Journal of Graph Algorithms and Applications
http://jgaa.info/ vol. 28, no. 1, pp. 179–224 (2024)
DOI: 10.7155/jgaa.v28i1.2933

Distance-Preserving Graph Compression Techniques

Amirali Madani[1] *Anil Maheshwari*[1]

[1]School of Computer Science,
Carleton University, Ottawa, Ontario, Canada

Submitted: June 2023 Accepted: February 2024 Published: May 2024

Article type: Regular paper Communicated by: M. Kaufmann

Abstract. We study the problem of distance-preserving graph compression for weighted paths and trees. The problem entails a weighted graph $G = (V, E)$ with non-negative weights and a subset of edges $E' \subset E$, which needs to be removed from G (with their endpoints merged as a supernode). The goal is to redistribute the weights of the deleted edges in a way that minimizes the error. The error is defined as the sum of the absolute differences of the shortest path lengths between different pairs of nodes before and after contracting E'. Based on this error function, we propose optimal approaches for merging any subset of edges in a path and a single edge in a tree. Previous works on graph compression techniques aimed at preserving different graph properties (such as the chromatic number) or solely focused on identifying the optimal set of edges to contract. However, our focus in this paper is on achieving optimal edge contraction (when the contracted edges are provided as input), specifically for weighted trees and paths.

1 Introduction and Related Work

Graphs have become increasingly relevant for solving real-world problems, leveraging their numerous characteristics [25, 27]. However, many of these graphs are incredibly large, consisting of trillions of edges and vertices, which poses scalability challenges for modern systems [7, 17, 20]. Consequently, graph compression techniques have garnered significant research interest in recent years, aiming to obtain a smaller graph while retaining the essential properties of the original input graph. Different names, such as graph compression [24], graph summarization [21], graph modification [10], and graph contraction [19], have been used in the literature to describe this problem, each within its specific context, leading to various proposed approaches. Regardless of the terminology or context, most of these problems focus on reducing the size of the graph while preserving a specific property [24], while some approaches aim to modify a graph to satisfy a given

This work is funded by the Natural Sciences and Engineering Research Council of Canada (NSERC).

E-mail addresses: amiralimadani@cmail.carleton.ca (Amirali Madani) anil@scs.carleton.ca (Anil Maheshwari)

property [15, 28]. Furthermore, graphs can be compressed in different ways, such as vertex deletions, edge deletions, and edge contractions. It is worth noting that many of these resulting graph modification problems are NP-hard, as indicated in [15].

A relevant problem that is commonly referred to as the *blocker* problem [19] is defined as follows. Given a graph G, integers k and d, an invariant $\pi : \mathcal{G} \to \mathbb{R}$, and some modification operations (such as edge contractions), a blocker problem asks whether there exists a set of at most k graph modification operators such that in the resulting graph $G^{'}$, $\pi(G') \leq \pi(G) - d$ holds. In recent years, blocker problems have been studied for various graph properties, such as the chromatic number [3, 9], maximum weight independent set and minimum weight vertex cover [4], maximum independent set [8, 9], the clique number [9], the total domination number [12], diameter [14], and maximum weight clique [22]. A lot of these blocker problems are defined as *contraction* problems, in which graphs can only be modified via edge contractions. More precisely, given a graph G, integers k and d, and an invariant $\pi : \mathcal{G} \to \mathbb{R}$, CONTRACTION($\pi$) asks whether there exists a set of at most k edge contractions that results in a graph G' with $\pi(G') \leq \pi(G) - d$. Galby *et al.* [11] studied the contraction problems in which a specific edge could be provided as input (in addition to π). As an important contribution, Galby *et al.* [11] proved that, unless **P=NP**, there exists no polynomial-time algorithm that decides whether contracting a given edge reduces the total domination number. Biedl *et al.* [6] studied the problem of flow preserving graph simplification, which is the problem of finding a set of edges whose removal does not change the maximum flow of the underlying network.

Shortest path queries are crucial to various domains, including search engines [13], networks [1, 16] and transportation [2, 30]. In a more relevant work to this paper, Bernstein *et al.* [5] studied a slightly different variant of CONTRACTION. In their work, Bernstein *et al.* [5] focused on compressing a given graph as much as possible, while permitting only a limited amount of distance distortion among any pair of vertices. Given a tolerance function $\varphi(x) = x/\alpha - \beta$, with $\alpha \geq 1$ and $\beta \geq 0$, Bernstein *et al.* [5] studied the problem of finding the maximum cardinality set of edges whose contraction results in a graph G' such that $\mathrm{d}_{G'}(u, v) \geq \varphi(\mathrm{d}_G(u, v))$ for all $u, v \in G$. However, in their work, they only focused on *finding* an optimal set instead of optimally *redistributing* the weights. More specifically, after finding the optimal set E', they set the weight of each edge $e \in E'$ to zero.

Unlike the work by Bernstein *et al.* [5], the work by Sadri *et al.* [24] focuses on optimally redistributing the weights. However, they do not provide any bound guarantees on the amount of error. Precisely, they assess the efficiency of their proposed approach by a set of experimental studies. Moreover, their weight redistribution approach for trees ignores the size of each subtree rooted at the endpoints of a given contracted edge. This is a key factor in deciding an optimal assignment, as we will show in this paper. More recently, Liang *et al.* [18] studied the problem of reachability-preserving graph compression techniques. There have also been other works related to graph compression for unweighted and weighted graphs, as listed in [23, 26]. Zhou *et al.* [29] proposed an efficient approach to remove a large portion of the edges in a network without affecting the overall connectivity by much. Ruan *et al.* [23] studied the minimum gate-vertex set discovery (MGS) problem. The MGS problem is concerned with finding the minimum cardinality set of vertices, designating them as gate vertices, using which every non-local pair of vertices (whose distance is above some threshold) is able to recover its distance in the original network. However, the work by Ruan et al. [23] only studies unweighted graphs.

Where our work stands in the literature: To the best of our knowledge, all existing works have either only focused on finding an optimal set of edges to contract or have not provided any bounds on the amount of error. We study a different problem: instead of choosing *which* optimal edge to contract, we are interested in finding out *how* to contract a given edge optimally. Even though we still study distance-preserving graph compression, our focus is mainly on optimally modifying the graph after a given edge has been contracted. Our primary modification operation is changing the edge weights of the graph. It is worth noting that this problem has received limited attention in the literature, with the closest existing work being the study by Sadri *et al.* [24]. Their approach involves solving a system of equations to determine the new edge weights in the resulting graph. However, their analysis of the problem has certain limitations. Firstly, they do not offer any optimal guarantees for their weight distribution technique. Furthermore, their weight redistribution method does not account for the sizes of the individual subgraphs connected to a given edge. In contrast, as we will demonstrate throughout this paper, the sizes of subtrees (particularly in the context of paths and trees) play a crucial role in achieving optimal weight redistribution.

The organization of the paper: The remainder of this paper is organized as follows. In Section 1.1, we present a summary of our main results along with some comments and details regarding each contribution. In Section 2.1, we describe the notation used in the paper, using which we formally define the scope of our paper in Section 2.2. In Section 3, we study the problem of distance-preserving graph compression for weighted paths, where we prove optimal approaches to contracting any set of k edges. In Section 4, we study the problem of graph compression for weighted trees, where we provide an optimal linear-time algorithm for contracting a single edge. We present the concluding remarks of this paper along with some potential avenues of future work in Section 5.

1.1 Contributions and Results

In Section 3, we study the problem of distance-preserving graph compression for weighted paths.

- As a warm-up, we prove an optimal bound for merging[1] a single edge in a path topology in Section 3.1, whose main result is stated in Theorem 1.

In Section 3.1, we present a method for transforming any weight redistribution for a given merged edge e^* to another redistribution in which only the weights of its neighbouring edges are altered.

- We present Algorithm 1 for merging any set of $k \leq \frac{n}{2}$ independent edges (edges that have no endpoints in common and induce a matching on the path) in a path of size n.

We note that Algorithm 1 produces suboptimal solutions when applied to a contiguous subpath (a connected subgraph) of the given input path. We relate this suboptimal performance to the distinction between merging two regular vertices and two *supernodes*. We thoroughly investigate this distinction in Lemma 3, where we present an optimal redistribution for merging two supernodes.

- Having the suboptimal performance of Algorithm 1 for merging subpaths in mind, in Section 3.3 we study the problem of finding the optimal redistribution for any connected subgraph of a given input path. The optimal method for contracting any contiguous subpath of the input path is presented in Theorem 2.

[1]As defined in Section 2, we use the terms *merging* and *contracting* interchangeably. They both refer to the act of contracting an edge or a set of edges.

- After studying the case of contiguous subpaths, we present Theorem 3 as another generalization of the case with a single edge (Theorem 1). When the edge set to be compressed consists of $k \leq \frac{n}{2}$ independent edges that induce a matching on the input path, Theorem 3 presents an optimal method for graph compression. Theorem 3 provides a correctness proof for Algorithm 1.

In Section 4, we study the problem of distance-preserving graph compression for the tree topology, where we present optimal approaches for merging a single edge in a weighted tree. To this end, we define a relevant problem, which we refer to as the *marking problem*. The objective of the marking problem is to minimize the error, as defined in Section 2, by marking a subset of the neighbouring edges of the merged edge e^* (with weight w^*). For merging an edge e^* with weight w^*, an edge e_i is said to be marked if its new weight $w'(e_i) = w(e_i) + w^*$. As a warm-up, in Section 4.3, we study the marking problem for a tree in which the neighbouring subtrees of e^* are of equal sizes. For such edges, we show that the optimal marking is achieved when all edges either to the left or right (but not both) of e^* are marked. In Section 4.4, we generalize the findings of Section 4.3 and present an optimal marking for any merged edge e^* in a weighted tree.

The definition of the marking problem implies that an edge can either be fully marked or unmarked. It is non-trivial to see whether *fractionally marking* the edges produces better results. Therefore, in Section 4.5, we thoroughly investigate the distinction between the marking problem (Definition 9) and the fractional marking problem (Definition 14) and conclude that any solution to the latter can be transformed into another solution to the former without worsening the error value.

- We present Algorithm 2, an $\mathcal{O}(|V|)$-time algorithm, for finding an optimal marking for e^* in a weighted tree.

2 Preliminaries

In this section, we first discuss the common notation (Section 2.1) and then present some additional definitions (Section 2.2) that help describe the scope of our paper. In Section 2.3, we present a simple number-theoretic lemma, which is later used in some of the proofs in this paper. Throughout this section, we use the path in Figure 1 as the running example of the definitions.

2.1 Notation

Let $G = (V, E)$ denote a graph with V and E as its sets of vertices and edges, respectively. With every edge $e \in E$, we associate a weight $w(e)$ $w : E \to \mathbb{R}_{\geq 0}$. We sometimes denote an edge e by (u, v), where $u, v \in V$ are referred to as the *endpoints* of e. Throughout this paper, we frequently denote edges and vertices using subscripts (for instance, e_i and v_i) and superscripts for merged edges (for instance e^*). When the context is clear, we sometimes abuse the weight notation and denote the weight of e_i and e^* by w_i and w^*, respectively. We denote the number of vertices by $n = |V|$ and a path of n vertices by P_n. Throughout this paper, we frequently use n_L and n_R in different contexts to denote different quantities. However, in most cases, we denote by n_L and n_R the number of vertices to the left and right of a given vertex of a path, respectively (including itself). For instance, in Figure 1-(a), n_L and n_R denote the number of vertices to the left of (and including) v'_3 and the right of (and including) v'_4, respectively. Formally, let G_1 be one of the two connected components of $H = G - \{e_3 = (v'_3, u_1), e^* = (u_1, v_1), e_4 = (v_1, v'_4)\}$ that is adjacent to

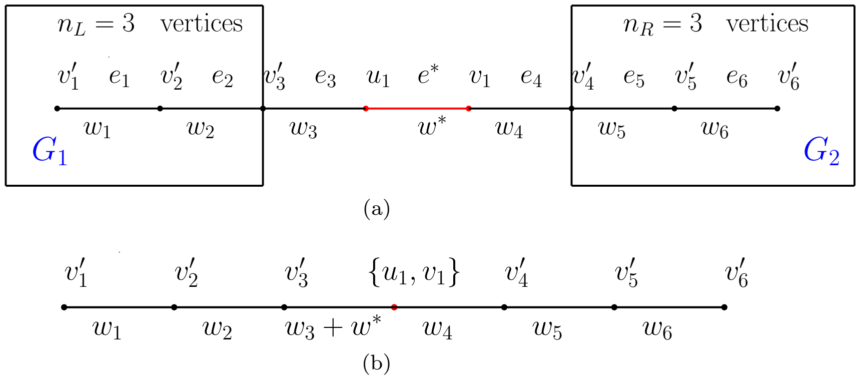


Figure 1: The path used as the running example in Section 2: (a) A path of 8 vertices, with regular edges denoted by $e_i, w_i = w(e_i)$, and the contracted edge is highlighted in red and denoted by $e^* = (u_1, v_1), w^* = w(e^*)$, and (b) The same path after contracting e^* and marking e_3 by setting $w'(e_3) = w_3 + w^*$. In this example, $n_L = n_R$, this is not always the case.

v'_3, and let G_2 be the connected component of H that is adjacent to v'_4. We have

$$n_L = |\{v|v \in G_1\}|\,, n_R = |\{v|v \in G_2\}|$$

For instance, in Figure 1-(a), $n_L = 3$ because G_1 includes vertices $\{v'_1, v'_2, v'_3\}$, and $n_R = 3$ because G_2 includes vertices $\{v'_4, v'_5, v'_6\}$. Therefore, in this paper, we assume that the graph is laid out in the plane and the edge to be merged (e^* in Figure 1-(a)) is horizontal. This assumption will simplify the description of our results.

2.2 Additional Definitions

We now provide some additional definitions for defining the scope of our paper.

Definition 1 *For a weighted graph $G = (V, E)$, the distance between two vertices $u, v \in V$, denoted by $d_G(u, v)$, is the length of the shortest weighted path between u and v in G.*

Definition 2 *A merged edge, or a contracted edge, is one whose endpoints are merged, and the edge itself is removed from the graph.*

For instance, $e^* = (u_1, v_1)$ in Figure 1-(a) (highlighted in red) is a contracted edge. After contracting e^*, the path of Figure 1-(a) is transformed into the one in Figure 1-(b).

Definition 3 *A supernode is a node containing a subset $V' \subset V$ of the nodes in the original graph, which is a result of a series of edge contractions. We denote the set of all supernodes by V_s.*

Definition 4 *For a supernode $v \in V_s$, the cardinality of v, denoted by $\mathcal{C}$, $\mathcal{C} : V_s \to \mathbb{N}$, is the number of regular vertices it contains.*

In the path of Figure 1-(b), $\{u_1, v_1\}$ is a supernode with cardinality 2.

Definition 5 *Let $G = (V, E)$ be a graph with weight function $w : E \to \mathbb{R}_{\geq 0}$, and let $e^* \in E$ be the merged edge. A weight redistribution is a new weight function $w' : E \to \mathbb{R}_{\geq 0}$ in which $w'(e_i) = w(e_i) + \epsilon_i$, $\forall e_i \in E$, $\epsilon_i \in \mathbb{R}$.*

In the path of Figure 1-(b), the weight redistribution sets the edge weights of Figure 1-(a) as $w'(e) = w(e) + w(e^*)$ if $e = e_3$, and $w'(e) = w(e)$ otherwise.

Definition 6 *With reference to a given merged edge e^* in a graph $G = (V, E)$ with the associated weight function $w : E \to \mathbb{R}_{\geq 0}$ and a new weight redistribution function $w' : E \to \mathbb{R}_{\geq 0}$, an edge e_i is said to be marked if $w'(e_i) = w(e_i) + w(e^*)$, unmarked if $w'(e_i) = w(e_i)$, and altered otherwise.*

As shown in Figure 1-(b), e_3 is marked and all other edges are unmarked.

Definition 7 *With reference to a set of merged edges $E_m \subset E$, the set of merged vertices V_m consists of all vertices with at least one endpoint in E_m, or $V_m = \{v | v, u \in V, \exists e = (u, v) \in E_m\}$. The set of unmerged vertices is defined as $\overline{V_m} = V - V_m$.*

In the path of Figure 1, we have $E_m = \{e^*\}$, $V_m = \{u_1, v_1\}$, and $\overline{V_m} = \{v'_1, v'_2, v'_3, v'_4, v'_5, v'_6\}$. With reference to a set of merged edges $E_m \subset E$ and a weight redistribution w', let G' be the resulting graph after contracting the edges in E_m and setting the new edge weights according to w'. The error associated with w' with respect to E_m is denoted by $|\Delta E|$ and calculated as:

$$|\Delta E| = \sum_{u \in V_m, v \in \overline{V_m}, \text{ or } u,v \in \overline{V_m}, u \neq v} |d_G(u, v) - d_{G'}(u, v)| \tag{1}$$

In other words, the error is equal to the sum of the absolute differences (between G and G') of all shortest path lengths between vertices u, v, at least one of which is in $\overline{V_m}$.
Returning to our example in Figure 1, the error function of Eq. (1) sums up the absolute values of the shortest path differences among the vertices of $\overline{V_m} = \{v'_1, v'_2, v'_3, v'_4, v'_5, v'_6\}$, and between the vertices of $\overline{V_m}$ and the vertices of $V_m = \{u_1, v_1\}$. As the final example, we now explain how the distance difference between one of the aforementioned pairs of vertices is calculated. In Figure 1, the shortest path value difference between v'_1 and u_1 changes from $w_1 + w_2 + w_3$ in G (Figure 1-(a)) to $w_1 + w_2 + w_3 + w^*$ in G' (Figure 1-(b)). The error induced by this change is thus equal to $|w_1 + w_2 + w_3 + w^* - w_1 - w_2 - w_3| = w^*$.

We are now ready to present the formal definition of our first studied problem:

Definition 8 ***Distance-Preserving Graph Compression****: Given a graph G, and a set of contracted edges E_m, the problem of distance-preserving graph compression is to find a weight redistribution w' for which $|\Delta E|$ is minimized.*

2.3 A Number-Theoretic Lemma

The following lemma is used in some of the proofs.

Lemma 1 *For all real numbers A, B, C, D, x, y, let $\alpha_1 = |x - A| + |x - A - B|$ and $\alpha_2 = |y - C| + |y - B - C|$. We have $\alpha_1 \geq B$ and $\alpha_2 \geq B$. Furthermore, $\alpha_1 = B$, $\alpha_2 = B$ for $A \leq x \leq A + B$ and $C \leq y \leq B + C$.*

Proof: Let us prove $\alpha_1 \geq B$; the other proof will be analogous. For the sake of contradiction, assume that $\alpha_1 < B$. We have four cases depending on whether the values inside the absolute value function ($x - A$ and $x - A - B$) are positive or negative. Note that $|a| = a$ when $a \geq 0$, and $|a| = -a$ otherwise.

- **Case 1:** $x < A$ and $x < A + B$:

  $$\alpha_1 = A - x + A + B - x < B \rightarrow A < x$$

  which contradicts the assumption ($x < A$).

- **Case 2:** $x < A$ and $x \geq A + B$: These two conditions imply that $B < 0$, we have:

  $$\alpha_1 = A - x + x - A - B < B \rightarrow 0 < 2B$$

  which is a contradiction since $B < 0$.

- **Case 3:** $x \geq A$ and $x < A + B$:

  $$\alpha_1 = x - A + A + B - x < B \rightarrow 0 < 0$$

  which is impossible.

- **Case 4:** $x \geq A$ and $x \geq A + B$:

  $$x - A + x - A - B < B \rightarrow x < A + B$$

  which contradicts the assumption.

Since we get a contradiction for every possible case, we have $\alpha_1 \geq B$ and $\alpha_1 = B$ for $A \leq x \leq A+B$. Similarly, we have $\alpha_2 \geq B$ and $\alpha_2 = B$ for $C \leq x \leq B + C$. □

We will also use the following corollary.

Corollary 1 *Given two real numbers x, z we have $|z| - |x| \leq |z - x|$.*

Proof: Using Lemma 1, we have the following for two real numbers x and z:

$$|z| \leq |x| + |z - x| \rightarrow |z| - |x| \leq |z - x|$$

□

3 Graph Compression for Paths

In this section, we study the problem of distance-preserving graph compression for a weighted path with non-negative weights. The paths in this section all have $n \geq 3$ vertices since the compression problem for a two-vertex path is trivial.

The remainder of this section is organized as follows. As a warm-up, we provide optimal bounds for merging a single edge in Section 3.1. In Section 3.2, we study the path compression problem for an edge connecting two supernodes, each consisting of a subset of nodes from the path. Two generalizations of the results of Section 3.1 for contracting any subpath (a contiguous subpath of the original graph) and any set of independent edges (that induce a matching in the original path) are provided in Section 3.3 and Section 3.4 respectively.

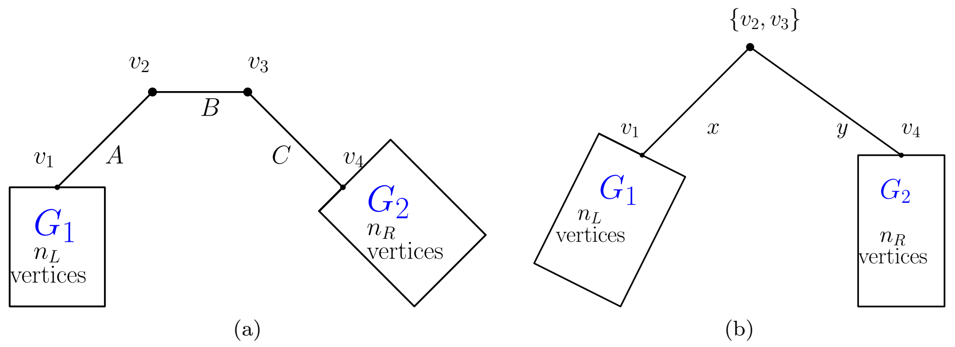


Figure 2: Merging a single edge $e^* = (v_2, v_3)$ with weight B in a path of n vertices, with $n_L \geq 0$, $n_R \geq 0$ vertices on the left of v_1 and the right of v_4 respectively ($n_L + n_R = n - 2$) (a) The original graph before merging v_2 and v_3 into a supernode. The neighbouring edges of e^* have weights A and C. (b) The modified path after merging v_2 and v_3 into a supernode.

3.1 A Tight Lower Bound for Merging One Edge

This section presents a tight lower bound on the optimal error (Eq. (1)) associated with merging a single edge in a path topology.

As seen in Figure 2, the edge between v_2 and v_3 is merged, and only the immediate edge weights are altered to x and y. Later in this section (Lemma 2), we show why it is sufficient to alter only the immediate edge weights (A and B in Figure 2) to get the minimum amount of error. Note that, for merging a single edge (Figure 2) we have:

$$n_L + n_R = n - 2 = \left|\overline{V_m}\right| \tag{2}$$

The following theorem is now presented:

Theorem 1 *Let $|\Delta E|$ be the error associated with merging a single edge $e^* = (v_2, v_3)$ (with weight B) in a path $P_n, n \geq 3$ (Figure 2). Furthermore, let $V_m = \{v_2, v_3\}$ and $\overline{V_m} = V - V_m$. We have $|\Delta E| \geq (n-2)B = |\overline{V_m}|B$. Moreover, this lower bound is tight and can be achieved by marking the left neighbour of the merged edge. If the merged edge has no left or right neighbour, the lower bound can be achieved by simply contracting the edge, and no further modifications (weight changes) are required.*

Proof: Figure 2 depicts the situation in which edge e^* with weight B is merged. We first assume that e^* has a left neighbour, and we handle the no-neighbour exception at the end of the proof. As seen in Figure 2-(b), let x and y denote the new edge weights of the neighbouring edges of e^*, and let G_1 (with n_L vertices) and G_2 (with n_R vertices) denote the subpaths rooted at v_1 and v_4 respectively. We denote the error by $|\Delta E|$ and classify it into different parts (in accordance with Eq. (1)):

1. The error between two vertices $u \in G_1, v \in G_2$ is $|x + y - A - B - C|$. The only affected portion of such a shortest path is the subpath between v_1 and v_4, the value of which changes from $A + B + C$ (in Figure 2-(a)) to $x + y$ (in Figure 2-(b)). Summing over all such pairs $u \in G_1, v \in G_2$, the total amount of error is $n_L n_R |x + y - A - B - C|$.

2. Between two vertices $u, v \in G_1$, there is no error, because the shortest path value between all such pairs of vertices is unchanged. Similarly, between two vertices $u, v \in G_2$, there exists no error.

3. The error between a vertex $u \in G_1$ and the vertices in $V_m = \{v_2, v_3\}$ is $|x - A| + |x - A - B|$. In the path from u to v_2, the only changed (with reference to edge weights) subpath is the subpath between v_2 and v_1 which changes from A (Figure 2-(a)) to x (Figure 2-(b)), inducing an error of $|x - A|$. Similarly, the error between some vertex $u \in G_1$ and v_3 is $|x - A - B|$ as that is the amount by which the weight of the subpath from v_1 to v_3 changes. The total amount of error between all vertices $u \in G_1$ and the vertices in $V_m = \{v_2, v_3\}$ is therefore $n_L(|x - A| + |x - A - B|)$.

4. By similar reasoning to the one provided above, the total amount of error between all vertices $u \in G_2$ and the vertices in $V_m = \{v_2, v_3\}$ is equal to $n_R(|y - C| + |y - B - C|)$.

Therefore, we can formulate $|\Delta E|$ as

$$\begin{aligned} |\Delta E| &= n_L(|x - A| + |x - A - B|) + n_R(|y - C| + |y - B - C|) + n_L n_R |x + y - A - B - C| \\ &= n_L \alpha_1 + n_R \alpha_2 + n_L n_R |x + y - A - B - C| \end{aligned}$$

where α_1 and α_2 are the values defined in Lemma 1. Using Lemma 1, we know $\alpha_1 \geq B$ and $\alpha_2 \geq B$. Thus,

$$|\Delta E| \geq B(n_L + n_R) + n_L n_R |x + y - A - B - C|$$

Using Eq. (2), we have

$$|\Delta E| \geq B(n - 2) + n_L n_R |x + y - A - B - C| \geq B(n - 2) = |\overline{V_m}| B$$

Which proves the first part of the theorem.

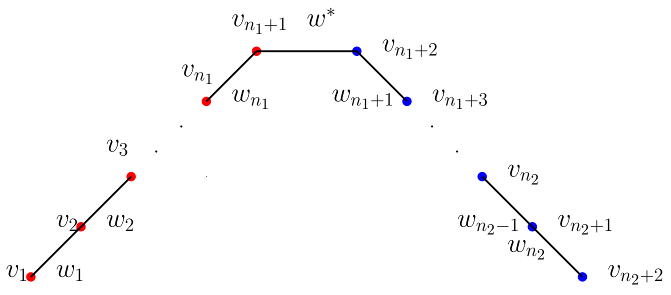
As for the second part, we now show that this lower bound is tight. By marking the left neighbouring edge of e^* (effectively setting $x = A + B$ and $y = C$) we get

$$\begin{aligned} |\Delta E| &= n_L(|x - A| + |x - A - B|) + n_R(|y - C| + |y - B - C|) + n_L n_R |x + y - A - B - C| \\ &= (n_L + n_R)B \\ &= (n - 2)B \\ &= |\overline{V_m}| B \end{aligned}$$

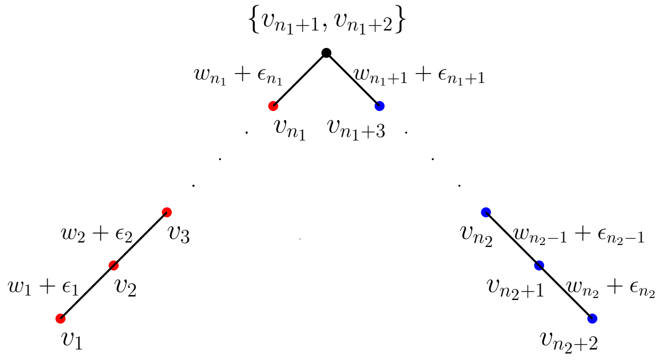
This analysis concludes the proof for the case where e^* has a left neighbour.

If e^* has no left neighbour, i.e., $n_L = 0$, and no shortest path crossing e^* is affected. For each shortest path starting from v_4 (and its right-side vertices) and terminating at v_2 and v_3, there is an error of $|y - C| + |y - B - C|$. According to Lemma 1, $|y - C| + |y - B - C|$ is minimized as long as $C \leq y \leq B + C$, which is the case if all edges are unmarked, i.e., $y = C$. We can use a similar argument if e^* has neither a right nor a left neighbour. □

Observe that marking the left neighbouring edge is not the only way of achieving the lower bound as it can also be achieved by marking the right neighbouring edge. In fact, any assignment of values to x and y such that $x = A + \epsilon_1$, $y = C + \epsilon_2$, $\epsilon_1 + \epsilon_2 = B$ will have the same impact. Therefore, for merging a single edge in a weighted path, the marked neighbour can be chosen arbitrarily, and the error value is *oblivious* to the marking direction. However, this observation (being oblivious to the marking direction) only holds for merging two regular nodes. As we will



(a) The original graph, before contracting $e^* = (v_{n_1+1}, v_{n_1+2})$ with weight w^*


$\{v_{n_1+1}, v_{n_1+2}\}$
$w_{n_1} + \epsilon_{n_1}$
$w_{n_1+1} + \epsilon_{n_1+1}$
v_{n_1}
v_{n_1+3}
$w_2 + \epsilon_2$
v_3
$w_1 + \epsilon_1$
v_2
v_1
v_{n_2}
$w_{n_2-1} + \epsilon_{n_2-1}$
v_{n_2+1}
$w_{n_2} + \epsilon_{n_2}$
v_{n_2+2}

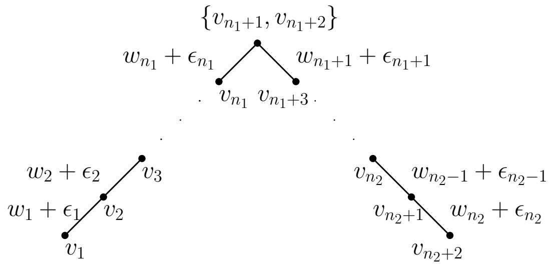
(b) An arbitrary weight redistribution which is transformed in the proof of Lemma 2 to the redistribution of Figure 4-(b).

Figure 3: The figure used in the proof of Lemma 2. The vertices of V_L and V_R are depicted in red and blue, respectively.

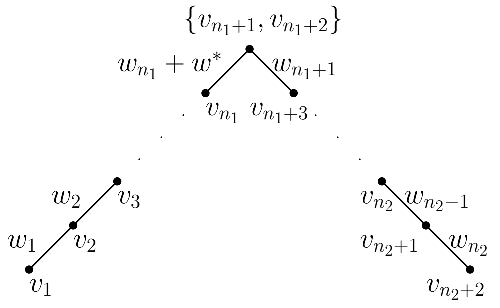
show in Lemma 3, for merging two supernodes the optimal error is obtained by marking the edge adjacent to the smaller node with respect to cardinality.

Theorem 1 assumes that to achieve the minimum amount of error, we have to alter only the immediate edges directly connected to the endpoints of the merged edge. We now prove the correctness of this assumption. We first define some notation. For any edge $e_i \in E$, we denote its new weight as $w'(e_i) = w(e_i) + \epsilon_i$ where ϵ_i is a real number (see Figure 3). This definition allows us to increase or decrease the weight of any given edge e_i by ϵ_i. We call this assignment of weights a *redistribution* for e^*. We refer to an edge e_i as *altered* if $\epsilon_i \neq 0$, and *unaltered* otherwise. Moreover, let $V_L, V_R \subset V$ be the vertices to the left and right of the merged edge respectively as depicted in Figure 3. Therefore, the problem is now to show that there exists an optimal solution, with only the immediate edges altered. For simplicity, we slightly abuse the notation and write $w(e_i)$ as w_i and $w(e^*)$ as w^*. In the following lemma, we show a construction for transforming any redistribution into another equivalent redistribution in which only the immediate edges are altered.

Lemma 2 *(See Figure 3 and Figure 4) For a merged edge e^* (in a weighted path) that has both left and right neighbouring edges, any weight redistribution can be transformed into another weight redistribution in which only the left neighbouring edge of e^* is altered (i.e., $\forall i \neq n_1$ in Figure 3, $\epsilon_i = 0$; see Figure 4). The error associated with this redistribution is no worse than that of the*


$\{v_{n_1+1}, v_{n_1+2}\}$
$w_{n_1} + \epsilon_{n_1}$
$w_{n_1+1} + \epsilon_{n_1+1}$
v_{n_1}
v_{n_1+3}
$w_2 + \epsilon_2$
v_3
$w_1 + \epsilon_1$
v_2
v_1
v_{n_2}
$w_{n_2-1} + \epsilon_{n_2-1}$
v_{n_2+1}
$w_{n_2} + \epsilon_{n_2}$
v_{n_2+2}

(a) Before applying the construction method of Lemma 2



(b) After applying the construction method of Lemma 2

Figure 4: The construction method of Lemma 2.

original one.

Proof: We prove the lemma by presenting a construction method for transforming any arbitrary weight redistribution to another one in which only the left neighbouring edge is altered. Furthermore, we show this transformation does not worsen the error. The illustration is mainly based on Figure 3 and Figure 4. Figure 3-(b) depicts an arbitrary weight redistribution for merging $e^* = (v_{n_1+1}, v_{n_1+2})$, which is transformed into another weight redistribution (depicted in Figure 4-(b)).

We now present a simple construction as follows. For illustration, see Figure 4.
Set $\epsilon_i = 0 \ \ \forall i \neq n_1$, and $\epsilon_{n_1} = w^*$. Note that this new redistribution may cause some parts of the error to increase. However, we will use Corollary 1 to provide an upper bound on any potential error increase and show that there will always be enough decrease in error to counterbalance the increase. In the original redistribution, the error between two vertices $v_i, v_j \in V_L (i < j)$ is:

$$\left|\sum_{k=i}^{j-1}(w_k + \epsilon_k) - \sum_{k=i}^{j-1} w_k\right| = \left|\sum_{k=i}^{j-1} \epsilon_k\right| \tag{3}$$

The indices i and j used in this proof are based on the ones depicted in Figure 3 and Figure 4. Assume that the path of Figure 3 has $n_2 + 1$ edges with $V_L = \{v_i | 1 \leq i \leq n_1 + 1\}$ and $V_R = \{v_i | n_1 + 2 \leq i \leq n_2 + 2\}$, with $n_1 = n_L$ and $n_2 = n_R$. For instance, v_1 and v_3 are two vertices from V_L in Figure 3 and the original shortest path length between v_1 and v_3 is $w_1 + w_2 = \sum_{k=1}^{2} w_k$ (Figure 3-(a)). In the original redistribution of Figure 3-(b) (which we transform into Figure 4-(b)), this length is $w_1 + \epsilon_1 + w_2 + \epsilon_2 = \sum_{k=1}^{2}(w_k + \epsilon_k)$, resulting in Eq. (3). A similar equation can also

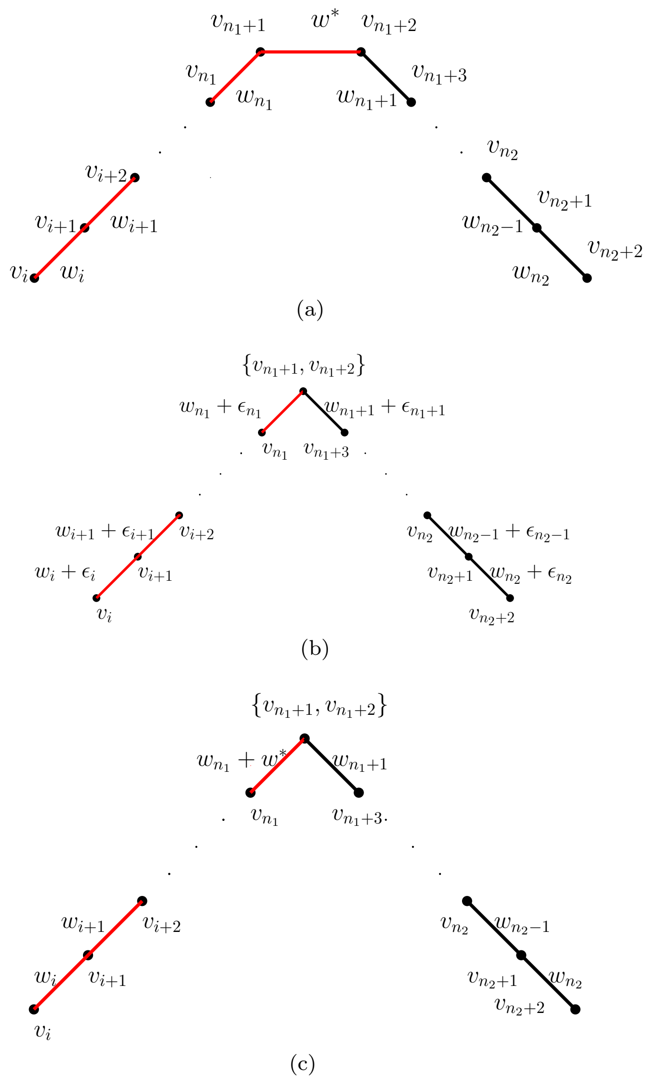


Figure 5: Case 2 in the proof of Lemma 2. (a) The original graph (b) The original weight redistribution (c) After applying the construction method of Lemma 2. The affected shortest path of Case 2 is highlighted in red.

be defined for any two vertices $v_i, v_j \in V_R$. Moreover, the error between a vertex $v_i \in V_L$ and a vertex $v_j \in V_R$ $(i < j)$ in the original redistribution is equal to:

$$\left|\sum_{k=i}^{j-2}(w_k + \epsilon_k) - w^* - \sum_{k=i}^{j-2} w_k\right| = \left|w^* - \sum_{k=i}^{j-2} \epsilon_k\right| \tag{4}$$

Transforming the weight redistribution (as shown in Figure 4) changes the error value. We break this change down into five different cases:

- **Case 1:** The error between two vertices $v_i, v_j \in V_L$ $(i < j,\ j \neq n_1 + 1)$ decreases by $-\left|\sum_{k=i}^{j-1} \epsilon_k\right|$, because after the construction this error is equal to zero (compare Figure 4-(b) with Figure 3-(a) for v_1 and v_3) and using Eq. (3), the change is equal to $0 - \left|\sum_{k=i}^{j-1} \epsilon_k\right|$.

- **Case 2:** The error between some vertex $v_i \in V_L, v_i \neq v_{n_1+1}$, and every vertex $v_j \in V_R$ decreases. Specifically, the error between v_i and v_{n_1+2} decreases by $-|w^* - \sum_{k=i}^{n_1} \epsilon_k|$ using Eq. (4) (see Figure 5).

- **Case 3:** The error between some vertex $v_i \in V_L, v_i \neq v_{n_1+1}$, and v_{n_1+1} changes by $|w^*| - |\sum_{k=i}^{n_1} \epsilon_k|$, because in the original redistribution, the error is equal to $|\sum_{k=i}^{n_1} \epsilon_k|$ (Eq. (3)) and in the new redistribution, it is equal to $|w^*|$. This change might lead to an increase in error; however, by using Corollary 1 and setting $z = w^*$ and $x = \sum_{k=i}^{n_1} \epsilon_k$ we have:

$$|w^*| - \left|\sum_{k=i}^{n_1} \epsilon_k\right| \leq \left|w^* - \sum_{k=i}^{n_1} \epsilon_k\right|$$

  In other words, if the construction causes an increase in error, it is at most equal to $|w^* - \sum_{k=i}^{n_1} \epsilon_k|$. However, from Case 2 we know that each such vertex v_i also has an error decrease of $-|w^* - \sum_{k=i}^{n_1} \epsilon_k|$, which will be enough to nullify this increase.

- **Case 4:** The error between some vertex $v_j \in V_R, v_j \neq v_{n_1+2}$, and v_{n_1+2} decreases by $-\left|\sum_{k=n_1+1}^{j-2} \epsilon_k\right|$ using Eq. (3).

- **Case 5:** The error between some vertex $v_j \in V_R, v_j \neq v_{n_1+2}$, and v_{n_1+1} changes by $|w^*| - \left|w^* - \sum_{k=n_1+1}^{j-2} \epsilon_k\right|$. This change may lead to an increase in error; however, we use Corollary 1 to bound this increase:

$$|w^*| - \left|w^* - \sum_{k=n_1+1}^{j-2} \epsilon_k\right| \leq \left|\sum_{k=n_1+1}^{j-2} \epsilon_k\right|$$

  Therefore, we have enough decrease from Case 4 to nullify this increase.

□

Corollary 2 *For a given merged e^* with left and right neighbouring edges, there exists an optimal redistribution in which only the left-neighbouring edge of e^* is marked and all other edges are unmarked.*

Based on Theorem 1, the algorithm for merging a set S of edges in a path $G = P_n$ is presented in Algorithm 1. Algorithm 1 continuously applies Theorem 1 and marks the left neighbouring edge of each edge $e^* \in S$ taken in an arbitrary order.

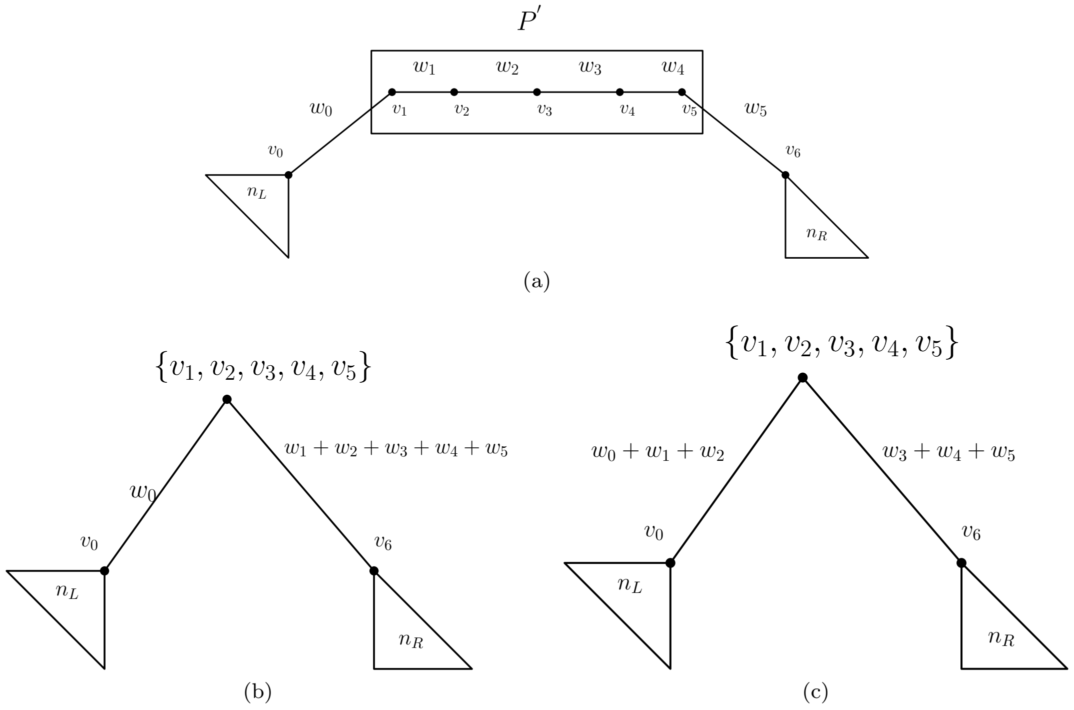


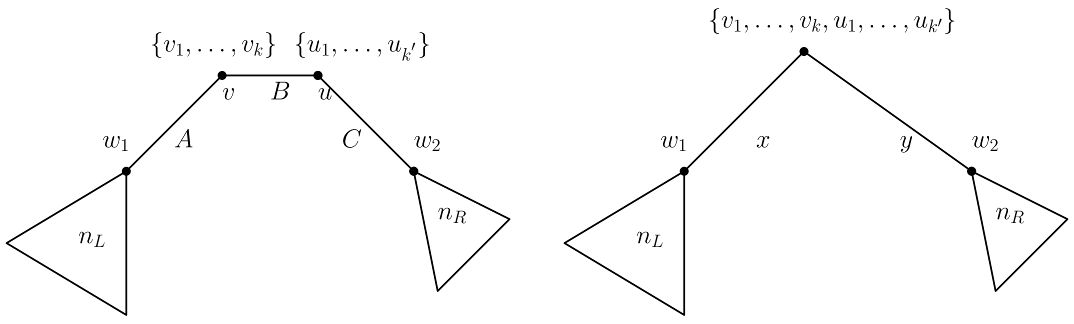
Figure 6: (a) An example of merging an entire subpath $P' \subset P$ with four edges, (b) A suboptimal solution generated by Algorithm 1, and (c) The optimal solution.

Algorithm 1 Graph Compression Algorithm for Independent Edges

```
 1: procedure PATH_COMPRESSION
 2:     Input: $G = P_n$ (A path of $n$ vertices) with weight function $w : E \to \mathbb{R}_{\geq 0}$, a set $E_m$ of edges to merge
 3:     $S \leftarrow E_m$
 4:     while $S$ is not empty do
 5:         Pick an arbitrary edge $e \in S$
 6:         if $e$ has a left neighbouring edge then
 7:             Let $e'$ be the left neighbouring edge of $e$
 8:             Set $w(e') \leftarrow w(e') + w(e)$                ▷ Mark $e'$
 9:         end if
10:         Remove $e$ from $G$ and merge its endpoints
11:         Remove $e$ from $S$
12:     end while
13: end procedure
```

Unfortunately, Algorithm 1 may produce suboptimal results when applied to specific kinds of inputs. Precisely, it may produce suboptimal results when merging a connected subpath of the given path. The reason behind this suboptimal performance lies in the difference between merging two regular nodes and two supernodes. An example of merging a subpath of size four is depicted in Figure 6-(a), for which Algorithm 1 may produce the suboptimal solution Figure 6-(b). Later in Section 3.3, we shall show that the optimal solution for this example is the one depicted in Figure 6-(c). Furthermore, in Theorem 3, we prove that when the input to Algorithm 1 consists of independent edges in G (if S induces a matching in G), it produces the optimal results.

3.2 Merging Supernodes



(a) Before merging the supernodes (b) After merging the supernodes

Figure 7: An example of merging two supernodes with cardinalities k and k'.

As seen in the previous section, Algorithm 1 may find suboptimal solutions when given an entire subpath of size k. The main reason behind this suboptimal performance lies in the difference between merging regular nodes and *supernodes*. Recall from Definition 3 that a supernode contains more than one node of the original graph. In this section, we show that merging two supernodes differs from merging two regular vertices, and we provide a generalized version of Theorem 1. Interestingly, we observe that unlike merging two regular nodes in which the error value was oblivious to the marking direction, for merging supernodes, this direction is directly affected by the cardinality (Definition 4) of each endpoint. In the following lemma, we shall see that for merging an edge $e^* = (u, v)$ connecting two supernodes u and v, the optimal solution is obtained by marking the edge adjacent to the lighter vertex (the one with the smaller cardinality) among u and v.

Lemma 3 *Suppose we have supernodes v and u (as shown in Figure 7), with $\mathcal{C}(v) = k$ and $\mathcal{C}(u) = k'$ (where $k \geq k'$), connected to vertices w_1 and w_2, respectively. The error incurred by merging the edge $e^* = (u, v)$ (with weight B) is at least $B \times k' \times (n - (k + k'))$. Furthermore, this lower bound can be achieved by marking the neighbouring edge adjacent to the smaller vertex among v and u in terms of cardinality ($e = (u, w_2)$ in Figure 7). If the smaller vertex, with reference to cardinality, has no neighbouring edge other than $e^* = (u, v)$, then the optimal error can be achieved by contracting e^* without any further modifications or weight changes.*

Proof: The analysis is similar to the case of merging regular vertices. We enumerate all possible error values and then deduce the optimal assignment. We first assume that u (the smaller vertex) is adjacent to another edge $e' \neq e^*$, and we handle the other case (only adjacent to e^*) later in the proof. We denote the error by $|\Delta E|$. Note that $n_L + n_R = |\overline{V_m}| = n - (k + k')$. Let x and y

denote the new weights of the edges adjacent to (v, u), we have:

$$|\Delta E| = \underbrace{n_L \times |x - A| \times k}_{\text{between the subpath of } w_1 \text{ and the vertices in } v} + \underbrace{n_L \times |x - A - B| \times k'}_{\text{between the subpath of } w_1 \text{ and the vertices in } u} + \underbrace{n_R \times |y - C| \times k'}_{\text{between the subpath of } w_2 \text{ and the vertices in } u} + \underbrace{n_R \times |y - B - C| \times k}_{\text{between the subpath of } w_2 \text{ and the vertices in } v} + \underbrace{n_L \times n_R \times |x + y - A - B - C|}_{\text{between the subpath of } w_1 \text{ and } w_2}$$

Because $k \geq k'$, we further simplify $|\Delta E|$ as:

$$\begin{aligned} |\Delta E| =& (k - k') \times n_L \times |x - A| + n_L \times k' \times \big(|x - A| + |x - A - B|\big) \\ &+ (k - k') \times n_R \times |y - B - C| + n_R \times k' \times \big(|y - C| + |y - B - C|\big) \\ &+ n_L \times n_R \times |x + y - A - B - C| \end{aligned} \tag{5}$$

Using Lemma 1, and the fact that $k - k' \geq 0$, we have:

$$|\Delta E| \geq n_L \times k' \times \underbrace{\big(|x - A| + |x - A - B|\big)}_{\geq B \text{ (Lemma 1)}} + n_R \times k' \times \underbrace{\big(|y - C| + |y - B - C|\big)}_{\geq B \text{ (Lemma 1)}} \tag{6}$$

$$\geq B \times k' \times \underbrace{(n_L + n_R)}_{= n - (k + k')} = B \times k' \times (n - (k + k')) \tag{7}$$

We can observe that this lower bound (Eq. (7)) is tight and can be achieved by setting $y = B + C$ and $x = A$ in Eq. (5).

Now if u is only adjacent to e^*, the lower bound can be achieved by just contracting e^* and leaving the weight function unchanged. To see why, suppose u is only adjacent to e^*. We have $n_R = 0$ and the error is equal to:

$$\begin{aligned} |\Delta E| =& (k - k') \times n_L \times |x - A| + n_L \times k' \times \big(|x - A| + |x - A - B|\big) \\ &+ (k - k') \times n_R \times |y - B - C| + n_R \times k' \times \big(|y - C| + |y - B - C|\big) \\ &+ n_L \times n_R \times |x + y - A - B - C| \\ =& (k - k') \times n_L \times |x - A| + n_L \times k' \times \big(|x - A| + |x - A - B|\big) \\ =& n_L \times k' \times B \text{ (by setting } x = A\text{, as no edge weights are changed.)} \end{aligned}$$

On the other hand, note that $n_L + n_R = n - (k + k')$, and $n_R = 0$ implies that $n_L = n - (k + k')$. We have:

$$|\Delta E| = n_L \times k' \times B = (n - (k + k')) \times k' \times B$$

and the lower bound is achieved without any weight changes.

It is worth noting that similar to Theorem 1, we assume that it is sufficient to only alter the neighbouring edges of the merged edge $e^* = (u, v)$. This proof for this assumption is almost identical to that of Lemma 2, where any arbitrary redistribution can be transformed into another redistribution in which only the edge adjacent to the smaller vertex ($e = (u, w_2)$) is marked. Then, similar to the proof of Lemma 2, the decrease in error is always sufficient to counterbalance any potential error increase. The only difference is that in the new claim, the decrease in error and any potential error increase are weighted by k and k', respectively. Since $k \geq k'$, the proof follows. □

Remark 1 *Lemma 3 is a generalization of Theorem 1. Thinking of each regular vertex as a supernode with cardinality one, we have $k = k' = 1$ and using Lemma 3, the error is equal to $B \times k' \times (n - (k + k')) = B \times (n - 2)$ by arbitrarily marking one of the neighbouring edges (since the endpoints have equal cardinalities).*

Remark 2 *Using Lemma 3, we can now explain the suboptimal performance of Algorithm 1 for edges that are not independent and form a contiguous subpath. For inputs of such kind, Algorithm 1 continuously marks the left neighbouring edges of all edges $e^* \in S$, potentially marking an edge adjacent to the heavier endpoint of some $e^* \in S$ along the way and violating the conditions of Lemma 3.*

In the next section, we study the problem of optimally merging an entire contiguous subpath of the path.

3.3 Merging Contiguous Subpaths

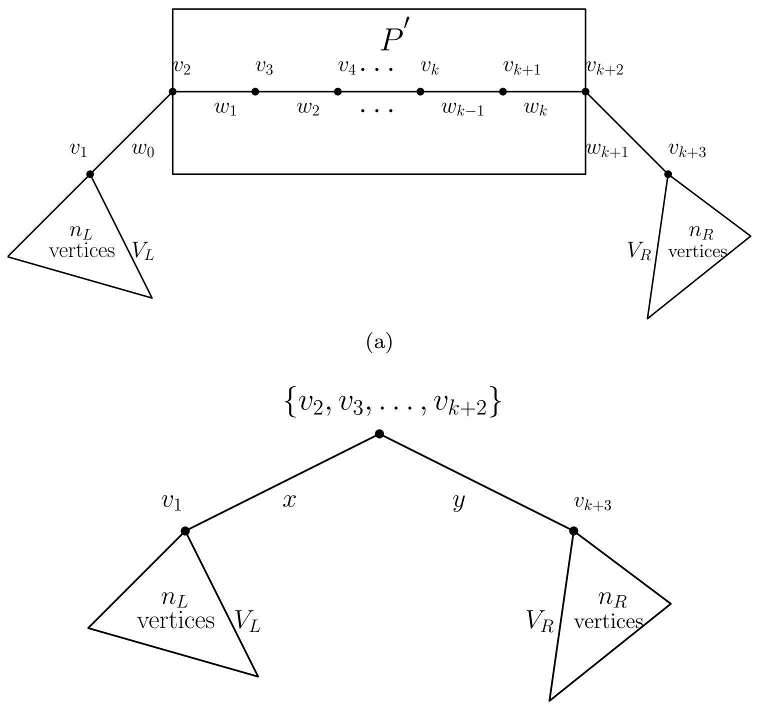


Figure 8: The figure used in the proofs of Section 3.3 (a) Before merging an entire subpath $P' \subset P$ with k edges, (b) After the merge.

This section presents an optimal way of merging any contiguous subpath (or connected subpath) of a given path. For convenience, we refer to contiguous subpaths as subpaths. Let $P^{'} \subseteq P$ be the desired subpath consisting of k edges (see Figure 8 for an illustration). Throughout this section, we assume k is even; otherwise, we can convert P' into an equivalent subpath of even length by adding a dummy edge of weight zero. As depicted in Figure 8, we assume P' partitions the set of vertices into two subsets, V_L and V_R, with n_L and n_R vertices, respectively. We denote the error associated with contracting P' by $\mathcal{E}$ and break it down into three components:

- $\mathcal{E}_L$, the error between the vertices in V_L and the ones inside P',
- $\mathcal{E}_R$, the error between the vertices in V_R and the ones inside P', and
- $\mathcal{E}_{LR}$ the error between the vertices of V_L and V_R.

With this in mind, we formulate $\mathcal{E}$ as:

$$\mathcal{E} = \mathcal{E}_L + \mathcal{E}_R + \mathcal{E}_{LR} \tag{8}$$

such that:

$$\mathcal{E}_L = n_L \times \left(\underbrace{|x - w_0|}_{\text{between vertices of } V_L \text{ \& } v_2} + \underbrace{|x - w_0 - w_1|}_{\text{between vertices of } V_L \text{ \& } v_3} + \cdots + \underbrace{|x - w_0 - w_1 - \cdots - w_k|}_{\text{between vertices of } V_L \text{ \& } v_{k+2}} \right) \tag{9}$$

$$\mathcal{E}_R = n_R \times \left(\underbrace{|y - w_{k+1}|}_{\text{between vertices of } V_R \text{ \& } v_{k+2}} + \underbrace{|y - w_{k+1} - w_k|}_{\text{between vertices of } V_R \text{ \& } v_{k+1}} + \cdots + \underbrace{|y - w_{k+1} - w_k - \cdots - w_1|}_{\text{between vertices of } V_R \text{ \& } v_2} \right) \tag{10}$$

and

$$\mathcal{E}_{LR} = n_L \times n_R \times |x + y - w_0 - w_1 - \cdots - w_{k+1}| \tag{11}$$

where x and y are the new weights of the neighbouring edges of P' (Figure 8-(b)).

We first prove the optimal solution for $\mathcal{E}_L$ and derive the optimal solution for $\mathcal{E}_R$ by symmetry. Let $\mathcal{E}_L^{(i)}$ denote the value of $\mathcal{E}_L$ when $x = w_0 + w_1 + \cdots + w_i$ for $0 \le i \le k$. We prove the following lemma using induction on i.

Lemma 4 $\mathcal{E}_L^{(i)} = n_L \times \big(\sum_{j=0}^{i} j\, w_j + \sum_{j=i+1}^{k} (k+1-j)\, w_j \big)$

Proof: For the base case, $\mathcal{E}_L^{(0)}$, assume $x = w_0$. By a simple replacement into Eq. (9) we get:

$$\mathcal{E}_L^{(0)} = n_L \times \big(w_1 + w_1 + w_2 + w_1 + w_2 + w_3 + \cdots + w_1 + w_2 + \cdots + w_k \big)$$

In other words, every w_j, $1 \le j \le k$, is repeated $k+1-j$ times, and:

$$\mathcal{E}_L^{(0)} = n_L \times \left(\sum_{j=1}^{k} (k+1-j)\, w_j \right)$$

Now assume the lemma holds for all $j < i+1$. By the inductive hypothesis, we have $\mathcal{E}_L^{(i)} = n_L \times \big(\sum_{j=0}^{i} j\, w_j + \sum_{j=i+1}^{k} (k+1-j)\, w_j \big)$.

We break Eq. (9) into $k+1$ *clauses*, such that $c_j = |x - w_0 - w_1 - \cdots - w_j|$ for $0 \le j \le k$. Going from $x = \sum_{j=0}^{i} w_j$ to $x = \sum_{j=0}^{i+1} w_j$, $\mathcal{E}_L^{(i)}$ first increases by $n_L \times \big((i+1)\, w_{i+1}\big)$ because there are $i+1$ clauses $c_0, c_1, \ldots, c_i$ that do not include w_{i+1}, and then decreases by $n_L \times \big((k-i)\, w_{i+1}\big)$

because there are $k-i$ clauses $c_{i+1},\dots,c_k$ that include w_{i+1} and were not covered by the previous assignment of x ($x=\sum_{j=0}^{i} w_j$). Therefore, we have:

$$\begin{aligned}
\mathcal{E}_L^{(i+1)} &= \mathcal{E}_L^{(i)} + n_L \times \left((i+1)\, w_{i+1} - (k-i)\, w_{i+1}\right) \\
&= n_L \times \left(\sum_{j=0}^{i} j\, w_j + \sum_{j=i+1}^{k} (k+1-j)\, w_j + (i+1)\, w_{i+1} - (k-i)\, w_{i+1} \right) \\
&= n_L \times \left(\sum_{j=0}^{i+1} j\, w_j + \sum_{j=i+2}^{k} (k+1-j)\, w_j \right)
\end{aligned}$$

□

The following lemma states that the optimal value of $\mathcal{E}_L$ is equal to $\mathcal{E}_L^{(\frac{k}{2})}$.

Lemma 5 *The optimal value of $\mathcal{E}_L$ is obtained when $x = w_0 + w_1 + \cdots + w_{\frac{k}{2}}$.*

Proof: It suffices to show the optimal value of $\mathcal{E}_L$ is equal to $\mathcal{E}_L^{(\frac{k}{2})}$. From the proof of Lemma 4, we know that $\mathcal{E}_L^{(i+1)} - \mathcal{E}_L^{(i)} = n_L \times \big((i+1)\, w_{i+1} - (k-i)\, w_{i+1}\big)$.
Therefore, $\mathcal{E}_L^{(i+1)} - \mathcal{E}_L^{(i)} < 0$ if:

$$i+1-k+i<0 \to 2i<k-1 \to i<\frac{k}{2}-\frac{1}{2} \xrightarrow[\text{since } k \text{ is even}]{} i \le \frac{k}{2}-1$$

In other words, $\mathcal{E}_L^{(\frac{k}{2})}$ is strictly better than (less than) any $\mathcal{E}_L^{(j)}, j \neq \frac{k}{2}$. Note that the optimal solution also cannot happen when $x = \epsilon + \sum_{j=0}^{\frac{k}{2}} w_j$ for some $0<\epsilon<w_{\frac{k}{2}+1}$, because in that case, the error would be equal to:

$$\mathcal{E}_L^{(\frac{k}{2})} + \left(\frac{k}{2}+1\right)\epsilon - \left(\frac{k}{2}\right)\epsilon > \mathcal{E}_L^{(\frac{k}{2})}$$

Using simple replacements, we can deduce that $\mathcal{E}_L^{(\frac{k}{2})}$ is also smaller than $\mathcal{E}_L$ when $x<w_0$ or $x>w_0+\cdots+w_k$. Let $\mathcal{E}_L^{(x<w_0)}$ denote the value of $\mathcal{E}_L$ for some $x<w_0$. For some $x<w_0$, all clauses in Eq. (9) have negative values. Recalling that $|x|=-x$ when $x<0$, we have:

$$\begin{aligned}
\mathcal{E}_L^{(x<w_0)} &= n_L \times (w_0 - x + w_0 + w_1 - x + \cdots + w_0 + w_1 + \cdots + w_k - x) \\
&= n_L \times \left(\left(\sum_{j=0}^{k} (k+1-j) w_j \right) - (k+1) \times x \right) \\
&\geq n_L \times \left(\sum_{j=1}^{k} (k+1-j) w_j \right), \text{ as } 0 \le x < w_0 \\
&= \mathcal{E}_L^{(0)} \text{ (see the proof of Lemma 4)} \\
&> \mathcal{E}_L^{(\frac{k}{2})}
\end{aligned}$$

The other case ($x > w_0 + \cdots + w_k$) can be handled analogously. □

Lemma 6 *The optimal value of $\mathcal{E}_R$ is obtained when $y = w_{\frac{k}{2}+1} + w_{\frac{k}{2}+2} + \cdots + w_{k+1}$.*

Proof: By symmetry and using Lemma 4 and Lemma 5. □

We now derive the following theorem, which states that the optimal way of contracting an entire subpath is by distributing the left and right halves of the edges in the subpath to the left and right neighbours, respectively.

Theorem 2 *Let $P' \subseteq P$ be a contiguous subpath of P (a weighted path on n vertices) consisting of k edges $\{e_1, \ldots, e_k\}$, and let e_0 and e_{k+1} be the left and right neighbouring edges of P' respectively. Furthermore, let $w_i = w(e_i)$ $\forall i \in \{0, \ldots, k+1\}$. The optimal error for contracting P' is obtained by setting $x = w_0 + w_1 + \cdots + w_{\frac{k}{2}}$ and $y = w_{\frac{k}{2}+1} + w_{\frac{k}{2}+2} + \cdots + w_{k+1}$, where x and y are the new edge weights of e_0 and e_{k+1} respectively (see Figure 8). If P' has no left neighbour (e_0 does not exist), the optimal error can be achieved by setting $y = w_{\frac{k}{2}+1} + w_{\frac{k}{2}+2} + \cdots + w_{k+1}$. If P' has no right neighbour (e_{k+1} does not exist), the optimal error can be achieved by setting $x = w_0 + w_1 + \cdots + w_{\frac{k}{2}}$. Finally, if P' has neither a left nor a right neighbour, the optimal error can be achieved by simply contracting P' and no further modifications (weight changes) are required.*

Proof: The case with both neighbours existing is immediate from Lemma 5, Lemma 6, Eq. (8), and the fact that $\mathcal{E}_{LR} = 0$ when $x = w_0 + w_1 + \cdots + w_{\frac{k}{2}}$ and $y = w_{\frac{k}{2}+1} + w_{\frac{k}{2}+2} + \cdots + w_{k+1}$. If P' has no left neighbour (e_0 does not exist), we have $n_L = 0$ and consequently $\mathcal{E}_{LR} = E_L = 0$. It follows that $\mathcal{E} = \mathcal{E}_R$ whose optimal value is obtained by setting $y = w_{\frac{k}{2}+1} + w_{\frac{k}{2}+2} + \cdots + w_{k+1}$ using Lemma 6. The other cases can be shown analogously.

To prove that it is sufficient to alter only the immediate neighbouring edges of P', we only provide a sketch to avoid repetition. The idea is very similar to the proof of Lemma 2 and Lemma 3. Suppose we have any arbitrary weight redistribution, which we transform to the one provided in this theorem. Let u be some vertex in V_L (as in Figure 8). In the original redistribution, let x be the length of the shortest path from u to the super vertex $v^* = \{v_2, v_3, \ldots, v_{k+2}\}$ in P' (Figure 8-(b)). It is easy to see that in the original distribution, the error between u and all of the vertices in v^* is equal to:

$$\mathcal{E}_1 = |x - w_0| + \cdots + |w - w_0 - \cdots - w_k|$$

For a fixed x, $\mathcal{E}_1$ corresponds to $\frac{\mathcal{E}_L}{n_L}$ (Eq. (9)). It is easy to see that in the new redistribution, the error between u and all vertices in v^* is equal to $\frac{\mathcal{E}_L^{(\frac{k}{2})}}{n_L}$. Therefore, using Lemma 5, we know that $\frac{\mathcal{E}_L^{(\frac{k}{2})}}{n_L} - \frac{\mathcal{E}_L}{n_L} \leq 0$ for any x, and this change in the weight redistribution cannot worsen the error associated with any $u \in V_L$. Other cases can be handled analogously. □

3.4 Merging a Set of Independent Edges

We now generalize the results of Section 3.1 by proving the correctness of Algorithm 1 for merging any set of independent edges. The proof of correctness consists of the following lemma and theorem, which are similar to Lemma 2 and Theorem 1, respectively.

Lemma 7 *For merging a set of independent edges E_m from a path on n vertices P_n, there exists an optimal redistribution in which for each $e \in E_m$, only its left neighbouring edge is marked. If $e' \in E_m$ is the leftmost edge on P_n, then this optimal solution is obtained by marking the left neighbouring edge of all edges in E_m except for e'.*

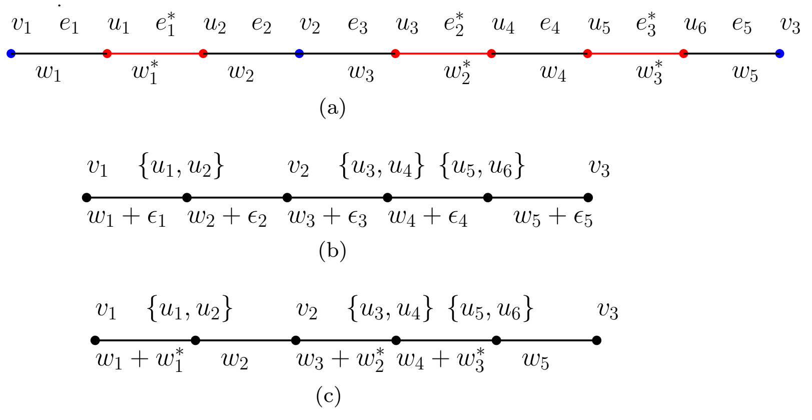

v_1 e_1 u_1 e_1^* u_2 e_2 v_2 e_3 u_3 e_2^* u_4 e_4 u_5 e_3^* u_6 e_5 v_3
w_1 w_1^* w_2 w_3 w_2^* w_4 w_3^* w_5
(a)
v_1 $\{u_1, u_2\}$ v_2 $\{u_3, u_4\}$ $\{u_5, u_6\}$ v_3
$w_1 + \epsilon_1$ $w_2 + \epsilon_2$ $w_3 + \epsilon_3$ $w_4 + \epsilon_4$ $w_5 + \epsilon_5$
(b)
v_1 $\{u_1, u_2\}$ v_2 $\{u_3, u_4\}$ $\{u_5, u_6\}$ v_3
$w_1 + w_1^*$ w_2 $w_3 + w_2^*$ $w_4 + w_3^*$ w_5
(c)

Figure 9: The figure used in the proof of Lemma 7. (a) The original graph. The vertices and edges in V_m and E_m are depicted in red and the vertices in $\overline{V_m}$ are depicted in blue. (b) An arbitrary weight redistribution which assigns $w'(e_i) = w(e_i) + \epsilon_i$ to every edge $e_i \in \overline{E_m} = E - E_m$ (c) Another weight redistribution that only marks the left neighbouring edge of each edge in E_m whose associated error is no worse than the one depicted in (b).

Proof: The proof is similar to the proof of Lemma 2, and we will provide a sketch using Figure 9. In Figure 9, the edges in E_m and the vertices in V_m are highlighted in red, and the vertices in $\overline{V_m}$ are depicted in blue. We assign an ordering to the vertices (of V_m and $\overline{V_m}$) and the edges (of E_m and $\overline{E_m}$) from left to right, as illustrated in Figure 9. Let v_i and u_j be the i-th and the j-th vertex in $\overline{V_m}$ and V_m respectively according to this ordering. Similarly, let e_i and e_j^* be the i-th and the j-th edge in $\overline{E_m} = E - E_m$ and E_m respectively. For convenience, we denote $w(e_i)$ and $w(e_i^*)$ by w_i and w_i^* respectively. Figure 9-(b) depicts some arbitrary weight redistribution in which the new weight of each edge e_i is set to $w(e_i) + \epsilon_i$. We shall show that the error associated with the weight redistribution of Figure 9-(c) (in which the left neighbours of E_m are marked) is no worse than that of Figure 9-(b). We again assume that all edges in E_m have left neighbours. First, observe how this new weight redistribution removes any error between the vertices in $\overline{V_m}$. For instance, in the path of Figure 9-(c), the shortest path value between $v_1, v_3 \in \overline{V_m}$ is the same as the one in the original path (Figure 9-(a)). Therefore, it suffices to study only the error between all pairs of vertices (u, v), $u \in V_m$, $v \in \overline{V_m}$. Using our ordering of edges, let $e_k^* = (u_j, u_{j+1}) \in E_m$ and let $v_i \in \overline{V_m}$ be a vertex to the left of e_k^* (we will explain how the other case can be handled analogously). Continuing with our example of Figure 9, let $e_k^* = e_3^* = (u_5, u_6)$ and $v_i = v_1$. Observe how between v_i (v_1 in Figure 9-(a)) and u_{j+1} (u_6 in Figure 9-(c)), there exists no error in the new redistribution as they have equal shortest path values in the original graph (Figure 9-(a)) and the new distribution (Figure 9-(c)). We show that, going from the distribution of Figure 9-(b) to the one in Figure 9-(c), any increase in the error between v_i and the left endpoint of e_k^* (u_j) can be nullified by the decrease in the error between v_i and u_{j+1}. The case where v_i is located on the right of e_k^* can be handled similarly.
For any $E' \subseteq E$ we define the following quantities:

$$\mathcal{W}(E') = \sum_{e \in E' \cap \overline{E_m}} w(e), \;\; \mathcal{W}^*(E') = \sum_{e \in E' \cap E_m} w(e), \;\; \mathcal{W}'(E') = \sum_{e_i \in \overline{E_m} \cap E'} \epsilon_i$$

where $\mathcal{W}'(E')$ denotes the sum of all ϵ_i's in the distribution of Figure 9-(b). Let $\pi_{v,u}$, $\pi'_{v,u}$, and $\pi''_{v,u}$ denote the shortest path values between v and u in the original graph (Figure 9-(a)), the first redistribution (Figure 9-(b)), and the second redistribution (Figure 9-(c)) respectively. Moreover, let $E^{(u,v)}$ denote the set of edges on the unique shortest path from u to v. We have:

$$\pi_{v_i,u_j} = \mathcal{W}(E^{(v_i,u_j)}) + \mathcal{W}^*(E^{(v_i,u_j)}) \tag{12}$$

$$\pi'_{v_i,u_j} = \mathcal{W}(E^{(v_i,u_j)}) + \mathcal{W}'(E^{(v_i,u_j)}) \tag{13}$$

$$\pi''_{v_i,u_j} = \mathcal{W}(E^{(v_i,u_j)}) + \mathcal{W}^*(E^{(v_i,u_j)}) + w_k^* \tag{14}$$

We provide some examples of these quantities in Example 1 for better readability. Note that:

$$\pi_{v_i,u_{j+1}} = \pi_{v_i,u_j} + w_k^*,\ \pi'_{v_i,u_j} = \pi'_{v_i,u_{j+1}},\text{ and } \pi''_{v_i,u_j} = \pi''_{v_i,u_{j+1}} \tag{15}$$

The error between v_i and u_{j+1} in the redistribution of Figure 9-(b) is:

$$\mathcal{E}_1^{v_i,u_{j+1}} = \left|\pi_{v_i,u_{j+1}} - \pi'_{v_i,u_{j+1}}\right| = \left|\pi_{v_i,u_j} + w_k^* - \pi'_{v_i,u_j}\right| = \left|w_k^* + \mathcal{W}^*(E^{(v_i,u_j)}) - \mathcal{W}'(E^{(v_i,u_j)})\right| \tag{16}$$

As mentioned before, the error between v_i and u_{j+1} in the weight redistribution of Figure 9-(c) is equal to zero:

$$\mathcal{E}_2^{v_i,u_{j+1}} = 0 \tag{17}$$

Therefore, transforming Figure 9-(b) into Figure 9-(c) changes the error between v_i to u_{j+1} by:

$$\Delta_{v_i,u_{j+1}} = \mathcal{E}_2^{v_i,u_{j+1}} - \mathcal{E}_1^{v_i,u_{j+1}} = -\left|w_k^* + \mathcal{W}^*(E^{(v_i,u_j)}) - \mathcal{W}'(E^{(v_i,u_j)})\right| \tag{18}$$

The error between v_i and u_j in the redistribution of Figure 9-(b) is:

$$\mathcal{E}_1^{v_i,u_j} = \left|\pi_{v_i,u_j} - \pi'_{v_i,u_j}\right| = \left|\pi'_{v_i,u_j} - \pi_{v_i,u_j}\right| = \left|\mathcal{W}'(E^{(v_i,u_j)}) - \mathcal{W}^*(E^{(v_i,u_j)})\right| \tag{19}$$

The error between v_i and u_j in the weight redistribution of Figure 9-(c) is equal to:

$$\mathcal{E}_2^{v_i,u_j} = \left|\pi''_{v_i,u_j} - \pi_{v_i,u_j}\right| = |w_k^*| \tag{20}$$

Transforming Figure 9-(b) into Figure 9-(c) changes the error between v_i to u_j by:

$$\Delta_{v_i,u_j} = \mathcal{E}_2^{v_i,u_j} - \mathcal{E}_1^{v_i,u_j} = |w_k^*| - \left|\mathcal{W}'(E^{(v_i,u_j)}) - \mathcal{W}^*(E^{(v_i,u_j)})\right| \leq \left|w_k^* - \mathcal{W}'(E^{(v_i,u_j)}) + \mathcal{W}^*(E^{(v_i,u_j)})\right| \tag{21}$$

using Corollary 1. Therefore, going from the first redistribution to the second one changes the error between the endpoints of $e_k^* = (u_j, u_{j+1})$ and v_i by:

$$\Delta_{v_i,u_j} + \Delta_{v_i,u_{j+1}} \leq \left|w_k^* - \mathcal{W}'(E^{(v_i,u_j)}) + \mathcal{W}^*(E^{(v_i,u_j)})\right| - \left|w_k^* + \mathcal{W}^*(E^{(v_i,u_j)}) - \mathcal{W}'(E^{(v_i,u_j)})\right| = 0$$

Since each e_k^* edge in E_m has exactly two endpoints, this concludes the proof for the first case (v_i is on the left of e_k^*). The other case can be handled analogously. □

Example 1 *Returning to our example of Lemma 7 and Figure 9, let* $e_k^* = (u_j, u_{j+1}) = e_3^* = (u_5, u_6)$*, and* $v_i = v_1$*. Then:*

- $E^{(v_1,u_5)} = \{e_1, e_1^*, e_2, e_3, e_2^*, e_4\}$
- $\mathcal{W}(E^{(v_1,u_5)}) = w_1 + w_2 + w_3 + w_4$
- $\mathcal{W}^*(E^{(v_1,u_5)}) = w_1^* + w_2^*$
- $\mathcal{W}'(E^{(v_1,u_5)}) = \epsilon_1 + \epsilon_2 + \epsilon_2 + \epsilon_3 + \epsilon_4$

Theorem 3 *Let* $|\Delta E|$ *be the optimal error resulting from merging a set of* k *independent edges* $e_1, e_2, \ldots, e_k$ *with respective weights* $w_1^*, w_2^*, \ldots, w_k^*$ *from a path on* n *vertices* P_n*. Let* (u_{2i-1}, u_{2i}) *be the endpoints of* $e_i \in E_m, 1 \leq i \leq k$*. Furthermore, let* $V_m = \{u_1, \ldots, u_{2k}\}$ *and* $\overline{V_m} = V - V_m$*. We have* $|\Delta E| = |\overline{V_m}|(w_1^* + \cdots + w_k^*) = (n-2k)(w_1^* + \cdots + w_k^*)$*. This optimal value can be achieved by marking the left neighbour of each edge in* E_m *after contraction. If the leftmost edge in* E_m *has no left neighbour, the optimal error can be achieved by marking the left neighbours of all other edges in* E_m*.*

Proof: Let $w' : E \to \mathbb{R}_{\geq 0}$ be the weight redistribution that marks the left neighbouring edge (if any) of each edge in E_m (Figure 9-(c)). That w' is optimal follows directly from Lemma 7. We now prove the error associated with w'.

Since the edges in E_m induce a matching on P_n, $|\overline{V_m}| = n - 2|E_m| = n - 2k$. Recall from the proof of Lemma 7 that in w', there exists no error between two vertices $v_1, v_2 \in \overline{V_m}$. Let us fix some $e_k^* \in E_m$. Using the proof of Lemma 7, we know that each vertex $v_i \in \overline{V_m}$ induces an error of w_k^* with exactly one endpoint of e_k^* (and no error with the other endpoint). Summing over all vertices $v_i \in \overline{V_m}$, we get that each edge $e_k^* \in E_m$ accumulates a total of $(n-2k)w_k^*$ in error. Summing again over all edges $e_k^* \in E_m$ yields the desired bound. □

4 Graph Compression for Trees

In this section, we study the problem of distance-preserving graph compression for weighted trees. Precisely, we study a relevant problem, referred to as *the marking problem*, for a tree $T = (V, E)$, $|V| = n$, and weight function $w : E \to \mathbb{R}_{\geq 0}$.

The remainder of this section is organized as follows. In Section 4.1, we formally define the marking problem. The adaptation of the error function (Eq. (1)) to the marking problem is thoroughly explained in Section 4.2. As a warm-up, we study a special case of the marking problem in Section 4.3, after which we generalize the results in Section 4.4 and present a linear-time algorithm for solving the marking problem in Algorithm 2. As the final component of this section, we thoroughly study the difference between the marking problem (Definition 9) and the fractional marking problem (Definition 14) in Section 4.5.

4.1 The Marking Problem for a Single Edge

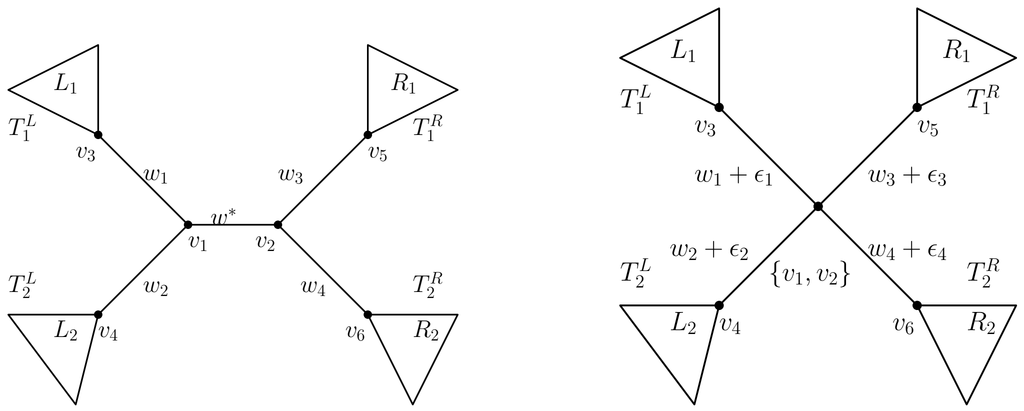
As seen in Section 3, for merging a single edge in a weighted path, marking one of the neighbouring edges produces the optimal amount of error. An important question is how to generalize this result to solve the same problem for weighted trees. We formally state the marking problem as:

Definition 9 ***The Marking Problem for Weighted Trees:*** *Given a contracted edge e^* in a weighted tree T, what subset of the neighbouring edges of e^* should we mark such that the error value of Eq. (1) is minimized over all such possible subsets?*

An example of the marking problem is depicted in Figure 10-(a), where edge e^* with weight w^* is contracted. As shown in Figure 10-(b), in the marking problem, the goal is to mark a subset of the neighbouring edges of e^*, by setting the new weight of each marked edge e_i to $w'(e_i) = w(e_i) + \epsilon_i, \epsilon_i \in \{0, w^*\}$, in a way that minimizes the error function of Eq. (1) over all such possible subsets. Note that the fractional case (when the weight of each marked edge e_i is set to $w'(e_i) = w(e_i) + \epsilon_i, \epsilon_i \in [0, w^*]$) is thoroughly studied in Section 4.5.

In the tree of Figure 10-(a), e^* has four neighbouring edges, namely $e_1 = (v_1, v_3)$, $e_2 = (v_1, v_4)$, $e_3 = (v_2, v_5)$, and $e_4 = (v_2, v_6)$. Different subsets of these neighbouring edges can be marked, for instance, in Figure 11-(a), $\{e_1, e_2\}$ is marked. In the remainder of this section, we may refer to each of these marked subsets as a *marking* for simplicity. For example, in Figure 11-(c), $\{e_1, e_3\}$ is a marking. An optimal marking is one that minimizes the error function of Eq. (1) over all possible markings.

Since for merging an edge in a weighted path marking one of the neighbouring edges gives the optimal amount of error, our intuition tells us that in a weighted tree, we have to mark all neighbouring edges on one side of the contracted edge e^*. As we shall show later, this intuition, though not completely correct, is optimal for specific kinds of input. To study the marking problem, we first present some definitions and observations using Figure 10 and Figure 11 as our running examples. We assume the tree is laid out in the plane and e^* (the edge to be merged) is horizontal. This assumption will simplify the description of our results.



(a) Before merging e^* (b) After merging e^*

Figure 10: The figure used in Section 4.1 for defining the marking problem. We denote by $T_i^L, i \in \{1, 2\}$, and $T_j^R, j \in \{1, 2\}$ the subtree rooted at the i-th edge to the left and the j-th edge to the right respectively. Moreover, $L_i = |\{v|v \in T_i^L\}|$ and $R_j = |\{v|v \in T_j^R\}|$ denote the number of vertices in each subtree.

Definition 10 *Let $T = (V, E)$ be a weighted tree with non-negative weights, and let $e^* = (v_1, v_2)$ be the merged edge with weight w^*, $V_m = \{v_1, v_2\}$, and $\overline{V_m} = V - V_m$. We denote by $\mathcal{L}$ the number* *of subtrees to the left of v_1 and by $\mathcal{R}$ the number of subtrees to the right of v_2. More formally, let $E' = E - e^*$. We have:*

$$V_L = \{u | (u, v_1) \in E'\}, \mathcal{L} = |V_L|$$

$$V_R = \{w | (v_2, w) \in E'\}, \mathcal{R} = |V_R|$$

For instance, in the tree of Figure 10, we have $V_L = \{v_3, v_4\}$ and $V_R = \{v_5, v_6\}$ and therefore $\mathcal{L} = \mathcal{R} = 2$.
Given $e^* = (v_1, v_2)$ in T, $T - \{v_1, v_2\}$ is a forest $\mathcal{F}$, the components of which are used in our analyses and defined as follows:

Definition 11 *Let T, $e^* = (v_1, v_2)$, V_L and V_R be as defined in Definition 10. Let $\mathcal{F}$ be the forest $T - \{v_1, v_2\}$. Furthermore, assume that the connected components of $\mathcal{F}$ are rooted at the vertices of V_L or V_R, and let C_L and C_R be the sets of components of $\mathcal{F}$ rooted at the vertices of V_L and V_R respectively. Then, we denote by $T_i^L, i \in \{1, \ldots, \mathcal{L}\}$ the i-th member of C_L, and by $T_j^R, j \in \{1, \ldots, \mathcal{R}\}$ the j-th member of C_R, given some arbitrary ordering on the members of C_L and C_R.*

In the tree of Figure 10, $\mathcal{L} = 2$, and C_L has two members (the subtrees rooted at v_3 and v_4). Given some arbitrary ordering on the members of C_L, T_1^L is the subtree rooted at v_3.
We also formally define the cardinality of the subtrees of Definition 11 as follows:

Definition 12 *Let $T_i^L, i \in \{1, \ldots, \mathcal{L}\}$ and $T_j^R, j \in \{1, \ldots, \mathcal{R}\}$ be as defined in Definition 11. We have $L_i = |\{v | v \in T_i^L\}|$ and $R_j = |\{v | v \in T_j^R\}|$. We refer to L_i as the cardinality of the i-th edge on the left and R_j as the cardinality of the j-th edge on the right.*

A few examples of marking the edges of Figure 10 are provided in Figure 11-(a) to Figure 11-(c). In Figure 11-(a) and Figure 11-(b), all edges on one side of e^* are marked, and in Figure 11-(c), a subset of edges from both sides are marked. Marking an edge could both increase and decrease the total amount of error. Before proceeding with the remainder of this section, we note the following lemma to justify our focus on minimizing the error between all pairs of vertices in $\overline{V_m}$.

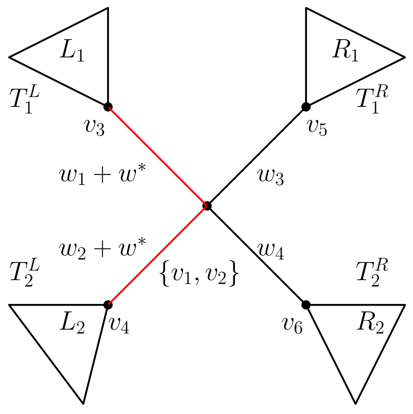
Lemma 8 *(See Figure 10) Let $e^* = (v_1, v_2)$ be the single merged edge in a weighted tree $T = (V, E)$, and let $\overline{V_m} = V - \{v_1, v_2\}$. Then, as long as every neighbouring edge of e^* is either marked or unmarked, the error between some vertex $u \in \overline{V_m}$ and the vertices in $\{v_1, v_2\}$ is minimized.*

Proof: This lemma is a direct result of Lemma 1 and Theorem 1. Let us fix some vertex $u \in T_2^L$ (see Figure 10-(b)), the error between u and the endpoints of e^*, v_1 and v_2, can be formulated as:

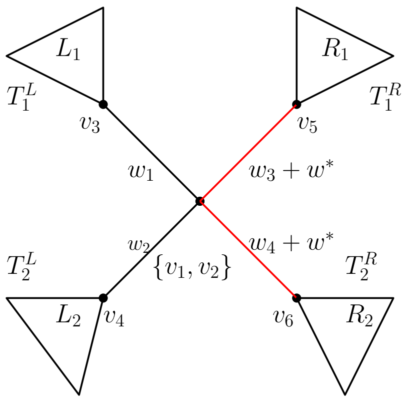
$$|\Delta E|' = \underbrace{|w_2 - (w_2 + \epsilon_2)|}_{\text{between } u \text{ and } v_1} + \underbrace{|w_2 + w^* - (w_2 + \epsilon_2)|}_{\text{between } u \text{ and } v_2} = |\epsilon_2| + |w^* - \epsilon_2| = |\epsilon_2| + |\epsilon_2 - w^*|$$

Using Lemma 1, we have $|\Delta E|' \geq w^*$, and $|\Delta E|' = w^*$ for $0 \leq \epsilon_2 \leq w^*$. Therefore, when (v_1, v_4) is either marked or unmarked, we have $\epsilon_2 \in \{0, w^*\}$, which satisfies the desired conditions. This analysis applies to all nodes $u \in \overline{V_m}$, thus the lemma follows. □

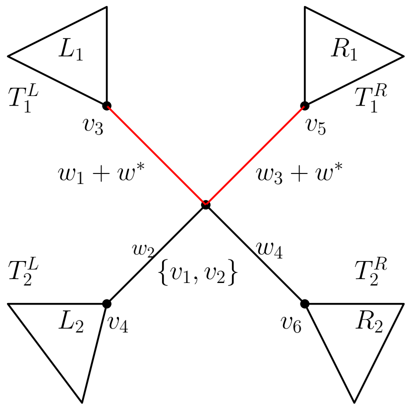
In the remainder of this section, we therefore only focus on minimizing the error between all pairs of vertices $u_1, u_2 \in \overline{V_m}$, because by the definition of the marking problem (Definition 9), the conditions of Lemma 8 are automatically satisfied.



(a) Both edges on the left are marked.



(b) Both edges on the right are marked.



(c) A subset of edges from both sides is marked.

Figure 11: The figure used in Section 4.2 for formulating the marking error, the marked edges are highlighted in red. We denote by $T_i^L, i \in \{1,2\}$, and $T_j^R, j \in \{1,2\}$ the subtree rooted at the i-th edge to the left and the j-th edge to the right respectively. Moreover, $L_i = |\{v|v \in T_i^L\}|$ and $R_j = |\{v|v \in T_j^R\}|$ denote the number of vertices in each subtree.

4.2 Formulating the Error

This section formally explains how marking a set of edges affects the error function. Using Figure 11, we first present some examples, which we generalize later in Observation 1. Throughout this section, we may sometimes refer to this error as *units of error*, where each unit is equal to w^*.

Example 2 *The error between v_3 and v_4 in Figure 11-(a) is equal to $|w_1+w^*+w_2+w^*-w_1-w_2| = 2w^*$. In the original graph (Figure 10-(a)), e^* does not appear on the unique path between v_3 and v_4, while in the modified graph (Figure 11-(a)), the weight of e^* appears twice. In the marking of Figure 11-(a), the total amount of error between all pairs of vertices $u_1 \in T_1^L, u_2 \in T_2^L$ is $L_1 \times L_2 \times 2w^*$.*

Example 3 *The error between v_3 and v_5 in Figure 11-(c) is $|w_1+w^*+w_3+w^*-w_1-w^*-w_3| = w^*$.*

Because in the original graph (Figure 10-(a)), e^* appears only once on the unique path from v_3 to v_5, while in the modified graph (Figure 11-(c)), the weight of e^* appears twice. The total amount of error between all pairs of vertices $u_1 \in T_1^L, u_2 \in T_1^R$ is $L_1 \times R_1 \times w^*$.

Example 4 *In Figure 11-(c), the error between v_5 and v_6 is $|w_3 + w^* + w_4 - w_3 - w_4| = w^*$. The total amount of error between all pairs of vertices $u_1 \in T_1^R$ and $u_2 \in T_2^R$ is $R_1 \times R_2 \times w^*$.*

Example 5 *In Figure 11-(c), the total amount of error between all pairs of vertices $u_1 \in T_1^L$ and $u_2 \in T_2^L$ is $L_1 \times L_2 \times w^*$.*

Example 6 *In Figure 11-(a), the error between v_3 and v_5 is $|w_1 + w^* + w_3 - w_1 - w^* - w_3| = 0$. The length of the unique path between v_3 and v_5 does not change compared with Figure 10-(a).*

Observation 1 *Between the vertices of two edges (vertices belonging to the subtree rooted at that edge) adjacent to the endpoints of e^*, there might exist some error. We classify this observation into the following cases:*

1. Let T_i^L and T_j^L be the subtrees adjacent to two distinct marked edges on the left. Then, the total amount of error between all pairs of vertices $u_1 \in T_i^L, u_2 \in T_j^L$ is $L_i \times L_j \times 2w^*$ (see Example 2).

2. Let T_i^R and T_j^R be the subtrees adjacent to two marked edges on the right. Then, the total amount of error between all pairs of vertices $u_1 \in T_i^R$ and $u_2 \in T_j^R$ is $R_i \times R_j \times 2w^*$.

3. Let T_i^L and T_j^R be the subtrees adjacent to two marked edges on the left and right, respectively. Then, the total amount of error between all pairs of vertices $u_1 \in T_i^L$ and $u_2 \in T_j^R$ is $L_i \times R_j \times w^*$ (see Example 3).

4. Let T_i^R and T_j^R be the subtrees adjacent to a marked edge and an unmarked edge on the right, respectively. Then, the total amount of error between all pairs of vertices $u_1 \in T_i^R$ and $u_2 \in T_j^R$ is $R_i \times R_j \times w^*$ (see Example 4).

5. Let T_i^L and T_j^L be the subtrees adjacent to a marked edge and an unmarked edge on the left, respectively. Then, the total amount of error between all pairs of vertices $u_1 \in T_i^L$ and $u_2 \in T_j^L$ is $L_i \times L_j \times w^*$ (see Example 5).

6. Let T_i^L be the subtree adjacent to a marked edge on the left, and T_j^R be the subtree adjacent to an unmarked edge on the right. Then, the total amount of error between all pairs of vertices $u_1 \in T_i^L$ and $u_2 \in T_j^R$ is equal to zero (see Example 6).

7. Let T_i^L be the subtree adjacent to an unmarked edge on the left, and T_j^R be the subtree adjacent to a marked edge on the right. Then, the total amount of error between all pairs of vertices $u_1 \in T_i^L$ and $u_2 \in T_j^R$ is equal to zero.

4.3 Equal-Sized Subtrees

We now investigate a special case where each subtree on the left has n_L vertices and each subtree on the right has n_R vertices, i.e., $L_i = n_L$, $1 \leq i \leq \mathcal{L}$, and $R_i = n_R$, $1 \leq i \leq \mathcal{R}$. Recall that every merged edge has two sides, left and right, one of which is designated as the *preferable side*.

A given side is preferable if it produces a smaller amount of error when fully marked compared to its fully marked counterpart. For example, if the left side is preferable, we have:

$$n_L^2 \times \mathcal{L}(\mathcal{L}-1) \leq n_R^2 \times \mathcal{R}(\mathcal{R}-1) \tag{22}$$

The above inequality compares the error between the marking with the left side fully marked and the right side fully unmarked (Figure 11-(a)), and the opposite marking with the right side fully marked and the left side fully unmarked (Figure 11-(b)). In the first marking, there exists no error between the left and the right sides (Observation 1, Case 6), but there are $\binom{\mathcal{L}}{2}$ distinct pairs of marked edges on the left, each inducing an error of $n_L \times n_L \times 2w^*$ (Observation 1, Case 1). Therefore, the total amount of error for the first marking is equal to $\binom{\mathcal{L}}{2} \times n_L \times n_L \times 2w^* = n_L^2 \times \mathcal{L}(\mathcal{L}-1) \times w^*$. The other marking can be analyzed analogously. Note that in the remainder of this section, we drop w^* from each error term, and each error term counts the error units, where each unit is equal to w^*. Therefore, all quantities are implicitly multiplied by w^* in the remainder of this section.

The following lemma states that, for a contracted edge e^* that has equal-sized subtrees on each side, the optimal solution is obtained by marking all edges on the preferable side of e^* and leaving the other side completely unmarked.

Lemma 9 *Given a merged edge e^* (in a weighted tree) with two sides left and right, such that the subtrees on each side have equal sizes, the optimal marking is obtained if one side (the preferable side) is fully marked and the other side is fully unmarked.*

Proof: By contradiction. This lemma assumes each subtree on the left and right side has n_L and n_R vertices respectively, i.e. $L_i = n_L,\ 1 \leq i \leq \mathcal{L}$, and $R_i = n_R,\ 1 \leq i \leq \mathcal{R}$. Without loss of generality, we assume the left side is preferable throughout this proof. Therefore, we have:

$$n_L^2 \times \mathcal{L}(\mathcal{L}-1) \leq n_R^2 \times \mathcal{R}(\mathcal{R}-1)$$

Let i and j denote the number of marked edges on the left and right, respectively. We define two functions, MARK_LEFT, which marks one of the edges on the left, and UNMARK_RIGHT, which unmarks one edge on the right. We will show that for all values $i < \mathcal{L}$ or $j > 0$, one can achieve smaller error values by applying a series of MARK_LEFT's and UNMARK_RIGHT's and ending up at $i = \mathcal{L}$ and $j = 0$, as desired. For a function $f \in \mathcal{F} = \{\text{MARK\_LEFT}, \text{UNMARK\_RIGHT}\}$, we define $\Delta(f)$ as the amount of change in the error value after applying f to the tree. Since we are interested in decreasing the error value using the functions in $\mathcal{F}$, in this proof, we will look for conditions under which $\Delta(\text{MARK\_LEFT}) \leq 0$ and $\Delta(\text{UNMARK\_RIGHT}) \leq 0$.

We begin by investigating MARK_LEFT. Note that this function sets $i \leftarrow i+1$ and $j \leftarrow j$. We observe the following:

1. Because we are marking a new edge, the total amount of error between the marked edges on the left changes by:
$$n_L^2 \times 2\left(\binom{i+1}{2} - \binom{i}{2}\right) = n_L^2 \times 2i$$

2. The total amount of error between the unmarked edges and the marked ones on the left changes by:
$$n_L^2 \times ((i+1)(\mathcal{L}-i-1) - i(\mathcal{L}-i)) = n_L^2 \times (\mathcal{L}-2i-1)$$

3. The total amount of error between the marked edges on the left and right changes by:
$$n_L n_R((i+1)j - ij) = n_L n_R \times j$$

4. The total amount of error between the unmarked edges on the left and right changes by:

$$n_L n_R \times ((\mathcal{L} - i - 1)(\mathcal{R} - j) - (\mathcal{L} - i)(\mathcal{R} - j)) = n_L n_R \times (j - \mathcal{R})$$

Therefore, Δ(MARK_LEFT) is equal to:

$$\begin{aligned}\Delta(\text{MARK\_LEFT}) &= n_L^2 \times (2i + \mathcal{L} - 2i - 1) + n_L n_R(j + j - \mathcal{R}) \\ &= n_L^2 \times (\mathcal{L} - 1) + n_L n_R(2j - \mathcal{R})\end{aligned}$$

Since we are looking for conditions under which $\Delta(\text{MARK\_LEFT}) \leq 0$, we have:

$$\Delta(\text{MARK\_LEFT}) \leq 0 \rightarrow n_L^2 \times (\mathcal{L} - 1) + n_L n_R(2j - \mathcal{R}) \quad \leq 0$$

Therefore,

$$\Delta(\text{MARK\_LEFT}) \leq 0 \text{ if } n_L \times (\mathcal{L} - 1) \leq n_R(\mathcal{R} - 2j)$$

Rearranging the terms, we have

$$j \leq \frac{\mathcal{R}}{2} + \frac{n_L(1 - \mathcal{L})}{2n_R} \tag{23}$$

A similar reasoning can be used for UNMARK_RIGHT. This function sets $i \leftarrow i$ and $j \leftarrow j-1$. We have:

1. The total amount of error between the marked edges on the right changes by:

$$n_R^2 \times 2\left(\binom{j-1}{2} - \binom{j}{2}\right) = n_R^2 \times (-2(j-1))$$

2. The total amount of error between the unmarked edges and the marked ones on the right changes by:

$$n_R^2 \times ((j-1)(\mathcal{R} - j + 1) - j(\mathcal{R} - j)) = n_R^2 \times (2j - \mathcal{R} - 1)$$

3. The total amount of error between the marked edges on the left and right changes by:

$$n_L n_R(i(j-1) - ij) = n_L n_R \times (-i)$$

4. The total amount of error between the unmarked edges on the left and right changes by:

$$n_L n_R \times ((\mathcal{L} - i)(\mathcal{R} - j + 1) - (\mathcal{L} - i)(\mathcal{R} - j)) = n_L n_R \times (\mathcal{L} - i)$$

Thus, we have:

$$\begin{aligned}\Delta(\text{UNMARK\_RIGHT}) &= n_R^2(-2(j-1) + 2j - \mathcal{R} - 1) + n_L n_R(\mathcal{L} - i - i) \\ &= n_R^2(1 - \mathcal{R}) + n_L n_R(\mathcal{L} - 2i)\end{aligned}$$

and

$$\Delta(\text{UNMARK\_RIGHT}) \leq 0 \text{ if } n_L(\mathcal{L} - 2i) \leq n_R(\mathcal{R} - 1)$$

Rearranging the terms, we have

$$i \geq \frac{\mathcal{L}}{2} + \frac{n_R(1 - \mathcal{R})}{2n_L} \tag{24}$$

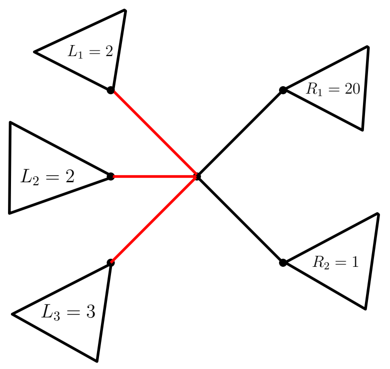
We conclude the proof by stating that whenever $i < \mathcal{L}$ or $j > 0$, one can achieve smaller error values by applying a series of MARK_LEFT's and UNMARK_RIGHT's and ending up at $i = \mathcal{L}$ and $j = 0$. When $j \leq \frac{\mathcal{R}}{2} + \frac{n_L(1-\mathcal{L})}{2n_R}$, Eq. (23) is satisfied. Therefore, we repeatedly apply MARK_LEFT until $i = \mathcal{L}$, at which point Eq. (24) is satisfied, and we repeatedly apply UNMARK_RIGHT until $j = 0$, as desired. Now suppose $j > \frac{\mathcal{R}}{2} + \frac{n_L(1-\mathcal{L})}{2n_R}$ edges are marked on the right side. If $i \geq \frac{\mathcal{L}}{2} + \frac{n_R(1-\mathcal{R})}{2n_L}$, Eq. (24) is satisfied, which allows us to repeatedly apply UNMARK_RIGHT until $j = 0$, at which point Eq. (23) is satisfied and we repeatedly apply MARK_LEFT until $i = \mathcal{L}$, as desired.

Assume $i < \frac{\mathcal{L}}{2} + \frac{n_R(1-\mathcal{R})}{2n_L}$ and $j > \frac{\mathcal{R}}{2} + \frac{n_L(1-\mathcal{L})}{2n_R}$, and both Eq. (23) and Eq. (24) are unsatisfied. We first apply $\frac{\mathcal{L}}{2} + \frac{n_R(1-\mathcal{R})}{2n_L} - i = \frac{\mathcal{L}n_L + n_R(1-\mathcal{R})}{2n_L} - i$ MARK_LEFT's, increasing the error by $(\frac{\mathcal{L}n_L+n_R(1-\mathcal{R})}{2n_L} - i)(n_L^2 \times (\mathcal{L}-1) + n_L n_R(2j - \mathcal{R}))$, at which point $i = \frac{\mathcal{L}n_L+n_R(1-\mathcal{R})}{2n_L}$ and $\Delta(\text{UNMARK\_RIGHT}) = 0$. Therefore, we set $j \leftarrow 0$ without changing the error (since $\Delta(\text{UNMARK\_RIGHT}) = 0$), and then we apply $\mathcal{L} - (\frac{\mathcal{L}n_L+n_R(1-\mathcal{R})}{2n_L}) = \frac{\mathcal{L}n_L - n_R(1-\mathcal{R})}{2n_L}$ MARK_LEFT's until $i = \mathcal{L}$, as desired. We now show that this sequence of MARK_LEFT's and UNMARK_RIGHT's results in an error value no worse than that of the original one:

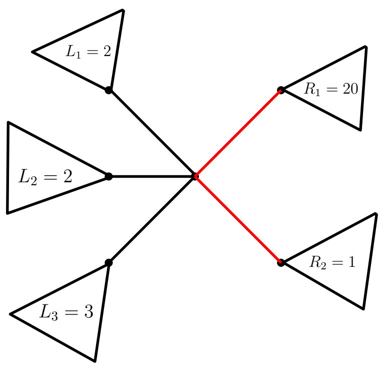
$$\begin{aligned}
\Delta = {} & \left(\underbrace{\frac{\mathcal{L}n_L + n_R(1-\mathcal{R})}{2n_L} - i}_{\leq \frac{\mathcal{L}n_L + n_R(1-\mathcal{R})}{2n_L}}\right) \underbrace{(n_L^2 \times (\mathcal{L}-1) + n_L n_R(2j - \mathcal{R}))}_{\leq n_L^2 \times (\mathcal{L}-1) + n_L n_R(\mathcal{R})} \quad + \\
& j \underbrace{\left(n_R^2(1-\mathcal{R}) + n_L n_R\left(\mathcal{L} - 2 \times \frac{\mathcal{L}n_L + n_R(1-\mathcal{R})}{2n_L}\right)\right)}_{=0} \quad + \\
& \frac{\mathcal{L}n_L - n_R(1-\mathcal{R})}{2n_L}(n_L^2 \times (\mathcal{L}-1) + n_L n_R(-\mathcal{R})) \\
\leq {} & \left(\frac{\mathcal{L}n_L + n_R(1-\mathcal{R})}{2n_L}\right)(n_L^2 \times (\mathcal{L}-1) + n_L n_R(\mathcal{R})) \quad + \\
& \frac{\mathcal{L}n_L - n_R(1-\mathcal{R})}{2n_L}(n_L^2 \times (\mathcal{L}-1) + n_L n_R(-\mathcal{R})) \\
= {} & n_L^2 \times \mathcal{L}(\mathcal{L}-1) - n_R^2 \times \mathcal{R}(\mathcal{R}-1) \\
\leq {} & 0
\end{aligned}$$

and we arrive at $i = \mathcal{L}$ and $j = 0$ while obtaining a smaller error value. □

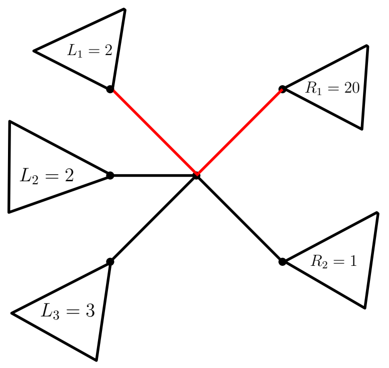
In the next section, we generalize Lemma 9 to the case in which different subtrees can have varying sizes.



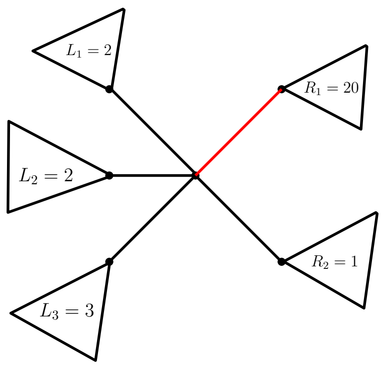
(a) A full marking of the edges on the left with error count 32.



(b) A full marking of the edges on the right with error count 40.



(c) A marking with edges marked on both sides with error count 75.



(d) The optimal solution with error count 27.

Figure 12: An example of tree compression in which the edges on each side have different-sized subtrees. The optimal solution does not have a full marking on any side. However, the optimal solution only has marked edges on one side (this is always the case as shown in Lemma 10).

4.4 Varying-Size Subtrees

As a generalization of Section 4.3, now assume the i-th subtree on the left ($1 \leq i \leq \mathcal{L}$) has L_i nodes, and the j-th subtree on the right ($1 \leq j \leq \mathcal{R}$) is of size R_j. We observe when each side has subtrees of different sizes, marking all edges on one side does not necessarily produce the optimal error. An example is depicted in Figure 12, where marking only one edge on the right produces the optimal amount of error. Although marking all edges on one side does not necessarily produce the optimal error, we observe that no optimal solution has markings on both sides, as the following lemma states. Similar to Section 4.3, we remove w^* from all calculations and expressions in this section. Therefore, all calculations in this section are implicitly multiplied by w^*.

Lemma 10 *Given a merged edge e^* (in a weighted tree) with two sides left and right, no optimal marking has marked edges on both sides.*

Proof: By contradiction. We assume there exists such an optimal marking, and we strictly improve its error by unmarking everything on one of the two sides (thus obtaining a contradiction).

Expanding the proof of Lemma 9, we define four operations, MARK_LEFT, UNMARK_LEFT, MARK_RIGHT, and UNMARK_RIGHT, for unmarking and marking edges on both ends. For a function

$$f \in \mathcal{F} = \{\text{MARK\_LEFT}, \text{UNMARK\_LEFT}, \text{MARK\_RIGHT}, \text{UNMARK\_RIGHT}\}$$

we define $\Delta(f)$ as the amount of change in the error value after applying f to the tree. Let $S_L = \sum_{i=1}^{\mathcal{L}} L_i$ and $S_R = \sum_{i=1}^{\mathcal{R}} R_i$ denote the total sum of all edge cardinalities on the left and right sides respectively. Furthermore, let S_{LM}, S_{LU}, S_{RM}, and S_{RU} denote the sum of the cardinalities of the marked and unmarked edges on the left and right sides, respectively. Note that $S_L = S_{LU} + S_{LM}$ and $S_R = S_{RU} + S_{RM}$.

First, we calculate $\Delta(\text{UNMARK\_RIGHT})$ and derive $\Delta(\text{UNMARK\_LEFT})$ by symmetry. Assume we are unmarking the i-th edge e_i on the right with cardinality R_i. We break the change in the error value down into four parts as follows:

1. The total amount of error between the marked edges on the right changes by:
$$-2 \times R_i \times (S_{RM} - R_i)$$
because between two marked edges on the right, there exist two units of error (equal to twice the weight of the merged edge e^*). Therefore, unmarking e_i relieves some of this error.

2. The total amount of error between the unmarked and the marked edges on the right changes by:
$$-R_i \times (S_{RU}) + R_i \times (S_{RM} - R_i)$$
because between a marked and an unmarked edge on the right, there exists one unit of error, and unmarking e_i relieves some error with other unmarked edges (the first part of the expression), making e_i an unmarked edge itself (the second part of the expression).

3. The total amount of error between e_i and the marked edges on the left changes by:
$$-R_i \times S_{LM}$$
because between two marked edges on the right and the left, there exists one unit of error, and unmarking e_i relieves some of this error.

4. The total amount of error between e_i and the unmarked edges on the left changes by:
$$R_i \times S_{LU}$$

Summing all four parts together, we get:

$$\Delta(\text{UNMARK\_RIGHT}) = R_i \times \Bigg(-(S_{RM} - R_i) - S_{RU} - S_{LM} + S_{LU} \Bigg) \tag{25}$$

By symmetry, we also have:

$$\Delta(\text{UNMARK\_LEFT}) = L_i \times \Bigg(-(S_{LM} - L_i) - S_{LU} - S_{RM} + S_{RU} \Bigg) \tag{26}$$

Next, we calculate $\Delta(\text{MARK\_LEFT})$ and derive $\Delta(\text{MARK\_RIGHT})$ by symmetry. Assume we are marking the i-th edge e_i on the left with cardinality L_i. We break the change in the error value into four parts:

1. The total amount of error between the marked edges on the left changes by:
$$2 \times L_i \times (S_{LM})$$
because between two marked edges on the left, there exist two units of error (equal to twice the weight of the merged edge e^*). Therefore, marking e_i introduces some error between e_i and all other marked edges on the left.

2. The total amount of error between the unmarked and the marked edges on the left changes by:
$$-L_i \times (S_{LM}) + L_i \times (S_{LU} - L_i)$$
because between a marked and an unmarked edge on the left, there exists one unit of error, and marking e_i relieves some error with all other marked edges on the left (the first part of the expression), making e_i a marked edge itself (the second part of the expression).

3. The total amount of error between e_i and the marked edges on the right changes by:
$$+L_i \times S_{RM}$$
because between two marked edges on the right and the left, there exists one unit of error, thus marking e_i introduces some error between e_i and all other marked edges on the right.

4. The total amount of error between e_i and the unmarked edges on the right changes by:
$$-L_i \times S_{RU}$$
because between two unmarked edges on the left and the right, there exists one unit of error, and marking e_i relieves some of this error.

Summing all four parts together, we get:
$$\Delta(\text{MARK\_LEFT}) = L_i \times \Big(S_{LM} + (S_{LU} - L_i) + S_{RM} - S_{RU} \Big) \tag{27}$$
By symmetry, we also have:
$$\Delta(\text{MARK\_RIGHT}) = R_i \times \Big(S_{RM} + (S_{RU} - R_i) + S_{LM} - S_{LU} \Big) \tag{28}$$
Now, we can complete the proof. For the sake of contradiction, assume that there exists an optimal marking M^* with edges marked on both sides. Therefore, we have $S_{LM} > 0$ and $S_{RM} > 0$. Without loss of generality, assume $S_{RU} \geq S_{LU}$ (see Figure 13-(a)). Using Eq. (25), we can unmark any edge on the right, say the i-th one e_i connected to R_i vertices, such that the change in the error value is equal to:
$$\Delta \quad = R_i \times \Big(\underbrace{-(S_{RM} - R_i)}_{<0} - S_{RU} \underbrace{-S_{LM}}_{<0} + S_{LU} \Big) < R_i (\underbrace{-S_{RU} + S_{LU}}_{\leq 0}) < 0$$
and we obtain a strictly better marking by unmarking e_i; therefore, the original marking could not have been optimal. After unmarking e_i, we again have $S_{RU} > S_{LU}$ and we can keep unmarking all edges on the right until the right side is fully unmarked, and we have a strictly better marking than M^* (see Figure 13). Note that the other case ($S_{RU} < S_{LU}$) can be handled symmetrically by fully unmarking the left side and repeatedly applying Eq. (26). □

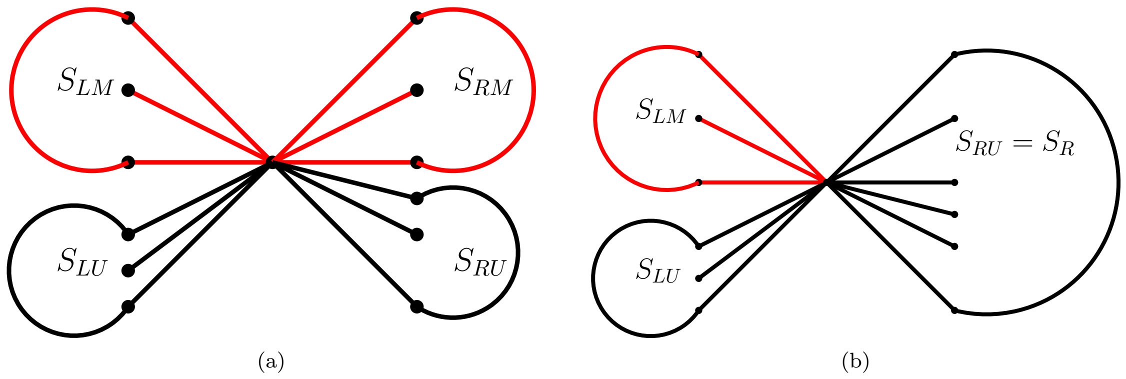


Figure 13: The example tree used in the proof of Lemma 10. (a) An arbitrary marking in which edges from both sides are marked, and more vertices are connected to the unmarked edges on the right ($S_{RU} \geq S_{LU}$). (b) A strictly better marking than (a) in which the heavier side is fully unmarked, as described in Lemma 10.

4.4.1 Partial Markings

In Lemma 10, we observed that no optimal marking has edges marked on both sides. In this section, we introduce the concept of partial markings, used to form optimal marking after merging a given edge e^*. A partial left (respectively right) marking, denoted by M_L (respectively, M_R), is a marking with all edges on the right (respectively, left) unmarked, and a subset of the edges on the left (respectively, right) marked. We call a partial marking *optimal* if its error count is less than any other partial marking for its respective side. Let M_L^* and M_R^* denote the optimal partial left and right markings, respectively. The following lemma is easy to prove.

Lemma 11 *After merging edge e^* in a weighted tree with non-negative weights, the optimal marking M^* is either M_L^* or M_R^*, depending on which one produces a smaller amount of error.*

Proof: Immediate from Lemma 10. □

Applying the results of Lemma 11, we can find an optimal marking M^* by finding the optimal partial markings M_L^* and M_R^*, comparing their respective error values, and choosing the one with the smaller error value as the optimal marking. The question is how to find the optimal partial markings, and it is answered in the following lemma.

Lemma 12 *The optimal partial marking M_L^* consists of all edges e_i (adjacent to L_i vertices) such that*

$$S_L - S_R \leq L_i \tag{29}$$

Similarly, the optimal partial marking M_R^* consists of all edges e_i (adjacent to R_i vertices) such that

$$S_R - S_L \leq R_i \tag{30}$$

Proof: We first prove Eq. (29) and derive Eq. (30) by symmetry. Suppose we are trying to construct an optimal partial marking for the left side. We can do so by keeping the right side

Algorithm 2 Graph Compression Algorithm for Trees

```
1: procedure TREE_COMPRESSION_SINGLE_EDGE
2:     Input: $T = (V, E)$ (A tree with $n$ vertices), an edge $e^* = (u, v)$ to be merged, the error function $\mathcal{E}(.)$
3:     Output: A marking of edges $M^*$ with the optimal amount of error
4:     Find all edges $E_L = \{(u, w) | (u, w) \in E, w \neq v\}$
5:     $\mathcal{L} \leftarrow |E_L|$, such that each edge $e_i$ in $E_L$ is connected to a subtree of size $L_i$ for all $i = \{1, \dots, \mathcal{L}\}$
6:     Find all edges $E_R = \{(v, w) | (v, w) \in E, w \neq u\}$
7:     $\mathcal{R} \leftarrow |E_R|$, such that each edge $e_i$ in $E_R$ is connected to a subtree of size $R_i$ for all $i = \{1, \dots, \mathcal{R}\}$
8:     $S_L \leftarrow \sum_{\forall e_i \in E_L} L_i$
9:     $S_R \leftarrow \sum_{\forall e_i \in E_R} R_i$
10:    Remove $e^*$ from $T$ and merge its endpoints
11:    $M_L^* \leftarrow \emptyset$, $M_R^* \leftarrow \emptyset$
12:    for each $e_i \in E_L$ do
13:        if $S_L - S_R \leq L_i$ then
14:            $M_L^* \leftarrow M_L^* \cup \{e_i\}$
15:        end if
16:    end for
17:    for each $e_i \in E_R$ do
18:        if $S_R - S_L \leq R_i$ then
19:            $M_R^* \leftarrow M_R^* \cup \{e_i\}$
20:        end if
21:    end for
22:    $M^* \leftarrow \text{argmin}(\mathcal{E}(M_L^*), \mathcal{E}(M_R^*))$
23:    Return $M^*$
24: end procedure
```

unmarked and marking edges on the left until the error can no longer be improved. Recall Eq. (27) from the proof of Lemma 10, we have to keep marking all edges e_i (with cardinality L_i) until the error can no longer be improved, the change in the error value at each step is equal to:

$$\Delta(\text{MARK\_LEFT}) = L_i \times \left(S_{LM} + (S_{LU} - L_i) + S_{RM} - S_{RU} \right)$$

At each step, to get an improvement, we must have $\Delta(\text{MARK\_LEFT}) \leq 0$:

$$(S_{LM} + (S_{LU} - L_i) + S_{RM} - S_{RU}) \leq 0 \xrightarrow{S_{LM}+S_{LU}=S_L} S_L - L_i + S_{RM} - S_{RU} \leq 0$$

However, since we are calculating a partial marking for the left side, we know by definition that the right side has to remain fully unmarked at all times, so we have $S_{RU} = S_R$ and $S_{RM} = 0$. Inserting these values in the above equation, we get the following inequality for edges that *improve* the partial left marking:

$$S_L - L_i - S_R \leq 0 \rightarrow S_L - S_R \leq L_i$$

Then, we can deduce that if an edge e_i on the left satisfies $S_L - S_R \leq L_i$, it must be marked in M_L^*. Conversely, if an edge on the left e_i is marked in M_L^*, it must satisfy $S_L - S_R \leq L_i$. To see why, assume M_L^* includes an edge e_i with $S_L - S_R > L_i$. Then, we can improve M_L^* by unmarking e_i (see Eq. (26)):

$$\Delta(\text{UNMARK\_LEFT}) = L_i \times \Big(-(S_{LM} - L_i) - S_{LU} - S_{RM} + S_{RU} \Big) = L_i \times \Big(L_i - S_L + S_R \Big) < 0$$

which contradicts the optimality of M_L^* and proves Eq. (29). The other inequality (Eq. (30)) can be proven analogously by applying Eq. (28). □

We present our linear-time algorithm for finding the optimal marking after merging an edge e^* in Algorithm 2.

Example 7 *As an example, let us demonstrate how Algorithm 2 finds the optimal marking for the tree of Figure 12. The optimal marking M_L^* consists of all edges on the left, because:*

- $7 - 21 \leq 2 = L_1$
- $7 - 21 \leq 2 = L_2$
- $7 - 21 \leq 3 = L_3$

On the other hand, the optimal marking M_R^* consists of only one edge on the right, because:

- $21 - 7 \leq 20 = R_1$
- $21 - 7 > 1 = R_2$

Moreover, because M_R^* has a better error count than M_L^*, Algorithm 2 returns M_R^* as the overall optimal marking M^* which is the correct answer as depicted in Figure 12.

Now, we summarize our result in the following theorem.

Theorem 4 *Algorithm 2 computes the optimal marking for a merged edge e^* in $\mathcal{O}(|V|)$ time.*

Proof: Immediate from Lemma 10, Lemma 11, and Lemma 12. □

4.5 Fractional Markings

In the previous section, we studied the marking problem under the assumption that each edge could either be fully marked or fully unmarked. In this section, we study a generalized version of the marking problem, called *the fractional marking problem* (to be defined momentarily). We show that Algorithm 2 does not err by assuming that each edge can either be fully marked or fully unmarked.

Definition 13 *With reference to a given merged edge e^* in a graph $G = (V, E)$ with the associated weight function $w : E \to \mathbb{R}_{\geq 0}$, and a new weight redistribution function $w' : E \to \mathbb{R}_{\geq 0}$, an edge e_i is said to be* fractionally marked *if $w'(e_i) = w(e_i) + c_i w(e^*)$ for some $c_i \in (0, 1)$. The edge e_i is* fully marked *if $c_i = 1$.*

Each neighbouring edge e_i has thus an assigned c_i, which denotes the (possibly fractional) amount by which it is marked. An edge e_i is marked by ϵ if its corresponding c_i is set to $c_i' = c_i + \epsilon$, and it is unmarked by ϵ if its corresponding c_i is set to $c_i' = c_i - \epsilon$.

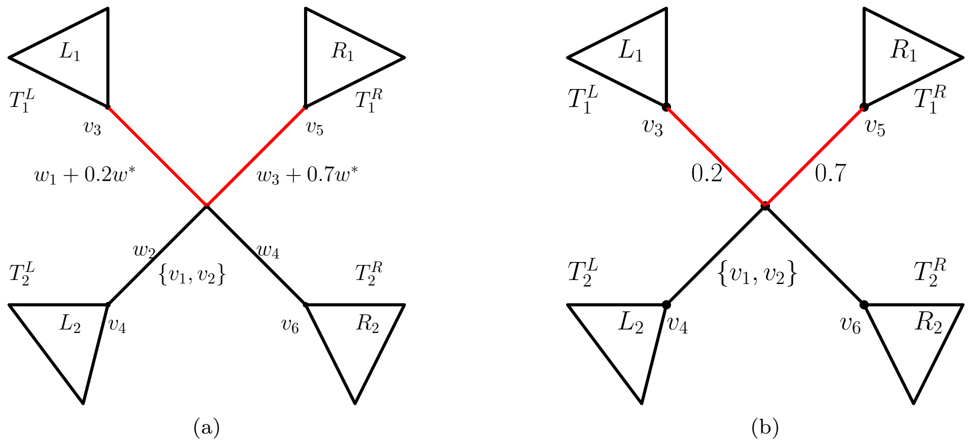


Figure 14: An extension of Figure 10 and Figure 11 as an example of fractional markings. (a) One edge from the left side and one from the right are fractionally marked. (b) A succinct representation of (a) in which each fractionally marked edge e_i is shown using its respective c_i (Definition 13).

Definition 14 ***The Fractional Marking Problem for Weighted Trees:*** *Given a contracted edge e^* in a weighted tree T with non-negative weights, what subset of the neighbouring edges of e^* should we fully mark or fractionally mark such that the error value of Eq. (1) is minimized over all such possible subsets?*

Similar to the previous section, we may omit some occurrences of w^* from our calculation for convenience. We borrow our previous running example (Figure 10 and Figure 11) and extend it to present an example of fractional markings in Figure 14. Figure 14-(a) depicts the tree of Figure 10 with two edges fractionally marked. Figure 14-(b) illustrates a succinct representation of Figure 14-(a), where each fractionally marked edge e_i is shown using its respective c_i (Definition 13) and the weights of the unmarked edges are omitted. We use this succinct version often in the remainder of this section.

As a warm-up, we first present a property of any optimal marking that has at least one fractionally marked edge.

Lemma 13 *Let M be an optimal marking for a contracted edge e^* (in a weighted tree) that has at least one fractionally marked edge e'. Then, M necessarily has marked edges on both sides.*

Proof: By contradiction. Suppose M is an optimal marking with a fractionally marked edge e', and suppose M is a partial left or right marking (Section 4.4.1) with marked edges only on the left or right, respectively. Without loss of generality, assume M is a partial left marking with a fractionally marked edge $e' = e_1$. As depicted in Figure 15-(a), assume e_1 has cardinality L_1 and marking value c_1 (Definition 13). We can obtain another marking M' by unmarking e_1 (Figure 15-(b)). Let $\mathcal{E}$ be the error function, then $\mathcal{E}(M) = \mathcal{E}(M') + \Delta_1(\text{MARK\_LEFT})$ and $\mathcal{E}(M) < \mathcal{E}(M')$ because M is an optimal marking. Therefore, $\Delta_1(\text{MARK\_LEFT}) < 0$ when marking e_1 back in M'. We now formulate $\Delta_1(\text{MARK\_LEFT})$ when marking e_1 in M' by c_1.

$$\Delta_1(\text{MARK\_LEFT}) = c_1 \times (X)$$

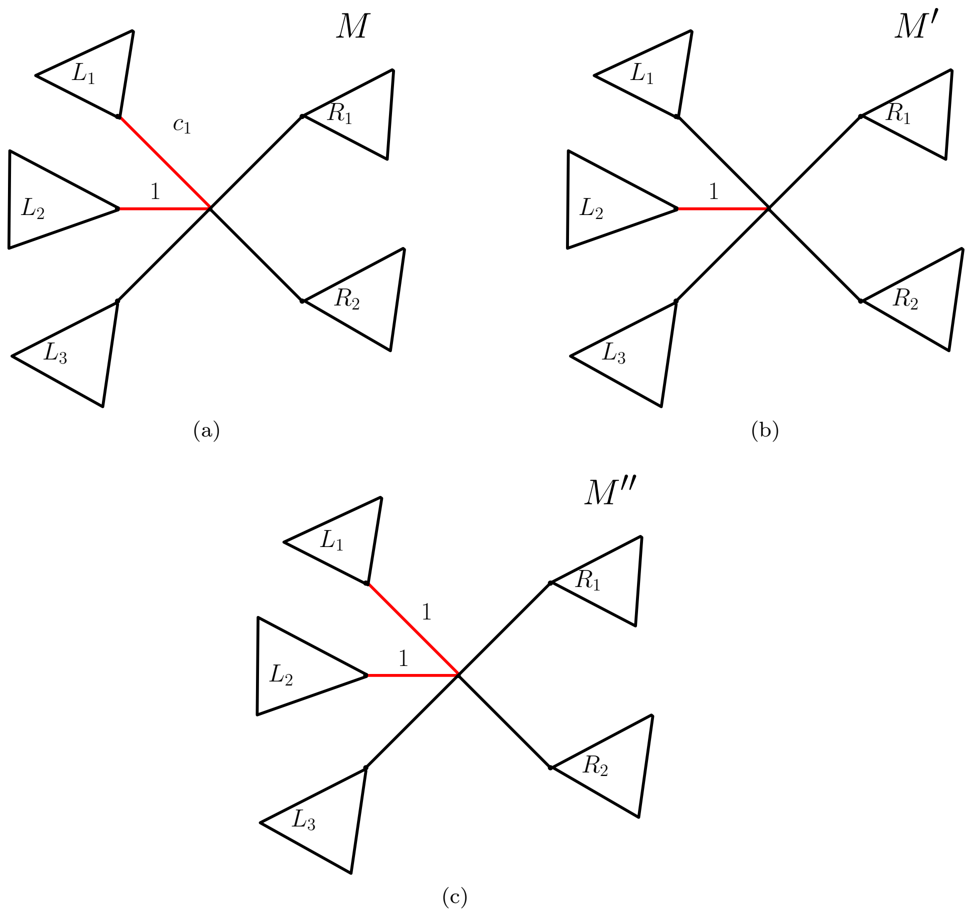


Figure 15: The example used in the proof of Lemma 13: (a) A hypothetical (for the sake of contradiction) optimal marking M with a fractionally marked edge e_1 with marking value c_1. (b) Another marking M' resulting from unmarking e_1 in M. (c) A new marking M'' strictly better than M ($\mathcal{E}(M'') < \mathcal{E}(M)$), yielding a contradiction and proving Lemma 13.

where $X = L_1 \times \left(S_{LM} + (S_{LU} - L_1) + S_{RM} - S_{RU}\right) = L_1 \times \left(S_{LM} + (S_{LU} - L_1) - S_R\right)$ because $S_{RM} = 0$ and $S_R = S_{RU}$. However, because $\Delta_1(\text{MARK\_LEFT}) < 0$, we have that $X < 0$ and we can fully mark e_1 in M' to get another marking M'' (Figure 15-(c)). The amount of error change of this mark operation is equal to:

$$\Delta_2(\text{MARK\_LEFT}) = (c_1) \times X + (1 - c_1) \times X$$

Noting that $\mathcal{E}(M) = \mathcal{E}(M') + \Delta_1(\text{MARK\_LEFT})$ we have:

$$\mathcal{E}(M'') = \mathcal{E}(M') + \Delta_2(\text{MARK\_LEFT}) = \underbrace{\mathcal{E}(M') + (c_1) \times X}_{=\mathcal{E}(M')+\Delta_1(\text{MARK\_LEFT})=\mathcal{E}(M)} + (1 - c_1) \times X < \mathcal{E}(M)$$

Therefore M'' is a strictly better marking than M, contradicting our assumption that M is an optimal marking. □

Lemma 13 states that an optimal marking with fractionally marked edges cannot be a partial left or right marking (as defined in Section 4.4.1). We now present the main result of this section.

Lemma 14 *Let M be an optimal marking for a contracted edge e^* (in a weighted tree) that contains both fractionally and fully marked edges. Then, M can be transformed into another optimal marking M' that contains no fractionally marked edges.*

Proof: We assume M is an optimal marking that contains fractionally marked edges. We consider several possible cases, and for each case, we present a transformation technique that does not worsen the marking with reference to the error function (Eq. (1)). The repeated application of these transformations converts M into another marking M' with no fractionally marked edges.

Let M be an optimal marking that contains at least one fractionally marked edge. Using Lemma 13, we may assume M contains marked edges on both sides. Throughout this proof, we use $0 < c_i \leq 1$ to denote the marking value for an edge e_i, such that e_i is fractionally marked if $0 < c_i < 1$ (see Definition 13).

- **Case 1:** M contains two marked edges e_1 (with cardinality L_1) and e_2 (with cardinality R_1) on the left and right respectively such that $c_1 + c_2 > 1$ (Figure 16-(a)).

Let $c_1 + c_2 = 1 + \epsilon$ for $\epsilon > 0$. We show that fractionally unmarking either e_1 or e_2 by ϵ does not worsen the error, and this change transforms M into another optimal marking M' in which $c_1' + c_2' = 1$. We generalize the proof of Lemma 10 and define $S_L = \sum_{i=1}^{\mathcal{L}} L_i$ and $S_R = \sum_{i=1}^{\mathcal{R}} R_i$ as the total sum of all edge cardinalities on the left and right sides, respectively. Let $S_L' = S_L - L_1$ and $S_R' = S_R - R_1$ and without loss of generality, assume $S_R' \geq S_L'$. We unmark e_2 by ϵ, setting $c_2' = c_2 - \epsilon$ (Figure 16-(b)). Note that we must necessarily have $c_2 \geq \epsilon$ because otherwise $c_1 + c_2 < 1 + \epsilon = c_1 + c_2$, which is a contradiction. We now show this operation does not worsen the marking. Between the vertices of e_2 and the vertices of all other edges on the right side, the error is reduced by $-\epsilon \times w^*$. Now, let $e_j \neq e_1$ be some edge on the left. The error between the vertices of e_2 and e_j is increased by at most $\epsilon \times w^*$. If e_j has marking value c_j, then this operation may increase the error between the vertices of e_2 and e_j by $|w^* - (c_2 \times w^* + c_j \times w^* - \epsilon \times w^*)| - |w^* - (c_2 \times w^* + c_j \times w^*)| \leq \epsilon \times w^*$ using Corollary 1. Therefore, this unmark operation changes the error by:

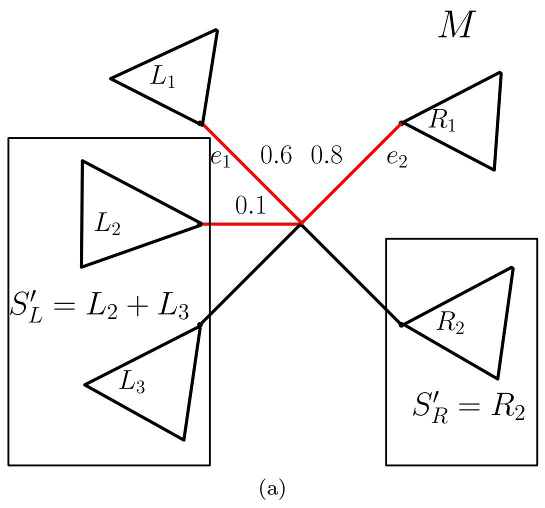
$$\Delta_1(\text{UNMARK\_RIGHT}) \leq R_1 \times \epsilon \times w^* \times \left(- S_R' \underbrace{-L_1}_{<0} + S_L' \right) < R_1 \times \epsilon \times w^* \times (\underbrace{-S_R' + S_L'}_{\leq 0}) \leq 0$$

- **Case 2:** For all pairs of marked edges e_1 and e_2 on the left and right respectively, $c_1 + c_2 \leq 1$ (Figure 17).

We consider two subcases:

- **Case 2-1:** There exist two edges e_i and e_j on one side (left or right) such that $c_i \neq c_j$ (Figure 17-(a)).

Figure 17-(a) is an example of such marking M in which $c_1 + c_2 \leq 1$ for any two edges e_1 and e_2 on opposite sides and at least two edges e_i and e_j on the left with $c_i \neq c_j$. Without loss of generality, we may assume $c_i < c_j$. Then, we can set $c_i' = c_j$ without increasing the error. Due to



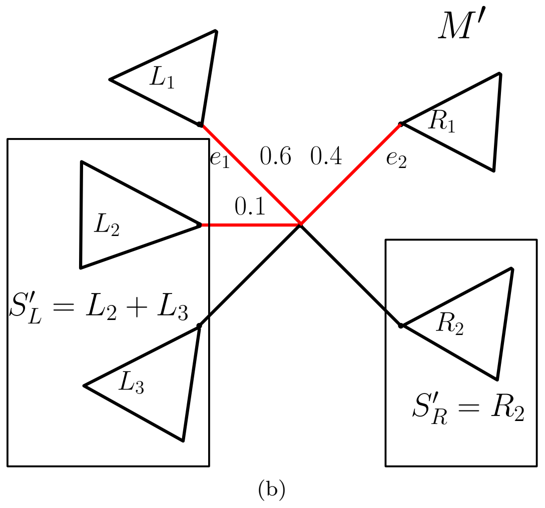


Figure 16: Case 1 in the proof of Lemma 14: (a) There exist two edges e_1 and e_2 such that $c_1 + c_2 = 0.6 + 0.8 = 1.4 > 1$, $S'_R = R_2 \geq S'_L = L_2 + L_3$. (b) Another marking M' where e_2 is unmarked by $\epsilon = 0.4$ such that $\mathcal{E}(M') \leq \mathcal{E}(M)$.

the properties of this subcase, we may get another marking M' by unmarking e_i (setting $c'_i = 0$) and then marking it by c_j to get a third marking M'' with $\mathcal{E}(M'') \leq \mathcal{E}(M)$. Similar to the proof of Lemma 13, we have:

$$\mathcal{E}(M'') = \underbrace{\mathcal{E}(M') + c_i \times X}_{=\mathcal{E}(M)} + (c_j - c_i) \times X$$

- **Case 2-2:** For all marked edges e_i (with $c_i > 0$) on the left $c_i = \epsilon_1$, and for all marked edges e_j (with $c_j > 0$) on the right $c_j = \epsilon_2$ ($\epsilon_1 + \epsilon_2 \leq 1$) (Figure 17-(b)).

For this case, we simply show that the error associated with the optimal partial marking (Lemma 12) is a lower bound on $\mathcal{E}(M)$, or $\min(\mathcal{E}(M^*_R), \mathcal{E}(M^*_L)) \leq \mathcal{E}(M)$. Without loss of generality, we assume

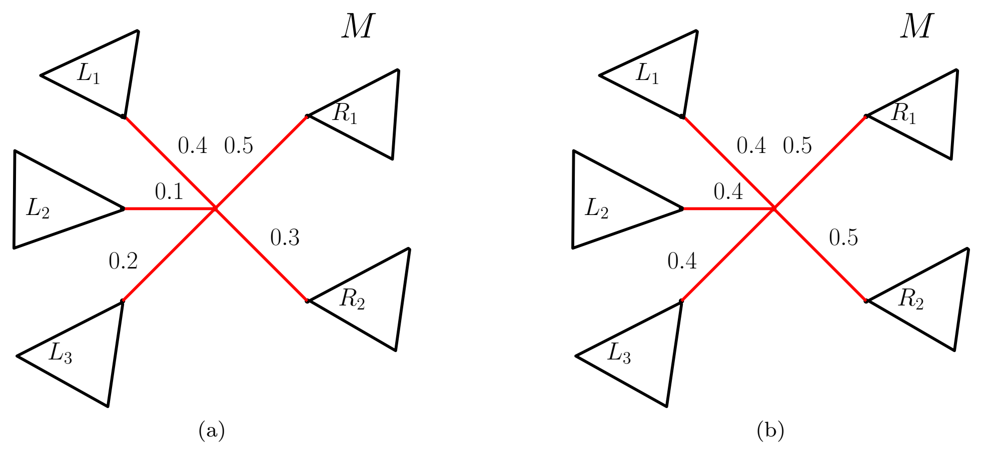


Figure 17: Case 2 in the proof of Lemma 14: (a) Case 2-1: For all edges e_1, e_2 on opposite sides $c_1 + c_2 \leq 1$, and there exist two edges e_i, e_j on one side with $c_i \neq c_j$ (for instance 0.4 and 0.1 on the left) (b) Case 2-2: For all edges e_1, e_2 on opposite sides $c_1 + c_2 \leq 1$, for all edges e_i on the left $c_i = \epsilon_1 = 0.4$, and for all edges e_j on the right $c_j = \epsilon_2 = 0.5$.

that $\min(\mathcal{E}(M_R^*), \mathcal{E}(M_L^*)) = \mathcal{E}(M_L^*)$.
Let E_L and E_R be the set of marked edges in M_L^* and M_R^*, respectively. From Lemma 12, we know that for each $e_i \in E_L$, $S_L - S_R \leq L_i$ and for each $e_i \in E_R$, $S_R - S_L \leq R_i$. We show that the set of marked edges in M is precisely equal to $E_L \cup E_R$. Let e_i be any marked (with reference to M) edge on the left, and let e_j be any marked edge on the right. Because $c_i + c_j \leq 1$, unmarking e_i by $\epsilon \leq c_i$ increases the error between the vertices of e_i and e_j by $\epsilon \times w^*$:

$$\underbrace{|w^* - (c_i \times w^* + c_j \times w^* - \epsilon \times w^*)|}_{=w^* - (c_i \times w^* + c_j \times w^* - \epsilon \times w^*) \text{ because } c_i + c_j \leq 1} - \underbrace{|w^* - (c_i \times w^* + c_j \times w^*)|}_{=w^* - (c_i \times w^* + c_j \times w^*) \text{ because } c_i + c_j \leq 1} = \epsilon \times w^*$$

Furthermore, unmarking e_i by ϵ decreases the error between the vertices of e_i and the vertices of all other edges on the left by $-\epsilon \times w^*$. Therefore, unmarking e_i by ϵ changes $\mathcal{E}(M)$ by:

$$\Delta_1(\text{UNMARK\_LEFT}) = L_i \times \epsilon \times w^* \times (-(S_L - L_i) + S_R) = L_i \times \epsilon \times w^* \times (-S_L + L_i + S_R)$$

Because M is an optimal marking, $\Delta_1(\text{UNMARK\_LEFT}) \geq 0$ and:

$$-S_L + L_i + S_R \geq 0 \rightarrow L_i \geq S_L - S_R$$

Conversely, we may assume that any edge e_i on the left satisfying $L_i \geq S_L - S_R$ is marked in M; because otherwise, we could improve M by marking e_i[2]. Similar reasoning can be applied to any marked edge e_i on the right.

[2]The proof for this claim is almost identical to the one provided in the proof of Lemma 12. Here, we omitted the details to avoid repetition.

We now conclude the proof. First, note that:

$$\mathcal{E}(M_L^*) = \underbrace{\mathcal{E}(M_0)}_{\text{The error associated with the empty marking}} + \underbrace{\sum_{e_i \in E_L} L_i \times w^* \times (S_L - L_i - S_R)}_{\text{The sum of all } \Delta(\text{MARK\_LEFT})\text{'s that transform } M_0 \text{ into } M_L^*}$$

and

$$\mathcal{E}(M_R^*) = \underbrace{\mathcal{E}(M_0)}_{\text{The error associated with the empty marking}} + \underbrace{\sum_{e_i \in E_R} R_i \times w^* \times (S_R - R_i - S_L)}_{\text{The sum of all } \Delta(\text{MARK\_RIGHT})\text{'s that transform } M_0 \text{ into } M_R^*}$$

where M_0 is a trivial marking with no marked edges.

On the other hand, M can be constructed by first marking all edges e_i in E_L by ϵ_1 and then marking all edges in E_R by ϵ_2.

$$\mathcal{E}(M) = \underbrace{\mathcal{E}(M_0)}_{\text{The error associated with the empty marking}} \underbrace{+\epsilon_1 \times \sum_{e_i \in E_L} L_i \times w^* \times (S_L - L_i - S_R)}_{\text{The sum of all } \Delta(\text{MARK\_LEFT})\text{'s by } \epsilon_1}$$
$$\underbrace{+\epsilon_2 \times \sum_{e_i \in E_R} R_i \times w^* \times (S_R - R_i - S_L)}_{\text{The sum of all } \Delta(\text{MARK\_RIGHT})\text{'s by } \epsilon_2}$$

From our assumption, $\mathcal{E}(M_L^*) \leq \mathcal{E}(M_R^*)$. We have $S_L - L_i - S_R \leq 0$ for all $e_i \in E_L$ and $S_R - R_i - S_L \leq 0$ for all $e_i \in E_R$. We get:

$$\mathcal{E}(M_L^*) \leq \mathcal{E}(M_R^*) \to \sum_{e_i \in E_L} L_i \times w^* \times (S_L - L_i - S_R) \leq \sum_{e_i \in E_R} R_i \times w^* \times (S_R - R_i - S_L)$$

On the other hand:

$$\begin{aligned}
\mathcal{E}(M) &= \mathcal{E}(M_0) + \epsilon_1 \times \sum_{e_i \in E_L} L_i \times w^* \times (S_L - L_i - S_R) + \epsilon_2 \times \sum_{e_i \in E_R} R_i \times w^* \times (S_R - R_i - S_L) \\
&\geq \mathcal{E}(M_0) + \epsilon_1 \times \sum_{e_i \in E_L} L_i \times w^* \times (S_L - L_i - S_R) + \epsilon_2 \times \sum_{e_i \in E_L} L_i \times w^* \times (S_L - L_i - S_R) \\
&= \mathcal{E}(M_0) + (\epsilon_1 + \epsilon_2) \times \sum_{e_i \in E_L} L_i \times w^* \times (S_L - L_i - S_R) \xrightarrow[S_L - L_i - S_R \leq 0]{\epsilon_1 + \epsilon_2 \leq 1} \\
&\geq \mathcal{E}(M_0) + \sum_{e_i \in E_L} L_i \times w^* \times (S_L - L_i - S_R) \\
&= \mathcal{E}(M_L^*) \qquad (31)
\end{aligned}$$

Thus, $\min(\mathcal{E}(M_L^*), \mathcal{E}(M_R^*))$ is a lower bound on the error of any such marking.

We now conclude the proof by stating that any optimal marking with fractionally marked edges can be transformed into another optimal marking with no fractionally marked edges. Let M be any such marking. If M satisfies the conditions of Case 1, we repeatedly apply the transformation method of Case 1 until it satisfies the conditions of Case 2-1. We then repeatedly apply the construction method of Case 2-1 until M satisfies the conditions of Case 2-2. Finally, if M satisfies the conditions of Case 2-2, we have already shown that $\mathcal{E}(M)$ is lower bounded by the optimal partial marking of Lemma 12, which has no fractionally marked edges. □

5 Conclusion and Open Problems

In this paper, we studied the problem of distance-preserving graph compression for weighted paths and trees. We first presented a brief literature review of some related work in this domain, noting that one particular aspect of the problem is understudied. More specifically, there has been little attention in the literature to the problem of optimally compressing a given set of edges. To address this, we presented optimal algorithms for compressing any set of k edges in a weighted path and for optimally compressing a single edge in a weighted tree. We tackled the problems in an incremental order of difficulty. For weighted paths, we first solved the problem of optimally compressing a single edge, then we generalized it to any set of k independent edges. Finally, we provided an optimal approach to compressing any contiguous subset of edges in a weighted path. We then generalized our scope to weighted trees, where we studied the problem of optimally compressing a single edge. To this end, we first studied the easier case in which the subtrees of both sides of the merged edge had equal sizes. Finally, we generalized our results to the case in which subtrees were of different sizes.
This research leads to several questions that require further exploration.

Problem 1 *How to optimally contract multiple edges in a tree?*

This problem includes the cases where the compressed edges form a contiguous subtree, when the edges form a matching, or when a combination of both cases happens. When multiple edges are to be contracted in a tree, many cases need to be considered in order to properly formulate the error function. Therefore, it may be worthwhile to see whether formulating the error when multiple edges are being contracted in a tree results in any interesting observations from a combinatorial optimization point of view, like the ones mentioned in Section 3 or Section 4.2.

There are two main reasons why the problem of contracting multiple edges in a tree is not as straightforward as its counterpart in a path. Firstly, the maximum degree in a tree is unbounded, whereas, in a path, the maximum degree is two. The second reason (which is a direct result of the first one) is that a node in an arbitrary tree can have many children. When an edge e^* with weight w^* is contracted in a path, there are only two groups of shortest paths that should not have w^* added to their values, the ones that lie to the left of e^* and the ones that lie to its right. However, as observed in Section 4.2, even for merging a single edge e^* in a tree, there are significantly more cases to consider. With no restrictions on the maximum degree of an arbitrary tree, any error unit enumeration technique (such as the ones employed in Section 3 or Section 4.2) could quickly become obsolete due to an explosion in the number of cases when many edges are contracted.

Problem 2 *Can we solve the distance-preserving graph compression problem for general graphs in polynomial time?*

The above problem would indeed be a natural extension of this paper. The complexity of the weight redistribution problem for general graphs is still unknown. However, it appears that the related problem of finding the contracted edges is unlikely to be solved in polynomial time. Bernstein *et al.* [5] showed that CONTRACTION (defined in Section 1) is NP-hard even if the underlying graph is just a weighted cycle. In a graph with cycles, some vertices are connected via multiple paths. Therefore, after merging a single edge, several shortest paths that traverse that edge may need to be rerouted using completely different edges, making the analysis much more difficult.

Problem 3 *Recall from Definition 7 that with reference to a set of merged edges $E_m \subset E$, the set of merged vertices V_m consists of all vertices with at least one endpoint in E_m, or $V_m = \{v | v, u \in V, \exists e = (u, v) \in E_m\}$. How could we find an optimal redistribution strategy that also minimizes the error between all pairs of vertices in V_m?*

Note that even if some weight redistribution minimized the error between two nodes in different supernodes, it would still be non-trivial to do the same for two vertices that are placed in a single supernode. Obviously, a trivial solution would be to store the shortest path weights between the vertices in one supernode as separate table entries. However, such an approach would defeat the whole purpose of graph compression, which is to reduce memory requirements.

Problem 4 *For the optimal weight redistribution problem, are there any better cost models (error functions)?*

As stated in Section 2, in this paper, we defined the error function as the sum of the absolute differences of the shortest path lengths between different pairs of nodes before and after redistributing the weights. However, exploring alternative cost functions that better capture the distance-based similarity between a modified graph and its original version can open up exciting research avenues. Investigating whether there exist other cost functions that provide a more accurate measure of closeness between graphs can lead to valuable research opportunities.

Acknowledgements

We thank the anonymous reviewers for providing valuable feedback on this paper.

References

[1] I. Abraham, D. Delling, A. V. Goldberg, and R. F. Werneck. A hub-based labeling algorithm for shortest paths in road networks. In *Experimental Algorithms: 10th International Symposium, SEA 2011, Kolimpari, Chania, Crete, Greece, May 5-7, 2011. Proceedings 10*, pages 230–241. Springer, 2011. doi:10.1007/978-3-642-20662-7_20.

[2] Y. Arfat, S. Suma, R. Mehmood, and A. Albeshri. Parallel shortest path big data graph computations of US road network using apache spark: survey, architecture, and evaluation. *Smart Infrastructure and Applications: Foundations for Smarter Cities and Societies*, pages 185–214, 2020. doi:10.1007/978-3-030-13705-2_8.

[3] C. Bazgan, C. Bentz, C. Picouleau, and B. Ries. Blockers for the stability number and the chromatic number. *Graphs and Combinatorics*, 31:73–90, 2015. doi:10.1007/S00373-013-1380-2.

[4] C. Bazgan, S. Toubaline, and Z. Tuza. The most vital nodes with respect to independent set and vertex cover. *Discrete Applied Mathematics*, 159(17):1933–1946, 2011. doi:10.1016/J.DAM.2011.06.023.

[5] A. Bernstein, K. Däubel, Y. Disser, M. Klimm, T. Mütze, and F. Smolny. Distance-preserving graph contractions. *SIAM Journal on Discrete Mathematics*, 33(3):1607–1636, 2019. doi:10.1137/18M1169382.

[6] T. C. Biedl, B. Brejová, and T. Vinař. Simplifying flow networks. In *Mathematical Foundations of Computer Science 2000: 25th International Symposium, MFCS 2000 Bratislava, Slovakia, August 28–September 1, 2000 Proceedings*, pages 192–201. Springer, 2000. doi:10.1007/3-540-44612-5_15.

[7] A. Ching, S. Edunov, M. Kabiljo, D. Logothetis, and S. Muthukrishnan. One trillion edges: Graph processing at facebook-scale. *Proceedings of the VLDB Endowment*, 8(12):1804–1815, 2015. doi:10.14778/2824032.2824077.

[8] M.-C. Costa, D. de Werra, and C. Picouleau. Minimum d-blockers and d-transversals in graphs. *Journal of Combinatorial Optimization*, 22:857–872, 2011. doi:10.1007/S10878-010-9334-6.

[9] Ö. Y. Diner, D. Paulusma, C. Picouleau, and B. Ries. Contraction and deletion blockers for perfect graphs and h-free graphs. *Theoretical computer science*, 746:49–72, 2018. doi:10.1016/J.TCS.2018.06.023.

[10] F. V. Fomin, S. Saurabh, and N. Misra. Graph modification problems: A modern perspective. In *Frontiers in Algorithmics: 9th International Workshop, FAW 2015, Guilin, China, July 3-5, 2015, Proceedings 9*, pages 3–6. Springer, 2015. doi:10.1007/978-3-319-19647-3_1.

[11] E. Galby, P. T. Lima, and B. Ries. Reducing the domination number of graphs via edge contractions and vertex deletions. *Discrete Mathematics*, 344(1):112169, 2021. doi:10.1016/J.DISC.2020.112169.

[12] E. Galby, F. Mann, and B. Ries. Blocking total dominating sets via edge contractions. *Theoretical Computer Science*, 877:18–35, 2021. doi:10.1016/J.TCS.2021.03.028.

[13] M. Kargar and A. An. Keyword search in graphs: finding r-cliques. *Proceedings of the VLDB Endowment*, 4(10):681–692, 2011. doi:10.14778/2021017.2021025.

[14] E. J. Kim, M. Milanič, J. Monnot, and C. Picouleau. Complexity and algorithms for constant diameter augmentation problems. *Theoretical Computer Science*, 904:15–26, 2022. doi:10.1016/J.TCS.2021.05.020.

[15] J. M. Lewis and M. Yannakakis. The node-deletion problem for hereditary properties is NP-complete. *Journal of Computer and System Sciences*, 20(2):219–230, 1980. doi:10.1016/0022-0000(80)90060-4.

[16] J. Li, T. Cai, K. Deng, X. Wang, T. Sellis, and F. Xia. Community-diversified influence maximization in social networks. *Information Systems*, 92:101522, 2020. doi:10.1016/J.IS.2020.101522.

[17] Q. Li, K. Zou, D. Kong, H. Guan, and X. Xie. Gpugraphx: A GPU-aided distributed graph processing system. In *Web Information Systems Engineering–WISE 2021: 22nd International Conference on Web Information Systems Engineering, WISE 2021, Melbourne, VIC, Australia, October 26–29, 2021, Proceedings, Part II 22*, pages 501–509. Springer, 2021. doi:10.1007/978-3-030-91560-5_38.

[18] Y. Liang, Y. Wang, K. Lei, M. Yang, Z. Lyu, et al. Reachability preserving compression for dynamic graph. *Information Sciences*, 520:232–249, 2020. doi:10.1016/J.INS.2020.02.028.

[19] P. T. Lima, V. F. dos Santos, I. Sau, and U. S. Souza. Reducing graph transversals via edge contractions. *Journal of Computer and System Sciences*, 120:62–74, 2021. doi:10.1016/J.JCSS.2021.03.003.

[20] H. Lin, X. Zhu, B. Yu, X. Tang, W. Xue, W. Chen, L. Zhang, T. Hoefler, X. Ma, X. Liu, et al. Shentu: processing multi-trillion edge graphs on millions of cores in seconds. In *SC18: International Conference for High Performance Computing, Networking, Storage and Analysis*, pages 706–716. IEEE, 2018. doi:10.1109/SC.2018.00059.

[21] Y. Liu, T. Safavi, A. Dighe, and D. Koutra. Graph summarization methods and applications: A survey. *ACM computing surveys (CSUR)*, 51(3):1–34, 2018. doi:10.1145/3186727.

[22] F. Mahdavi Pajouh, V. Boginski, and E. L. Pasiliao. Minimum vertex blocker clique problem. *Networks*, 64(1):48–64, 2014. doi:10.1002/NET.21556.

[23] N. Ruan, R. Jin, and Y. Huang. Distance preserving graph simplification. In *2011 IEEE 11th International Conference on Data Mining*, pages 1200–1205. IEEE, 2011. doi:10.1109/ICDM.2011.57.

[24] A. Sadri, F. D. Salim, Y. Ren, M. Zameni, J. Chan, and T. Sellis. Shrink: Distance preserving graph compression. *Information Systems*, 69:180–193, 2017. doi:10.1016/J.IS.2017.06.001.

[25] O. Sporns. Graph theory methods: applications in brain networks. *Dialogues in clinical neuroscience*, 2022. doi:10.31887/DCNS.2018.20.2/osporns.

[26] H. Toivonen, F. Zhou, A. Hartikainen, and A. Hinkka. Compression of weighted graphs. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 965–973, 2011. doi:10.1145/2020408.2020566.

[27] N. Trinajstic. *Chemical graph theory*. CRC press, 2018. doi:10.1201/9781315139111.

[28] T. Watanabe, T. Ae, and A. Nakamura. On the NP-Hardness of edge-deletion and-contraction problems. *Discrete Applied Mathematics*, 6(1):63–78, 1983. doi:10.1016/0166-218X(83)90101-4.

[29] F. Zhou, S. Malher, and H. Toivonen. Network simplification with minimal loss of connectivity. In *2010 IEEE international conference on data mining*, pages 659–668. IEEE, 2010. doi:10.1109/ICDM.2010.133.

[30] C. J. Zhu, K.-Y. Lam, and S. Han. Approximate path searching for supporting shortest path queries on road networks. *Information Sciences*, 325:409–428, 2015. doi:10.1016/J.INS.2015.06.045.