Approximating k-center in planar graphs

David Eisenstat* Philip N. Klein* Claire Mathieu†

Abstract

We consider variants of the metric k-center problem. Imagine that you must choose locations for k firehouses in a city so as to minimize the maximum distance of a house from the nearest firehouse. An instance is specified by a graph with arbitrary nonnegative edge lengths, a set of vertices that can serve as firehouses (i.e., centers) and a set of vertices that represent houses. For general graphs, this problem is exactly equivalent to the metric k-center problem, which is APX-hard. We give a polynomial-time bicriteria approximation scheme when the input graph is a planar graph.

We also give polynomial-time bicriteria approximation schemes for several generalizations: if, instead of all houses, we wish to cover a specified proportion of the houses; if the candidate locations for firehouses have rental costs and we wish to minimize not the number of firehouses but the sum of their rental costs; and if the input graph is not planar but is of bounded genus.

1 Introduction

We consider the *k-center* problem and its variants. Imagine you must choose locations for firehouses in a city so that all (or a specified proportion of) houses are within a five-minute drive of some firehouse; the goal is to minimize the number of firehouses or the sum of their acquisition costs. Alternatively, you have a budget for acquiring firehouse locations and want to minimize the travel time to the farthest house served.

The *metric k-center* problem is as follows: given a metric space (V, d), find a k-element subset M of points that minimizes $\max_{v \in V} d(M, v)$. Even for the Euclidean plane and L_2 distances, it is NP-hard to approximate this objective to within a factor better than 1.822 [10] (within a factor better than 2 for L_1 and L_∞). Hochbaum and Shmoys [12] and Gonzalez [11] gave a polynomial-time 2-approximation algorithm that works for an arbitrary metric space and showed that no polynomial-time $(2 - \epsilon)$-approximation algorithm exists unless P=NP (even for the metric space arising from distances in a planar graph with unit-length edges). There is thus not much room for improvement here.[1]

The *r-dominating-set* problem is as follows: given a metric space (V, d) and a distance threshold r, find a minimum-size set M of points such that every point in V is within distance r of some point in M. This is a special case of *set cover* and can be approximated to within a factor of $1 + \ln|V|$. In the Euclidean plane, however, a better approximation can be achieved. Hochbaum and Maass [13] gave a polynomial-time approximation scheme for a more general problem, a special case of which is *covering by disks* (where the disk radius is specified in the input).

Metric spaces arising from distances in planar graphs Since road maps are nearly planar, it is natural to consider the problems of k-center and r-dominating set when the metric space arises from distances in a planar graph with edge lengths (and the planar graph is included as part of the input). Unfortunately, as mentioned above, this restriction does not help in obtaining a more accurate polynomial-time approximation algorithm for k-center, and so far it remains open whether there is such an algorithm for r-dominating set.

Constant r and constant k However, adding an additional restriction helps a great deal. A planar graph of diameter c has treewidth $O(c)$ [4, 5]. This suggests restricting attention to metric spaces arising from planar graphs with unit-length edges and restricting r to be a constant. Under this restriction, Demaine, Fomin, Hajiaghayi, and Thilikos [8] give a linear-time approximation scheme for r-dominating set.

The k-center problem involves minimizing the radius, and the r-dominating set involves minimizing the number of centers. The *(k, r)-center problem* is to find a set of at most k centers such that every other vertex is within distance r of at least one of these centers. Demaine, Fomin, Hajiaghayi, and Thilikos [8] gave an algorithm for the (k, r)-center problem that, for constant k and constant r, runs in polynomial time.

Unbounded k and r What can be achieved for k-center, r-dominating set, and (k, r)-center if k and r are unbounded and the metric space is that of a given planar graph with arbitrary edge lengths? We give a *bicriteria* polynomial-time approximation scheme. That is, the algorithm approximates both the number

*Brown University. Research supported in part by National Science Foundation Grant CCF-0964037.

†École Normale Supérieure and Brown University. Research supported in part by ANR RDAM and National Science Foundation Grant CCF-0964037.

[1] Feder and Greene [10] improve the running time for the Euclidean case.

of centers and the radius. In addition, the algorithm handles graphs of any bounded genus. More precisely,

THEOREM 1.1. *For every fixed $\epsilon > 0$, there is a polynomial-time algorithm such that, for every bounded-genus input graph with nonnegative edge lengths, for every input numbers k and r,*

- *if there is a k-element set M^* of centers for which every vertex is within distance r of M^**
- *then the algorithm outputs a set M of centers of size at most $(1+\epsilon)k$ for which every vertex is within distance $(1+\epsilon)r$ of M.*

Not all vertices are equally good locations for firehouses. Some might not be available to acquire; some might be disallowed due to neighborhood objections; some might be more costly than others. We therefore consider an assignment of *costs* to vertices, and we seek a low-cost set M.

Also, not all locations in a map need to be near firehouses. As pointed out by Charikar, Khuller, Mount, and Narasimhan [6] it is unrealistic to require that every vertex in a graph be served by a facility. For commercial applications especially it might be economically sensible to disregard some potential customers. Charikar et al. cite a Washington Post article from 2000, in which an executive states that "there's a Kmart within six miles of 88 percent of the U. S. population."[2] We therefore consider an assignment of *weights* to vertices and seek to maximize the weight of vertices within a given distance from the set M.

Our bicriteria approximation scheme can handle costs and weights without approximating in weight.

THEOREM 1.2. *For every fixed $\epsilon > 0$, there is a polynomial-time algorithm such that, for every bounded-genus input graph with nonnegative edge lengths, nonnegative vertex costs, and arbitrary vertex weights, for every input numbers C, r,*

- *if there is a set M^* of vertices of cost at most C such that the vertices within distance r of M^* have total weight w*
- *then the algorithm returns a set M of cost at most $(1+\epsilon)C$ such that the vertices within distance $(1+\epsilon)r$ have total weight at least w.*

As a corollary, we get an algorithm that can handle *penalties* on vertices, to be added to the costs if these vertices are not within the required distance of the centers.

[2]If the article were more recent, it would no doubt mention instead Starbucks or Walmart.

COROLLARY 1.1. *For every $\epsilon > 0$, there is a polynomial-time algorithm such that, for every bounded-genus input graph with nonnegative edge lengths, nonnegative vertex costs, and vertex penalties, for every input number r,*

- *if there are sets M^* and U^* of vertices such that every vertex not in U^* is within distance r of M^**
- *then the algorithm returns two sets M and U such that every vertex not in U is within distance $(1+\epsilon)r$ of M and*

$$\begin{aligned}&\text{cost}(M) + \text{penalty}(U)\\ &\quad\le (1+\epsilon)\text{cost}(M^*) + \text{penalty}(U^*).\end{aligned}$$

1.1 Techniques

Metric shifting Baker [4] gave approximation schemes for several optimization problems in planar graphs using *shifting*, a method in which a problem on the whole graph is reduced to solving the problem on several graphs with small breadth-first-search depth and returning the best solution thereby obtained. (A similar technique was the basis of the scheme of Hochbaum and Maass [13].) This technique was directly applied by Demaine et al. [7] to graphs with unit-length edges and constant radius r. Here we extend the shifting method to handle arbitrary lengths and arbitrary radius r. The extension is obvious; instead of yielding subgraphs with small breadth-first search depth, it yields subgraphs with small metric radius.

Clusters bounded by short paths For the subsequent dynamic program, we break the graph into clusters such that each cluster's boundary vertices lie on a small number of shortest paths, based on applying a lemma of Lipton and Tarjan [15] to a shortest-path tree. This idea was used by Arora, Grigni, Karger, Klein, and Woloszyn [2] in an approximation scheme for TSP in planar graphs with edge lengths, and by Thorup [17] in an approximate distance oracle for planar graphs and in follow-up work, e.g., [14], which included the extension to bounded-genus graphs.

Lipschitz configurations In the dynamic program, a configuration assigns numbers to the boundary vertices of a cluster. In order to keep the number of configurations small, we use only a coarse selection of integers and need to impose a continuity constraint on the numbers. This idea was introduced by Arora, Rao, and Raghavan [3]. It leads to approximation error in the distances.

1.2 Notation and terminology

Although for historical reasons we have discussed k-center, our algorithms do not take the desired number of centers

as a parameter. Our algorithms clearly can be used to address the k-center problem.

The reader should be warned that we use k in this paper as a variable in the dynamic program and that occurrences of k in the paper do not refer to the parameter in the k-center problem.

Let G be a graph. We denote the edge set by $E(G)$ and the vertex set by $V(G)$. If G is an embedded graph, we denote the face set by $F(G)$. If G has nonnegative edge lengths and there is a shortest-path tree with maximum distance L, we say G has *radius* L.

For a set S of vertices of a graph G, $\delta_G(S)$ is the set of *boundary edges* of S, i.e., the set of edges e such that e has one endpoint in S and one endpoint not in S. Such a set is called a *cut*. For a set S of edges of G, ∂S is the set of *boundary vertices* of S, i.e., the set of vertices v such that v belongs to at least one edge in S and at least one edge not in S. For a set S of faces of an embedded graph G, $\delta_G(S)$ is the set of edges e such that e bounds a face in S and a face not in S, and $\partial_G S$ is the set of vertices of G that lie on edges of $\delta_G(S)$.

We define the *ball-cover problem in graphs* as follows: given a pair (G, W) where G is a graph with nonnegative edge lengths and W is a subset of vertices of G, find a minimum-size set M of vertices such that every vertex of W is within distance 1 of some vertex in M. We refer to the vertices in M as *centers*, and we refer to the vertices in W as *clients*.

In this definition, we specify a distance threshold of 1, but this choice can be made without loss of generality since the edge lengths can be scaled.

We say that a set of centers *covers within distance* d a set of clients if every client is within distance d of a center. For a graph G with edge lengths, we use $d_G(u, v)$ to denote the u-to-v distance.

2 Metric shifting

We assume for notational convenience that ϵ^{-1} is an integer. In this section, we extend Baker's shifting technique to reduce a ball-cover instance (G, W) to a collection of ball-cover instances (G_j^σ, W_j^σ) such that each graph G_j^σ has radius at most $2\epsilon^{-1} + 3$. These instances are solved by the subroutine of Step *, described in greater detail later.

For each value of σ in $\{0, 1, 2, \ldots, \epsilon^{-1} - 1\}$, the graphs $G_0^\sigma, G_1^\sigma, G_2^\sigma, \ldots$ overlap slightly. The client sets $W_0^\sigma, W_1^\sigma, W_2^\sigma, \ldots$ partition W and thus do not overlap. The solutions to the instances $(G_0^\sigma, W_0^\sigma), (G_1^\sigma, W_1^\sigma), (G_2^\sigma, W_2^\sigma), \ldots$ are combined to form a solution M^σ. The best of the solutions $\{M^\sigma : \sigma \in \{0, 1, 2, \ldots, \epsilon^{-1} - 1\}\}$ is returned.

The graph G_j^σ is obtained from G by deleting vertices whose distance from s is at least some threshold, and contracting to a single vertex (called s) all vertices whose distance is less than some lesser threshold, so it is planar. The edges incident to s in G_j^σ all are assigned length 1, so every path in G_j^σ that passes through s has length at least 2.

LEMMA 2.1. *If there exists a set M^* of at most k centers covering all clients within distance 1, then the output M of* **Algorithm Main** *is a set of at most $(1+\epsilon)k$ centers covering all clients within distance $1+\epsilon$.*

Proof. Fix a value of σ in $\{0, 1, 2, \ldots, \epsilon^{-1} - 1\}$. Let $S_j^\sigma = V_j^\sigma \cap V_{j+1}^\sigma$ and let $S^\sigma = \bigcup_j S_j^\sigma$. For every σ, the only vertices common to some pair of $G_0^\sigma, G_1^\sigma, G_2^\sigma, \ldots$ belong to S^σ, and every such vertex is common to exactly two of these graphs. This implies that

$$\sum_j |M^* \cap V_j^\sigma| \leq |M^*| + |M^* \cap S^\sigma|.$$

The sets $\{S^\sigma : \sigma \in \{0, 1, 2, \ldots, \epsilon^{-1} - 1\}\}$ are disjoint, so

$$\sum_\sigma |M^* \cap S^\sigma| \leq |M^*|.$$

Therefore, there is a value of σ for which $|M^* \cap S^\sigma| \leq (1+\epsilon)|M^*|$. Let $\bar{\sigma}$ be such a value. Then

$$\sum_j |M^* \cap V_j^{\bar{\sigma}}| \leq (1+\epsilon)|M^*|. \tag{2.1}$$

By the correctness of Step *, the solution $M_j^{\bar{\sigma}}$ obtained for input $(G_j^{\bar{\sigma}}, W_j^{\bar{\sigma}})$ has cardinality at most $|M^* \cap V_j^{\bar{\sigma}}|$. Combining with 2.1, we obtain $|M^{\bar{\sigma}}| \leq (1+\epsilon)|M^*|$. □

We have now reduced the problem to instances (G', W') where G' has radius at most $2\epsilon^{-1} + 3$. To implement Step *, we first use the embedding to find a recursive partition of the graph, then apply a dynamic-programming algorithm.

3 Recursive partition

In this section, we define a small-depth recursive partition of the faces of a planar embedded triangulated graph, such that each cluster of faces is bounded by at most 4 short simple paths (Corollary 3.1, used in the algorithm presented in section 4.1.)

We adapt the notion of *carving* [16]. A *recursive partition* of a set S is an abstract binary tree $\mathcal{T}$ in which each node x is labeled with a subset $S(x)$ of S, called the *cluster* associated with x, such that

- the root x is labeled with S, and

Algorithm Main

input: graph G with nonnegative edge lengths, subset W of vertices
output: a set of centers M that satisfies Lemma 2.1

Let s be a vertex of G.
Compute a shortest-path tree in G with root s.
For every shift $\sigma \in \{0, 1, 2, \ldots, \epsilon^{-1} - 1\}$,
  For every $j \geq 0$,
    Let intervals $\mathcal{I} = [2j\epsilon^{-1} - 2\sigma,\ 2(j+1)\epsilon^{-1} - 2\sigma)$
      and $\mathcal{I}^+ = [2j\epsilon^{-1} - 1 - 2\sigma,\ 2(j+1)\epsilon^{-1} + 1 - 2\sigma)$.
    Let $W_j^\sigma = \{v \in W : d_G(s,v) \in \mathcal{I}\}$.
    Let $V_j^\sigma = \{v \in V(G) : d_G(s,v) \in \mathcal{I}^+\}$.
    Let E_j^σ be the restriction of $E(G)$ to $V_j \times V_j$.
    Let G_j^σ denote the graph with vertices $\{s\} \cup V_j^\sigma$ and
      edges $E_j^\sigma \cup \{sv : v \in V_j,\ v$'s parent not in $V_j\}$,
        where the extra edges sv have length 1.
    For input (G_j^σ, W_j^σ), call **Algorithm DP** to find a set M_j^σ of centers
      covering all clients in W_j^σ within distance $1+\epsilon$ in G_j^σ, such that the number of centers
        is at most the minimum required to cover W_j^σ within distance 1.
  $M^\sigma \leftarrow \bigcup_j M_j^\sigma$.
$M \leftarrow$ best of M^σ over all values of σ.
Output M.

- for each node x of $\mathcal{T}$ with children x_1 and x_2, $S(x) = S(x_1) \cup S(x_2)$ and $S(x_1) \cap S(x_2) = \varnothing$, and
- each leaf x is labeled with a singleton: $|S(x)| = 1$.

The *depth* of the recursive partition is the depth of the tree.

LEMMA 3.1. *Let T be a tree of degree at most 3. There is a recursive partition of $V(T)$ with depth $O(\log|V(T)|)$ such that, for each cluster C of vertices, the subforest of T induced by C is a tree, and the set $\delta_T(C)$ of boundary edges of C in T has cardinality at most 4.*

Proof. We proceed top-down and give an algorithm to decompose each cluster C into two child clusters C_1 and C_2. The abstract tree $\mathcal{T}$ can be inferred from the recursive partition. The algorithm chooses an edge e of $T[C]$, the subtree of T induced by C. Then $T[C] - e$ consists of two trees, T_1 and T_2, and we take $C_j = V(T_j)$.

There are two cases. If $|\delta_T(C)| \leq 3$, we choose e to be a balanced separator of $T[C]$, i.e., an edge for which $|C_j|$ is between $\frac{1}{3}|C|$ and $\frac{2}{3}|C|$. If $|\delta_T(C)| = 4$, we choose e so as to separate the endpoints of two of the edges in $\delta_T(C)$ from the endpoints of the other two, so that $|\delta_T(C_j)| \leq 3$ for $j = 1$ and $j = 2$.

Since, for each leaf x of the resulting recursive partition, Case 1 applies at least to every other ancestor of x, it follows that the depth is $O(\log|V(T)|)$. □

COROLLARY 3.1. *Let G be an n-face planar embedded triangulated graph with edge lengths. Then there is a recursive partition of the faces $F(G)$ of G that has depth $O(\log n)$ and such that each cluster is bounded by vertices lying on at most 8 shortest paths.*

Proof. Let T be a shortest-path tree of G from some arbitrary starting vertex. Let G^* be the planar dual of G. Let T^* be the set of edges e of G^* such that the edge in G corresponding to e is not in T. A classical result [1] is that the edges of T^* form a spanning tree of G^*. Apply Lemma 3.1 to T^* to get a recursive partition of the set of vertices of T^*, i.e., the set of faces of G.

Let C be a cluster in this partition. Then C consists of the vertices of a subtree of T^* bounded by at most 4 edges. Let $e_1, \ldots, e_\ell$ be those edges. Each edge e_j belongs to T^* and therefore does not belong to T. It follows from well known properties of planar graph duality (See, e.g., `http://planarity.org`.) that in G the faces in C are bounded by the fun-

damental cycles[3] of $e_1, \ldots, e_\ell$. The vertices of the fundamental cycle lie on two paths from the shortest-path tree. □

4 Dynamic programming

As with algorithms based on tree-width, once we have reduced the problem to small-radius graphs with a good recursive partition, dynamic programming can be used to find a near-optimal solution. Since the dynamic program is a bit technical, we ease into it by presenting first a slow but exact version.

4.1 Exact distances: a slow dynamic program With every vertex $v \in V$, we associate an arbitrary face $\phi(v)$ incident to v. A cluster C is responsible for all clients in $W \cap \phi^{-1}(C)$.

Let M be a set of centers. For each cluster C, we define two functions $\partial C \to [0,\ 1] \cup \{\infty\}$. One, called i^M, maps each boundary vertex $v \in \partial C$ to the distance from v to the nearest center in $M \cap \phi^{-1}(C)$, replacing every distance greater than 1 by ∞. The other, e^M, maps each boundary vertex $v \in \partial C$ to the distance from v to the nearest center in $M - \phi^{-1}(C)$, replacing every distance greater than 1 by ∞. We also set $k = |M \cap \phi^{-1}(C)|$.

Formally, an *interface* for a cluster C is a function $i : \partial C \to [0,\ 1] \cup \{\infty\}$. A *configuration* $\kappa = (i, e, k)$ for a cluster C consists of two interfaces i, e and an integer k.

On the following page we give a dynamic program that populates a boolean table indexed by clusters and associated configurations. The table is filled in bottom-up order of the recursive cluster partition, in such a way that the entry associated to cluster C and configuration $\kappa = (i, e, k)$ of C is true if and only if there exists a set of centers $M_C \subseteq \phi^{-1}(C)$ that *conforms* to κ in the sense that

1. M_C has cardinality k, and
2. $i^{M_C} = i$, and
3. every vertex $v \in W \cap \phi^{-1}(C)$ is within distance 1 of M_C or within distance $1 - e(w)$ of some vertex $w \in \partial C$.

To that end, let C_0 be a parent cluster with children C_1, C_2. For every $j \in \{0, 1, 2\}$, let $\kappa_j = (i_j, e_j, k_j)$ be a configuration for C_j. We say that these configurations are *compatible* if the following conditions hold.

[3] Let T be a spanning tree of G. For every edge e of G not in T, joining e with the simple path in T between its endpoints forms a simple cycle, called the *fundamental cycle of* e with respect to T.

Budgets $k_0 = k_1 + k_2$.

Parent's Incoming Assignments For every boundary vertex $w \in \partial C_0$ with $i_0(w) \neq \infty$, we have $i_0(w) = \min\{d(w, v) + i_j(v) : j \in \{1, 2\},\ v \in \partial C_1 \cup \partial C_2\}$.

Children's Outgoing Assignments For every $j \in \{1, 2\}$ and every boundary vertex $v \in \partial C_j$ with $e_j(v) \neq \infty$, we have $e_j(v) = \min\{\min\{d(v, w) + e_0(w) : w \in \partial C_0\}, \min\{d(v, w) + i_{3-j}(w) : w \in \partial C_{3-j}\}\}$.

The correctness of **Algorithm DP** follows from Lemmas 4.1 and 4.2, whose proofs are omitted.

LEMMA 4.1. *Let M be a set of centers such that every vertex of W is within distance 1 of M. For every cluster C, let $M_C = M \cap \phi^{-1}(C)$ and let $\kappa_C = (i^{M_C}, e^{M_C}, |M_C|)$. Then M_C conforms to κ_C, and, for every parent cluster C_0 with children C_1, C_2, configurations $(\kappa_{C_0}, \kappa_{C_1}, \kappa_{C_2})$ are compatible.*

LEMMA 4.2. *Let C_0 be a parent cluster with children C_1, C_2. For $j \in \{0, 1, 2\}$, let κ_{C_j} be a configuration for C_j. Let M_{C_1} and M_{C_2} be sets of centers conforming to κ_{C_1} and κ_{C_2} respectively. If configurations $(\kappa_{C_0}, \kappa_{C_1}, \kappa_{C_2})$ are compatible, then $M_{C_0} = M_{C_1} \cup M_{C_2}$ conforms to κ_{C_0}.*

4.2 Approximate distances: a polynomial-time dynamic program We now turn our focus to modifying the dynamic program so that it is polynomial-time. We observe that, for the dynamic program to be correct, it suffices by Lemma 4.1 to consider what we call Lipschitz interfaces. To limit further the number of interfaces in need of consideration, we modify our definitions and algorithms to tolerate small one-sided errors in the distances specified by an interface.

The following elementary lemma enables us to bound the amount of storage necessary to approximate a 1-Lipschitz function defined on an interval $[a, b]$.

LEMMA 4.3. *Let $\delta > 0$ and let $f : [a, b] \to [0, 1+\epsilon]$ be a 1-Lipschitz function: $\forall x, y,\ |f(x) - f(y)| \leq |x - y|$. Then there exists a function g approximating f well in the sense of* (4.2) *such that g is constant on each interval $(a + (j-1)\delta, a + j\delta]$ and, for every x, $g(x)$ is an integer multiple of δ and, at each point of discontinuity, g changes by $\pm\delta$.*

$$\forall x \qquad f(x) < g(x) < f(x) + 3\delta \tag{4.2}$$

Proof. Define $g(a + j\delta)$ to be the smallest integer multiple of δ that is greater than or equal to $f(a +$

Algorithm DP

input: triangulation G with nonnegative edge lengths and radius $O(\epsilon^{-1})$, set of clients $W \subseteq V$,
recursive cluster partition, map ϕ from vertex to incident face
output (exact): the minimum cardinality of a set of centers M
such that every vertex in W is within distance 1 of M
output (approximate): the cardinality of a set of centers M that satisfies Theorem 4.1

For each cluster C in bottom-up order,
For each configuration $\kappa = (i, e, k)$ of C,
Case C leaf (single face): set $T[C, \kappa]$ to true if there exists a set of centers $M_C \subseteq \phi^{-1}(C)$ such that M_C conforms to κ.
Case C parent with children C_1, C_2: set $T[C, \kappa]$ to true if there exist configurations κ_1, κ_2 such that $T[C_1, \kappa_1]$ and $T[C_2, \kappa_2]$ both are true, and $(\kappa, \kappa_1, \kappa_2)$ are compatible.
Output the minimum k such that for the root cluster C_r,
there is a configuration $\kappa = (i, e, k)$ where $T[C_r, k]$ is true.

$j\delta) + \delta$. For every point x in the interval $I = (a + (j-1)\delta, a + j\delta]$, define $g(x) = g(a + j\delta)$. By construction, g satisfies the first two conditions.

Since f has Lipschitz constant 1,

$$\big|f(a + j\delta) - f\big(a + (j-1)\delta\big)\big| \le \delta,$$

so there is at most one integer multiple of δ between $f(a + (j-1)\delta) + \delta$ and $f(a + j\delta) + \delta$, and the third condition holds:

$$g(a + j\delta) - g\big(a + (j-1)\delta\big) \in \{0, \pm\delta\}.$$

For every x in the interval $(a + (j-1)(b-a)/\delta, a + j(b-a)/\delta]$, by the Lipschitz condition $f(x) < f(a + j\delta) + \delta$ and the definition of g, we have $g(x) \ge f(a + j\delta) + \delta$, so the first part of (4.2) holds.

Similarly, by the Lipschitz condition $f(x) > f(a + j\delta) - \delta$ and the definition of g, we have $g(x) < f(a + j\delta) + 2\delta$, so the second part of (4.2) holds. □

Definition 4.1. *A nonnegative real-valued function whose domain is a subset of vertices of a graph is* Lipschitz *if, for every v, w in the domain such that $i(v) + d(v, w) \le 1$, we have $i(w) \le i(v) + d(v, w)$.*

Lemma 4.4. *Let $\delta > 0$, and let P be a shortest path in a graph such that P has length at most L. Then there exists a collection $\mathcal{I}$ of Lipschitz functions on the vertices of P of cardinality $2^{O(L\delta^{-1})}$ such that, for every Lipschitz function $\imath$ on $V(P)$ that is bounded by L, there exists a function $\tilde{\imath} \in \mathcal{I}$ such that*

$$\forall v \in V(P) \quad i(v) \le \tilde{\imath}(v) < i(v) + 3\delta.$$

Proof. Fix a function i from vertices of P to $[0, L]$. We proceed as follows to define $\tilde{\imath}$. Let s be the first vertex of P. Define the function f such that, for every v in $V(P)$,

$$f\big(d(s, v)\big) = i(v).$$

Since i is Lipschitz, we can extend f to a continuous and piecewise affine function $f : [0, d(s_j, t_j)] \to [0, L]$ that has Lipschitz constant 1.

Let $a = 0$ and $b = d(s, t)$ where t is the last vertex of P, and let g approximate f in the sense of Lemma 4.3. Note that $b \le L$. Every function g defined by Lemma 4.3 can be encoded by its initial value $g(a)$, which is an integer multiple of δ belonging to the interval $[0, L]$, and by its successive increments $g(a+j\delta) - g(a+(j-1)\delta) \in \{0, \pm\delta\}$. There are $O(L/\delta)$ possibilities for the initial value, three possibilities for each increment, and $O(d(s, t)/\delta) \le L\delta^{-1}$ increments, for a total of $O(L/\delta)\, 3^{L\delta^{-1}} = 2^{O(L\delta^{-1})}$ possibilities for g.

Now define $\tilde{\imath}$ as follows: for every vertex v of P, let $\tilde{\imath}(v) = g(d(s, v))$. By Lemma 4.3, we have $0 < g\big(d(s, v)\big) - f(d(s, v)) < 3\delta$, from which the requisite property of $\tilde{\imath}$ readily follows. □

The following lemma establishes the existence of small collections $\mathcal{I}_C$ that suffice to prove a weakened version of the guarantee from Lemma 4.1.

Lemma 4.5. *Let $\delta > 0$ and let C be a cluster belonging to a graph of radius $O(\epsilon^{-1})$. Then there exists a collection $\mathcal{I}_C$ of interfaces of cardinality $2^{O(\epsilon^{-1}\delta^{-1})}$ such that, for every Lipschitz interface i, there exists an interface $\tilde{\imath} \in \mathcal{I}_C$ satisfying*

$$\forall v \in \partial C \quad i(v) \le \tilde{\imath}(v) < \max\{i(v) + 3\delta, 1 + 2\epsilon\}$$

Proof. By Corollary 3.1, the boundary ∂C can be partitioned into a constant number of shortest paths

$P_1, P_2, \ldots, P_{O(1)}$ such that, for every j, P_j is a path of length $O(\epsilon^{-1})$ between some pair of boundary vertices $s_j, t_j \in \partial C$. Given an interface i, for every j, let i_j be the function defined on the vertices of P_j that agrees with i except that if $i(v) = \infty$ then $i_j(v) = 1 + 2\epsilon$. We apply Lemma 4.4 to all such maps i_j to construct a collection of approximate maps $\tilde{i}_j$. We then define $\mathcal{I}_C$ to be the Cartesian product of these collections. The bound on the size of $\mathcal{I}_C$ follows from Lemma 4.4.

□

Our modified definitions of interface, compatibility, and conformance are as follows. An interface no longer gives an exact distance but an upper bound on that distance, so its codomain is extended slightly to $[0,\ 1+\epsilon] \cup \{\infty\}$. Moreover, the value of an interface for a parent cluster is obtained not just by summing a distance with the value inherited from a child cluster but also by rounding the result up to a multiple of δ, with the effect that every level of the tree hierarchy introduces an additive error. Formally, an *interface* for a cluster C is a function $i : \partial C \to [0,\ 1+\epsilon] \cup \{\infty\}$. Let C_0 be a parent cluster with children C_1, C_2. For every $j \in \{0, 1, 2\}$, let $\kappa_j = (i_j, e_j, k_j)$ be a configuration. These configurations are *compatible* if the following conditions hold.

Budgets $k_0 = k_1 + k_2$.

Parent's Incoming Assignments For every boundary vertex $w \in \partial C_0$ with $i_0(w) \neq \infty$, we have $i_0(w) \geq \min\{d(w, v) + i_j(v) : j \in \{1, 2\},\ v \in \partial C_1 \cup \partial C_2\}$.

Children's Outgoing Assignments For every $j \in \{1, 2\}$ and every boundary vertex $v \in \partial C_j$ with $e_j(v) \neq \infty$, we have $e_j(v) \geq \min\{\min\{d(v, w) + e_0(w) : w \in \partial C_0\}, \min\{d(v, w) + i_{3-j}(w) : w \in \partial C_{3-j}\}\}$.

For a cluster C, a configuration $\kappa = (i, e, k)$ of C, and a set of centers $M_C \subseteq \phi^{-1}(C)$, we say that M_C *conforms* to κ if

1. M_C has cardinality k, and
2. $i(M_C) \leq i$, and
3. every vertex $v \in W \cap \phi^{-1}(C)$ is within distance $1 + \epsilon$ of M_C or within distance $1 + \epsilon - e(w)$ of some vertex $w \in \partial C$.

The following lemma is the analog of Lemma 4.1. Let round(x) denote the least integer multiple of δ greater than or equal to x.

LEMMA 4.6. *Let $h = \max_{\text{cluster } C} \text{depth}(C)$, where $\text{depth}(C)$ is the number of clusters $C' \supsetneq C$. Let $\delta = \epsilon/(6h + 4)$. Let M be a set of centers such that every vertex of W is within distance 1 of M. For every cluster C, let $M_C = M \cap \phi^{-1}(C)$ and let $k_C = |M_C \cap \phi^{-1}(C)|$. For every $w \in \partial C$, let $\hat{i}_C(w) = \text{round}(i_C^{M_C}(w)) + \big(h - \text{depth}(C)\big) \cdot 3\delta$ and $\hat{e}_C(w) = \text{round}(e_C^{M_C}(w)) + \big(h + 1 + \text{depth}(C)\big) \cdot 3\delta$. This defines a configuration $\kappa_C = (\hat{i}_C, \hat{e}_C, k_C)$. Then M_C conforms to κ_C, and, for every parent cluster C_0 with children C_1, C_2, configurations $(\kappa_{C_0}, \kappa_{C_1}, \kappa_{C_2})$ are compatible.*

Proof. To prove that M_C conforms to κ_C, we observe first that (1) and (2) hold by the definition of our configurations. As for (3), v is within distance $1 < 1 + \epsilon$ of M_C or within distance 1 of some vertex $x \in M - M_C$. In the latter case, the shortest path from v to x intersects ∂C at some vertex w, which implies that $d(v, w) + d(w, x) = d(v, x)$. We bound

$$\begin{aligned}\hat{e}_C(w) &\leq d(w, x) + (h + 1 + \text{depth}(C))3\delta + \delta \\ &\leq d(w, x) + (3(2h + 1) + 1)\delta \\ &= d(w, x) + \epsilon,\end{aligned}$$

and so (3) holds.

Now we prove compatibility between configurations for a parent-child triplet C_0, C_1, C_2. The Budgets condition is trivial. For the Parent's Incoming Assignments condition, recall that

$$\begin{aligned}i_0^M(w) = \min\{&d(w, v) + i_j^M(v) \\ &: j \in \{1, 2\},\ v \in \partial C_1 \cup \partial C_2\} \\ \hat{i}_0(w) \geq\ &i_0^M(w) + \big(h - \text{depth}(C_0)\big) \cdot 3\delta.\end{aligned}$$

On the other hand,

$$\hat{i}_j^M(v) \leq \delta + i_j^M(v) + \big(h - \text{depth}(C_1)\big) \cdot 3\delta.$$

For $j \in \{1, 2\}$, we have $\text{depth}(C_0) = \text{depth}(C_j) - 1$, so

$$\begin{aligned}&d(w, v) + \hat{i}_j^M(v) \\ &\quad \leq d(w, v) + i_j^M(v) + \big(h - \text{depth}(C_0)\big) \cdot 3\delta - 2\delta.\end{aligned}$$

For the Children's Outgoing Assignments condition, observe that

$$\begin{aligned}e_j^M(v) = \min\{&\min\{d(v, w) + e_0^M(w) : w \in \partial C_0\}, \\ &\min\{d(v, w) + i_{3-j}^M(w) : w \in \partial C_{3-j}\}\}.\end{aligned}$$

If the minimum is reached by the first expression, then the argument is similar (thanks to the $+\text{depth}(C)$ instead of $-\text{depth}(C)$ in our definition of

$\hat{e}$). If the minimum is reached by the second expression, then the additive term is at least 3δ more on the left than on the right side (thanks to the $+1$ in our definition of $\hat{e}$), and the rounding cannot make up the difference, so together we obtain

$$\hat{e}_j(v) \geq \min\{\min\{d(v,w) + \hat{e}_0(w) : w \in \partial C_0\}, \min\{d(v,w) + \hat{\imath}_{3-j}(w) : w \in \partial C_{3-j}\}\}.$$

This proves compatibility of configurations. □

The following lemma is the analog of Lemma 4.2 but in the approximate setting.

LEMMA 4.7. *Let C_0 be a parent cluster with children C_1, C_2. For $j \in \{0,1,2\}$, let κ_j be a configuration for C_j. Let M_{C_1} and M_{C_2} be sets of centers conforming to κ_{C_1} and to κ_{C_2} respectively. If configurations $(\kappa_{C_0}, \kappa_{C_1}, \kappa_{C_2})$ are compatible, then $M_{C_0} = M_{C_1} \cup M_{C_2}$ conforms to κ_{C_0}.*

Proof. We need to verify that M_0 has the correct cardinality, that $i(M_0) \leq i_{C_0}$, and that every vertex $v \in W \cap \phi^{-1}(C_0)$ is within distance $1+\epsilon$ of M_0 or within distance $1 + \epsilon - e_{C_0}(w)$ of some vertex $w \in \partial C_0$. The first property follows as before from the definition of compatibility of $(\kappa_{C_0}, \kappa_{C_1}, \kappa_{C_2})$ and the conformance of M_{C_1}, M_{C_2} to $\kappa_{C_1}, \kappa_{C_2}$ respectively. Verifying the second and third properties is tedious but straightforward. □

The dynamic program is then exactly analogous to the one given in the exact case, with 1 replaced by $1+\epsilon$ and all definitions of interface, configuration, conforming, and compatibility replaced by their approximate versions. We refer to this algorithm as **Algorithm DP Approx**.

THEOREM 4.1. **Algorithm DP Approx** *takes as input a triangulation G of radius $O(1/\epsilon)$, a recursive partition of G, and a set of clients $W \subseteq V(G)$. In time $n^{O(\epsilon^{-2})}$, it gives as output a set M of centers that covers W within distance $1+\epsilon$, and such that any set of centers that covers W within distance 1 must have cardinality at least $|M|$.*

5 Extensions

We show how the algorithm of the previous section can be extended to handle center costs, vertex weights or penalties, and bounded-genus graphs.

5.1 Center costs The algorithm can be modified to handle arbitrary center costs. The budgets appearing in configurations now are expressed in terms of total cost, not cardinality. Standard rounding techniques ensure that costs are scaled to integers of polynomial magnitude.

In more detail: the algorithm takes the center costs as an additional input. These costs may be of wildly different orders of magnitude, so we introduce an outer loop to try all possibilities for $c_{\max}$, the maximum cost of a center included in the solution, against which we calibrate the rounding. We modify the meaning of conformance of a set of centers M_C to a configuration $\kappa = (i, e, k)$ by deleting the requirement that $|M_C| = k$ and adding the requirements that for every center $v \in M_C$, we have $\text{cost}(v) \leq c_{\max}$ and that the following equality holds:

$$\sum_{v \in M_C} \left\lceil \frac{\text{cost}(v)}{\epsilon c_{\max}/n} \right\rceil = k.$$

Observe that, given (1), the left-hand side is at most n^2/ϵ. For every solution M, regardless of $c_{\max}$, we can derive the error bounds

$$0 \leq (\epsilon c_{\max}/n) \sum_{v \in M} \left\lceil \frac{\text{cost}(v)}{\epsilon c_{\max}/n} \right\rceil - \sum_{v \in M} \text{cost}(v) \leq \epsilon c_{\max},$$

so when $c_{\max}$ is equal to the true maximum, we achieve the requisite approximation guarantees.

5.2 Client weights Clients can be assigned weights, and the algorithm can be required to choose centers so that a given amount of client weight is (approximately) within a given distance of a chosen center. To handle this, we modify the dynamic program and the algorithm that calls it. In the original algorithm, each dynamic-programming table entry $T[C, \kappa]$ was true or false, indicating whether there was a solution for the given cluster and configuration in which all clients are covered. Under the modification, each entry is a number, the maximum weight of clients that can be covered. We show the new pseudocode, with Δ marking lines that have been changed.

Since the dynamic program now produces a budget/weight trade-off table, the algorithm that calls it must optimize the allocation of centers. Again, lines that have changed are preceded by Δ.

5.3 Client penalties Clients can be assigned penalties, and the algorithm can be allowed to pay the penalty for a client vertex in lieu of covering it. Since the algorithm for weights does not approximate in the weights, we set the weight of each vertex to be minus its penalty and interpret the final budget/weight trade-off table in a different but obvious way.

5.4 Bounded-genus graphs For every integer constant $g \geq 0$, the algorithm can be extended to handle graphs embedded on a surface of genus g. Using metric shifting (Section 2) and scaling, we obtain

Algorithm DP-Weights

input: triangulation G with nonnegative edge lengths and radius $O(\epsilon^{-1})$, set of clients $W \subseteq V$,
recursive cluster partition, map ϕ from vertex to incident face,
Δ vertex weights
Δ *output:* a budget/weight trade-off table: for every budget k, a quantity that is
Δ at most the maximum weight of vertices in W within distance $1+\epsilon$ of centers of cost k and
Δ at least the maximum weight of vertices in W within distance 1 of centers of cost k

For each cluster C in bottom-up order,
For each configuration $\kappa = (i, e, k)$ of C,
Δ Case C leaf (single face): set $T[C, \kappa]$ to the maximum of
Δ $\sum_{v \in W \cap \phi^{-1}(C),\, d(v,M_C) \le 1+\epsilon \text{ or } \exists w \in \partial C \text{ s.t. } d(v,w)+e(w) \le 1+\epsilon} \text{weight}(v)$
Δ over all sets of centers $M_C \subseteq \phi^{-1}(C)$ such that M_C conforms to κ.
Δ Case C parent with children C_1, C_2: set $T[C, \kappa]$ to the minimum of $T[C_1, \kappa_1] + T[C_2, \kappa_2]$
Δ over all configurations κ_1, κ_2 such that $(\kappa, \kappa_1, \kappa_2)$ are compatible.
Δ Output, for every budget k, the maximum of $T[C_r, \kappa]$ over all configurations $\kappa = (i, e, k)$,
Δ where C_r is the root cluster.

a collection of instances (G, W) where G is bounded-genus with radius at most $3 + 2/\epsilon$, and finding an approximate solution of the original instance is reduced to finding an approximate solution for each of the new instances. To each of these new instances, we apply the following algorithm, whose running time is bounded by a function of the form $f_1(g, \epsilon)\, n^{f_2(\epsilon)}$.

Let T be a shortest-path tree in G. By the tree-cotree decomposition [9], there is a set of $2g$ edges not in T such that cutting the surface along the fundamental cycles of these edges with respect to T yields a genus 0 surface. The vertices of these cycles belong to $O(g)$ shortest-paths. Because of the radius bound on G, each shortest path has length $O(1/\epsilon)$. For each of these shortest paths P, the algorithm selects $O(1/\epsilon^2)$ vertices of P, called *portals*, such that each vertex on the path is within distance $\epsilon/3$ of some portal. When the surface is cut along the fundamental cycles, the vertices on these cycles, and in particular the portals, get duplicated. Thus each portal gives rise to at most $2g$ "clones" in the graph G' resulting from the cutting.

Suppose that, in some solution, P is a shortest path from a center to vertex v. If P passes through none of the fundamental cycles, then it exists as is in G'. Otherwise, the path can be perturbed by introducing a detour along the last cycle crossed, to the portal nearest to the crossing point. This increases the length of the path by at most $2\epsilon/3$.

Now we continue with the description of the algorithm. Conceptually, we introduce another internal/external interface pair, for the portals. In greater detail: for each of the $O(g)$ shortest paths P, consider a collection $\mathcal{I}_P$ of discretized Lipschitz functions (in the sense of Lemma 4.4) from the portals of P to $[0, 2\epsilon^{-1} + 3]$, with maximum upward error $\epsilon/3$. Take the Cartesian product of all these families. We add to the existing dynamic program an outer loop over this collection that guesses the approximately correct assignment of distances to the portals. The number of iterations is bounded by a fixed function of g and ϵ. For a particular set of centers $M \subseteq V$, the "correct" guess for p satisfies for every portal v

$$d_G(v, M) \le p(v) \le d_G(v, M) + \epsilon/3,$$

similar to the correct guess for e.

Given p, the algorithm attempts to cover the clients $W' = \{v : v \in W, \nexists w \text{ s.t. } p(w) + d_G(w, v) \le 1 + \epsilon\}$ in G', via a slight extension of the dynamic program for planar graphs. Specifically, we enlarge the definition of a configuration by a set of portals Q. Conformance of a set of centers M to a configuration requires additionally that, for every $v \in Q$ with $p(v) < \infty$, there exists an center $w \in M$ satisfying $d_G(w, v) \le p(v)$. Compatibility requires additionally that $Q = Q_1 \cup Q_2$. When the table for the root cluster is examined, only configurations with Q equal to all portals are examined.

Now we sketch a proof of correctness. The output M of the dynamic program is feasible because, for every client $v \in W$, either $v \in W'$ or $v \notin W'$. If $v \notin W'$ then, by the same proof of correctness as in the planar case, there exists a center $w \in M$ such that $d_G(w, v) \le d_{G'}(w, v) \le 1 + \epsilon$. Otherwise, there exists a portal w with $p(w) + d_G(w, v) \le 1 + \epsilon$. By

Algorithm Main-Weights

input: graph G with nonnegative edge lengths, subset W of vertices,
Δ minimum total weight $w_{\min}$
Δ *output:* a budget k such that, for radius 1, at least k centers are required to cover weight $w_{\min}$, and,
Δ for radius $1+\epsilon$, at most k centers are required

Let s be a vertex of G.
Compute a shortest-path tree in G with root s.
For every shift $\sigma \in \{0,\ 1,\ 2,\ \ldots,\ \epsilon^{-1}-1\}$,
Δ Initialize a budget/weight trade-off table T_0^σ with one entry $T_0^\sigma[0]=0$.
For every $j \geq 0$,
Let intervals $\mathcal{I} = [2j\epsilon^{-1}-2\sigma,\ 2(j+1)\epsilon^{-1}-2\sigma)$
and $\mathcal{I}^+ = [2j\epsilon^{-1}-1-2\sigma,\ 2(j+1)\epsilon^{-1}+1-2\sigma)$.
Let $W_j^\sigma = \{v \in W : d_G(s,v) \in \mathcal{I}\}$.
Let $V_j^\sigma = \{v \in V(G) : d_G(s,v) \in \mathcal{I}^+\}$.
Let E_j^σ be the restriction of $E(G)$ to $V_j \times V_j$.
Let G_j^σ denote the graph with vertices $\{s\} \cup V_j^\sigma$ and
edges $E_j^\sigma \cup \{sv : v \in V_j,\ v$'s parent not in $V_j\}$,
where the extra edges sv have length 1.
Δ For input (G_j^σ, W_j^σ), find the trade-off table U_j^σ using **Algorithm DP-Weights**.
Δ For every k, let $T_{j+1}^\sigma[k]$ be the maximum of $T_j^\sigma[k_1] + U_j^\sigma[k_2]$
Δ over all k_1, k_2 such that $k_1 + k_2 = k$.
Δ $T \leftarrow$ best of T_{j+1}^σ over all values of σ, where j takes on its maximum relevant value.
Δ Output the minimum budget k such that $T[k] \geq w_{\min}$.

the new definitions of conformance and compatibility, there exists a center x such that $d_G(x,w) \leq p(w)$, implying that

$$\begin{aligned} d_G(x,v) &\leq d_G(x,w) + d_G(w,v) \\ &\leq p(w) + d_G(w,v) \leq 1+\epsilon. \end{aligned}$$

Conversely, suppose that M is an optimal solution with coverage radius 1. We show that the dynamic program finds a solution at least as good. The correct guess for p is as mentioned previously. For every cluster C, we choose configurations as in the planar case and set

$$Q_C = \{v : v \text{ is a portal, } d_G(v,\ M \cap \phi^{-1}(C)) \leq p(v)\}.$$

Compatibility is clear, and Q_{C_r} is all portals, where C_r is the root cluster. Every client whose shortest path to the nearest center crosses a fundamental cycle in G is excluded from W' because, even after making a detour to a portal of round-trip distance at most $2\epsilon/3$ and incurring error of $\epsilon/3$ on the value of p at that portal, the total distance is at most $1+\epsilon$. The remaining clients are covered in G' by M.

References

[1] Martin Aigner, Günter M. Ziegler, and Karl Heinrich Hofmann. *Proofs from The Book*. Springer, Berlin, New York, 1998. Autre tirage : 1999 (corrigé).

[2] Sanjeev Arora, M. Grigni, D. R. Karger, Philip N. Klein, and A. Woloszyn. A polynomial-time approximation scheme for weighted planar graph TSP. In *Proceedings of the 9th Symposium on Discrete Algorithms*, pages 33–41, 1998.

[3] Sanjeev Arora, Prabhakar Raghavan, and Satish Rao. Approximation schemes for euclidean k-medians and related problems. In *Proceedings of the thirtieth annual ACM symposium on Theory of computing*, STOC '98, pages 106–113, New York, NY, USA, 1998. ACM.

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

[5] Hans L. Bodlaender. Some classes of graphs with bounded treewidth. *Bulletin of the EATCS*, 36:116–125, 1988.

[6] Moses Charikar, Samir Khuller, David M. Mount, and Giri Narasimhan. Algorithms for facility location problems with outliers. In *SODA*, pages 642–651, 2001.

[7] Erik D. Demaine, Fedor V. Fomin, Mohammad Taghi Hajiaghayi, and Dimitrios M. Thilikos. Bidimensional parameters and local treewidth. In

LATIN, pages 109–118, 2004.

[8] Erik D. Demaine, Fedor V. Fomin, Mohammad Taghi Hajiaghayi, and Dimitrios M. Thilikos. Fixed-parameter algorithms for (k, r)-center in planar graphs and map graphs. *ACM Transactions on Algorithms*, 1(1):33–47, 2005.

[9] David Eppstein. Dynamic generators of topologically embedded graphs. In *Proc. 14th Symp. Discrete Algorithms*, pages 599–608. ACM and SIAM, January 2003.

[10] Tomás Feder and Daniel Greene. Optimal algorithms for approximate clustering. In *Proceedings of the twentieth annual ACM symposium on Theory of computing*, STOC '88, pages 434–444, New York, NY, USA, 1988. ACM.

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

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

[13] D. S. Hochbaum and W. Maass. Fast approximation algorithms for a nonconvex covering problem. *J. Algorithms*, 8:305–323, 1987.

[14] Ken-ichi Kawarabayashi, Philip N. Klein, and Christian Sommer. Linear-space approximate distance oracles for planar, bounded-genus, and minor-free graphs. In *Proceedings of the 38th International Colloquium on Automata, Languages and Programming, ICALP 2011*, pages 135–146, 2011.

[15] R. J. Lipton and R. E. Tarjan. A separator theorem for planar graphs. *SIAM Journal on Applied Mathematics*, 36(2):177–189, 1979.

[16] P. Seymour and R. Thomas. Call routing and the ratcatcher. *Combinatorica*, 14(2):217–241, 1994.

[17] Mikkel Thorup. Compact oracles for reachability and approximate distances in planar digraphs. *Journal of the ACM*, 51(6):993–1024, 2004.