Embedding Planar Graphs into Low-Treewidth Graphs with Applications to Efficient Approximation Schemes for Metric Problems

Eli Fox-Epstein* Philip N. Klein† Aaron Schild‡

Abstract

We show that, for any $\epsilon > 0$, there is a deterministic embedding of edge-weighted planar graphs of diameter D into bounded-treewidth graphs. The embedding has additive error ϵD. We use this construction to obtain the first *efficient* bicriteria approximation schemes for weighted planar graphs addressing k-Center (equivalently d-Domination), and a metric generalization of independent set, d-independent set. The approximation schemes employ a metric generalization of Baker's framework that is based on our embedding result.

1 Introduction

An *approximation scheme* for an optimization problem is a family of algorithms $\{A_\epsilon : \epsilon > 0\}$ such that algorithm A_ϵ returns a solution that is within a factor $1+\epsilon$ of optimal. It is a *polynomial-time* approximation scheme (PTAS) if each algorithm in the family runs in polynomial time, and an efficient PTAS (EPTAS) if there is a fixed degree d such that every algorithm in the scheme has running time $O(n^d)$.

In 1983, Baker [1] introduced a framework for obtaining very efficient (linear time) approximation schemes for certain optimization problems in planar graphs. Her framework gave rise to linear-time approximation schemes for minimum-weight dominating set, maximum-weight independent set, minimum vertex cover, maximum triangle matching, and several others.

Baker's approach can handle only problems characterized by very local properties. A *dominating set* in a graph is a set S of vertices such that every vertex in the graph is within one hop of a vertex in S. An *independent set* is a set of vertices no two of which are within one hop. These are the two problems that best illustrate the ideas underlying Baker's framework.

The key to Baker's results, is a property now known as *bounded local treewidth* (a.k.a. *the diameter-treewidth property* [8]). *Treewidth* is a measure of the structural complexity of a graph and many problems can be solved in polynomial time on graphs of bounded treewidth.[1] A family of graphs has bounded local treewidth if the treewidth of every graph in the family is upper-bounded by a function of the graph's diameter. In particular, a graph whose diameter is no more than a constant has treewidth no more than a constant. In modern parlance, Baker showed that planar graphs have bounded local treewidth, and used this result to reduce the approximation of local optimization problems to exact solution of these problems on bounded-treewidth graphs.

The reduction works for problems characterized by local properties—properties involving only vertices separated by a constant number of hops—because the structures in question can be in a sense isolated to subgraphs of constant diameter.

In this paper, we seek to extend Baker's framework to handle problems characterized by properties involving vertices separated by a given *distance* in an edge-weighted graph. We call these the *metric generalizations* of the problems addressed by Baker. Our goal is to give *efficient* PTASs for such metric generalizations.

Consider the two most representative problems addressed by Baker: Minimum-Weight Dominating Set and Maximum-Weight Independent Set. The metric generalization of the first is: given a graph G with vertex weights and edge lengths, and given a number $d > 0$, find a minimum-weight set S of vertices such that every vertex of G is within distance d of some vertex in S. Such a set S is said to be an *d-dominating set*. The d-Domination problem models a scenario in which one must select locations for facilities, e.g. clinics or firehouses, so that every client (vertex) is within a prescribed travel time from some facility. A closely

*Google. Research funded by NSF Grants CCF-1409520 and CCF-9640347

†Brown University. Research funded by NSF Grants CCF-1409520 and CCF-9640347, with additional support from the Radcliffe Institute of Advanced Study, Harvard University

‡University of California, Berkeley. Research funded by NSF Grants CCF-15-39577 and CCF-9640347. Part of this research was done while visiting Brown University.

[1] Due to space limits, we omit the formal definitions of treewidth and of branchwidth, a measure that is within a constant factor of treewidth.

related problem is k-CENTER, in which the goal is to minimize d such that there exist k vertices such that every vertex is within distance d of one of these k vertices.

The second problem to generalize is MAXIMUM-WEIGHT INDEPENDENT SET. Its metric generalization is MAXIMUM-WEIGHT d-INDEPENDENT SET: given a graph G with vertex weights and edge lengths, and given a number $d > 0$, find a maximum-weight set S of vertices such that every pair of vertices in S are a distance greater than d from each other.[2]

For unweighted planar graphs and constant d, Baker's approach can be generalized to obtain linear-time approximation algorithms for these generalizations. Indeed, such approximation schemes have been given [6, 9]. The challenge we address is to obtain efficient approximation schemes for the case of arbitrary edge lengths and arbitrary d.

Evidence (see Section 1.1) indicates that such efficient approximation schemes do not exist, so we slightly lower our sights: we give *bicriteria* approximation schemes, where both the weight of the solution *and* the distance d are approximated to within a $1 + \epsilon$ factor.

THEOREM 1.1. *There is an efficient bicriteria PTAS for d-DOMINATION in planar graphs with vertex weights and edge lengths. That is, there is a fixed c such that for any $\epsilon > 0$ there is an $O(n^c)$ algorithm that, given a planar graph with vertex weights and edge-lengths, and given a distance d, returns a set S of vertices such that*

- *every vertex is within distance $(1 + \epsilon)d$ of S, and*
- *the weight of S is at most $1 + \epsilon$ times the minimum weight of an d-dominating set.*

The problem d-DOMINATION can be generalized by specifying a subset of vertices to be clients (so not every vertex needs to be near a facility) and further generalized by assigning a mass to each vertex and requiring only that, say, 80% of the client mass is required to be near a facility. Our efficient bicriteria PTAS can handle this generalization.

A bicriteria PTAS was previously given [7]; the same paper introduced the notion of a metric generalization of Baker's framework. However, the PTAS was not an *efficient* PTAS: the degree of the polynomial increased with $1/\epsilon$.

Our bicriteria EPTAS can also be interpreted as addressing k-CENTER, in which the input specifies a number k, and the goal is to minimize d such that there is a set of k vertices that d-dominate all vertices. Our EPTAS can find d such that there is a set of $(1 + \epsilon)k$ vertices that d-dominate all vertices, and such that no set of k vertices can $(1 - \epsilon)d$-dominate all vertices.

THEOREM 1.2. *There is an efficient bicriteria PTAS for d-independent set problem in planar graphs with vertex weights and edge lengths. That is, there is a fixed c such that for any $\epsilon > 0$, there is an $O(n^c)$ algorithm that, given a planar graph with vertex weights and edge-lengths, and given a distance d, returns a set S of vertices such that*

- *every pair of vertices in S are at a distance greater than $(1 - \epsilon)d$, and*
- *the weight of S is at least $1 - \epsilon$ times the maximum weight of a set $\hat{S}$ of vertices every pair of which are at distance at least d.*

We can similarly obtain efficient bicriteria approximation schemes for metric generalizations of other problems addressed by Baker, e.g. maximum triangle matching. All these new schemes follow from our main result, an embedding of planar graphs into bounded-treewidth graphs with additive error a bounded fraction of the input graph's radius.

THEOREM 1.3. *There is a polynomial-time algorithm that, given an edge-weighted planar graph and given a number $\epsilon > 0$, outputs an embedding of the graph into a planar graph of treewidth poly$(1/\epsilon)$ with additive error $\epsilon\, D$, where D is the diameter of the input graph.*

This theorem represents our attempt to generalize the notion of *bounded local treewidth* to graphs with edge lengths.[3] Each of our approximation schemes consists in

- using a variant of Baker's approach, *metric shifting* (from [7]) to reduce the problem to several instances in bounded-diameter planar graphs,
- using the embedding of Theorem 1.3 to further reduce each instance to one in a bounded-treewidth graph, and
- using dynamic programming to obtain an approximately optimal solution for each instance.

The running time of Step 1 is linear. The running time of Step 2 is polynomial with some fixed degree. (We have not tried to optimize this running time.) For Step 3, in the case of (k, r)-center, Katsikarelis

[2] One might call this the *franchise* problem. It is said that Colonel Sanders promised the early franchisees for Kentucky Fried Chicken that no franchisees would be within a given distance of each other. It is called the *d-scattered set* in [15].

[3] Admittedly, in our current proof, the treewidth is bounded by a polynomial of very high degree in $1/\epsilon$. There is some irony in the fact that our approach to achieving an efficient PTAS yields an algorithm that is inefficient in the constant's dependence on ϵ. However, with this paper we are introducing the problem and some fundamental techniques. Although the result presented here is highly theoretical, it could illuminate the path towards a more practical result.

et al. [14] gave a fixed-parameter tractable bicriteria approximation scheme, parameterized by the treewidth of the input graph. The running time is a fixed polynomial in the graph size times $(\mathrm{tw}/\epsilon)^{O(\mathrm{tw})}$ where tw is the treewidth. It is not hard to obtain a similar result for d-independent set. (We will describe such algorithms in the full version.)

In Section 2.1, we illustrate the metric Baker framework by reducing bicriteria approximation of d-domination and d-independent set in planar graphs with edge-lengths to approximation in such graphs but where the radius is $O(d\epsilon^{-1})$. Most of the paper is devoted to proving Theorem 1.3.

1.1 Related work

Embeddings Metric embeddings of bounded dimensional metric spaces have been of great recent interest. Talwar [17] showed that a metric of *bounded doubling dimension* and aspect ratio Δ can be probabilistically approximated with $1+\epsilon$ error by a family of treewidth-κ metrics, where κ is bounded by a function that is polylogarithmic in Δ. Feldmann et al. [10] build on this result to obtain a similar embedding theorem for graphs of bounded highway dimension. Chan and Gupta [5] showed a similar result for graphs of bounded correlation dimension, though the treewidth of the approximating graphs is $\tilde{O}(\sqrt{n})$. These results can be used to obtain (superpolynomial-time) approximation algorithms for NP-complete problems like the traveling salesman problem.

Unfortunately, planar graphs neither have low doubling dimension nor low highway dimension. Indeed, results as strong as those of Talwar [17] and Feldmann et al. [10] are not possible for planar graphs: Chakrabarti et al. [4] showed a result that implies that unit-weight planar graphs cannot be embedded into distributions over $o(\sqrt{n})$-treewidth graphs with better than $O(\log n)$ distortion. (In an earlier result, Carroll and Goel showed [3] that any embedding of planar graphs into a distribution over bounded-treewidth graphs has (relative) distortion $\Omega(\log n)$.)

As mentioned earlier, Eisenstat et al. [7] introduced a kind of metric version of Baker's framework, reducing bicriteria approximation of k-center in bounded-genus graphs to approximation in bounded-genus bounded-radius graphs. However, their method for addressing the problem in the latter graphs used shortest-path separators rather than metric embeddings into low-treewidth graphs, and the resulting approximation scheme is inefficient. During this research Eisenstat (personal communication) asked whether any planar graph with diameter D has a minor that has bounded treewidth and preserves distances up to ϵD.

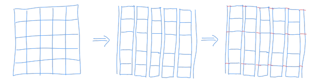

Figure 1: Transforming a grid into a graph with low branchwidth.

It is this question that inspired our work towards our main result.

More on k-center and d-domination and d-independent set As mentioned earlier, our bicriteria EPTAS can be used to address k-center. For this problem, Hochbaum and Shmoys [13] and Gonzalez [11] gave a 2-approximation algorithm for arbitrary graphs. This is best possible; for any $\epsilon > 0$, $2-\epsilon$-approximation is NP-hard [12], even for planar graphs [16]. We therefore cannot expect a PTAS for k-center in planar graphs, so are willing to settle for a bicriteria EPTAS.

Assuming the Exponential-Time Hypothesis (ETH), there is no $f(k)n^{o(\sqrt{k})}$ algorithm to find a solution of size k (when one exists) to d-Dominating Set or d-Independent Set (see [15]). Suppose there were a (single-criteria) EPTAS for d-Domination or d-Independent Set in planar graphs, and that its running time was $O(n^c)$ where c is a constant independent of ϵ. For $k \le c^3$, one can choose ϵ small enough so that the EPTAS would find the optimal solution. This would be an $f(k)n^{o(\sqrt{k})}$ algorithm, refuting ETH. Thus we do not expect a single-criteria EPTAS for either of these problems.

Prior to the work of Katsikarelis et al. [14], there was other work on dynamic programs for k-center and d-domination for unweighted graphs and fixed k or fixed d in graphs of bounded treewidth: Demaine et al. [6], and Borradaile and Le [2].

Marx and Pilipczuk [15] have given algorithms that run in $n^{O(\sqrt{k})}$ for finding the optimal solution to d-dominating set and d-independent set in planar graphs with edge-lengths when the vertices are unweighted and the size of the optimal solution is at most k.

1.2 Informal discussion of the embedding techniques

Why try to embed a planar graph into a bounded-treewidth graph with additive error bounded by ϵ times the diameter of the input graph? Consider a regular $\sqrt{n} \times \sqrt{n}$ grid, which has diameter $O(\sqrt{n})$.

For each column, cut along the column, turning a single path into two parallel paths. Each vertex of the original column now corresponds to two copies. Think of the space between these paths as a *channel* filled with water. A boat could efficiently move up and down the channels to reach any vertex. So far it is impossible to travel between the left and the right of each column, so the new metric is far from the old

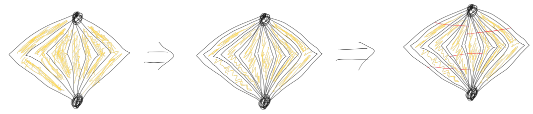

Figure 2: Channelizing the bars of the cage, and then adding links to approximately preserve bar-to-bar distances.

one. In order to partially restore the metric, at rows $1, \sqrt{n}\epsilon + 1, 2\sqrt{n}\epsilon + 1, 3\sqrt{n}\epsilon + 1, \ldots$ add artificial edges between the two copies of each vertex. Think of these artificial edges as *drawbridges*. Each drawbridge slightly impedes the movement of the boat. However, each channel only has $1/\epsilon$ drawbridges so every vertex is still just a small number of drawbridges away from every other. At the same time, the addition of the artificial edges makes the new metric much closer to the original one; relative to the original, distances have increased by at most ϵ times the diameter.

Our approach to the embedding makes use of four basic techniques. The first, which has been used in many approximation algorithms, is the notion of *portalization* or, equivalently, ϵ-nets.

The other three techniques are new. One *cages*, addresses the grid-like structure of planar graphs One technique addresses the recursive-nesting structure of planar graphs. One technique helps achieve bounded treewidth despite the recursion.

Cages The first technique involves a subgraph we call a *cage*, depicted in Figure 2. Between the bars of the cage the graph can be arbitrarily complicated. In order to reduce the treewidth, the algorithm turns bars of the cage into channels. Using a result of Tamaki (Theorem 3.1), we can show that the channels enable us to bound the treewidth of the cage. However, the channels alone would destroy the metric. The algorithm also uses a procedure Bars2Bars to select a subset of shortest paths between bars in order to approximately preserve all bar-to-bar shortest paths. Each channel is intersected by a constant number (depending on ϵ) of selected paths, which only increases the treewidth by a constant factor.

Recursive nesting and detour cost bump The algorithm must recursively operate on the subgraphs embedded between bars of a cage. However, the nesting could be arbitrarily deep, and we can only afford a constant depth of nesting for two reasons: each level of nesting increases the approximation error, and each level of nesting increases the branchwidth of the resulting graph.

To bound the effective nesting, we use a measure we call *detour cost*. Consider two paths P_1 and P_2. Under certain conditions, if P_2's detour cost is not much more than P_1's then P_2 can be approximated using a subpath of P_1. Therefore the algorithm needs to worry about P_2 only if its detour cost exceeds that of P_1 by at least ϵ. This increase in detour cost allows us to show limited recursion depth because we show that no path has very large detour cost.

2 Applications of our embedding result

In this section, we illustrate the use of the embedding in obtaining approximation schemes using a metric version of Baker's framework.

2.1 Metric adaptation of Baker's framework

We outline how bicriteria approximation for d-domination and d-independence in a graph G can be reduced to the same problems in a graph (obtained from G by edge deletions and contractions) whose radius is $O(\epsilon^{-1})$.

We closely follow the treatment in [7], which described this reduction for d-domination. Assume for simplicity of presentation that ϵ^{-1} is an integer and that $d = 1$.

For each of the two problems, we define families of intervals. Let σ be a value in $\{0, 1, 2, \ldots, \epsilon^{-1} - 1\}$. For $j = 0, 1, \ldots$, we define intervals $\mathcal{I}_{j,\sigma}$ and $\mathcal{I}^+_{j,\sigma}$:

- For 1-domination, $\mathcal{I}_{j,\sigma} = [2j\epsilon^{-1} - 2\sigma,\, 2(j+1)\epsilon^{-1} - 2\sigma)$ and $\mathcal{I}^+_{j,\sigma} = [2j\epsilon^{-1} - 1 - 2\sigma,\, 2(j+1)\epsilon^{-1} + 1 - 2\sigma)$.
- For 1-independence, $\mathcal{I}_{j,\sigma} = [j\epsilon^{-1} + 1 - \sigma, (j+1)\epsilon^{-1} - \sigma)$ and $\mathcal{I}^+_{j,\sigma} = [j\epsilon^{-1} - \sigma, (j+1)\epsilon^{-1} + 1 - \sigma)$.

We now describe the two reduction algorithms. Compute shortest-path distances rooted at a vertex s. Define U^σ_j to be the set of vertices whose distances from s lie in $\mathcal{I}_{j,\sigma}$, and define V^σ_j to be the set of vertices whose distances from s lie in $\mathcal{I}^+_{j,\sigma}$. Let G^σ_j to be the subgraph of G induced by V^σ_j, with extra edges $\{sv$: v's parent in shortest-path tree is not in $V^\sigma_j\}$ assigned length 1. The radius of G^σ_j is at most $2\epsilon^{-1} + 3$ in case of d-domination and at most $\epsilon^{-1} + 2$ in case of d-independence.

For each $\sigma \in \{0, 1, \ldots, \epsilon^{-1} - 1\}$, the algorithm uses a subroutine to compute an approximation to the appropriate problem for each of the instances $(G^\sigma_0, U^\sigma_0), (G^\sigma_1, U^\sigma_1), \ldots$ then finds the union of the solutions to these instances, and returns whichever union is the best.

For the analysis, we assume that the subroutine has the following guarantee:

- (Generalized 1-domination) The subroutine returns a set that $(1 + \epsilon)$-dominates all vertices of U and has weight at most the minimum weight of a set of vertices that 1-dominates all vertices of U.

- (Generalized 1-independendence) The subroutine returns a $(1-\epsilon)$-independent set whose weight is at least the maximum weight of a 1-independent set.

Let $W_j^\sigma = V_j^\sigma \cap V_{j+1}^\sigma$ for $\sigma \in \{0, 1, \dots, \epsilon^{-1} - 1\}$ and $j = 0, 1, \dots$. Let $W^\sigma = \bigcup_j W_j^\sigma$.

The proofs of the following, like those of Baker, use average arguments.

LEMMA 2.1. *Let M^* be a set of vertices of G that 1-dominates all vertices in G. The reduction algorithm returns a set that $1+\epsilon$ dominates all vertices in G and has weight at most $1+\epsilon$ times that of M^*.*

LEMMA 2.2. *Let M^* be a set of 1-independent vertices of G. The reduction algorithm returns a $(1-\epsilon)$-independent set whose weight is at least $1-\epsilon$ times that of M^*.*

3 Preliminaries

For a path P, the first vertex of P is denoted $\mathsf{start}(P)$ and the last vertex is denoted $\mathsf{end}(P)$. The vertices of P other than $\mathsf{start}(P)$ and $\mathsf{end}(P)$ are *internal vertices*. The length of P is denoted $\mathsf{length}(P)$. The reverse path is denoted P^R. The path obtained by concatenating paths P and Q is denoted PQ. The u-to-v distance in G is denoted $\mathrm{dist}_G(u, v)$ or just $\mathrm{dist}(u, v)$.

The radius of a graph with respect to a vertex r is the maximum over all vertices v of the v-to-r distance.

We say two intersecting shortest paths P and Q are *uniquely intersecting* if their intersection is a single subpath (possibly consisting of just one vertex) of each of the paths. More generally, we say a set of shortest paths satisfies *unique intersection* if every pair of intersecting paths are uniquely intersecting. Let $\mathcal{S}$ be such a set. We say a path P is uniquely intersecting with respect to $\mathcal{S}$ if $\mathcal{S} \cup \{P\}$ satisfies the property. Given such a set $\mathcal{S}$ and given vertices u and v, there exists a u-to-v path that is uniquely intersecting with respect to $\mathcal{S}$.

3.1 Radial radius and branchwidth For any planar embedded graph, the corresponding *radial graph* is an embedded bipartite planar graph whose vertices correspond to the vertices and faces of the original graph; two vertices of the radial graph are adjacent if they correspond to a vertex and a face of the original graph that are incident. The lengths of the radial graph's edges are all one.

THEOREM 3.1. (TAMAKI [18]) *The branchwidth of a planar graph is bounded by the radius of its radial graph with respect to a vertex.*

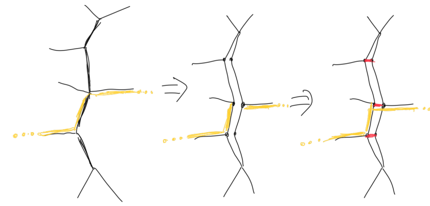

Figure 3: Channelizing a shortest path: slicing the path, duplicating edges and internal vertices; and then rejoining, adding zero-length artificial edges between pairs of duplicated vertices. A path Q that crosses the channelized path corresponds to a path $\widehat{Q}$ after channelization.

We refer to a path in the radial graph as a *radial path*. The *size* of a radial path is its number of edges.

4 Strategy

In this section, we outline the high-level strategy for proving Theorem 1.3. To prove this theorem, we must construct an embedding $\phi(\cdot)$ of the input graph G_{in} into a bounded-branchwidth graph such that, for every pair u, v of vertices of G_{in},

(4.1)
$$\mathrm{dist}_G(u, v) \le \mathrm{dist}_H(\phi(u), \phi(v)) \le \mathrm{dist}_G(u, v) + \epsilon D$$

where D is the diameter of G_{in}.

We state lemmas about the algorithm that we only prove later once we give details of the algorithm. We show that, once those lemmas are proved, the theorem follows.

4.1 Root and radius of the input graph The algorithm designates an arbitrary vertex r of the input graph, which we call the *root*. The algorithm then scales the edge-lengths so that the farthest vertex from r is at distance 1. As a result, the diameter is between 1 and 2. It therefore suffices to establish Inequality 4.1 with ϵD replaced by ϵ. For notational convenience, our proof establishes the inequality with ϵD replaced by a quantity that is at most $c\epsilon$ for some constant c. To obtain the desired quantity, it suffices to provide the algorithm with a parameter $\epsilon' = \epsilon/c$.

Using a simple transformation, we assume that there is a zero-length self-loop that encloses all of the graph except r. We also assume that every edge is a shortest path between its endpoints; other edges can be removed.

4.2 Channelizing The algorithm for Theorem 1.3 maintains a planar embedded graph G and a set LINKS of shortest paths, called *links*. Initially G is the input graph G_{in}. The algorithm repeatedly modifies the graph by performing an operation we call *channelizing*.

Given a shortest path P, channelizing P consists of the following steps:

Slice Replace P with two paths P^+ and P^- that are internally disjoint. Each edge e of P is replaced with two copies, e^+ and e^-, which are each assigned the length of the original edge e. Each internal vertex v of P is replaced with two copies, v^+ and v^-. Every edge in the unmodified graph that is not in P but is incident to an internal vertex v of P remains incident to one of the two copies of v in the modified graph. In the modified graph, the two paths P^+ and P^- form the boundary for a new face. The face is called a *channel* (the channel of P), and the paths P^+ and P^- are called the the *banks* of the channel. The vertices of the banks are called *bank vertices* of the channel.

Rejoin Add zero-length artificial edges. For each vertex v that is replaced with copies v^+ and v^-, an artificial edge is added with endpoints v^+ and v^-. These are the artificial edges of the channel.

Each path Q in G before channelization of P corresponds to an equal-length path $\widehat{Q}$ after, one possibly using an artificial edge of the channel.

Note that if some of the endpoints of Q are internal vertices of P then Q corresponds to two or even four paths after channelization, because those endpoints are duplicated.

We will generally avoid further discussion of the distinction between Q and $\widehat{Q}$. In particular, the description of the algorithm will not explicitly include updating of the paths in LINKS.

4.3 Drawbridges and travel by channel The artificial edges of a channel that also belong to LINKS are called *drawbridges* of the channel. The *cost* of a channel is its number of drawbridges.

THEOREM 4.1. (CHANNEL COST THEOREM) There is a constant c such that every channel has cost $O(\epsilon^{-c})$.

A *concatenation* of channels of size k is a sequence $C_1 e_1 C_2 e_2 \cdots e_{k-1} C_k$ of alternating channels C_i and artificial edges e_i such that, for $i = 1, \dots, k-1$, e_i is incident to bank vertices of C_i and C_{i+1}.

CHANNEL GLOBAL TRAVEL THEOREM. There is a constant d such that, for every channel C, there is a concatenation of channels of size $O(\epsilon^{-d})$ in which the first channel is C and the last channel ends at r.

At any time in the algorithm's execution, we define $G[\text{LINKS}]$ to be the subgraph of G consisting of (i) banks of all channels in G, together with (ii) all edges belonging to links, i.e. paths in LINKS. Note in particular that the only artificial edges in $G[\text{LINKS}]$ are those in LINKS.

Let $\widehat{G}$ and $\widehat{\text{LINKS}}$ denote respectively the graph G and the set LINKS when the algorithm finishes.

LEMMA 4.1. In $\widehat{G}$, every vertex is on the bank of some channel.

It follows from the Channel Cost Theorem and the Channel Global Travel Theorem combined with Lemma 4.1 that the radius of the radial graph of $\widehat{G}[\widehat{\text{LINKS}}]$ is $O(\epsilon^{-c-d})$ where c and d are the constants in those theorems. By Theorem 3.1, we obtain

COROLLARY 4.1. The branchwidth of $\widehat{G}[\widehat{\text{LINKS}}]$ is $O(\epsilon^{-c-d})$.

Each vertex v of G_{in} (called an *original* vertex) maps to at least one vertex $\hat{v}$ in G. It is clear that channelization does not change distances, and therefore that, for any original vertices u and v, the $\hat{u}$-to-$\hat{v}$ distance in $\widehat{G}[\widehat{\text{LINKS}}]$ is at least the u-to-v distance in G_{in}. We will show an approximate converse.

THEOREM 4.2. For each pair u, v of original vertices, the $\hat{u}$-to-$\hat{v}$ distance in $\widehat{G}[\widehat{\text{LINKS}}]$ exceeds the u-to-v distance in G_{in} by $O(\epsilon)$.

This shows that the mapping $v \mapsto \hat{v}$ is a metric embedding of G_{in} into $\widehat{G}[\widehat{\text{LINKS}}]$ with an additive error of $O(\epsilon)$. This together with Corollary 4.1 will prove Theorem 1.3.

4.4 r-path A rootward r-rooted shortest-path tree is maintained throughout the algorithm. For each vertex v, the v-to-r path in the r-rooted shortest-path tree is called the *r-path* of v, and is denoted $\text{rpath}(v)$.

The algorithm ensures that the set $\mathcal{P}$ of r-paths of vertices together with the set of banks satisfies the unique intersection property. Just as with links, in describing the algorithms we do not explicitly describe updating the r-paths after a channelization.

In a slight abuse of terminology, if P is a subpath of the r-path of v, we refer to P as an r-path. If a path that is channelized is an r-path, the resulting channel is said to be an *r-path* channel, and its banks are called *r-path* banks. The other channels and banks are called *non-r-path* channels and banks.

At any given time in the algorithm's execution, the *r-path prefix* of a vertex v is the minimal prefix of v's r-path whose end is a vertex of a non-r-path bank, and that non-r-path bank is called the *crossroad* of v.

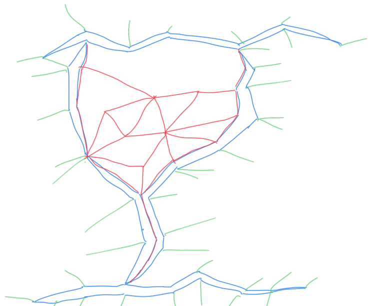

Figure 4: The blue lines are banks of channels. The top and bottom channels are non-r-path channels. The left and right channels are r-path channels. The red lines form a territory. Note that the territory includes the adjacent bank of each of the r-path channels, continuing down to where the r-paths meet the bottom channel (which is a *crossroad*).

4.5 Territories and operations and crossroads

Consider some moment in the execution of the algorithm. Let G be the current graph. Let H be the subgraph of G consisting of the banks and artificial edges of all channels. A face of $G - H$ is called a *region*. A connected component K of $G - H$ is said to be the *interior of a territory* of G, and its vertices are *internal vertices* of the territory. Let f be the region in which K is embedded; the boundary of f consists of maximal subpaths of banks, and each such maximal subpath is called a *side* of the territory.

The territory itself is the subgraph obtained from K by adding

- edges of G incident to K and vertices adjacent to K, and
- r-path prefixes of all the internal vertices and all the vertices adjacent to them.

The vertices of the territory that are not internal vertices are called *external* vertices of the territory.

Suppose uv is an edge that does not belong to a bank and is not artificial, such that u and v belong to banks. Then uv and the r-path prefixes of u and v form a *degenerate* territory. It has no internal vertices.

The algorithm proceeds by divide-and-conquer on territories. Channelizing a path in τ adds to the subgraph of H and thereby gives rise to smaller territories (called *subterritories* of τ). We identify τ with the corresponding subgraph in the modified graph.

Applying an *operation* to τ consists of performing certain channelizations and also adding paths in τ to Links, thereby designating them aslinks. We say the channels created in the operation are *owned* by τ.

The subterritories resulting from applying an operation to τ are considered the children of τ. The algorithm then applies operations to these child territories, and so on. The execution of the algorithm defines a rooted tree, the *territory tree*, in which the nonroot nodes are the territories resulting from operations.

To start the process, we define the initial territory to be the whole graph and define the initial operation to be a special operation that channelizes the self-loop enclosing the whole graph except r (see Figure **??**). We refer to the resulting channel as the *root channel.* It is a non-r-path channel. Of its two banks, we refer to the *inner* bank and the *outer* bank. The outer bank is part of the boundary of the face of G containing r. The inner bank is the crossroad of the territories that are children of the root territory.

The process ends when there are no territories remaining. At this point, there are no interior vertices, which proves Lemma 4.1.

Recall that we defined the crossroad of a vertex v to be the first non-r-path bank encountered by the r-path of v. The following definitions reflect the fact that this changes over time as new paths are channelized. If v is a vertex of a territory τ, we define the *crossroad of v with respect to τ* to be the first non-r-path bank encountered by rpath(v) that existed before the operation on τ. Similarly, we define the *r-path prefix of v with respect to τ* to be the prefix of v's r-path that ends at the intersection of that r-path with the crossroad of v with respect to τ.

The *crossroads* of a territory are the crossroads of all vertices of the territory with respect to the territory. The set of *crossroads* of a vertex v is

$\{$crossroad of v with respect to $\tau : \tau$ a territory containing v that is in territory tree$\}$

and the set of *r-path prefixes* of v is

$\{r$-path prefix of v with respect to $\tau \; : \tau$ a territory containing v that is in the territory tree$\}$

4.6 Approximation and admissibility For an error parameter $\delta > 0$, we say a shortest path P in G is *δ-approximated* by a path P' in G[Links] if P' and P have the same start and end, and the length of P' exceeds the length of P by at most δ. We say more briefly that P is δ-approximated in G[Links] if P is δ-approximated by some path in G[Links]. We leave out the parameter δ and say *approximated* for the case where $\delta = O(\epsilon)$, where ϵ is the error parameter provided as input to the algorithm. We say that a path P' is an approximation of P if P is approximated by P'.

We also introduce a weaker notion of approximation. We say P is *near-approximated* in G[Links] if P is approximated in the graph obtained from G[Links] by adding approximations to the r-path prefixes of P's endpoints. We say P is near-approximated with respect to a territory τ that contains P if P is near-approximated in the graph $G[\textsc{Links}]$ that exists just before the operation on τ.

Near-Approximation Lemma. Let u, v be vertices. Then in $\widehat{G}$ there is a shortest u-to-v path P and a decomposition of P into a constant number of subpaths joined by artificial edges $P = P_1 e_1 P_2 e_2 \cdots e_{k-1} P_k$ such that each subpath P_i is near-approximated in $\widehat{G}[\widehat{\textsc{Links}}]$.

The artificial edges $e_1, \ldots, e_{k-1}$ in the decomposition might not be in $\widehat{\textsc{Links}}$. However, the following lemma ensures this is not a problem.

Lemma 4.2. (Nearby-Drawbridge Lemma) For any vertex u on the bank of a channel, there is a vertex v on the same bank such that the u-to-v distance is $O(\epsilon^2)$ and there is a link crossing the channel at v.

Lemma 4.3. At the end of the algorithm, each r-path prefix is approximated.

We can now prove Theorem 4.2, which shows that every shortest path P is approximated. Consider the decomposition of the Near-Approximation Lemma. Each subpath P_i is near-approximated in $\widehat{G}[\widehat{\textsc{Links}}]$, which means that it is approximated in the graph obtained by adding approximations to the r-path prefixes of the endpoints of P_i. By Lemma 4.3, such approximations exist in $\widehat{G}[\widehat{\textsc{Links}}]$, so each subpath P_i is approximated by some path P'_i. By the Nearby-Drawbridge Lemma (Lemma 4.2), each artificial edge e_i in the decomposition of the Near-Approximation Lemma can be replaced by a path Q_i with the same endpoints that is of length $O(\epsilon^2)$. We obtain a path $P' = P'_1 Q_1 P'_2 Q_2 \cdots Q_{k-1} P'_k$ that approximates P. This completes the proof of Theorem 4.2.

We say a path is *admissible* with respect to a territory τ if (a) it is an r-path or (b) it is a subpath of a path in τ that is *not* near-approximated in the graph $G[\textsc{Links}]$ that exists just before the operation on τ.

Lemma 4.4. If a path P is first added to Links or is channelized in the operation on territory τ then P is admissible with respect to τ.

Lemma 4.5. If P is added to Links by an operation on τ and is subsequently channelized by an operation on τ' then τ is τ' or its parent or grandparent.

4.7 Detour cost Let P be a u-to-v shortest path. An alternative u-to-v route takes a shortest path from u to the root r, then a shortest path from r to v. The difference between the length of this route and the length of the shortest path P is called the *detour cost* of P, written $\mathsf{detour}(P)$. Because the graph G has radius at most one with respect to r, the detour cost is at most two.

Proposition 4.1. Let P be a shortest path. Let R_1 and R_2 be prefixes of $\mathsf{rpath}(\mathsf{start}(P))$ and $\mathsf{rpath}(\mathsf{end}(P))$, respectively, and let S be a shortest $\mathsf{end}(R_1)$-to-$\mathsf{end}(R_2)$ path. If $\mathsf{detour}(P) \le \mathsf{detour}(S) + \epsilon$ then $\mathsf{length}(R_1 \circ S \circ R_2^{\mathsf{rev}}) \le \mathsf{length}(P) + \epsilon$

Using the triangle inequality, we can show

Proposition 4.2. The detour cost of P is no more than the detour cost of any subpath of P.

4.8 Using detour cost to show limited nesting Define the *level* of a path P to be $\lceil \mathsf{detour}(P)/\epsilon \rceil$. (The level of a channel is defined to be the level of the path channelized to form it.) Because detour cost is at most two, we obtain:

Corollary 4.2. No level is greater than $2/\epsilon$.

We say a territory is *simple* if it has only one crossroad and is adjacent to at most two r-path banks.

Proposition 4.3. The children of the root territory are simple.

The proposition shows that every nonroot territory has a simple ancestor.

Corollary 4.3. Let P be a non-r-path that is admissible with respect to a simple territory τ. Then P's level is greater than that of the common crossroad of τ.

Proof. P is a subpath of a path P' in τ that is not near-approximated in G^-_τ. By Proposition 4.1, the detour cost of P' exceeds that of the common crossroad of τ by more than ϵ. By Proposition 4.2, the detour cost of P is at least that of P.

Suppose non-r-path P is first added to Links or channelized during the operation on nonroot territory $\hat{\tau}$. The *predecessor territory* of P is defined to be the closest simple ancestor of $\hat{\tau}$, and the *predecessor* of P is defined to be the crossroad of that closest simple ancestor. These notions are extended in a natural way to the case where P is a channel or the bank of a channel.

Corollary 4.4. For non-r-path bank P, the level of P is greater than that of its predecessor.

Proof. By Lemma 4.4, the path channelized to form the channel of P was admissible with respect to the territory τ whose operation first added it to LINKS or channelized it, and therefore admissible with respect to the leafmost simple ancestor of τ. The corollary then follows by Corollary 4.3.

LEMMA 4.6. (SIMPLICITY LEMMA) For any territory τ, τ or its parent or grandparent is simple.

COROLLARY 4.5. Let P be a non-r-path bank, let τ be the owner of the channel of P, and let τ' be the predecessor territory of P. Then τ' is among the five closest ancestors of τ.

Proof. Let $\hat{\tau}$ be the ancestor of τ in which C is first added to LINKS or channelized. By Lemma 4.5, $\hat{\tau}$ is τ or its parent or grandparent. Let τ' be the leafmost simple ancestor of $\hat{\tau}$. By Lemma 4.6, τ' is $\hat{\tau}$ or its parent or grandparent.

COROLLARY 4.6. Each vertex has $O(\epsilon^{-1})$ distinct crossroads.

Proof. Let $P_1, \ldots, P_k$ be the crossroads of v in the order in which they became banks. For each crossroad P_i, let τ_i be the rootmost territory such that P_i is v's crossroad with respect to τ_i. It follows that the parent of τ_i is the owner of P_i's channel. Let $\hat{\tau}_i$ be the predecessor territory of P_i, and let $\hat{P}_i$ be the predecessor of P_i. Then $\hat{P}_i$ is P_j for some $j < i$. By Corollary 4.5, $i \leq j + 5$. By Corollary 4.4, it follows by induction that, for each i, the level of P_i is at least $i/5$. By Corollary 4.2, no level is greater than $2/\epsilon$, so this completes the proof.

4.9 Approximation of r-path prefixes

The following result is proven in the full version of the paper.

r-PATH SUBPATH APPROXIMATION LEMMA. Let v be a vertex, and let P_1 and P_2 be two distinct r-path prefixes of v that are consecutive (i.e. one operation changed the prefix from P_1 to P_2). Write $P_1 = P_2 Q$. When the algorithm ends, Q is $O(\epsilon^2)$-approximated.

We can now prove Lemma 4.3, which states that every r-path prefix is approximated. Let v be a vertex. Let $P_1, \ldots, P_k$ be v's crossroads in the order in which they appeared. For $i = 1, \ldots, k-1$, write $P_i = P_{i+1} Q_i$. The r-Path Subpath Approximation Lemma implies, via an induction on $i-j$ that $Q_I Q_{i-1} \cdots Q_j$ is $c\epsilon^2(i-j)$-approximated for some constant c. Consider finally P_k. Lemma 4.1 ensures that v is on a bank when the algorithm terminates. If this is a non-r-path bank then P_k is trivial. If it is an r-path bank then P_k is a subpath of this bank. In either case, P_k is 0-approximated. It follows that, for $i = 1, \ldots, k$, $P_k Q_{k-1} Q_{k-2} \cdots Q_i$, which is P_i, is $c\epsilon^2(i-j)$-approximated, and is therefore, by Corollary 4.6, $O(\epsilon)$-approximated.

4.10 Channel Global Travel Recall The Channel Global Travel Theorem. It suffices to bound the size of the concatenation when C is a non-r-path channel because each r-path channel ends on a non-r-path channel. The idea of the proof is to use induction on the level of C. In each operation that channelizes a non-r-path C, we will observe that there is a small concatenation of channels starting at C and leading to a channel of smaller level. This gives us the induction step.

4.11 Channel Cost Define the *level* of a territory τ to be the minimum over all interior vertices v of τ of the level of v's crossroad.

LEMMA 4.7. (ISOLATION LEMMA) For any territory τ, there are $O(1)$ proper ancestors of τ that have the same level as τ and create links that overlap τ.

LEMMA 4.8. (SPARSITY LEMMA) There is a constant c such that, for any nonroot territory τ, the operation on the parent of τ creates $O(\epsilon^{-c+1})$ links that overlap τ.

We can now prove the Channel Cost Theorem (Theorem 4.1). An induction on level using Lemmas 4.7 and 4.8 shows that, for any territory τ, if the level of τ is k then $O(\epsilon^{-c+1}k)$ links overlapping τ were created by operations on proper ancestors of τ. By Corollary 4.2, the total number of links created by operations on proper ancestors of τ that overlap τ is $O(\epsilon^{-c})$.

5 High-level algorithm

In this section, we state lemmas concerning the operations, and we show, based on the lemmas, that paths are approximated. We also start to prove some of the lemmas stated in Section 4.

5.1 Non-r-path boundary segments Let τ be a territory. Consider the state of the graph G just before the operation on τ. Consider the subgraph of G consisting of the inner bank of the root channel, together with banks of non-r-path channels. The territory τ is embedded in one face of this subgraph, called the *non-r-path boundary face* of τ. The face's boundary is called the *non-r-path boundary* of τ. Each maximal subpath of the boundary that is a subpath of a single crossroad is called a *boundary segment* of τ.

5.2 Age of crossroads and messiness Let τ be a territory and let $\hat{\tau}$ be an ancestor of τ. We say $\hat{\tau}$ is a *pre-simple ancestor* of τ if τ has an ancestor that is simple and is a child of $\hat{\tau}$. Note in particular that τ cannot be its own pre-simple ancestor.

Let τ be a territory. We say a crossroad of τ is *mature* with respect to τ if it is a crossroad of a pre-simple ancestor of τ. We say it is *aged* if it is a crossroad of a pre-simple ancestor of a pre-simple ancestor of τ. Note that an aged crossroad is mature.

LEMMA 5.1. Each territory has at most one mature crossroad.

Proof. Let τ^* be the closest simple ancestor of τ. Then τ^* has one crossroad, so any descendant of τ^* has at most one crossroad that was a crossroad of the parent of τ^*.

We say τ is *messy* with respect to $\hat{\tau}$ if

- τ has an aged crossroad and
- operations on some descendants of $\hat{\tau}$'s closest pre-simple ancestor designated links that intersect τ.

We say that τ is *messy* if τ is messy with respect to itself.

LEMMA 5.2. If τ is not messy with respect to its parent, it is not messy.

Proof. Let $\hat{\tau}$ be the parent of τ. Suppose τ is not messy with respect to $\hat{\tau}$. Case 1: τ has no aged crossroad. In this case, τ is not messy with respect to any ancestor, including itself.

Case 2: τ' has an aged crossroad but does not intersect any links designated by operations on descendants of the closest pre-simple ancestor of $\hat{\tau}$. This implies that τ does not intersect any links designated by operations on descendants of the closest pre-simple ancestor of τ itself.

5.2.1 To-boundary near-approximation We say a side of τ is *to-boundary near-approximated* in G[LINKS] if every shortest path in τ from an internal vertex to that side is near-approximated in G[LINKS]. We say a territory is to-boundary near-approximated if every side is to-boundary near-approximated.

5.3 Operations Given a degenerate territory (one consisting of a single edge and no internal vertices), the operation channelizes the edge. The operation yields no children. For nondegenerate territories, there are three operations: SIMPLIFY, CONNECT2BOUNDARY, and ISOLATE. The algorithm follows the following rules. Let τ be a nondegenerate territory τ, and let τ' be its parent.

if τ' was operated on by	then the operation on τ is
special operation	SIMPLIFY
CONNECT2BOUNDARY	ISOLATE if τ is messy else SIMPLIFY
ISOLATE	SIMPLIFY
SIMPLIFY	CONNECT2BOUNDARY

The following lemma therefore implies the Simplicity Lemma (Lemma 4.6).

LEMMA 5.3. Every child resulting from a SIMPLIFY operation is simple.

LEMMA 5.4. Every child resulting from an ISOLATE operation is not messy.

LEMMA 5.5. SIMPLIFY only channelizes r-paths.

The proofs of Lemmas 5.3, 5.4, and 5.5 will be apparent when the operations are described.

The only operation that channelizes a path previously added to LINKS is ISOLATE, and only if that path was added to LINKS during the previous CONNECT2BOUNDARY or SIMPLIFY, which proves Lemma 4.5.

5.4 Proof of Isolation Lemma (Lemma 4.7) Let τ_0 be a territory. For $i = 1, 2, \ldots$, let τ_i be a child of τ_{i-1} with the same level as τ_0. We prove by induction on i that operations on at most four territories among $\tau_0, \ldots, \tau_{i-2}$ designate links that intersect τ_i. This holds trivially if $i \leq 5$. Otherwise, by Lemma 4.6, there exists $j \in \{i-3, i-2, i-1\}$ such that τ_j is τ_{i-1}'s closest simple ancestor, and there exists $k \in j-3, j-2, j-1$ such that τ_k is τ_{j-1}'s closest simple ancestor. The number of territories among $\tau_0, \ldots, \tau_{j-2}$ whose operations designate links intersecting τ_i is at most the number that designate links intersecting τ_j, which is at most four by the inductive hypothesis. Consider the links designated by the operations on $\tau_{j-1}, \ldots, \tau_{i-2}$. If none intersect τ_{i-1} then none intersect τ_i, and we are done. If some intersect τ_{i-1} then τ_{i-1} is messy, so ISOLATE is applied to τ_{i-1}, so τ_i is not messy with respect to τ_{i-1}, so none intersect τ_i, and again we are done.

5.5 Approximation In this section, we address the near-approximation and approximation of paths. Let P be a shortest path, chosen to minimize the number of crossings of channels in the output graph among all shortest paths with the same start and end. Let τ be a territory that includes P.

LEMMA 5.6. If the operation on τ is ISOLATE then P is divided into at most three subpaths by the paths channelized in the operation.

LEMMA 5.7. Suppose P is admissible with respect to τ, and let v_1 and v_2 be the first and last vertices of P that are on paths channelized in the operation on τ. If τ is operated on by SIMPLIFY or CONNECT2BOUNDARY then the v_1-to-v_2 subpath of P is $O(\epsilon^2)$-approximated after the operation.

LEMMA 5.8. Suppose P starts at an internal vertex of τ and ends at a side of τ. If the operation on τ is CONNECT2BOUNDARY then P is near-approximated after the operation.

LEMMA 5.9. Let v be an internal vertex of τ and an internal vertex of a child τ' of τ. Let P be v's r-path prefix with respect to τ, let P' be v's r-path prefix with respect to τ', and write $P = P'Q$. If the operation on τ is CONNECT2BOUNDARY then Q is $O(\epsilon^2)$-approximated after the operation.

LEMMA 5.10. Suppose P is a subpath of an r-path and P starts and ends at external vertices. If the operation on τ is SIMPLIFY then P is a subpath of the bank of a channel after the operation.

5.6 Proof of the Near-Approximation Lemma

Let P be a u-to-v path that is uniquely intersecting with respect to all banks. We show in the full version that P can be decomposed as $P = P_1e_1P_2e_2 \cdots e_{k-1}P_k$ where k is $O(1)$ so that there are corresponding territories $\tau_1, \ldots, \tau_k$ such that, for $i = 1, \ldots, k$, P_i is in τ_i and one of the following holds:

1. The operation on τ_i is SIMPLIFY or CONNECT2BOUNDARY and each endpoint of P_i is on a path channelized in that operation; or
2. the operation on τ_i is CONNECT2BOUNDARY and one endpoint of P_i is an endpoint of P and the other is an external vertex of τ_i.

The near-approximations of the subpaths P_i then follow from Lemmas 5.7 and 5.8.

6 The operations

A key part of the analysis is showing that the operations preserve the following invariants:

- **Boundary Segment Invariant:** Each territory is adjacent to at most $g(\epsilon^{-1})$ boundary segments.
- **Mature Boundary Segment Invariant:** Each territory has at most one boundary segment that is a subpath of a mature crossroad.
- **Single-Bank Invariant:** Each territory is adjacent to at most one bank of each channel.

The key contributor to the number of boundary segments is the ISOLATE operation. This operation channelizes paths that were previously designated as links by SIMPLIFY and CONNECT2BOUNDARY. However, the ISOLATE operation only operates on a territory if it has an aged crossroad. Therefore SIMPLIFY is designed to ensure that few links are designated near a mature crossroad.

6.1 Cages and Bars2Bars

Two noncrossing shortest paths P_1 and P_2 form a *bar* if their starting vertices are the same vertex or are the two endpoints of an edge. In accordance with the tradition in graph theory of mixing metaphors, we refer to the two paths as the *arms* of the bar, and to the shared edge or vertex as the bar's *center*. The *ends* of a bar are defined to be the ends of the two arms. To channelize a bar is to channelize each of the two arms (and, if the center is an edge, that edge). For vertices u and v, we say the bar is a (u, v)-*bar* if one arm is a subpath of a shortest path to u and the other arm is a subpath of a shortest path to v. A *degenerate* bar is one in which one of the arms is degenerate (i.e. consists of a single vertex).

Note that a bar naturally defines a path, the concatenation of: the reverse of P_1, the edge if it exists, and P_2. For notational convenience we sometimes identify the path with the bar. Every subpath of a bar (called a subbar) is a (possibly degenerate) bar.

Let D_1D_2 be a non-self-crossing cycle, and let u, v be vertices. A (u, v)-*cage with respect to* (D_1, D_2) is a family $\mathcal{B}$ of uniquely intersecting, mutually noncrossing (u, v)-bars that are enclosed by D_1D_2 such that

- each bar has an end on D_1 and an end on D_2,
- if each of two bars has the same vertex v as its two ends (and therefore forms a cycle) then one of the bars encloses the other.

A family $\mathcal{P}$ of shortest paths is said to be a set of *bar-to-bar paths* with respect to $\mathcal{B}$ if

- each path in $\mathcal{P}$ starts and ends at a bar of $\mathcal{B}$, and
- the union of $\mathcal{P}$ with the arms of the bars satisfies the unique intersection property.

LEMMA 6.1. (BARS2BARS LEMMA) There is a polynomial-time algorithm BARS2BARS that, given a cage $\mathcal{B}$, a set $\mathcal{P}$ of bar-to-bar paths, and an error parameter $\delta \in (0, 1)$, channelizes the bars of the cage and adds to LINKS some subpaths of paths belonging to $\mathcal{P}$ such that

- (Nearby Drawbridge) Each vertex on the bank of a channel is with distance δ of a drawbridge.
- (Approximation) All paths $P \in \mathcal{P}$ are δ-approximated after applying Bars2Bars.
- (Sparsity) For each bar $B \in \mathcal{B}$, Bars2Bars adds at most $h(1/\delta)$ paths to LINKS that intersect B.

Here $h(\cdot)$ is a polynomial to be determined. Most of the channelizations are done by calls to BARS2BARS; for these, the lemma's *Nearby Drawbrige* part supports the Nearby-Drawbridge Lemma (Lemma 4.2). Other channelizations are done in ISOLATE.

The bound on the number of links intersecting each new channel implies a bound on the number of links intersecting each resulting new territory. Note also that, when the bars of a cage are channelized, each resulting territory is adjacent to at most two banks arising from channelization of those bars, and that these banks preserve the Single-Bank Invariant.

6.2 δ-Crossbars Let τ be a territory. Adding a *δ-crossbar* to τ to connect a given set of sides of τ is as follows: for each of those sides, select a δ-net of vertices to designate as *portals*. For each pair u, v of portals, let P be a u-to-v shortest path that is uniquely intersecting with respect to banks and links. If P is admissible, add it to LINKS (designate P as a link).

Suppose in a territory τ the algorithm applies BARS2BARS to a set of cages. If a resulting subterritory connects banks resulting from channelizing bars from two different cages, we say those cages are *adjacent*. *Joining* two adjacent cages with a δ-crossbar means adding a crossbar connecting portals from each of those two banks.

Similarly, if a side of τ is connected via a resulting subterritory to a bank resulting from channelizing a bar, we say the side is adjacent to the bar. Joining that bar to that side means adding a crossbar connecting the two banks. Joining a cage to the sides of τ means joining each bar of the cage to each of the adjacent sides.

In all the operations, paths are added to LINKS by invoking BARS2BARS or by creating crossbars. In either case, the paths added are admissible, supporting Lemma 4.4.

6.3 Simplify The purpose of SIMPLIFY is to produce territories that are simple.

Applying Simplify to a simple territory First suppose the input territory τ is already simple. In this case, the operation selects an internal vertex v and channelizes its r-path prefix. In each resulting subterritory τ', the new channel C might violate the Single-Bank Invariant; if so, the operation re-establishes the invariant in τ' as follows.

Because τ is simple, it has one crossroad D. Consider the non-r-path boundary of τ. It includes a subpath of D. We write it as $D_1D_2D_3$ where D_1D_2 is the subpath of D, and the end of D_1 coincides with the end of channel C.

Let uv be an edge whose vertices are internal to τ'. Because the r-path prefixes of u and v end on B, these prefixes, together with the edge uv and a subpath of B, form a cycle. We say uv is *bichromatic* if that cycle encloses the channel C. The corresponding bar is formed by uv together with those r-path prefixes. Let $\mathcal{B}$ be the set of bars corresponding to bichromatic edges. Because the bars are built from r-path prefixes, they are mutually noncrossing. These bars form an (r, r) cage with respect to (D_1, D_2D_3).

The operation calls BARS2BARS on this cage with $\delta = \epsilon$. Lemma 6.1 implies that Lemma 5.7 is satisfied. No non-r-path channels are formed (supporting Lemma 5.5), and channelization of the bars corresponding to bichromatic edges implies that each child τ'' of τ' lacks bichromatic edges, which shows that τ'' is simple. The channelizations done by BARS2BARS are of r-path subpaths and the links added by BARS2BARS are all admissible, so the operation is consistent with Lemma 4.4. The number of links formed is $h(1/\epsilon)$.

Applying Simplify to a nonsimple territory Now suppose τ is not simple. Again, let uv be an edge whose vertices are internal to τ. Now we say uv is *bichromatic* if the r-path prefixes of u and v end on distinct boundary segments. The bar corresponding to a bichromatic edge uv consists of the edge and those two prefixes. If u is itself on a boundary segment, the bar is degenerate. As before, the bars corresponding to bichromatic edges are mutually noncrossing.

Overview Unlike the case when τ starts out simple, in this case we need multiple cages to account for all the bichromatic edges. In particular, there is a cage for each boundary segment, and a call to BARS2BARS for each cage, resulting in channelization of the bars of that cage and the designation of links. Because each bar is channelized, no child territory contains a bichromatic edge, which implies that each child territory is simple.

We use the invariant to bound the number of cages by $g(\epsilon^{-1})$. The operation also must add δ-crossbars between cages in order to approximate paths that cross multiple cages. Furthermore, the error parameter δ (also passed to BARS2BARS) must be set so that a path crossing multiple cages is approximated with a total error of $O(\epsilon)$. We therefore set the error parameter δ to $\epsilon/g(\epsilon^{-1})$. Finally, as mentioned at the beginning of Section 6, care is needed to ensure that not too many links be introduced in the vicinity of a mature crossroad.

Part 1 Say an arm of a bar and the bar itself are *mature* if the arm ends at a mature crossroad.

Part 1 of the operation is as follows. Initialize

R with the set of bars corresponding to bichromatic edges. Consider the boundary segments of τ, starting with the mature crossroad if τ has one (here we use the Mature Boundary Segment Invariant). For each boundary segment S in turn, assign to B_S the set of bars in R with ends in S, and remove these bars from R. The bars in B_S form a cage with respect to (D_1, D_2) where $D_1 = S$ and $D_1 D_2$ is τ's entire non-r-path boundary. The cage B_S corresponding to the mature crossroad S, if it exists, is called the *mature cage*.

Part 2 Before describing Part 2, we explain the motivation behind it. Let P be any shortest path in τ that starts and ends at bars and that is not approximated with respect to τ. Let P_0 be the maximal prefix of P that ends at a bar of the same cage at which it starts. For $i = 1, 2, \ldots$, let Q_i be the minimal subpath of P that starts at the end of P_{i-1} and stops at a vertex belonging to a different cage (Q_i might be a trivial path) and let P_i be the maximal subpath of P that starts at the end of Q_i and ends at a bar of the same cage. This process stops at $i = k$ when no such Q_k exists.

Then the decomposition $P = P_0 Q_1 \cdots Q_{k-1} P_k$ has the following properties:

- No internal vertex of Q_i belongs to any cage.
- No bichromatic edge belongs to Q_i and therefore the endpoints of Q_i are on arms that end at the same boundary segment.
- $k + 1$ is at most the number of cages.
- Every P_i and every Q_i is a subpath of non-near-approximated path P and is therefore admissible.

In Part 2, if the mature cage exists, the Simplify operation first invokes Bars2Bars on the mature cage to ϵ-approximate all admissible paths between mature bars. Then, for each remaining cage, the operation invokes Bars2Bars on that cage to $\epsilon/g(\epsilon^{-1})$-approximate all admissible paths between the bars of that cage.

As a consequence, by Lemma 6.1, the bars are channelized and each path P_i is $\epsilon/g(\epsilon^{-1})$-approximated in G[Links], or is ϵ-approximated if P_i starts and ends at bars of the mature cage. It is still necessary that each Q_i be $\epsilon/g(\epsilon^{-1})$-approximated. This is the job of Part 3.

Part 3 Part 3 is as follows. Consider each pair of cages for which there exists a cage-to-cage path that has no internal vertices belonging to cages. Every bichromatic edge belongs to a bar, so such a path must connect two bars whose ends lie on a common boundary segment. By planarity, there are only two bars B_1 and B_2, one from each of the two cages, such that such a path connects B_1 and B_2. Add a δ-crossbar joining B_1 and B_2 where δ is ϵ if one of the cages is mature and is $\epsilon/g(\epsilon^{-1})$ otherwise.

Channel-to-channel approximation Now we complete the proof that P is near-approximated. The crossbars ensure that each Q_i is $\epsilon/g(\epsilon)$-approximated (or ϵ-approximated, if an endpoint of Q_i belongs to a mature cage). This shows that Simplify is consistent with Lemma 5.7.

The links added by Bars2Bars and the crossbar links are all admissible, so the operation is consistent with Lemma 4.4.

Link sparsity We must bound the number of links intersecting each child τ'. By Lemma 6.1, each call to Bars2Bars designates at most $h(\epsilon/g(\epsilon))$ links that intersect τ', for a total of at most $g(\epsilon) \cdot h(\epsilon/g(\epsilon))$. Each portal-to-portal link passes through a single child (and touches the boundaries of at most two others), and each bar has length at most two so the number of portals on each bar is at most $2\epsilon/g(\epsilon)$. Therefore each child τ' is intersected by at most $(2\epsilon/g(\epsilon))^2$ portal-to-portal links. Thus in total each child is intersected by at most $g(\epsilon) \cdot h(1, \epsilon/g(\epsilon)) + g(\epsilon) \cdot h(\epsilon/g(\epsilon))$ links. Since $h(\cdot)$ and $g(\cdot)$ are polynomials, this gives the bound in the Sparsity Lemma (Lemma 4.8).

Link sparsity for children with aged crossroads The following will help us in showing the Boundary Segment Invariant is preserved:

Lemma 6.2. Let τ' be a child of a territory τ operated on by Simplify. If τ' has an aged crossroad then the operation introduced at most $h(\epsilon)$ links intersecting τ'.

Proof. If τ' included a path to an immature arm then that path would contain a bichromatic edge, a contradiction. Thus no path between immature bars intersects τ'. This shows that only the first call to Bars2Bars could have introduced links that intersect τ', and by Lemma 6.1 and the choice of error parameter used in that call, at most $h(1, \epsilon)$ links resulting from that call intersect τ'. Moreover, no crossbar path from Part 3 intersects τ'.

Channel travel Because Simplify creates only r-path channels, it is not relevant to the inductive proof of the Channel Global Travel Theorem (Theorem **??**), except to note that each channel created ends at a crossroad.

Other properties The Mature Boundary Segment Invariant is preserved because Simplify only channelizes r-paths. Channelization of degenerate bars (corresponding to an edge uv where u is on a non-r-path boundary segment) implies Lemma 5.10.

6.4 Connect2Boundary

The *crossing* semiorder and crossing-net Consider a graph. Fix vertices r and s. We define a relation on other vertices. For notational convenience, for this section we assume shortest paths are unique, and we represent the unique u-to-v shortest path by $P(u,v)$. We say that $u \leq_{\chi_{rs}} v$ iff $P(v,r)$ intersects $P(u,s)$. The next two propositions show that $\leq_{\chi_{rs}}$ is a semiorder.

PROPOSITION 6.1. *If $u \leq_{\chi_{rs}} v$, then $\mathsf{detour}(P(u,s)) \leq \mathsf{detour}(P(v,s))$.*

PROPOSITION 6.2. *If $u \leq_{\chi_{rs}} v$ and $\mathsf{detour}(P(u,s)) = \mathsf{detour}(P(v,s)$ then $v \leq_{\chi_{rs}} u$*

PROPOSITION 6.3. *Given a graph, two vertices r and s, and a subset A of the other vertices, there is an algorithm to select a subset X of A such that*

1. *Vertices of X are pairwise incomparable with respect to $\leq_{\chi_{rs}}$, and*
2. *for any vertex v in A but not in X, there is a vertex u in X such that $u \leq_{\chi_{rs}} v$.*

Proof. The algorithm is a simple greedy covering algorithm. Initialize X to the empty set. Then repeat the following step: among all vertices $v \in A$ for which there is no vertex $u \in X$ for which $u \leq_{\chi_{rs}} v$, choose v to minimize $\mathsf{detour}(P(u,s))$. The algorithm terminates when no candidate vertices v remain.

The termination condition proves Property 2. We prove Property 1 by contradiction. Suppose $u, v \in X$ are comparable. Assume without loss of generality that u was the first of the two to be added to X. Because v was a candidate for addition, $u \not\leq_{\chi_{rs}} v$. Since u and v are comparable, therefore, $v \leq_{\chi_{rs}} u$. By Proposition 6.1, $\mathsf{detour}(P(v,s)) \leq \mathsf{detour}(P(u,s))$. Because u was chosen by the algorithm before v, it must be that $\mathsf{detour}(P(v,s)) = \mathsf{detour}(P(u,s))$. By Proposition 6.2, therefore $u \leq_{\chi_{rs}} v$, a contradiction.

Part 1 of Connect2Boundary Let τ_0 be the territory to which the operation is applied. Part 1 of the CONNECT2BOUNDARY operation is to select portals from among the external vertices. The set of portals should consist of an ϵ^2-net on each adjacent r-path bank, and also, if the crossroad is immature, an ϵ^2-net on the crossroad. Because τ_0 is simple, it is adjacent to at most three banks, so the number of portals is $O(\epsilon^{-2})$.

Part 2 For each portal s, and for each interval $\mathcal{I}$ among the intervals $[0,\epsilon), [\epsilon, 2\epsilon), \ldots, [2-\epsilon, 2]$, the operation defines $A_s^{\mathcal{I}}$ to be the set of vertices v of τ_0 such that $\mathsf{detour}(P(v,s)) \in \mathcal{I}$ and $P(v,s)$ is admissible. The operation then uses the algorithm of Proposition 6.3 to select a subset $X_s^{\mathcal{I}}$ of $A_s^{\mathcal{I}}$.

Consider the pairs $(s, \mathcal{I})$ in some arbitrary order. Let k be the number of pairs, and note $k = O(\epsilon^{-3})$. For $i = 1, 2, \ldots, k$, let $(s, \mathcal{I})$ be the i^{th} pair, and let $\mathcal{B}_i$ be the set of bars whose arms are the shortest v-to-s path and v's r-path prefix, chosen to preserve the unique intersection property. We claim that the arms in $\mathcal{B}_i$ are mutually noncrossing. Certainly the r-paths do not cross and the shortest paths to s can be chosen so as not to cross. Because each pair $u, v \in X_s^{\mathcal{I}}$ are incomparable with respect to $\leq_{\chi_{rs}}$, the path heading to r does not cross the path heading to s.

For any i, consider a component K obtained from the interior of τ_0 by deleting vertices of $\mathcal{B}_i$. Then K is adjacent to at most two bars of $\mathcal{B}_i$. We call this the *single-cage limited-adjacency property*. Next, consider the graph consisting of bars in $\mathcal{B}_1, \mathcal{B}_2, \ldots, \mathcal{B}_i$. Each bar consists of two shortest paths; let $\mathcal{P}$ be the set of all these shortest paths, and note that $|\mathcal{P}| = 2i$. Say a vertex v of this graph is *special* if there exist $P, Q \in \mathcal{P}$ such that v is the first or last vertex of P that is also on Q. By the unique intersection property, the number of special vertices is at most $4i^2$.

Combining this bound with the single-cage limited-adjacency property (and the fact that τ_0 is simple), it follows that if the CONNECT2BOUNDARY operation were to channelize the bars in $\mathcal{B}_1, \mathcal{B}_2, \ldots, \mathcal{B}_i$, each resulting territory would have $O(i^2)$ adjacent sides. We call this the *multi-cage limited-adjacency property*. The operation cannot channelize all those bars at once because the arms cross each other. Instead, in the next section we describe how to iteratively do the equivalent.

Part 3 Part 3 consists of a sequence of iterations, each of which consists of some channelizations and designation of links, dividing territories into subterritories. For iteration $i = 1, 2, \ldots, k$, we proceed as follows:

Consider each territory τ resulting from the previous iterations. Let $\mathcal{B}_i^{\tau}$ be the set of intersections of the bars of $\mathcal{B}_i$ with τ. For each adjacent side S of τ, form a cage $\mathcal{B}_i^{\tau,S}$ consisting of those bars in $\mathcal{B}_i^{\tau}$ that have ends on S and that have not already been assigned to a cage. (The fact that these bars all have ends on S ensures that they indeed form a cage.) Call BARS2BARS on each cage in turn to ϵ^2/k^3-approximate all admissible bar-to-bar paths. The multi-cage limited-adjacency property ensures that the number of adjacent sides of τ (and therefore the number of cages) is $O(k^2)$.

Next, δ-crossbars are added for $\delta = O(\epsilon^2/k^3)$ to connect between adjacent pairs of cages and between each cage and immature adjacent sides. This completes the description of an iteration of Part 3.

Note: only the final territories are considered the children of τ for the purpose of defining the territory tree.

Number of adjacent boundary segments Initially, the territory τ_0 has at most adjacent sides. Each iteration adds $O(k^2)$ sides, and there are k iterations, so the number of sides of each territory resulting from CONNECT2BOUNDARY is $O(k^3)$, which is $O(\epsilon^{-9})$.

Channel-to-channel approximation

LEMMA 6.3. For any vertex u on a bank of a bar from $\mathcal{B}_i$ and any vertex v on a bank of a bar from $\mathcal{B}_j$, if the shortest u-to-v path in τ was admissible before the operation then it is $O((i+j)\epsilon^2/k)$-approximated in $G[\text{LINKS}]$ after the operation.

The proof is a straightforward induction on $i+j$, using the δ-approximation guarantee of the BARS2BARS Lemma (Lemma 6.1), the $O(k^2)$ bound on the number of cages for a territory τ, and the crossbar construction.

Lemma 6.3 in turn proves Lemma 5.9 and shows that CONNECT2BOUNDARY is consistent with Lemma 5.7.

To-boundary near-approximation We show that τ is to-boundary near-approximated after the operation, proving Lemma 5.8. First, note that if τ has a mature crossroad, then that crossroad is already to-boundary near-approximated before the operation. Consider an adjacent side that is not a mature crossroad, and let x be a vertex on that side.

By the choice of portals in Part 1, there is a portal s on that side such that the x-to-s distance along the side is at most ϵ^2. Let v be an internal vertex. If the shortest v-to-s path in τ is near-approximated before the operation, we are done, so assume it is not. By the claim from Part 2, one of the cages contains a bar with an arm $P(u,s)$ such that a prefix of v's r-path together with a suffix of $P(u,s)$ form a v-to-s path whose length exceeds $P(v,s)$ by at most ϵ.

Let w be the first vertex on v's r-path that belongs to a bank of a new channel. Note that w comes no later in v's r-path than the intersection with $P(u,s)$. Therefore, the v-to-s path consisting of the shortest v-to-w path followed by the shortest w-to-s path has length exceeding $\text{dist}(v,s)$ by at most ϵ. By Lemma 6.3, the shortest w-to-s path is in turn $O(\epsilon^2)$-approximated in $G[\text{LINKS}]$. We conclude that $P(v,s)$ is near-approximated in $G[\text{LINKS}]$.

Channel travel CONNECT2BOUNDARY is applied to a simple territory τ_0. By an induction, for each channel C introduced in iteration i, there is a concatentation of $O(i)$ channels where the first channel is C and the last channel ends at the common crossroad of τ_0. By Corollary 4.3, any path admissible with respect to τ_0 has level greater than the common crossroad of τ_0. This supports using induction on level to prove the Channel Global Travel Theorem (Theorem **??**). The proof must also take into account the channelizations done by ISOLATE, discussed later.

Link sparsity Consider Iteration i of Part 3. It includes $O(k^2)$ calls to BARS2BARS applied to a territory τ. Each call designates $h(1/\delta)$ links, where $\delta = \epsilon^2/k^3$. There are k iterations. Therefore the total number of links designated by all these calls is $k \cdot h(k^3/\epsilon^2)$, which is $O(\epsilon^{-3}h(\epsilon^{-11}))$.

We must also account for the links contributed by crossbars. There are $O(k^2)$ cages and $O(k^2)$ sides, so the number of portals is $O(k^2/\delta)$, so the number of links is $O(k^4/\delta^2)$, which is $O(\epsilon^{-16})$. This is consistent with the Sparsity Lemma (Lemma 4.8).

Other properties Because in Part 1 no portals are selected for a mature side, no non-r-path bar ends at a point on the interior of the mature side. This shows that the operation preserves the Mature Boundary Segment invariant.

A path is channelized only if admissible, and all paths included in crossbars or designated as links calls to BARS2BARS are admissible, as required by Lemma 4.4.

6.5 Isolate Let τ be a territory. Let $\mathcal{L}$ be the set of links designated by descendants of τ's closest pre-simple ancestor. Recall that ISOLATE is only applicable to τ if τ is messy, i.e. if τ has an aged crossroad C and if $\mathcal{L}$ intersects τ.

Say an internal vertex v of τ is aged if the crossroad of v is C and the r-path prefix of v does not go through a vertex of $\mathcal{L}$. Let $\hat{\tau}$ be the interior of τ. Say a connected component of $\hat{\tau}-\mathcal{L}$ is aged if it contains an aged vertex. Let H be the planar embedded subgraph of G consisting of banks of channels and paths in $\mathcal{L}$. Each component of $\hat{\tau}-\mathcal{L}$ is embedded in some face of H. We say a face is aged if an aged component of $\hat{\tau}-\mathcal{L}$ is embedded in that face. The boundary of such a face consists of subpaths of banks of channels and subpaths of paths in $\mathcal{L}$. The ISOLATE operation channelizes each subpath of a path in $\mathcal{L}$ that is part of the boundary of an aged face, and designates as links an ϵ-net of artificial edges belonging to each of the resulting channels in order to satisfy the Nearby Drawbridge Lemma (Lemma 4.2).

Properties Note that, as a consequence,

- each aged component becomes the interior of a single child territory,

- only these child territories have an aged crossroad, and
- each such child territory intersects no links (because no link is embedded in an aged face).

Therefore no child territory is messy, proving Lemma 5.4. Because every link is admissible, the paths channelized are admissible, satisfying Lemma 4.4.

We observed that each territory resulting from CONNECT2BOUNDARY has $O(\epsilon^{-9})$ sides. Due to ISOLATE, the number of sides could increase by at most the number of links intersecting a territory, which we saw is $O(\epsilon^{-3}h(\epsilon^{-11}) + \epsilon^{-16})$. This is an upper bound on the number of adjacent boundary segments because each r-path side connects to a single boundary segment. Thus we can preserve the Boundary Segment Invariant by choosing the function $g(\cdot)$ to be at least this quantity. Moreover, the upper bound on the number of new channels shows that this operation obeys the Channel Global Travel Theorem (Theorem **??**).

We prove Lemma 5.6. We denote by $L_1, \ldots, L_k$ the subpaths of paths in $\mathcal{L}$ that were channelized by the ISOLATE operation. For $i = 1, \ldots, k$, if P crosses L_i then write $P = P_iQ_iR_i$ or $P = R_iQ_iP_i$ where Q_i is a subpath of L_i and P_i includes some aged vertices. Let $\hat{P}$ be the intersection $\bigcap_i P_i$. It is connected and is therefore a subpath of P. We write $P = Q\hat{P}R$, which proves Lemma 5.6.

7 The Bars2Bars algorithm

7.1 Preliminaries for the subroutines

7.1.1 Linear Nesting First, we state some straightforward properties of this ordering that encapsulate our use of planarity in constructing the subroutines. We defer the proofs of Propositions 7.1,7.2, and 7.3 to the full version. A cage $\mathcal{B}$ has a natural ordering of bars $B_1, B_2, \ldots, B_k$ of bars induced by enclosure with respect to the cycle D_1D_2. Let $\mathcal{B}_P$ denote the subset of bars that a bar-to-bar path P intersects. We start by showing that a bar-to-bar path intersects an interval of bars.

PROPOSITION 7.1. *For a cage $\mathcal{B} = \{B_1, B_2, \ldots, B_k\}$ and bar-to-bar x-to-y path P with vertices $v_1, v_2, \ldots, v_t$, $\mathcal{B}_P = \{B_i, B_{i+1}, \ldots, B_j\}$ for some $i \leq j$.*

We use this proposition to prove a more general result that also reasons about the order of intersection. For a cage $\mathcal{B}$ with bar-to-bar path P with vertices $v_0, v_1, \ldots, v_t$ in order, call a *bar assignment* a list of pairs $(v_{i_0}, B_{i_0}), (v_{i_1}, B_{i_1}), \ldots, (v_{i_s}, B_{i_s})$ in which $v_{i_0} = v_0$, $v_{i_s} = v_t$, and for which either $v_{i_j} = v_{i_{j+1}}$ or $v_{i_{j+1}} = v_{i_j+1}$ for all $j \in \{0, 1, \ldots, s-1\}$. Notice that a bar assignment may assign vertices to multiple bars: this is because bars may share vertices. We now show that a "continuous" bar assignment exists:

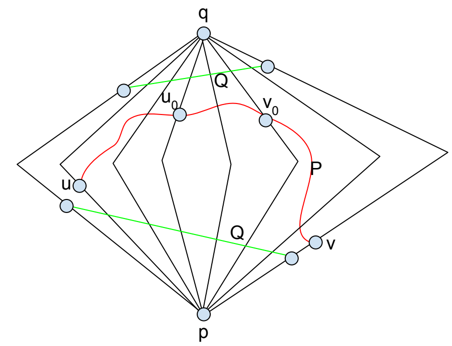

Figure 5: Depiction of Proposition 7.3, with both possibilities for Q drawn in green.

PROPOSITION 7.2. *For a cage $\mathcal{B} = \{B_0, \ldots, B_k\}$, any bar-to-bar path $\mathcal{P}$ has a bar assignment $\{(v_{i_j}, B_{i_{j+1}})\}_{j=0}^{s-1}$ in which B_{i_j} and $B_{i_{j+1}}$ are equal or consecutive for all j; i.e. $B_{i_{j+1}} \in \{B_{i_j-1}, B_{i_j}, B_{i_j+1}\}$.*

In Bars2Bars, we divide paths up into ones that cross each bar at most once, as such paths are easier to approximate. We say that P is $\mathcal{B}$-*increasing* (resp. $\mathcal{B}$-decreasing) if P has a bar assignment in which $B_{i_{j+1}} \in \{B_{i_j+1}, B_{i_j}\}$ for all j (resp. $B_{i_{j+1}} \in \{B_{i_j-1}, B_{i_j}\}$). P is $\mathcal{B}$-*monotone* if it is $\mathcal{B}$-increasing or $\mathcal{B}$-decreasing. Notice that subpaths of $\mathcal{B}$-monotone paths are also $\mathcal{B}$-monotone. Paths constrain one another topologically:

PROPOSITION 7.3. *Let $\mathcal{B}$ be an s-t-cage. Consider two shortest paths P and Q for which the arms of $\mathcal{B}$, P, and Q together form a uniquely intersecting set of shortest paths. Suppose that both endpoints u and v of P are on the $p \in \{s,t\}$ arm of a bar in $\mathcal{B}$ and that P intersects the $q \neq p \in \{s,t\}$ side of some bar. Let u_0 and v_0 denote the closest vertices to u and v on P respectively that are on q-arms of bars.*

Suppose that Q does not intersect P and that Q intersects bars containing u_0, v_0, u, and v. Then at least one of the following occurs: (1) Q intersects an arm containing u_0 and an arm containing v_0, or (2) Q intersects an arm containing u and an arm containing v.

Furthermore, any shortest path P can be split into a small number of monotone segments. This does not require planarity directly; it only requires Proposition 7.2 and for P to uniquely intersect each bar.

PROPOSITION 7.4. *Let $\mathcal{B}$ be a cage and let P be a shortest path that uniquely intersects the bars of $\mathcal{B}$. Then, there exists a decomposition of P into $O(1)$ edge-disjoint* subpaths $\{P_i\}_i \leftarrow$ MONOTONESPLIT$(P, \mathcal{B})$ for which each P_i is $\mathcal{B}$-monotone.

7.1.2 Bucketing All of the subroutines start by bucketing the input set of shortest paths $\mathcal{P}$ with respect to their lengths and the distance to endpoints of two cages. Given a set of paths $\mathcal{P}$ with start and end chosen, four vertices s_0,t_0,s_1, and t_1, and an error parameter $\delta \in (0,1)$, $\mathsf{Buckets}(\mathcal{P}, s_0, t_0, s_1, t_1, \delta)$ returns a partition of $\mathcal{P}$ into δ^{-5} sets $\{\mathcal{P}_{a,b,c,d,e}\}_{a,b,c,d,e\in\{0,1,\dots,\lceil\delta^{-1}\rceil-1\}}$, where each $\mathcal{P}_{a,b,c,d,e}$ is the set of shortest paths $P \in \mathcal{P}$ with start point x and end point y and the following properties:

- (Length bucketing) $\text{dist}(x,y) \in (a\delta, (a+1)\delta]$
- (s endpoint distance bucketing) $\text{dist}(s_0,x) \in (b\delta,(b+1)\delta]$ and $\text{dist}(s_1,y) \in (c\delta,(c+1)\delta]$
- (t endpoint distance bucketing) $\text{dist}(t_0,x) \in (d\delta,(d+1)\delta]$ and $\text{dist}(t_1,y) \in (e\delta,(e+1)\delta]$

The subscripts in the partition can be abstracted away due to the following proposition, which says that cage-to-cage shortest paths within a bucket approximate one another. Fix an (s_0,t_0)-cage $\mathcal{B}_0$ and an (s_1,t_1)-cage $\mathcal{B}_1$. For a $\delta > 0$, a $\mathcal{B}_0$-to-$\mathcal{B}_1$-path P, and a set of $\mathcal{B}_0$-to-$\mathcal{B}_1$ paths $\mathcal{P}$, let $\mathcal{P}_\delta(P)$ denote the union of the buckets $B \in \mathsf{Buckets}(\mathcal{P}, s_0, t_0, s_1, t_1)$ that contain a subpath of P. We now show that any path in $\mathcal{P}_\delta(P)$ can be used to approximate P:

PROPOSITION 7.5. *Let $\mathcal{B}_0$ and $\mathcal{B}_1$ be s_0-t_0 and s_1-t_1-cages respectively, $\mathcal{P}$ be a set of $\mathcal{B}_0$-to-$\mathcal{B}_1$-paths, $\delta > 0$, P be an x-y $\mathcal{B}_0$-to-$\mathcal{B}_1$ path, P_x be an arm in $\mathcal{B}_0$ containing x, and P_y be an arm in $\mathcal{B}_1$ containing y. Then*

$$P_x \cup P_y \cup Q$$

contains a path that 3δ-approximates P for any $Q \in \mathcal{P}_\delta(P)$.

We only apply the above proposition in two cases:

1. $\mathcal{B}_0 = \mathcal{B}_1$. We use this case to approximate bar-to-bar paths in a cage.
2. $\mathcal{B}_1$ is a single shortest path B with endpoints s_0 and t_0. We use this case to approximate shortest paths from bars in a cage $\mathcal{B}_0$ to vertices on B.

Proof. Let w and z be arbitrary vertices in the sets $P_x \cap Q$ and $P_y \cap Q$ respectively. Let Q_0, Q_1, and Q_2 be the $x-w$ subpath of P_x, the $w-z$ subpath of Q, and the $z-y$ subpath of P_y respectively. To prove the proposition, it suffices to show that the concatenation of Q_0, Q_1, and Q_2 3δ-approximates P.

Let p be the endpoint of P_x and q be the endpoint of P_y. Let $\sigma_p \in \{-1,1\}$ be 1 if x is between w and p along P_x and -1 otherwise. Similarly, let $\sigma_q \in \{-1,1\}$ be 1 if y is between z and q along P_y and -1 otherwise. Consider the detour cost-related quantity

$$\mathsf{detour}'(a,b) := \sigma_p \text{dist}(a,p) + \sigma_q \text{dist}(b,q) + \text{dist}(a,b)$$

defined for any pair of vertices a, b with a on a bar in $\mathcal{B}_0$ and b on a bar in $\mathcal{B}_1$. For a path R with start point a_0 and end point b_0, define $\mathsf{detour}'(R) := \mathsf{detour}(a_0,b_0)$. For any two vertices c, d on a shortest path between a and b in the order $a-c-d-b$, note that $\mathsf{detour}'(a,b)$ is $\sigma_p\text{dist}(a,p)+\sigma_q\text{dist}(b,q)+\text{dist}(a,b)$, which is $(\sigma_p\text{dist}(a,p) + \text{dist}(a,c)) + (\sigma_q\text{dist}(b,q) + \text{dist}(d,b)) + \text{dist}(c,d)$.

For each $u \in \{p,q\}$ and all vertices v, v', $\sigma_u\text{dist}(v,u) + \text{dist}(v,v') \le \sigma_u\text{dist}(v',u)$ by the triangle inequality. Therefore, $\mathsf{detour}'(a,b) \ge \sigma_p\text{dist}(c,p) + \sigma_q\text{dist}(d,q) + \text{dist}(c,d) = \mathsf{detour}'(c,d)$.

Furthermore, since detour' is a sum of three signed path lengths, detour' differs by at most 3δ for endpoint pairs of paths in the same bucket. Therefore, since Q is in a bucket for a subpath R of P, $\mathsf{detour}'(w,z) \le \mathsf{detour}'(Q) \le \mathsf{detour}'(R) + 3\delta \le \mathsf{detour}'(x,y) + 3\delta$. Therefore, the length of the detour path $Q_0Q_1Q_2$ is

$$\begin{aligned}
&\text{dist}(x,w) + \text{dist}(w,z) + \text{dist}(z,y)\\
&\quad= (-\sigma_p\text{dist}(p,w) + \text{dist}(x,w))\\
&\qquad +(\text{dist}(w,z) + \sigma_p\text{dist}(p,w) + \sigma_q\text{dist}(q,z))\\
&\qquad +(-\sigma_q\text{dist}(q,z) + \text{dist}(y,z))\\
&\quad= -\sigma_p\text{dist}(p,x) + \mathsf{detour}'(w,z) - \sigma_q\text{dist}(q,y)\\
&\quad\le -\sigma_p\text{dist}(p,x) + \mathsf{detour}'(x,y) + 3\delta - \sigma_q\text{dist}(q,y)\\
&\quad= \text{dist}(x,y) + 3\delta
\end{aligned}$$

7.1.3 Sparse Covers To exploit Proposition 7.5, it helps to add a subset of each bucket that "sparsely" intersects the bars.

PROPOSITION 7.6. *Given an cage $\mathcal{B}$ and a set of bar-to-bar paths $\mathcal{P}$, there is a subset $\mathcal{Q}$ such that*

- *(Coverage) Each bar intersected by some path in $\mathcal{P}$ is intersected by some path in $\mathcal{Q}$*
- *(Sparsity) For each $P \in \mathcal{P}$, the bars $\mathcal{B}_P$ intersected by P are intersected by $O(1)$ paths in $\mathcal{Q}$.*

There is a greedy set-cover algorithm to find $\mathcal{Q}$: each path in $\mathcal{P}$ stands for the set of bars that it intersects:

- $\mathsf{SparseCover}(\mathcal{P}, \mathcal{B})$:
- $\mathcal{Q} \leftarrow \emptyset$
- While there exists a path in $\mathcal{P}$ that intersects an bar in $\mathcal{B}$ that does not intersect any path in $\mathcal{Q}$
  - $\mathcal{B}' \leftarrow$ the set of bars intersected by paths in $\mathcal{Q}$

– $P \leftarrow$ the path in $\mathcal{P}$ that intersects the most bars in $\mathcal{B} - \mathcal{B}'$
– Add P to $\mathcal{Q}$

The algorithm terminates because $\mathcal{Q}$ increases in size in each iteration. The while loop condition ensures the *Coverage* condition upon termination. To establish the *Sparsity* condition, focus on a path $P \in \mathcal{P}$. Any path added to $\mathcal{Q}$ that intersects $\mathcal{B}_P$ must intersect an endpoint bar of P; otherwise P would have been a better choice. The first two paths added that intersect $\mathcal{B}_P$ must intersect both endpoint bars of P; also by the greedy condition. $\mathcal{B}_Q \not\subseteq B_{Q'}$ and $\mathcal{B}_{Q'} \not\subseteq \mathcal{B}_Q$ for any $Q \neq Q' \in \mathcal{Q}$ by the greedy condition. Therefore, P is a better choice than any other third path that could intersect P, so $\mathcal{B}_P$ is covered after adding three paths to $\mathcal{Q}$ that intersect $\mathcal{B}_P$.

7.1.4 Crossbars Let $\mathcal{P}$ be a set of shortest paths between pairs of bars in a set $\mathcal{B}$. Let $\delta(0, 1)$. $\mathsf{Crossbars}(\mathcal{P}, \mathcal{B}, \delta)$ goes through each pair of adjacent bars B_i, B_{i+1} and arbitrarily chooses subpaths of paths in $\mathcal{P}$ that only intersect the face between B_i and B_{i+1} that are not yet δ-approximated to add to a set $\mathcal{Q}$ until all such paths are δ-approximated. $\mathcal{Q}$ contains at most $16\delta^{-2}$ paths that intersect any given bank for a channelized cage $\mathcal{B}$, as any two paths with endpoints within distance $\delta/2$ of one another must $2(\delta/2) = \delta$-approximate one another. Each path has two endpoints chosen from a set of size $2/(\delta/2) = 4/\delta$, leading to at most $(4/\delta)^2 = 16/\delta^2$ possible paths intersecting any given bank.

7.2 Bars to P_0 Now we approximate all bar-to-P_0 paths for an arbitrary shortest path P_0. In particular, P_0 may or may not be the bank of a channel.

PROPOSITION 7.7. *There is a polynomial time algorithm* $\mathsf{Bars2Path}(P_0, \mathcal{B}, \mathcal{P}, \delta)$ *that, when given a path P_0, a channelized cage $\mathcal{B}$, a set of shortest paths $\mathcal{P}$ closed under taking subpaths, and an error parameter $\delta \in (0, 1)$, adds paths to* LINKS *with the following properties:*
- *(Subpaths)* Bars2Path *only adds subpaths of paths in $\mathcal{P}$ to* LINKS.
- *(Approximation) All bar-to-P_0 shortest paths $P \in \mathcal{P}$ are δ-approximated after applying* Bars2Path.
- *(Sparsity) For each bar $B \in \mathcal{B}$,* Bars2Path *adds at most $O(\delta^{-5})$ paths to* LINKS *that intersect B.*

The algorithm Bars2Path approximates bar-to-P_0 paths in two stages. In the first stage, Bars2Path approximates paths in each bucket from one arm of each bar to P_0. However, the algorithm has little control over which arm is dealt with. To handle the remaining paths, Bars2Path makes an "unapproximated prefix" for each

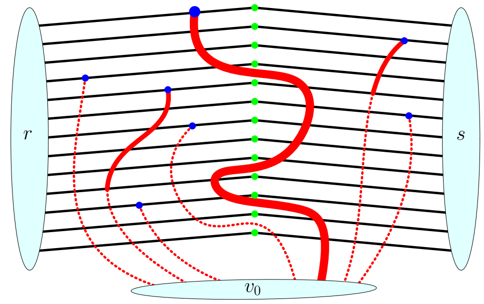

Figure 6: Processing of one bucket of paths during Stage 1 of Bars2Path. The boldest red path is the one path added from the bucket during Stage 1. The other solid red paths are the subpaths that Stage 2 processes.

vertex v. Bars2Path then adds a sparse cover for each bucket of prefixes and crossbars to get from the end of a prefix to the path approximating it.

- $\mathsf{Bars2Path}(P_0, \mathcal{B}, \mathcal{P}, \delta)$ for
  – a shortest path P_0 with endpoints s' and t'
  – an $s - t$ cage $\mathcal{B}$
  – a set of shortest paths $\mathcal{P}$ closed under taking subpaths
  – and a parameter $\delta \in (0, 1)$:
- (Stage 1:)
- $\{\mathcal{P}_i\}_i \leftarrow \mathsf{Buckets}(\mathcal{P}, s, t, s', t'\delta/100)$.
- For each i,
  – Add $\mathsf{SparseCover}(\mathcal{P}_i, \mathcal{B})$ to LINKS.
- Add $\mathsf{Crossbars}(\mathcal{P}, \mathcal{B}, \delta/100)$ to LINKS.
- (Stage 2:)
- For each path $P_{vv'} \in \mathcal{P}$ for v on a bar in $\mathcal{B}$ and $v' \in P_0$
  – Let x be first vertex on the v–v' path $P_{vv'}$ for which the $x - v'$ subpath of $P_{vv'}$ is $\delta/20$-approximated.
  – Let $Q_{vv'}$ be the $v - x$ subpath of $P_{vv'}$.
- For each i,
  – Let $\mathcal{Q}_i$ be the set of nonempty (edge-containing) paths $Q_{vv'}$ for vertices v that are starting points of paths in $\mathcal{P}_i$ and $v' \in P_0$.
  – Add $\mathsf{SparseCover}(\mathcal{Q}_i, \mathcal{B})$ to LINKS.

Now we analyze this algorithm. The sparsity of LINKS follows from the number of buckets and from the sparsity guarantee of SparseCover. For the approximation guarantee, consider vertices v on a bar B and $v' \in P_0$ for which $P_{vv'} \in \mathcal{P}_i$. $P_{vv'}$ is $\delta/100$-approximated if v happens to be on the arm of the bar that $\mathsf{SparseCover}(\mathcal{P}_i, \mathcal{B})$ intersects. Otherwise, by the coverage guarantee of $\mathsf{SparseCover}(\mathcal{Q}_i, \mathcal{B})$, there is some $Q_{uu'} \in \mathsf{SparseCover}(\mathcal{Q}_i, \mathcal{B})$ that intersects a bar containing v. The core of the proof is in showing that $Q_{uu'}$ intersects the arm containing v. If $Q_{uu'}$ does not intersect the arm containing v, we show that $Q_{uu'}$ contains a non-endpoint vertex y whose $y - u'$ path is $\delta/20$-approximated by Stage 1 paths, a contradiction.

7.3 Bars To Bars Now we prove Lemma 6.1. The algorithm $\mathsf{Bars2Bars}$ still uses $\mathsf{SparseCover}$ and $\mathsf{Buckets}$, but also uses $\mathsf{Bars2Path}$ in order to topologically constrain the paths that Proposition 7.5 is applied to:

- $\mathsf{Bars2Bars}(\mathcal{B}, \mathcal{P}, \delta)$ for
  - an $s-t$ cage $\mathcal{B}$
  - a set of bar-to-bar shortest paths $\mathcal{P}$ closed under taking bar-to-bar subpaths
  - and a parameter $\delta \in (0, 1)$:
- Channelize each bar in $\mathcal{B}$ and delete all but $O(1/\delta)$ evenly-spaced drawbridges.
- $\mathcal{R} \leftarrow \emptyset$
- For each $P \in \mathcal{P}$,
  - Add MonotoneSplit$(P, \mathcal{B})$ to $\mathcal{R}$.
- Reset $\mathcal{P}$ to $\mathcal{R}$.
- Add $\mathsf{Crossbars}(\mathcal{P}, \mathcal{B}, \delta/100)$ to Links.
- For each $p \in \{s, t\}$,
  - Let $\mathcal{P}_p$ be the set of subpaths of paths in $\mathcal{P}$ with both endpoints on p-arms.
  - Let $\mathcal{Q}_p$ be the set of subpaths P of paths $\mathcal{P}$ of paths in P with the property that P only intersects p-arms of a bar.
  - $\{\mathcal{P}_{pi}\}_i \leftarrow \mathsf{Buckets}(\mathcal{P}_p, s, t, \delta/100)$
  - $\{\mathcal{Q}_{pi}\}_i \leftarrow \mathsf{Buckets}(\mathcal{Q}_p, s, t, \delta/100)$
  - For each bucket $\mathcal{L} \in \{\mathcal{P}_{pi}\}_i \cup \{\mathcal{Q}_{pi}\}_i$
    * $\mathcal{K} \leftarrow \mathsf{SparseCover}(\mathcal{L}, \mathcal{B})$
    * Add $\mathcal{K}$ to Links
    * For each path $P \in \mathcal{K}$
      · Let B_a and B_b be bars containing the endpoints of P.
      · For each $j \in \{a, b\}$, let A_{js} and A_{jt} be the s and t-side arms respectively of B_j.
      · Let $\mathcal{B}_0$ denote the bars that P intersects.
      · Let $\mathcal{L}_0$ be the set of paths in $\mathcal{L}$ that intersect a bar in $\mathcal{B}_0$.
      · Let $\mathcal{B}_1$ be the set of bars in $\mathcal{B}$ that intersect a path in $\mathcal{L}_0$ and the two adjacent bars.
      · Let $\mathcal{P}'$ be the set of paths in $\mathcal{P}$ that only intersect bars in $\mathcal{B}_1$.
      · For each arm $A \in \{A_{as}, A_{at}, A_{bs}, A_{bt}\}$, run $\mathsf{Bars2Path}(A, \mathcal{B}, \mathcal{P}', \delta/100)$.
      · Run $\mathsf{Bars2Path}(P, \mathcal{B}, \mathcal{P}', \delta/100)$.

Now we give an overview of the analysis. Note that the four places that add Links only add subpaths of $\mathcal{P}$ to Links by Proposition 7.6 and the definition of $\mathsf{Crossbars}$. This completes the *Subpaths* guarantee of Proposition 6.1. To show the *Sparsity* guarantee, we just need to bound the number of different $\mathcal{B}'$s that contain a bar. This can be bounded using the *Sparsity* guarantee of Proposition 7.6 applied to $\mathcal{K}$.

For the *Approximation* guarantee, we start by arguing that paths $Q \in \mathcal{Q}_p$ for $p \in \{s, t\}$ are $\delta/20$-approximated. We break this analysis up into two

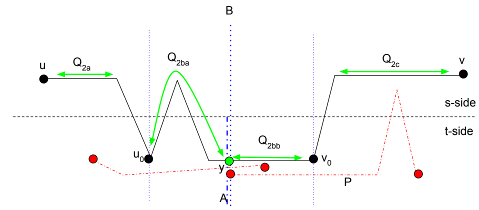

Figure 7: The paths in Case 1.

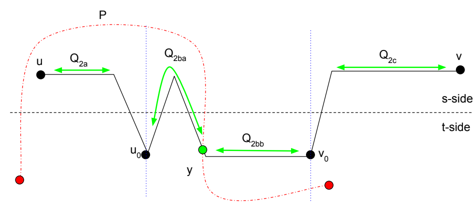

Figure 8: The paths in Case 2.

cases. Let $\mathcal{L}$ be the bucket in $\mathsf{Buckets}(\mathcal{Q}_p, s, t, \delta/100)$ containing Q. If Q intersects an arm A of a bar B containing an endpoint of a path $P \in \mathsf{SparseCover}(\mathcal{L}, \mathcal{B})$, then break Q up into two segments at some intersection point with B and notice that $\mathsf{Bars2Path}(A, \mathcal{B}, \mathcal{P}', \delta/100)$ $\delta/100$-approximates each of those segments, for a total of $\delta/50$ error. Otherwise, Q does not intersect a bar containing an endpoint of a path in $\mathsf{SparseCover}(\mathcal{L}, \mathcal{B})$. In particular, the *Coverage* property ensures that some path $P \in \mathsf{SparseCover}(\mathcal{L}, \mathcal{B})$ intersects bars containing the endpoints of Q. Since $\mathcal{L} \subseteq \mathcal{Q}_p$, it only contains paths that intersect p-arms of bars. Therefore, Proposition 7.5 implies that $P \cup \mathcal{B}$ $3\delta/100$-approximates Q.

Now we argue that any path $Q \in \mathcal{P}$ is δ-approximated after applying $\mathsf{Bars2Bars}$. Q can be broken up into three segments Q_0, Q_1, and Q_2, where Q_1 is $\delta/100$-approximated by a crossbar, $Q_0 \in \mathcal{Q}_s \cup \mathcal{Q}_t$, and $Q_2 \in \mathcal{P}_s \cup \mathcal{P}_t$. By the previous paragraph, Q_0 is $\delta/20$-approximated. Therefore, we just need to focus on Q_2. Divide Q_2 up into three segments Q_{2a}, Q_{2b}, and Q_{2c}, where Q_{2a} and Q_{2c} only consist of vertices on p-arms and the endpoints of Q_{2b} are on $q \in (\{s, t\} - p)$-arms. The following cases are illustrated in figures.

Case 1: $\mathcal{K}$ contains a path P with an endpoint bar B that contains an arm A for which (1) B intersects Q_2 and (2) P intersects a bar that Q_{2b} intersects. In this case, split Q_2 into two segments using a vertex $y \in A \cap Q_2$. By construction of $\mathcal{P}'$, subsegments Q_{2ba} and Q_{2bb} of these segments that together contain Q_{2b} are each $\delta/100$-approximated using $\mathsf{Bars2Path}(A, \mathcal{B}, \mathcal{P}', \delta/100)$, leaving two subsegments of Q_{2a} and Q_{2c} to approximate. By the previous paragraph, these subsegments are both $\delta/20$-approximated. Therefore Q_2 is approximated.

Case 2: $\mathcal{K}$ contains a path P that (1) intersects

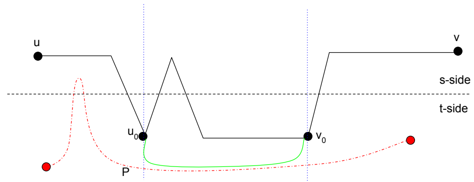

Figure 9: The paths in Case 3. The green path approximates a segment containing Q_{2b}.

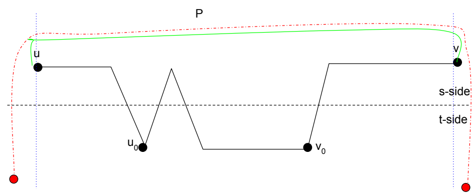

Figure 10: The paths in Case 4. The green path approximates Q_2.

Q_2 and (2) intersects a bar that Q_{2b} intersects. In this case, split Q_2 into two segments using a vertex $y \in P \cap Q_2$. Subsegments of both of these segments that together contain Q_{2b} are each $\delta/100$-approximated using $\mathsf{Bars2Path}(P, \mathcal{B}, \mathcal{P}', \delta/100)$. The remaining two segments of Q_2 are subsegments of Q_{2a} and Q_{2c} and are therefore $\delta/20$-approximated.

If neither of these cases holds, Proposition 7.2 implies that a single path $P \in \mathcal{K}$ intersects all bars that Q_2 intersects; in particular both endpoint bars of Q_{2b} and Q_2. Furthermore, Proposition 7.3 applies with the parameter settings $P \leftarrow Q_2$ and $Q \leftarrow P$ and splits the remaining proof into two more cases:

Case 3: P intersects both endpoint arms for Q_{2b}. In this case, P and bars $3\delta/100$-approximate Q_{2b} because P and Q_{2b} are in the same bucket (Proposition 7.5). Therefore, Q_2 is also approximated.

Case 4: P intersects both endpoint arms for Q_2. In this case, P and bars $3\delta/100$-approximate Q_2 because P is in a bucket for a subpath of Q_2 (Proposition 7.5).

Thus, Q_2 and in turn Q are δ-approximated.

References

[1] B. S. Baker. Approximation algorithms for NP-complete problems on planar graphs. *Journal of the ACM*, 41(1):153–180, 1994.

[2] G. Borradaile and H. Le. Optimal dynamic program for r-domination problems over tree decompositions. In *11th International Symposium on Parameterized and Exact Computation*, pages 8:1–8:23, 2016.

[3] D. E. Carroll and A. Goel. Lower bounds for embedding into distributions over excluded minor graph families. In *12th Annual European Symposium on Algorithms*, pages 146–156. 2004.

[4] A. Chakrabarti, A. Jaffe, J. R. Lee, and J. Vincent. Embeddings of topological graphs: Lossy invariants, linearization, and 2-sums. In *49th Annual IEEE Symposium on Foundations of Computer Science*.

[5] T.-H. H. Chan and A. Gupta. Approximating tsp on metrics with bounded global growth. In *Nineteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 690–699, 2008.

[6] E. D. Demaine, F. V. Fomin, M. T. Hajiaghayi, and D. M. Thilikos. Fixed-parameter algorithms for (k, r)-center in planar graphs and map graphs. *ACM Transactions on Algorithms*, 1(1):33–47, 2005.

[7] D. Eisenstat, P. N. Klein, and C. Mathieu. Approximating k-center in planar graphs. In *25th Annual ACM-SIAM Symposium On Discrete Algorithms*, pages 617–627, 2014.

[8] D. Eppstein. Diameter and treewidth in minor-closed graph families. *Algorithmica*, 27(3-4):275–291, 2000.

[9] H. Eto, T. Ito, Z. Liu, and E. Miyano. Approximability of the distance independent set problem on regular graphs and planar graphs. In *10th International Conference on Combinatorial Optimization and Applications*, pages 270–284, 2016.

[10] A. E. Feldmann, Wai S. F., J.. Könemann, and I. Post. A $(1+\varepsilon)$-embedding of low highway dimension graphs into bounded treewidth graphs. In *42nd International Colloquium on Automata, Languages, and Programming*, pages 469–480. 2015.

[11] T. F Gonzalez. Clustering to minimize the maximum intercluster distance. *Theoretical Computer Science*, 38:293–306, 1985.

[12] D. S. Hochbaum. When are NP-hard location problems easy? *Annals of Operations Research*, 1(3):201–214, 1984.

[13] D. S. Hochbaum and D. B. Shmoys. A best possible heuristic for the k-center problem. *Mathematics of Operations Research*, 10(2):180–184, 1985.

[14] I. Katsikarelis, M. Lampis, and V. Th. Paschos. Structural parameters, tight bounds, and approximation for (k, r)-center. In *28th International Symposium on Algorithms and Computation)*, pages 50:1–50:13, 2017.

[15] D. Marx and M. Pilipczuk. Optimal parameterized algorithms for planar facility location problems using Voronoi diagrams. In *23rd Annual European Symposium on Algorithms*, pages 865–877, 2015.

[16] J. Plesník. On the computational complexity of centers located in a graph. *Aplikace matematiky*, 25(6):445–452, 1980.

[17] K. Talwar. Bypassing the embedding: algorithms for low dimensional metrics. In *36th Annual ACM Symposium on Theory of Computing*, pages 281–290, 2004.

[18] H. Tamaki. A linear time heuristic for the branch-decomposition of planar graphs. In *11th Annual European Symposium on Algorithms*, pages 765–775, 2003.