Approximation algorithms for edge-dilation k-center problems

Jochen Könemann[1], Yanjun Li, Ojas Parekh, and Amitabh Sinha

Carnegie Mellon University, Pittsburgh, PA 15213-3890, USA.
{jochen,yanjun,odp,asinha}@andrew.cmu.edu

Abstract. In an ideal point-to-point network, any node would simply choose a path of minimum latency to send packets to any other node; however, the distributed nature and the increasing size of modern communication networks may render such a solution infeasible, as it requires each node to store global information concerning the network. Thus it may be desirable to endow only a small subset of the nodes with global routing capabilites, which gives rise to the following graph-theoretic problem.

Given an undirected graph $G = (V, E)$, a metric l on the edges, and an integer k, a k-center is a set $\Pi \subseteq V$ of size k and an assignment π_v that maps each node to a unique element in Π. We let $\mathtt{d}_\pi(u, v)$ denote the length of the shortest path from u to v passing through π_u and π_v and let $\mathtt{d}_l(u, v)$ be the length of the shortest u, v-path in G. We then refer to $\mathtt{d}_\pi(u, v)/\mathtt{d}_l(u, v)$ as the *stretch* of the pair (u, v). We let the stretch of a k-center solution Π be the maximum stretch of any pair of nodes $u, v \in V$. The minimum edge-dilation k-center problem is that of finding a k-center of minimum stretch.

We obtain combinatorial approximation algorithms with constant factor performance guarantees for this problem and variants in which the centers are capacitated or nodes may be assigned to more than one center. We also show that there can be no $5/4 - \epsilon$ approximation for any $\epsilon > 0$ unless $\mathcal{P} = \mathcal{NP}$.

1 Introduction

In this paper we consider the following graph-theoretic problem: we are given an undirected edge-weighted graph $G = (V, E, l)$, (l is the (metric) edge-weight function), and a parameter $k > 0$. We want to find a set $\Pi \subseteq V$ of k center nodes and assign each node $v \in V$ to a unique center $\pi_v \in \Pi$.

Let the *center distance* between nodes $u, v \in V$ be defined as

$$\mathtt{d}_\pi(u, v) = \mathtt{d}_l(u, \pi_u) + \mathtt{d}_l(\pi_u, \pi_v) + \mathtt{d}_l(\pi_v, v)$$

where $\mathtt{d}_l(u, v)$ denotes the shortest path distance between nodes u and v. The *stretch* for a pair of nodes $u, v \in V$ is then defined as the ratio $\mathtt{d}_\pi(u, v)/\mathtt{d}_l(u, v)$ of center distance and shortest path distance. We let the stretch of a solution $(\Pi, \{\pi_v\}_{v \in V})$ be the maximum stretch of any pair of nodes $u, v \in V$. The goal in the *minimum edge-dilation k-center problem* (MEDKC) is to find a set $\Pi \subseteq V$ of cardinality at most k and an assignment $\{\pi_v\}_{v \in V}$ of nodes to centers of minimum stretch.

A closely related problem is that of finding a *k-center* in a given graph $G = (V, E)$. Here, we want to find a set of nodes $C \subseteq V$ of cardinality at most k such that the maximum

[1] This material is based upon work supported by the National Science Foundation under Grant No. 0105548.

distance from any node to its closest center is as small as possible. This problem admits a 2-approximation in the undirected setting [2, 5, 11] and it is well-known that there cannot exist a $2 - \epsilon$ approximation for any $\epsilon > 0$ unless $\mathcal{P}$=$\mathcal{NP}$ [6, 10]. We adapt techniques used for the k-center problem to minimize the bottleneck stretch of any pair of nodes $u, v \in V$. Our main result is the following:

Theorem 1. *There is a polynomial-time algorithm that computes a feasible solution $\Pi \subseteq V$ to the* `MEDKC` *problem such that for every two vertices $u, v \in V$ we have $\mathtt{d}_\pi(u, v)/\mathtt{d}_l(u, v) \leq 4 \cdot \mathtt{opt} + 3$, where* `opt` *is the optimal stretch. On the negative side, no $5/4-\epsilon$ approximation can exist for any $\epsilon > 0$ unless $\mathcal{P}$=$\mathcal{NP}$.*

The `multi-MEDKC` problem is a natural extension of the `MEDKC` problem. Here, each vertex is allowed to keep a set of centers $\pi_v \subseteq \Pi$. We redefine the center distance between nodes u and v as

$$\mathtt{d}^m_\pi(u, v) = \min_{\pi_1 \in \pi_u, \pi_2 \in \pi_v} \mathtt{d}_l(u, \pi_1) + \mathtt{d}_l(\pi_1, \pi_2) + \mathtt{d}_l(\pi_2, v).$$

Again, the task is to find a set of center nodes $\Pi \subseteq V$ of cardinality at most k that minimizes the maximum stretch, now defined as $\mathtt{d}^m_\pi(u, v)/\mathtt{d}_l(u, v)$.

Theorem 2. *Given an undirected graph $G = (V, E)$ and a non-negative length function l on E, there is a polynomial-time algorithm that computes a solution Π to the* `multi-MEDKC` *problem such that for every two vertices $u, v \in V$ we have $\mathtt{d}^m_\pi(u, v)/\mathtt{d}_l(u, v) \leq 2 \cdot \mathtt{opt} + 1$.*

Subsequently, we extend the result in Theorem 1 to a natural capacitated version of the `MEDKC` problem (denoted by `C-MEDKC`): each potential center location $v \in V$ has an associated capacity U_v. We now want to find a minimum-stretch center set $\Pi \subseteq V$ of size at most k and an assignment $\{\pi_v\}_{v \in V}$ of nodes to centers such that the set $\pi_i^{-1} = \{v \in V : \pi_v = i\}$ has size at most U_i for all $i \in \Pi$. We adapt facility location techniques from [12] in order to obtain the following bicriteria result:

Theorem 3. *Given an instance of the* `C-MEDKC` *problem, there is a polynomial-time algorithm that computes a center set $\Pi = \{\pi_1, \ldots, \pi_{2k}\}$ and an assignment of nodes to centers such that $|\pi_i^{-1}| \leq 2U_i$ for all $1 \leq i \leq 2k$. The stretch of the solution is at most $12 \cdot \mathtt{opt} + 1$ where* `opt` *is the stretch of an optimum solution which places no more than k centers and obeys all capacity constraints.*

The problem motivation comes from (distributed) *routing* in computer networks. Here, a host v keeps information about routing paths to each other host u locally in its *routing table*. The entry for node v in u's routing table consists of the next node on the routing path from node u to node v. Clearly, we can ensure shortest-path routing if we allow each node to store $O(n)$ entries in its routing table.

Considering the size of modern computer networks that often connect millions of nodes, we can hardly ask each node to store information for every other host in the network. For this reason, modern routing protocols like OSPF[8] allow a subdivision of a network into areas. Now, each node keeps an entry for every other node in the same area. Routing between nodes in different areas is done via a backbone network of area border routers that interconnects the areas.

We can formalize the above problem as follows: We allow each node to store up to $O(B)$ entries in its routing table, where B is a constant representing the memory available at each node. These are the nodes with which it can directly communicate. In addition, we install a supporting backbone network of k center nodes. Each node is allowed to keep an additional entry in its routing table for the center node π_v that it is assigned to. Whenever node v needs to compute a route to node u that is not in its routing table, it has to route via its center π_v. As before, we assume that routing among center nodes is along shortest paths.

The problem now is to place k center nodes and configure the routing tables of each of the nodes in V such that the maximum stretch of any path is minimum. We refer to this problem as `MEDKC` *with bounded routing table space* (`B-MEDKC`). We obtain the following theorem.

Theorem 4. *Given an instance of the* `B-MEDKC` *problem, we can find in polynomial time a center set Π and an assignment $\{\pi_v\}_{v \in V}$ of nodes to centers that achieves stretch $O(\rho \cdot \mathtt{opt})$ where* `opt` *denotes the optimum stretch of any* `B-MEDKC` *solution and ρ is the performance guarantee of any algorithm for the* `MEDKC` *problem.*

We note that the last result is closely related to work on *compact routing schemes* (see [1] and the references therein). Cowen [1] shows that if we allow $O(n^{2/3} \log^{4/3} n)$ table space at each node, we can achieve a solution where the routing path between any pair of nodes $u, v \in V$ is at most three times as long as the shortest u, v-path in G. Notice that this contrasts our results since we are comparing the stretch that we achieve with the minimum possible stretch.

Finally, our problem is related to that of designing *graph spanners*. In the unweighted version, first considered in [9], we are given an undirected edge-weighted graph $G = (V, E)$. A subgraph $H = (V, E_H)$ of G is called an α-spanner if we have $\mathtt{d}_H(u, v) \leq \alpha\, \mathtt{d}_G(u, v)$ for every pair of nodes $u, v \in V$. The literature on spanners is vast and includes variants such as degree-bounded spanners, sparse spanners, additive graph spanners as well as hardness results (see [3] and the references therein).

2 Hardness

We first show that the basic `MEDKC` problem is $\mathcal{NP}$-hard. Hardness of the extensions follow because each of the extensions is a strict generalization of the basic problem.

Theorem 5. *The minimum edge-dilation k-center problem is $\mathcal{NP}$-hard. Furthermore, unless $\mathcal{NP} = \mathcal{P}$, there can be no $5/4 - \epsilon$ approximation for any $\epsilon > 0$.*

Proof. The proof is by reduction from *minimum vertex-dominating set* (`MVDS`). In `MVDS` we are given an undirected graph $G = (V, E)$ and we want to find a set $S \subseteq V$ of minimum cardinality such that for all $v \in V$, either $v \in S$ or there is a $u \in S$ such that $vu \in E$. This problem is known to be $\mathcal{NP}$-hard [4].

Suppose we are given an instance of the `MVDS` problem: $G_1 = (V_1, E_1)$. We construct an edge-weighted auxiliary graph $G = (V, E, l)$ from G_1. For each node $v \in V_1$, let V contain two copies v_1 and v_2, along with an edge $v_1 v_2$ of length 1. For each edge $uv \in E_1$, we let $u_1 v_1$ of length 1 be in E. Finally, we include edge $u_1 v_1$ of length 2 in E if the shortest path between u and v in G_1 has at least 3 edges.

It is not difficult to see that if there exists a vertex dominating set in G_1 of cardinality at most k then the optimum stretch of the `MEDKC` instance given by G is at most 4 (locate the centers exactly at the positions of the vertex dominating set). Also, if there exists no vertex dominating set in G_1 with size less than or equal to k, then for any center set $\Pi \subseteq V$ of cardinality k we can always find a vertex $v^* \in V_1$ such that its copies v_1^* and v_2^* satisfy $\mathtt{d}_\pi(v_1^*, v_2^*)/\mathtt{d}_l(v_1^*, v_2^*) \geq 5$. $\square$

3 The basic MEDKC problem

In this section, we prove Theorem 1. We first develop a combinatorial lower-bound and use it to compute an approximate solution to the MEDKC problem. We then give an algorithm that computes an approximate solution to the proposed lower bound.

3.1 A lower-bound: covering edges with vertices

For each pair $u, v \in V$, consider the set

$$S_{uv}^{\alpha} = \{w \in V \, : \, \mathtt{d}_l(u,w) + \mathtt{d}_l(v,w) \leq \alpha \cdot \mathtt{d}_l(u,v)\}. \tag{1}$$

It is clear that any optimum solution Π to MEDKC needs to have at least one node from $S_{uv}^{\mathtt{opt}}$ for all pairs $u, v \in V$.

The *minimum-stretch vertex cover problem* (MSVC-α) for a given graph $G = (V, E, l)$ and a parameter $\alpha > 0$ is to find a set $C \subseteq V$ of minimum cardinality such that $S_{uv}^{\alpha} \cap C \neq \emptyset$ for all pairs $u, v \in V$. Let k_α denote the cardinality of an optimal solution to MSVC-α. The following lemma is immediate.

Lemma 1. *Suppose there is a solution with stretch α for a given instance of the* MEDKC *problem. Then $k_\alpha \leq k$.*

3.2 Computing an approximate MEDKC solution

Given an instance of MEDKC, we first compute the smallest α such that the associated MSVC-α instance has a solution of cardinality at most k.

Lemma 2. *Given an instance of* MEDKC*, let* opt *be the minimum possible stretch of any solution. We can then efficiently compute $\alpha \leq 2\,\mathtt{opt} + 1$ such that $k_\alpha \leq k$.*

Our algorithm to locate a set of center nodes $\Pi \subseteq V$ is now straightforward: Let α be as in Lemma 2 and let Π be a solution to the respective instance of MSVC-α. For each vertex $v \in V$, we assign v to the closest center in Π, i.e. $\pi_v = \operatorname{argmin}_{u \in \Pi} \mathtt{d}_l(v,u)$.

Proof of Theorem 1. Let u and v be an arbitrary pair of vertices in V. We want to bound $\mathtt{d}_\pi(u,v)$. Let c_{uv} be the node that covers the pair u, v in the MSVC-α solution.

It follows from our choice of π_v and π_u that $\mathtt{d}_l(u,\pi_u) \leq \mathtt{d}_l(u,c_{uv})$ and $\mathtt{d}_l(v,\pi_v) \leq \mathtt{d}_l(v,c_{uv})$. Hence,

$$\mathtt{d}_\pi(u,v) \leq 2(\mathtt{d}_l(u,\pi_u) + \mathtt{d}_l(v,\pi_v)) + \mathtt{d}_l(u,v) \leq (2\alpha+1)\mathtt{d}_l(u,v)$$

Using Lemma 2 we can bound $(2\alpha+1)\mathtt{d}_l(u,v)$ by $4\,\mathtt{opt} + 3$. □

3.3 Solving MSVC-α

We now proceed by giving a proof of Lemma 2. We first show how to compute a solution APX to MSVC-$(2\alpha+1)$ of cardinality at most k_α.

For a vertex v and a subset of the edges $\overline{E} \subseteq E$ define $l_v(\overline{E}) = \min_{e \in \overline{E}} l_e$ to be the minimum length of any edge $e \in \overline{E}$ that is incident to v. Also, let $S_{\alpha}^{-1}(\overline{E}, v) = \{e \in \overline{E} \, : \, v \in S_e^{\alpha}\}$ be the subset of edges in $\overline{E}$ that are covered by vertex $v \in V$.

In the following we let $\alpha' = 2\alpha + 1$ and we say that a set $C \subseteq V$ *covers* edge $e \in E$ if $S_e^{\alpha'} \cap C \neq \emptyset$. Our algorithm starts with $C = \emptyset$ and repeatedly adds vertices to C until all edges in the graph are covered. More formally, in iteration i, let the remaining uncovered set of edges be $\overline{E}$ and let $\overline{V} \subseteq V$ be the set of vertices that have positive degree in $\overline{E}$. Let $e_i \in \overline{E}$ be the shortest edge in $\overline{E}$. We then choose v_i as one of the endpoints of e_i. Subsequently we remove $S^{-1}_{\alpha'}(\overline{E}, v_i)$ from $\overline{E}$ and continue.

Lemma 3. *If the above algorithm terminates with a feasible solution $C \subseteq V$ for a given instance of* MSVC-$(2\alpha+1)$ *then we must have $k_\alpha \geq |C|$.*

Proof. Assume for the sake of contradiction that there exists a set $C^* \subseteq V$ such that $|C^*| < |C|$ and for all $e \in E$, there exists $v_e \in S^\alpha_e \cap C^*$.

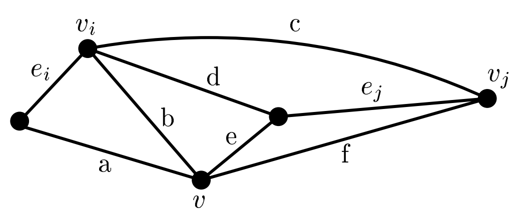

Fig. 1. Center $v \in C^*$ covers both e_i and e_j.

Recall the definition of v_i and e_i. There must exist $1 \leq i < j \leq |C|$ and a node $v \in C^*$ such that $v \in S^\alpha_{e_i} \cap S^\alpha_{e_j}$. In the following, refer to Figure 1. By definition, we must have $a + b \leq \alpha \cdot l_{e_i}$ and $e + f \leq \alpha \cdot l_{e_j}$. Using this along with triangle inequality yields $c + d \leq \alpha \cdot l_{e_i} + \alpha \cdot l_{e_j} + l_{e_i}$.

The right hand side of the last inequality is bounded by $(2\alpha+1)l_{e_j}$ by our choice of v_i. This contradicts the fact that e_j remains uncovered in iteration i. □

Proof of Lemma 2. The optimum stretch α^* of any instance has to be in the interval $[1, \mathtt{diam}(G)]$. We can use binary search to find the largest α in this interval such that the above algorithm returns a solution of cardinality at most k. Our algorithm produces a solution with stretch $2\alpha + 1$ and it follows from Lemma 3 that $\alpha^* \leq \alpha$. □

4 Choosing among many centers – multi-MEDKC

In the multi-MEDKC setting, we allow each node v to keep a set of center nodes $\pi_v \subseteq \Pi$. For each pair of nodes $u, v \in V$, we allow u and v to use the center nodes $\pi^v_u \in \pi_u$ and $\pi^u_v \in \pi_v$ that minimize

$$\mathtt{d}_l(u, \pi^v_u) + \mathtt{d}_l(\pi^v_u, \pi^u_v) + \mathtt{d}(\pi^u_v, v).$$

Notice that in an optimum solution, triangle inequality will always enforce $\pi^v_u = \pi^u_v$. Hence this problem has a solution with stretch α iff MSVC-α has a solution of cardinality at most k. This, together with Lemma 2, immediately yields Theorem 2.

A more interesting version of multi-MEDKC occurs when we restrict π_v for each node $v \in V$. For example, we might require that there is a global constant ρ such that every client node can only communicate with centers within distance ρ. We call this the *ρ-restricted* multi-MEDKC *problem.* We assume we are always given a “large enough” ρ, otherwise the problem is not meaningful.

We omit the proof of the following lemma since it is similar to that of Lemma 2.

Lemma 4. *Given an instance of ρ-restricted* `multi-MEDKC`*, let* `opt` *be its optimal stretch. We can then efficiently compute $\alpha \le 2\,\texttt{opt} + 1$ such that $k_\alpha \le k$.*

This shows that we can still use `MSVC`-α as a basis to construct a low-stretch center set. The following is again an immediate corollary of Theorem 1.

Corollary 1. *There is a polynomial time algorithm to solve the ρ-restricted* `multi-MEDKC` *problem that achieves a stretch of at most $4 \cdot \texttt{opt} + 3$.*

5 Capacitated center location

We now come to the capacitated version of the basic `MEDKC` problem. Here, we want to find a minimum-stretch center set (and assignment of nodes to centers) of cardinality at most k such that the number of nodes that are assigned to center i is no more than U_i, specified in the input.

5.1 A modified lower bound

For each node v, we define $l_v = \min_{uv \in E} l_{uv}$. For a given stretch $\alpha \ge 1$ let S_v^α be the set of nodes whose distance from v is at most $\alpha \cdot l_v$, i.e. $S_v^\alpha = \{u \in V \,:\, \mathtt{d}_l(v,u) \le \alpha \cdot l_v\}$. The optimum solution must have a center node in S_v^α in order to cover the shortest edge incident to v.

We now need to find a set of vertices $C \subseteq V$ of minimum cardinality such that $C \cap S_v^\alpha \ne \emptyset$ for all $v \in V$. Additionally, we require that each node v is assigned to exactly one center node π_v and that the sets $\pi_i^{-1} = \{u \in V \,:\, \pi_u = i\}$ have size at most U_i for all $i \in V$. Let this problem be denoted by $\texttt{MSVC2} - \alpha$.

Lemma 5. *Suppose there is a solution with stretch α for a given instance of the* `C-MEDKC` *problem. Then there must be a solution C to the associated $\texttt{MSVC2}-\alpha$ instance with $|C| \le k$.*

We model the lower-bound by using an integer programming formulation. We then solve the LP relaxation of the model and round it to an integer solution using ideas from [12]. Finally, we prove that this solution yields a solution for the original instance of `C-MEDKC` with low stretch.

5.2 A facility location type LP

In the IP, we have a binary indicator variable y_i for each $i \in V$ that has value 1 iff we place a center node at i. Additionally, we have variables x_{iv} that have value 1 iff $\pi_v = i$. The following IP formulation models $\texttt{MSVC2} - \alpha$.

$$\min \sum_{i \in V} y_i \tag{IP}$$

$$\text{s.t} \sum_{i \in S_v^\alpha} x_{iv} \ge 1 \qquad \forall v \in V \tag{2}$$

$$\sum_{v} x_{iv} \le U_i\, y_i \quad \forall i \in V \tag{3}$$

$$x_{iv} \le y_i \qquad \forall i, v \in V \tag{4}$$

$$x_{iv}, y_i \in \{0,1\} \quad \forall i, v \in V \tag{5}$$

We refer to the LP relaxation of the above IP as (LP).

It follows from Lemma 5 that if there is a feasible solution for C-MEDKC with stretch opt , then (LP) with $\alpha =$ opt has a solution with value at most k.

We next show how to round a solution (x^0, y^0) of (LP) to a solution $(\overline{x}, \overline{y})$ of cost at most twice the cost of the original solution and such that $\overline{y}$ is binary. All capacity constraints are violated by at most a factor of two. Moreover, if node v is assigned to facility i, i.e. $\overline{x}_{iv} > 0$, then the distance between i and v is not too large, i.e. $i \in S_v^{3\alpha}$. Finally, we show how to assign each node v to a unique center π_v, and prove that this solution to C-MEDKC has low stretch.

5.3 Algorithm details

Starting with a fractional solution (x^0, y^0) of (LP), we iteratively modify it in order to finish with $(\overline{x}, \overline{y})$ which satisfies the conditions above. We refer to the solution at the beginning of iteration j as (x^j, y^j).

We call a center i *fractionally opened* if $y_i^0 > 0$. In the course of the algorithm we *open* a subset of the set of fractionally opened centers. The indicator variables for open center nodes are rounded to one. We let O^j be the set of open centers at the beginning of iteration j. Initially, let O^0 be the empty set.

The procedure maintains the following invariants for all iterations $1 \le j \le t$:

(I1) $\sum_i y_i^j \le 2 \sum_i y_i^0$
(I2) $\sum_i x_{iv}^j \ge 1/2$ for all nodes $v \in V$
(I3) $\sum_v x_{iv}^j \le U_i y_i^j$ for all $i \in V$

We say that a node is *satisfied* if in iteration j we have $\sum_{i \in O^j} x_{iv}^j \ge 1/2$. Let S^j denote the set of satisfied nodes in iteration j. Our algorithm stops in iteration t when no unsatisfied nodes remain. We then increase the assignment of nodes j to open centers $i \in O^t$ such that the final solution satisfies the demand constraints (2).

A detailed description of an iteration follows. An iteration j starts by selecting an unsatisfied node v_j of minimum l_v value. We let $I(v_j)$ be the set of centers that fractionally serve v_j and have not yet been opened, i.e.

$$I(v_j) = \{i \in V \setminus O^j \,:\, x_{iv_j}^j > 0\} = \{i_1, \ldots, i_p\}.$$

W.l.o.g., assume that $U_{i_1} \ge U_{i_2} \ge \cdots \ge U_{i_p}$. We now open the first $\gamma = \left\lceil \sum_{i \in I(v_j)} y_i \right\rceil$ centers from $I(v_j)$ and close all fractional centers in $\{i_{\gamma+1}, \ldots, i_p\}$, i.e. $O^{j+1} = O^j \cup \{i_1, \ldots, i_\gamma\}$. Hence $y_i^{j+1} = 1$ for all $i \in \{i_1, \ldots, i_\gamma\}$ and $y_i^{j+1} = 0$ otherwise. We let $y_i^{j+1} = y_i^j$ for all $i \notin I(v_j)$.

Notice that a variable y_i^0 for a fractionally opened center node i is modified exactly once by the procedure outlined above. This modification happens whenever i is either opened or closed. It follows from this observation that

$$\sum_{i \in I(v_j)} y_i^0 = \sum_{i \in I(v_j)} y_i^j \ge 1/2$$

where the last inequality is a consequence of the fact that v_j is unsatisfied with respect to y^j. Therefore,

Lemma 6. *Let v_j be the unsatisfied node chosen in iteration j of our algorithm and let y^0 and y^{j+1} be defined as before. We then must have $\sum_{i\in I(v_j)} y_i^{j+1} \leq 2\sum_{i\in I(v_j)} y_i^0$.*

This shows that invariant (I1) is preserved throughout the algorithm.

It remains to modify x^j and obtain x^{j+1} so that invariants (I2) and (I3) are maintained. Specifically, we have to modify x^j such that no node is assigned to closed centers and all capacity constraints are satisfied with respect to y^{j+1}.

For an arbitrary node v, let $\omega_v = \sum_{i\in I(v_j)} x_{iv}^j$ be the assignment of v to centers from $I(v_j)$. Let Ω^j be the set of unsatisfied nodes that are attached to $I(v_j)$, i.e.

$$\Omega^j = \{v \in V \setminus S^j \,:\, \omega_v > 0\}.$$

We now (fractionally) assign the nodes from Ω^j to centers $i_1, \dots, i_\gamma$ such that for all $1 \leq l \leq \gamma$ at most U_{i_l} nodes are assigned to y_{i_l} and no node is assigned to any node in $\{i_{\gamma+1}, \dots, i_p\}$. The existence of such an assignment (and hence the validity of (I3)) follows from the following lemma (implicit in [12]):

Lemma 7. *Let v_j, γ and $I(v_j)$ be defined as above. Then, $\sum_{i\in I(v_j)} U_i y_i^j \leq \sum_{l=1}^{\gamma} U_{i_l}$.*

We do not reassign nodes v that were satisfied with respect to (x^j, y^j). Hence, a satisfied node v might lose at most $1/2$ of its demand that was assigned to now closed centers. This entails invariant (I2).

At termination time t, all nodes are satisfied. We now obtain a solution $\overline{x}$ that satisfies the demand constraints (2) of (LP) by scaling x^t appropriately. For all $i \in O^t$, $v \in V$, let $\overline{x}_{iv} = x_{iv}^t / \sum_{i\in O^t} x_{iv}^t$. Invariant (I2) implies the following lemma.

Lemma 8. *Let i be a center opened by the above algorithm. Then, $\sum_v \overline{x}_{iv} \leq 2U_i$.*

It remains to show that whenever we have $\overline{x}_{iv} > 0$ it must be that $v \in S_v^{3\alpha}$. The proof of this lemma is similar to that of Lemma 3. It crucially uses the ordering in which the algorithm considers vertices and triangle inequality. We omit the details from this extended abstract.

Lemma 9. *Let $\overline{x}, \overline{y}$ be the solution computed by the preceding algorithm. We must have $i \in S_v^{3\alpha}$ whenever $\overline{x}_{iv} > 0$.*

An observation from [13] enables us to assign each node v to a unique center π_v without increasing the violation of any of the capacity constraints.

Lemma 10. *Let $(\overline{x}, \overline{y})$ be feasible for (2), (4) and (5) such that for all $i \in V$ we have $\sum_{v\in V} \overline{x}_{iv} \leq 2U_i\overline{y}_i$ and $\overline{y}$ is binary. Then, there exists an integral feasible solution (x, y) such that $\sum_{v\in V} x_{iv} \leq 2U_i y_i$ for all $i \in V$ and $\sum_i y_i \leq \sum_i \overline{y}_i$.*

We let $\pi_v = i$ iff $x_{iv} = 1$ and prove Theorem 3.

Proof of Theorem 3. We only need to show that for any $u, v \in V$, we have $\mathtt{d}_\pi(u,v) \leq (12\alpha+1)\cdot\mathtt{d}_l(u,v)$. Theorem 3 then follows from Lemma 5 and the fact that we can perform a binary search to find the right estimate for α.

Let us estimate $\mathtt{d}_\pi(u,v)$: From triangle inequality, we obtain that

$$\mathtt{d}_\pi(u,v) \leq 2(\mathtt{d}_l(u,\pi_u) + \mathtt{d}_l(v,\pi_v)) + \mathtt{d}_l(u,v). \tag{6}$$

It follows from Lemma 9 that we must have $\mathtt{d}_l(u,\pi_u) \leq 3\alpha\cdot l_u \leq 3\alpha\cdot\mathtt{d}(u,v)$ and $\mathtt{d}_l(v,\pi_v) \leq 3\alpha\cdot l_v \leq 3\alpha\cdot\mathtt{d}(u,v)$. Hence we obtain together with (6) that $\mathtt{d}_l(u,v) \leq (12\alpha+1)\mathtt{d}_l(u,v)$. $\square$

6 Bounded routing table space

In this section we prove Theorem 4. Our algorithm preprocesses the input graph and then relies on our procedure for MEDKC from Section 3.

Preprocessing: For each node v, let B_v denote the set of B closest nodes to v in V. Let $r_v = \max_{u \in B_v} \mathtt{d}_l(u, v)$ be the radius of B_v.
Initialize $\overline{V}$ by V and let $T = \emptyset$. Now, as long as $\overline{V}$ is non-empty, find $u = \mathrm{argmin}_{w \in \overline{V}}\, r_w$, remove B_u from $\overline{V}$ and add u to T. For all nodes w that we removed from $\overline{V}$, let $\mathtt{wit}(w) = u$ and let $\mathtt{wit}(u) = u$.

Center location: Using a slight modification of the MEDKC algorithm of Section 3, solve the following extension of the MEDKC problem on G. Instead of minimizing the stretch over all u, v pairs, we minimize the stretch only over pairs $u, v \in T \times T$. We use $\mathtt{opt}_1$ to denote the stretch of the computed solution and denote the center of terminal $v \in T$ by π_v.

Solution to B-MEDKC: Finally, for $v \in V \setminus T$, let $\pi_v = \pi_{\mathtt{wit}(v)}$. Also, we keep $B_{\mathtt{wit}(v)}$ as the routing table of each node $v \in V$.

6.1 Analysis

We first relate the optimum stretch $\mathtt{opt}$ for the B-MEDKC instance to $\mathtt{opt}_1$.

Lemma 11. $\mathtt{opt}_1 \le 6 \cdot \mathtt{opt} + 3$.

Proof. Let $u, v \in T$ be any two terminal nodes. Let $R_v \subseteq V$ be the routing table content of node v in the optimum solution. There must exist a node $u' \in B_u$ such that $u' \notin R_v$. Suppose the optimum solution assigns center π^*_w to each node $w \in V$.

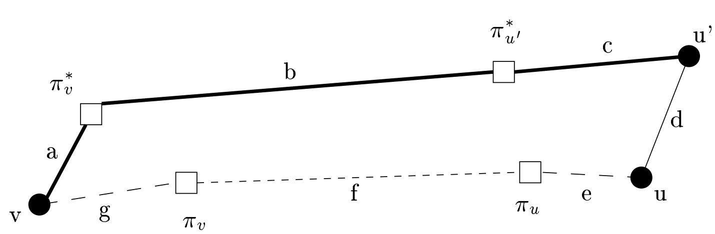

Fig. 2. Bounding the center distance between u and v.

Let $\pi_v = \mathrm{argmin}_{\pi \in \Pi^*} \mathtt{d}_l(v, \pi)$. In the following we bound $\mathtt{d}_\pi(v, u)$. For brevity, we use letters a-f to denote certain inter-point distances (see Figure 2).

$$
\begin{aligned}
\mathtt{d}_\pi(v, u) &\le 2g + a + b + c + d + 2e \\
&\le 3 \cdot \mathtt{d}_{\pi^*}(v, u') - b + 3d \qquad (7)\\
&\le 3 \cdot \mathtt{d}_{\pi^*}(v, u') + 3r_u \qquad (8)
\end{aligned}
$$

where (7) follows from the fact that $g \le a$ and $e \le c + d$ by our choice of π_u and π_v, respectively. Inequality (8) follows from the fact that $u' \in B_u$.

Assume w.l.o.g, that u has been picked before v in the preprocessing phase (the other case is symmetrical). Then we must have $r_u \leq \mathtt{d}_l(v,u)$. Using (8) yields

$$\mathtt{d}_\pi(v,u) \leq 3 \cdot \mathtt{d}_{\pi^*}(v,u') + 3 \cdot \mathtt{d}_l(v,u). \tag{9}$$

Finally notice that $\mathtt{d}_{\pi^*}(v,u') \leq \mathtt{opt} \cdot \mathtt{d}_l(v,u') \leq \mathtt{opt} \cdot (\mathtt{d}_l(v,u) + r_u) \leq 2\,\mathtt{opt} \cdot \mathtt{d}_l(v,u)$. This together with (9) finishes the lemma. □

Lemma 12. *For all pairs of nodes $u, v \in V$, we must have $\mathtt{d}_\pi(v,u) \leq (4\rho \cdot \mathtt{opt}_1 + 3)\mathtt{d}_l(v,u)$, where ρ denotes the performance ratio of an algorithm for* MEDKC*.*

Proof. We assume that $u \notin B_{\mathtt{wit}(v)}$ since we're done trivially otherwise. We first bound the distance between $\mathtt{wit}(v)$ and $\mathtt{wit}(u)$.

$$\mathtt{d}_l(\mathtt{wit}(v), \mathtt{wit}(u)) \leq r_{\mathtt{wit}(v)} + r_{\mathtt{wit}(u)} + \mathtt{d}_l(v,u) \leq r_v + r_u + \mathtt{d}_l(v,u) \leq 4 \cdot \mathtt{d}_l(v,u)$$

where the last inequality uses $r_u \leq \mathtt{d}_l(v,u)$ and $r_v \leq \mathtt{d}_l(v,u) + r_u$. It follows that $\mathtt{d}_\pi(v,u) \leq r_v + r_u + \rho\,\mathtt{opt}_1 \cdot \mathtt{d}_l(\mathtt{wit}(v), \mathtt{wit}(u)) \leq (4\rho\,\mathtt{opt}_1 + 3)\mathtt{d}_l(v,u)$. □

The proof of Theorem 4 follows using Lemmas 11 and 12.

7 Open problems

An apparent open question is to develop a unicriteria approximation algorithm for the capacitated case (maybe based on the ideas in [7]). Furthermore, an interesting remaining problem is to extend Theorem 4 to the case where we do not have a backbone network. In other words, how close to the best possible stretch can we get given limited routing table space B? A possible direction would be to explore stronger combinatorial lower-bounds and explore the merit of LP techniques.

References

1. L. Cowen. Compact routing with minimum stretch. *J. Algorithms*, 38(1):170–183, 2001.
2. M. E. Dyer and A. M. Frieze. A simple heuristic for the p-centre problem. *Operation Research Letters*, 3(6):285–288, 1985.
3. M. Elkin and D. Peleg. $(1+\epsilon, \beta)$-spanner constructions for general graphs. In *Proceedings, ACM Symposium on Theory of Computing*, pages 173–182, 2001.
4. M. R. Garey and D. S. Johnson. *Computers and Intractability: A guide to the theory of NP-completeness*. W. H. Freeman and Company, San Francisco, 1979.
5. D. Hochbaum and D. Shmoys. A best possible approximation algorithm for the k-center problem. *Mathematics of Operations Research*, 10(2):180–184, 1985.
6. W. L. Hsu and G. L. Nemhauser. Easy and hard bottleneck location problems. *Discrete Appl. Math.*, 1:209–216, 1979.
7. S. Khuller and Y. Sussmann. The capacitated K-center problem. *SIAM Journal on Discrete Mathematics*, 13(2):403–418, 2000.
8. J. Moy. OSPF version 2. IETF RFC 2328, 1998. (http://www.ietf.org/rfc.html).
9. David Peleg and Jeffrey D. Ullman. An optimal synchronizer for the hypercube. *SIAM Journal on Computing*, 18(4):740–747, 1989.
10. J. Plesnik. On the computational complexity of centers locating in a graph. *Aplikace Matematiky*, 25:445–452, 1980.
11. J. Plesnik. A heuristic for the p-center problem in graphs. *Discrete Applied Mathematics and Combinatorial Operations Research and Comp. Sci.*, 17:263–268, 1987.
12. D. Shmoys, E. Tardos, and K. Aardal. Approximation algorithms for facility location problems. In *ACM Symp. on Theory of Computing*, pages 265–274, 1997.
13. D. B. Shmoys and E. Tardos. An improved approximation algorithm for the generalized assignment problem. *Math. Programming*, A62:461–474, 1993.