpoints move along a fixed pair of straight-line edges for each of the trajectories prescribed by the constraints in Table 1. This trajectory yields the length of the currently diametral paths and the lengths of the candidates for diametral paths as functions of $d_T(p,p')$, assuming that the path-state does not change. The first position for p' where the current diameter meets the length of any candidate path marks the next path-state change event, if this occurs before the next vertex event. Due to the preparations in Section 6.1 and the following, there are only a constant number of candidates for diametral paths that we need to take into account at each event of the discretized algorithm.

Lemma 16 *For a geometric tree with n vertices and straight-line edges, the path state changes $O(n)$ times during Phase II of the continuous algorithm.*

Proof. In Phase II, the speed of p and q is determined by two or three different types of diametral paths, i.e., by x-pq-y and x-pq-▲, or by x-pq-y and x-pq-• and x-T-•, or x-pq-y and x-T-▲, or by x-pq-y and ▲-p-○ and ▲-q-○. For any path state $\mathfrak{S}$ during Phase II, only the subset $\mathfrak{S}'$ of $\mathfrak{S}$ that leads to the least increase in the continuous diameter determines the speed of p and q. The other types of paths cease to be diametral instantaneously, i.e., we transit from $\mathfrak{S}$ to $\mathfrak{S}'$. Some transitions between the path states in Phase II are impossible. For instance, suppose we perform an x-shift from pq to a new position $p'q'$ such that the path state remains $\{x$-pq-y, ▲-p-○, ▲-q-○$\}$ until we reach $p'q'$ and where the path state changes upon reaching $p'q'$. Then $T + p'q'$ cannot be in path state $\{x$-pq-y, ▲-p-○, ▲-q-○, x-pq-•, x-T-•$\}$, due to the following. From Table 1 we know $d_T(q,q') = d_T(p,p') + \frac{1}{3}(|pq| - |p'q'|)$. As pq moves to $p'q'$, the length of the paths x-pq-• and x-T-• changes by $d_T(p',p) + \frac{2}{3}(|pq| - |p'q'|)$, since

$$\begin{aligned} d_{T+pq}(x,\bar{p}) - d_{T+p'q'}(x,\bar{p}') &= (d_T(p',p) + |pq| - |p'q'| + d_T(q,q'))/2 \\ &= \frac{d_T(p',p) + |pq| - |p'q'| + d_T(p,p') + \frac{1}{3}(|pq| - |p'q'|)}{2} \\ &= d_T(p',p) + 2(|pq| - |p'q'|)/3 \ . \end{aligned}$$

Since $d_T(p,p') > 0$, the paths of type x-pq-• and x-T-• shrink at a faster rate than the diametral paths of type x-pq-y, ▲-p-○, and ▲-q-○. Therefore, x-pq-• and x-T-• cannot become diametral when the shortcut reaches $p'q'$. Figure 25 illustrates the transitions between the path states that may occur during Phase II.

We bound the number of visits to the path state $\{x$-pq-y, x-pq-▲$\}$ by k, where $k = O(n)$ is the number of secondary $\mathcal{B}$-sub-trees of T. When we are in path state $\{x$-pq-y, x-pq-$s_j\}$ for some $j = 1, 2, \ldots, k$, then q lies midway along the path from y to s_j, i.e., $d_T(q,y) = d_T(q,s_j)$. When we transit from $\{x$-pq-y, x-pq-$s_j\}$ to any other path state, q will begin to move with non-zero speed towards a. Hence, x, s_j ceases to be diametral and cannot become diametral again during Phase II. Therefore, we take at most k green transitions in Figure 25.

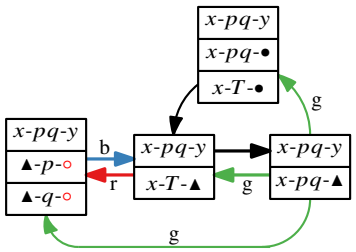


Figure 25: The transitions between the path states during Phase II. Transitory states, such as $\{x$-pq-y, x-pq-▲, x-pq-•, x-T-•$\}$ have been omitted. The red transition (r) is possible when the shortcut grows; the blue transition (b) is possible when the shortcut shrinks. When we take any green transition (g) from $\{x$-pq-y, x-pq-$s_j\}$ for $j = 1, 2, \ldots, k$ then x, s_j ceases to be diametral and cannot become diametral again.

We bound the number of visits to the pair state $\{x$-pq-y, ▲-p-○, ▲-q-○$\}$ by $O(n)$. Once the green transitions are exhausted, we can only enter $\{x$-pq-y, ▲-p-○, ▲-q-○$\}$ with the red transition from $\{x$-pq-y, x-T-▲$\}$. This transition is only possible when the shortcut is growing. Therefore, we pay $O(n)$ visits to $\{x$-pq-y, ▲-p-○, ▲-q-○$\}$, since the shortcut switches at most $2n$ times between shrinking and growing during an x-shift.

After $O(n)$ visits to the path states $\{x$-pq-y, x-pq-▲$\}$ and $\{x$-pq-y, ▲-p-○, ▲-q-○$\}$, we can no longer take any red or green transitions in Figure 25. We encounter $O(n)$ path state changes during Phase II, as the remaining transitions form an acyclic digraph. □

Therefore, we can simulate Phase II in $O(n \log n)$ time, since Phase II consists of $O(n)$ events that we can process in $O(\log n)$ time per event using Corollary 14.

6.4. Simulating Phase III

In Phase III, the continuous algorithm balances a diametral path with endpoints of type x-■ and a diametral path with endpoints of type ■-y. This leads to the following.

Lemma 17 *Let pq and $p'q'$ be two shortcuts for a geometric tree T where we reach $p'q'$ from pq with an out-shift. If $T + pq$ and $T + p'q'$ are in the same path state, then we can express the distance of q and q' and the change in diameter as stated in Table 2.*

x-■	■-y	$d_T(q, q')$	$\operatorname{diam}(T + pq) - \operatorname{diam}(T + p'q')$
x-pq-▲	▲-pq-y	$d_T(p, p')$	$\lvert pq\rvert - \lvert p'q'\rvert$
x-pq-▲	•-y	$d_T(p, p') + \frac{1}{3}(\lvert pq\rvert - \lvert p'q'\rvert)$	$\frac{2}{3}(\lvert pq\rvert - \lvert p'q'\rvert)$
x-pq-▲	▲-T-y	$d_T(p, p') + \lvert pq\rvert - \lvert p'q'\rvert$	0
x-•	▲-pq-y	$d_T(p, p') - \frac{1}{3}(\lvert pq\rvert - \lvert p'q'\rvert)$	$\frac{2}{3}(\lvert pq\rvert - \lvert p'q'\rvert)$
x-•	•-y	$d_T(p, p')$	$\frac{1}{2}(\lvert pq\rvert - \lvert p'q'\rvert)$
x-•	▲-T-y	$d_T(p, p') + \lvert pq\rvert - \lvert p'q'\rvert$	0
x-T-▲	▲-pq-y	$d_T(p, p') - (\lvert pq\rvert - \lvert p'q'\rvert)$	0
x-T-▲	•-y	$d_T(p, p') - (\lvert pq\rvert - \lvert p'q'\rvert)$	0
x-T-▲	▲-T-y	$d_T(p, p')$	0

Table 2: The distance between q and q' with the change in diameter during an out-shift from pq to $p'q'$ while balancing the diametral paths. Here, x-• stands for x-pq-• and x-T-•; and •-y stands for •-pq-y and •-T-y.

Proof. We establish these results using the same approach as in Lemma 15. As pq moves to $p'q'$, as in Figure 26, the potential diametral paths change as follows. For $r \in \{p, q\}$, $\bar{r}$ and $\bar{r}'$ are the farthest point from r along $C(p, q)$ and $C(p', q')$, respectively.

$$
\begin{aligned}
x\text{-}pq\text{-}▲: &\quad d_{T+pq}(x, s_i) - d_{T+p'q'}(x, s_i) = d_T(p', p) + |pq| - |p'q'| - d_T(q', q) \\
x\text{-}pq\text{-}•, x\text{-}T\text{-}•: &\quad d_{T+pq}(x, \bar{x}) - d_{T+p'q'}(x, \bar{x}') = \frac{d_T(p', p) + |pq| - |p'q'| - d_T(q', q)}{2} \\
x\text{-}T\text{-}▲: &\quad d_{T+pq}(x, s_j) - d_{T+p'q'}(x, s_j) = 0 \\
▲\text{-}pq\text{-}y: &\quad d_{T+pq}(s_l, y) - d_{T+p'q'}(s_l, y) = d_T(q', q) + |pq| - |p'q'| - d_T(p', p) \\
•\text{-}pq\text{-}y, •\text{-}T\text{-}y: &\quad d_{T+pq}(y, \bar{y}) - d_{T+p'q'}(y, \bar{y}') = \frac{d_T(q', q) + |pq| - |p'q'| - d_T(p', p)}{2} \\
▲\text{-}T\text{-}y: &\quad d_{T+pq}(s_o, y) - d_{T+p'q'}(s_o, y) = 0
\end{aligned}
$$

These changes equate for paths that remain diametral during the outwards shift. For instance, if the paths x-pq-•, x-T-•, and ▲-pq-y remain diametral as pq moves to $p'q'$, then $d_{T+pq}(x, \bar{x}) - d_{T+p'q'}(x, \bar{x}') = \operatorname{diam}(T + pq) - \operatorname{diam}(T + p'q') = d_{T+pq}(s_l, y) - d_{T+p'q'}(s_l, y)$ and, thus, $d_T(p, p') = d_T(q, q') + \frac{1}{3}(|pq| - |p'q'|)$, since

$$3d_T(p, p') = 3d_T(p, p') + 2 \cdot 0$$

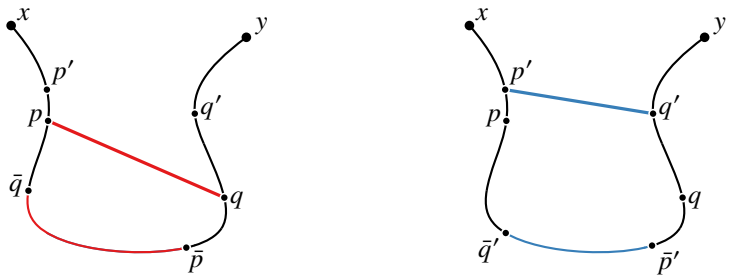


Figure 26: The situation at the start (left) and at the end (right) of an out-shift from pq to $p'q'$.

$$\begin{aligned}
&= 3d_T(p,p') + 2\left[d_{T+pq}(s_l,y) - d_{T+p'q'}(s_l,y)\right] \\
&\qquad - 2\left[d_{T+pq}(x,\bar{x}) - d_{T+p'q'}(x,\bar{x}')\right] \\
&= 3d_T(p,p') + 2\left[d_T(q',q) + |pq| - |p'q'| - d_T(p',p)\right] \\
&\qquad - \left[d_T(p',p) + |pq| - |p'q'| - d_T(q',q)\right] \\
&= 3d_T(q,q') + |pq| - |p'q'| \quad .
\end{aligned}$$

In this case, the diameter changes by $\frac{2}{3}(|pq| - |p'q'|)$, since

$$\begin{aligned}
\operatorname{diam}(T + pq) - \operatorname{diam}(T + p'q') &= d_{T+pq}(s_l,y) - d_{T+p'q'}(s_l,y) \\
&= d_T(q',q) + |pq| - |p'q'| - d_T(p',p) \\
&= d_T(q',q) + |pq| - |p'q'| - d_T(q,q') - \frac{|pq| - |p'q'|}{3} \\
&= \frac{2}{3}(|pq| - |p'q'|) \quad .
\end{aligned}$$

In the same manner, we express $d_T(q,q')$ and the change in diameter in terms of $d_T(p,p')$ and $|pq| - |p'q'|$ for the remaining path states listed in Table 2. □

Similar to Phase I and II, there are $O(n)$ events where the shortcut meets a vertex or starts to shrink or grow. Even though we can rule out certain transitions between path states in Phase III, the path state might change $\Omega(n^2)$ times.

Lemma 18 *For every $l \in \mathbb{N}$, there exists a geometric tree T_l with $n = 8l + 5$ vertices where at least $3l^2 - 2l = \Omega(n^2)$ path state changes occur during Phase III.*

Proof (Sketch). We construct T_l with the following objectives in mind.

1. The geometric tree T_l consists of a backbone path $\mathcal{B}$ from x to y with $2l$ pendant edges $r_1s_1, r_2s_2, \ldots, r_{2l}s_{2l}$, where $r_i \in \mathcal{B}$ and $s_i \notin \mathcal{B}$ for each $i = 1, 2, \ldots, 2l$.
2. During Phase III, the shortcut grows and shrinks at least l times.
3. For every stretch where pq only grows or only shrinks, pq changes between being useful and useless for the $2l$ pairs (x, s_{l+1}), (x, s_{l+2}), $\ldots$ (x, s_{2l}) and (s_1, y), (s_2, y), $\ldots$ (s_l, y). These changes occur at l events. At the j-th event, for $j = 1, 2, \ldots, l$, the shortcut switches between being useful and useless for (x, s_{l+j}) and (s_j, y).

4. For $j = 1, 2, \ldots, l$, the pairs (x, s_{l+j}) and (s_j, y) are the only diametral pairs of $T + pq$ when the shortcut switches between being useful and useless for them.

Figure 27 illustrates our construction for $l = 4$. When the shortcut grows, $\bar{q}$ passes through r_l, r_{l-1}, ..., r_1 in this order while $\bar{p}$ passes through r_{l+1}, r_{l+2}, ..., r_{2l} with $\bar{q}$ reaching r_{l+j} at the same time as $\bar{p}$ reaches r_{l-j+1} for $j = 1, 2, \ldots, l$. By choosing sufficiently short lengths for the edges $r_i s_i$ and by spacing them out sufficiently far apart, we ensure that x-pq-s_{l+j} and s_{l-j+1}-pq-y are diametral immediately before $\bar{q}$ reaches r_j and $\bar{p}$ reaches s_{l+j} and that x-T-s_{l+j} and s_{l-j+1}-T-y are diametral shortly after $\bar{q}$ passed r_{l-j+1} and $\bar{p}$ passed r_{l+j}. We ensure that the pairs $(x, \bar{p})$ and $(\bar{q}, y)$ become diametral when $\bar{q}$ and $\bar{p}$ move from r_{l-j+1} to r_{l-j} and from r_{l+j} to r_{l+j+1}, respectively. By balancing the lengths of the stretches where the shortcut only grows with those where the shortcut only shrinks, we ensure that the above happens in reverse when the shortcut shrinks.

Figure 28 illustrates the path state transitions for T_l during Phase III . We count three path states for each of the l moments when $\bar{q}$ and $\bar{p}$ pass through r_j and r_{2l-j+1}, respectively. This results in at least $3l - 2$ path state changes for each time the shortcut is growing or shrinking, since the first and last state in each sequence do not incur a path state change. Since the shortcut switches at least l times between growing and shrinking, we count at least $l \cdot (3l - 2) = 3l^2 - 2l = \Omega(n^2)$ path state changes during Phase III. □

We circumvent this issue by ignoring certain superfluous path state events.

Suppose that the shortcut is growing while the path x-T-s_i, for some $i = 1, 2, \ldots, k$, is a diametral path of type x-T-▲. In this situation a path state change may occur where the path x-pq-$\bar{x}$ or a path x-pq-s_j for some $j > i$ might become a diametral path of type x-pq-● and x-pq-▲, respectively. There is no need to recognize this path state change, since the diameter cannot decrease before the shortcut becomes useful for (x, s_i) again. Until then, we ignore any changes in the diametral path for pairs of type x-■. With this modification, the shortcut might leave the trajectory of the continuous algorithm. This does not compromise optimality, as we uphold the same invariants: the path x-T-s_i certifies that no x-shift leads to a better shortcut, even if it is no longer diametral.

In Phase III, the path state has two components: the diametral path of type x-■ and the diametral path of type ■-y. We modify Phase III as follows. We ignore changes to the x-■-component when x-T-s_i has become diametral for some $i \in \{1, 2, \ldots, k\}$ until pq becomes useful for (x, s_i), and we ignore changes to the ■-y-component when s_j-T-y has become diametral for some $j \in \{1, 2, \ldots, k\}$ until pq becomes useful for (s_j, y). Figure 30 illustrates the modified Phase III for the geometric tree T_l from the construction of the $\Omega(n^2)$ bound on the number of the path state changes from Lemma 18.

Lemma 19 *For a geometric tree with n vertices and straight-line edges, the path state changes $O(n)$ times during the modified Phase III.*

Proof. In Phase III, the speed of p and q is determined by one type of diametral path for a diametral pair of type x-■ and by one type of diametral path for a diametral pair of type ■-y, as indicated in Table 2. Certain path state transitions require the shortcut to shrink, others require it to grow. Suppose, for instance, that the shortcut shrinks as we move from pq to $p'q'$, i.e., $|p'q'| < |pq|$. In this case, we cannot transit from {x-pq-●, x-T-●,●-pq-y, ●-T-y} to {x-pq-▲, x-pq-●, x-T-●, ●-pq-y, ●-T-y} as the shortcut moves from pq to

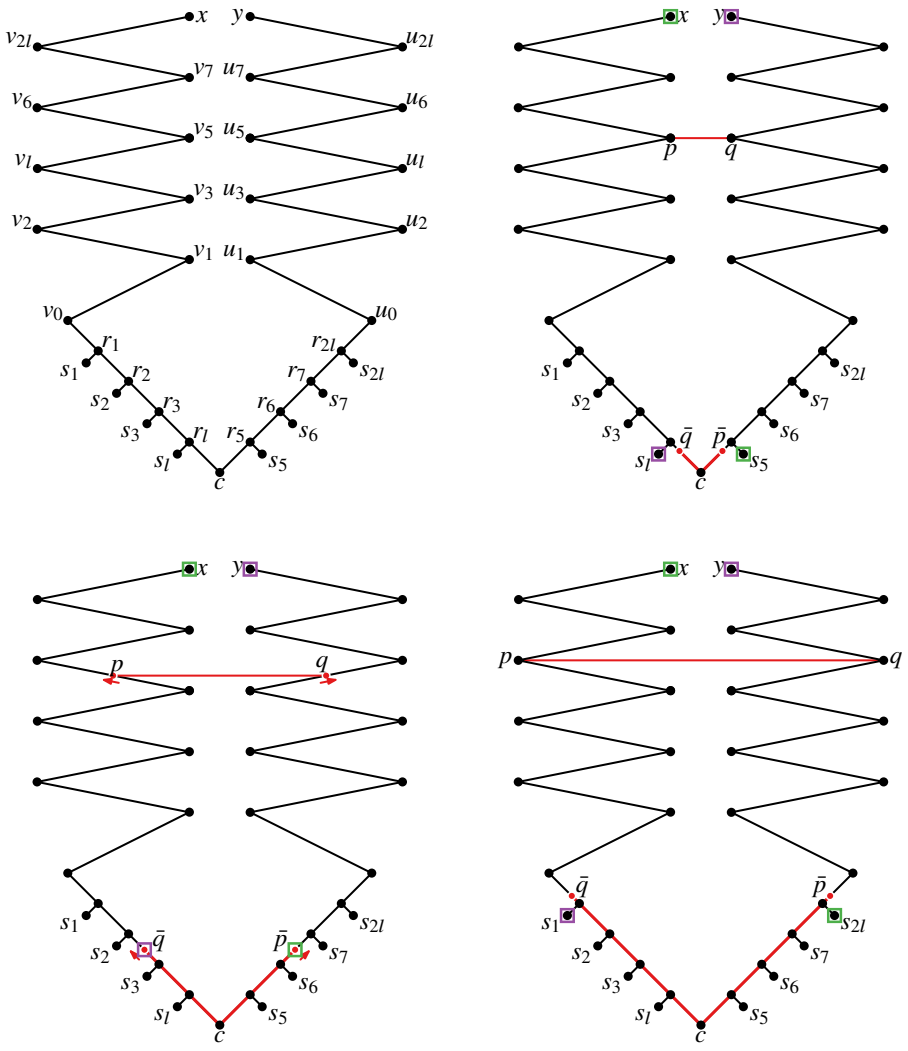


Figure 27: The tree T_l for $l = 4$ with three positions for the shortcut pq where the shortcut grows during an out-shift. The diametral pairs of $T + pq$ are marked with squares of matching color. The sets of diametral paths of $T + pq$ are $\{x\text{-}pq\text{-}s_5, s_4\text{-}pq\text{-}y\}$, $\{x\text{-}pq\text{-}\bar{p}, x\text{-}T\text{-}\bar{p}, \bar{q}\text{-}pq\text{-}y, \bar{q}\text{-}T\text{-}y\}$, and $\{x\text{-}T\text{-}s_8, s_1\text{-}T\text{-}y\}$.

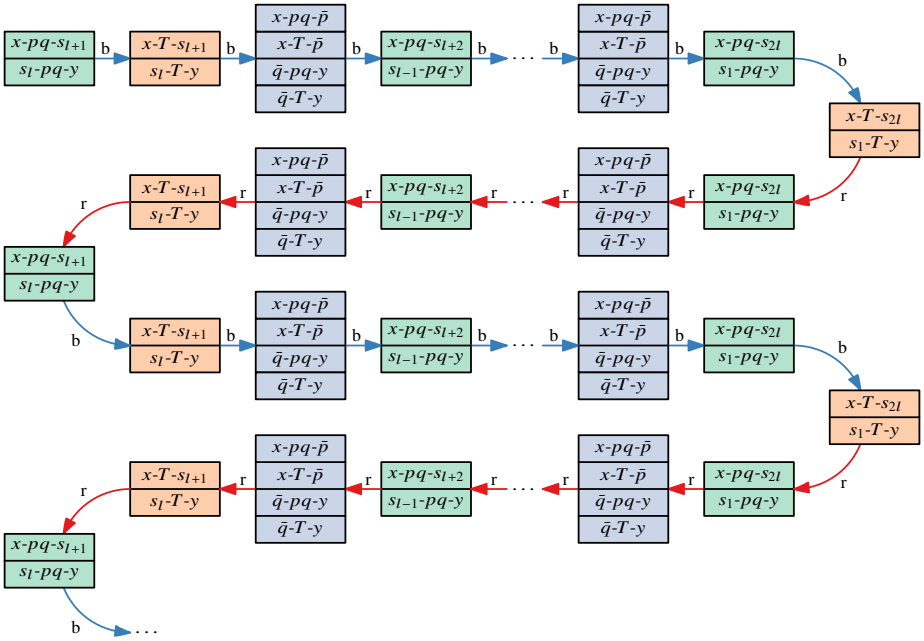


Figure 28: The changes in the set of diametral paths for the tree T_l during Phase III. Blue transitions (b) occur while the shortcut is growing and red transitions (r) occur while the shortcut is shrinking.

$p'q'$, since then $d_T(q,q') = d_T(p,p')$ and, thus, any path of type x-pq-▲ shrinks by $d_{T+pq}(x,s_i) - d_{T+p'q'}(x,s_i) = d_T(p',p) + |pq| - |p'q'| - d_T(q',q) = |pq| - |p'q'|$ whereas the diameter only shrinks by $\frac{1}{2}(|pq| - |p'q'|)$. Figure 31 illustrates the path state transitions during Phase III for the path states that do not contain x-T-▲ or ▲-T-y.

Recall that the $\mathcal{B}$-sub-trees $S_1, S_2, \ldots, S_k$ are numbered in the order along the backbone from a to b. After x-T-s_i has become a diametral path of type x-T-▲ in the modified Phase III, none of the paths x-T-s_j with $i < j$ will be registered as diametral paths. If x-T-s_j does become diametral after x-T-s_i has become diametral, then the shortcut is useless for (x, s_i), since it is useless for (x, s_j). Therefore, we do not register that x-T-s_j becomes diametral, because we are still waiting for the shortcut to become useful for (x, s_i), since $d_T(x,s_i) = d_{T+pq}(x,s_i) \leq \operatorname{diam}(T + pq)$.

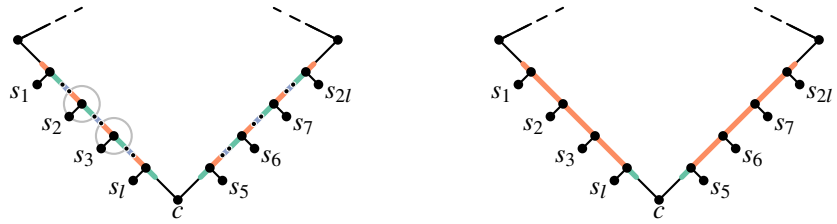


Figure 29: The sections of T_l that are swept by $\bar{q}$ or $\bar{p}$ coloured according to the corresponding path state from Figure 28 for the unmodified Phase III on the left and from Figure 30 for the modified Phase III on the right.

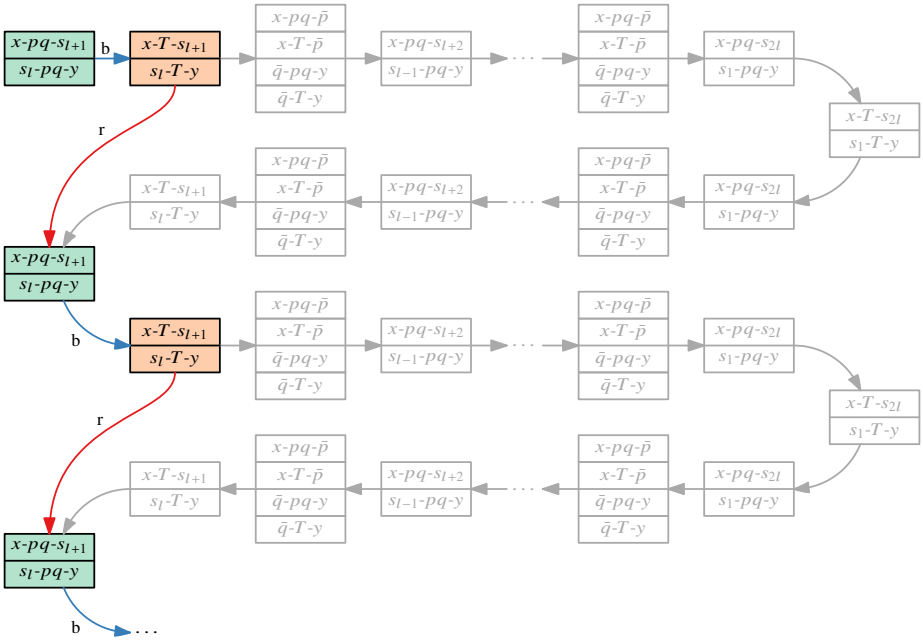


Figure 30: The modified Phase III compared with the original Phase III for the tree T_l with $l = 4$. Gray states and transitions are no longer visited by the modifed Phase III. Blue transitions (b) occur while the shortcut is growing and red transitions (r) occur while the shortcut is shrinking. The number of path state changes decreased to $l - 1$ with one state change for each time the shortcut changes between growing and shrinking.

We argue that the modified Phase III visits the path states containing x-T-▲ at most $k + n$ times. Figure 32 illustrates the possible changes in the x-■-component of the path state during the modified Phase III. When the shortcut grows, we only enter a path state containing x-T-▲ when the path x-T-s_i that was most recently noted a diametral path of type x-T-▲ becomes diametral again. In this case, the x-■-component only changes away from x-T-▲ after the shortcut has began to shrink again. Since the shortcut is growing at most n times during the modified Phase III, we register at most $O(n)$ path state changes of this kind. When the shortcut shrinks, we only enter a path state containing x-T-▲ when a path x-T-s_l becomes diametral that has not been diametral before. As argued above, this may occur at most k times, since $l < i$, where x-T-s_i was the most recent diametral path of type x-T-▲. Likewise, we argue that the modified Phase III visits the path states containing ▲-T-y at most $k + n$ times. Once we have exhausted the at most $2k + 2n$ visits to path states containing x-T-▲ or ▲-T-y, the remaining path state transitions form acyclic digraphs when the shortcut is shrinking and when the shortcut is growing, as depicted in Figure 31. Since the shortcut changes at most $2n$ times between shrinking and growing, we register at most $2k + 4n$ events where the path state changes. Therefore, we process $O(n)$ path state events throughout the modified Phase III. □

We detect path state events during Phase III in the same fashion as in Phase I and II, except for the following two differences. First, the detection of diametral paths of type x-pq-▲ and ▲-pq-y changes, since x-pq-y is no longer diametral. Second, we

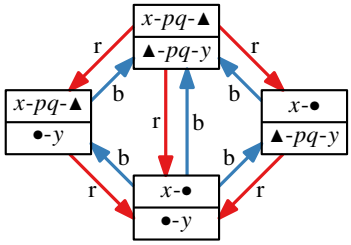


Figure 31: The path states encountered during Phase III, excluding the states containing x-T-▲ or ▲-T-y and any transitions to these states. Red transitions (r) occur while the shortcut is shrinking; blue transitions (b) occur while the shortcut is growing.

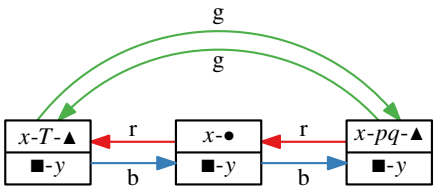


Figure 32: A simplified view on the path states encountered during Phase III. Only the changes in the path type for x-■ are shown. Red transitions (r) may occur while the shortcut is shrinking; blue transitions (b) may occur while the shortcut is growing. Further restrictions apply for green transitions (g).

need to detect when a diametral path that connects a diametral pair of type ■-■ appears, since this marks the end of Phase III. This concerns diametral paths of type ▲-T-▲ and ▲-pq-▲, since we already detect ▲-pq-● and ▲-T-● when monitoring the candidates for diametral paths with an endpoint on the simple cycle $C(p,q)$ in $T + pq$.

x*-*pq*-▲, ▲-*pq*-*y There are two cases in which a path x-pq-s_i, for $i \in \{1,2,\ldots,k\}$ becomes a diametral path of type x-pq-▲ during the modified Phase III.

In the first case, the path x-T-s_i is a diametral path of type x-T-▲ and the shortcut is about to become useful for (x, s_i), which we detect as described in Section 6.1.

In the second case, the shortcut is growing and the path x-pq-$\bar{p}$ is a diametral path of type x-pq-●. The point $\bar{p}$ is moving towards y when the shortcut is growing and s_i is a leaf of a secondary $\mathcal{B}$-sub-tree attached to the path from q to $\bar{p}$. This implies that we have $d_T(\bar{q}, r_i) = d_T(r_i, s_i)$ in this case. We can detect when this type of path state event occurs by comparing the distance $d_T(r_i, s_i)$ with

$$d_T(\bar{q}, r_i) = d_T(\bar{q}, p) - d_T(p, r_i) = (d_T(p,q) - |pq|)/2 - d_T(p, r_i) \quad .$$

▲-***T***-▲ We ignore diametral paths of type ▲-T-▲. Suppose, during the modified Phase III, s_i-T-s_j, for $i, j \in \{1,2,\ldots,k\}$, becomes a diametral path of type ▲-T-▲ for some shortcut position $\hat{p}\hat{q}$. Even if we fail to register this path state event, we still report an optimal shortcut, since we report the shortcut that yields the smallest encountered continuous diameter including the continuous diameter of $T + \hat{p}\hat{q}$.

▲-***pq***-▲ We cannot ignore an event where a diametral path of type ▲-pq-▲ appears, since the length of these paths depends on pq. Moreover, we cannot afford to check whether a diametral path of type ▲-pq-▲ is about to appear when processing the other events. We perform the detection of such events as a post-processing step instead. It is sufficient to find the first position $\hat{p}\hat{q}$ where some path s_i-pq-s_j, for $i, j \in \{1,2,\ldots,k\}$, becomes a diametral path of type ▲-pq-▲: If we shift outwards from $\hat{p}\hat{q}$, then s_i-pq-s_j will remain diametral while increasing in length. We proceed as follows. First, we simulate the modified Phase III without attempting to detect if a diametral path of type ▲-pq-▲ appears. We record the sequence

of edge pairs that we visit during this simulation. As argued in Lemma 19, this sequence contains $O(n)$ edge pairs. After the simulation, we perform a binary search for $\hat{p}\hat{q}$ in the sequence of visited edge pairs. The binary search for $\hat{p}\hat{q}$ takes $O(n \log n)$ time, since we can determine the largest path of type ▲-pq-▲ in $O(n)$ time for a fixed position of the shortcut, as shown in Lemma 20.

Lemma 20 *For every augmented tree $T + pq$ with n vertices and straight-line edges, we can determine the length of the longest paths of type ▲-pq-▲ in $O(n)$ time.*

Proof. Every path of type ▲-pq-▲ has length $d_T(s_i, p) + |pq| + d_T(q, s_j)$ for some $i, j = 1, 2, \ldots, k$ with $i < j$ such that pq is useful for (s_i, s_j). This means that s_i is a leaf of one of the secondary $\mathcal{B}$-sub-tees $S_{i_L}, S_{i_L+1}, \ldots, S_{i_H}$ that are attached to the path from p to $\bar{q}$, and s_j is a leaf of one of the secondary $\mathcal{B}$-sub-tees $S_{j_L}, S_{j_L+1}, \ldots, S_{j_H}$ attached to the path from $\bar{p}$ to q. We prove that the matrix M with entries

$$M_{j,i} = \begin{cases} d_T(s_i, p) + |pq| + d_T(q, s_j) & \text{, if } pq \text{ is useful for } (s_i, s_j) \quad \text{for } i_L \le i \le i_H \\ 0 & \text{, otherwise} \qquad\qquad\qquad\qquad \text{and } j_L \le j \le j_H \end{cases},$$

is totally monotone. This means we have to show that $M_{j_1,i_1} < M_{j_1,i_2}$ implies $M_{j_2,i_1} < M_{j_2,i_2}$ for all indices i_1, i_2, j_1, j_2 with $i_L \le i_1 < i_2 \le i_R$ and $j_L \le j_1 < j_2 \le j_R$.

Suppose we have $M_{j_1,i_1} < M_{j_1,i_2}$. Then $M_{j_1,i_2} > 0$, because all entries of M are non-negative. This means that pq is useful for (s_{i_2}, s_{j_1}). Therefore, the shortcut pq is also useful for (s_{i_1}, s_{j_1}), for (s_{i_1}, s_{j_2}), and for (s_{i_2}, s_{j_2}), due to the relative positions of r_{i_1} ,r_{i_2}, r_{j_1}, and r_{j_2}, as shown in Figure 33. Hence, $M_{j,i} = d_T(s_i, p) + |pq| + d_T(q, s_j)$ for $i \in \{i_1, i_2\}$ and $j \in \{j_1, j_2\}$. With this observation, $M_{j_1,i_1} < M_{j_1,i_2}$ implies $d_T(s_{i_1}, p) < d_T(s_{i_2}, p)$, since $d_T(s_{i_1}, p) + |pq| + d_T(q, s_{j_1}) = M_{j_1,i_1} < M_{j_1,i_2} = d_T(s_{i_2}, p) + |pq| + d_T(q, s_{j_1})$. This implies $M_{j_2,i_1} < M_{j_2,i_2}$, because

$$M_{j_2,i_1} = d_T(s_{i_1}, p) + |pq| + d_T(q, s_{j_2}) < d_T(s_{i_2}, p) + |pq| + d_T(q, s_{j_2}) = M_{j_2,i_2} \quad .$$

Thus, the matrix M is totally monotone. We can access any entry $M_{j,i}$ of M in constant time after $O(n)$ pre-processing: We determine $d_T(s_1, r_1)$, $d_T(s_2, r_2)$, …, $d_T(s_k, r_k)$ as well as $d_T(a, r_1)$, $d_T(a, r_2)$, …, $d_T(a, r_k)$ in advance. The shortcut pq is useful for (s_i, s_j) precisely when $d_T(r_i, p) + |pq| + d_T(q, r_j) < d_T(r_i, r_j)$, which we can check in constant time. Computing $d_T(s_i, p) + |pq| + d_T(q, s_j)$ also takes constant time. Therefore, we can determine a largest entry in M—and, thus, a longest path of type ▲-pq-▲—in $O(n)$ time using the SMAWK Algorithm [18] without constructing M explicitly. □

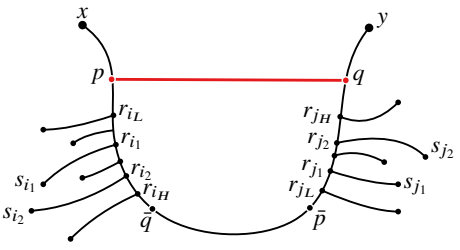


Figure 33: The relative positions of the secondary $\mathcal{B}$-sub-trees with indices $i_L, i_1, i_2, i_H, j_L, j_1, j_2$, and j_H when determining the longest ▲-pq-▲ path.

In conclusion, we can discretize all phases of the continuous algorithm—with some modifications that do not impact optimality—with $O(n)$ events that we can process in $O(n \log n)$ total time, followed by a post-processing step that takes $O(n \log n)$ time.

Theorem 21 *For any geometric tree T with n straight-line edges, we determine a shortcut pq that minimizes the continuous diameter of $T + pq$ in $O(n \log n)$ time.* □

7. Conclusion

We discussed the problem of minimizing the continuous diameter when augmenting a geometric tree with a single shortcut. A natural extension of this problem would be to minimize the continuous diameter when augmenting a geometric network with multiple shortcuts. For instance, given a number $k \geq 2$, we would like to characterize the trees where at least k shortcuts are required to reduce the continuous diameter.

Even though our construction was specified for geometric trees whose edges are straight-line segments, the structural results extend to trees that are embedded into an arbitrary metric space and whose edges are arbitrary rectifiable curves. The algorithmic results extend to more geometric trees with more general edges, e.g., for algebraic curves, the running time of the algorithm increases proportional to the number of times the shortcut grows and shrinks while its endpoints slide along a pair of edges.

References

[1] S. L. Hakimi, Optimum Locations of Switching Centers and the Absolute Centers and Medians of a Graph, Operations Research 12 (3) (1964) 450–459.

[2] G. Y. Handler, Minimax Location of a Facility in an undirected Tree Graph, Transportation Science 7 (3) (1973) 287–293.

[3] V. Chepoi, Y. Vaxès, Augmenting Trees to Meet Biconnectivity and Diameter Constraints, Algorithmica 33 (2) (2002) 243–262.

[4] C.-L. Li, S. McCormick, D. Simchi-Levi, On the Minimum-Cardinality-Bounded-Diameter and the Bounded-Cardinality-Minimum-Diameter Edge Addition Problems, Operations Research Letters 11 (5) (1992) 303–308.

[5] A. A. Schoone, H. L. Bodlaender, J. van Leeuwen, Diameter Increase Caused by Edge Deletion, Journal of Graph Theory 11 (3) (1987) 409–427.

[6] F. Frati, S. Gaspers, J. Gudmundsson, L. Mathieson, Augmenting Graphs to Minimize the Diameter, Algorithmica 72 (4) (2015) 995–1010.

[7] Y. Gao, D. R. Hare, J. Nastos, The Parametric Complexity of Graph Diameter Augmentation, Discrete Applied Mathematics 161 (10-11) (2013) 1626–1631.

[8] E. Oh, H.-K. Ahn, A Near-Optimal Algorithm for Finding an Optimal Shortcut of a Tree, in: 27th International Symposium on Algorithms and Computation (ISAAC 2016), 59:1–59:12, 2016.

[9] U. Große, J. Gudmundsson, C. Knauer, M. Smid, F. Stehn, Fast Algorithms for Diameter-Optimally Augmenting Paths, in: Automata, Languages, and Programming: 42nd International Colloquium, ICALP 2015, Kyoto, Japan, July 6-10, 2015, Proceedings, Part I, 678–688, 2015.

[10] H. Wang, An Improved Algorithm for Diameter-Optimally Augmenting Paths in a Metric Space, in: Proceedings of the 15th International Symposium on Algorithms and Data Structures, WADS 2017, St. John's, NL, Canada, July 31 – August 2, 2017, 545–556, 2017.

[11] M. Farshi, P. Giannopoulos, J. Gudmundsson, Improving the Stretch Factor of a Geometric Network by Edge Augmentation, SIAM Journal on Computing 38 (1) (2008) 226–240.

[12] J. Luo, C. Wulff-Nilsen, Computing Best and Worst Shortcuts of Graphs Embedded in Metric Spaces, in: Proceedings of the 19th International Symposium on Algorithms and Computation, ISAAC 2008, Gold Coast, Australia, December 15–17, 2008, 764–775, 2008.

[13] J.-L. De Carufel, C. Grimm, A. Maheshwari, M. Smid, Minimizing the Continuous Diameter when Augmenting Paths and Cycles with Shortcuts, in: Proceedings of the 15th Scandinavian Symposium and Workshops on Algorithm Theory (SWAT 2016), 27:1–27:14, 2016.

[14] J. Cáceres, D. Garijo, A. González, A. Márquez, M. L. Puertas, P. Ribeiro, Shortcut Sets for Plane Euclidean Networks, Electronic Notes in Discrete Mathematics 54 (2016) 163–168.

[15] B. Yang, Euclidean Chains and their Shortcuts, Theoretical Computer Science 497 (2013) 55–67.

[16] U. Große, J. Gudmundsson, C. Knauer, M. Smid, F. Stehn, Fast Algorithms for Diameter-Optimally Augmenting Paths and Trees, arXiv:1607.05547, 2016.

[17] J. Fischer, V. Heun, Space-Efficient Preprocessing Schemes for Range Minimum Queries on Static Arrays, SIAM Journal on Computing 40 (2) (2011) 465–492.

[18] A. Aggarwal, M. M. Klawe, S. Moran, P. Shor, R. Wilber, Geometric Applications of a Matrix-Searching Algorithm, Algorithmica 2 (1) (1987) 195–208.