ELSEVIER

Information Processing Letters 56 (1995) 67-71

Information Processing Letters

On a generalization of the p-Center Problem

S.O. Krumke [1]

Department of Computer Science, University of Würzburg, Am Hubland, 97074 Würzburg, Germany

Received 7 March 1995; revised 31 July 1995
Communicated by H. Ganzinger

Abstract

We study a generalization of the p-Center Problem, which we call the *α-Neighbor p-Center Problem* (p-CENTER$^{(\alpha)}$). Given a complete edge-weighted network, the goal is to minimize the maximum distance of a client to its α nearest neighbors in the set of p centers. We show that in general finding a $O(2^{\text{poly}(|V|)})$-approximation for p-CENTER$^{(\alpha)}$ is NP-hard, where $|V|$ denotes the number of nodes in the network. If the distances are required to satisfy the triangle inequality, there can be no polynomial time approximation algorithm with a $(2-\varepsilon)$ performance guarantee for any fixed $\varepsilon > 0$ and any fixed $\alpha \leqslant p$, unless P = NP. For this case, we present a simple yet efficient algorithm that provides a 4-approximation for $\alpha \geqslant 2$. If $\alpha = 1$, our algorithm basically falls back to the algorithm presented in [2] and has a relative performance guarantee of 2.

Keywords: Algorithms; Approximation algorithms; Location problems

1. Introduction and basic definitions

The p-Center Problem (p-CENTER for short) is one of the classical location problems. The objective is to select a set of p centers such that the maximum distance of a non-center to its nearest center is minimized. The problem is used e.g. to model the placement of emergency facilities such as fire stations or hospitals, where the aim is to have a minimum guaranteed response time between a client and its center.

In this paper, we study a generalization of p-CENTER, which we call the *α-Neighbor p-Center Problem* (p-CENTER$^{(\alpha)}$). Given again an edge-weighted network, the target is now to minimize the maximum distance of a client to its α nearest neighbors in the set of p centers. For $\alpha = 1$, p-CENTER$^{(\alpha)}$ is identical to p-CENTER.

Recall that an approximation algorithm A for a minimization problem is said to have a performance guarantee of $K > 0$, if given any instance I of the problem it returns a solution $A(I)$ of value at most K times the optimal function value, i.e. if $A(I)/OPT(I) \leqslant K$ for any instance I.

We show that in general finding a $O(2^{\text{poly}(|V|)})$ approximation for p-CENTER$^{(\alpha)}$ is NP-hard. If the distances are required to satisfy the triangle inequality, there can be no polynomial time approximation algorithm with a $(2-\varepsilon)$ performance guarantee for any fixed $\varepsilon > 0$ and any fixed $\alpha \leqslant p$, unless P = NP.

Hochbaum and Shmoys [2] have developed an approximation algorithm for p-CENTER for the case when the distances in the graph obey the triangle inequality. Their algorithm has performance ratio 2. Moreover, it is shown in [2] that this is the best approximation ratio possible, i.e., that there can be no polynomial time approximation algorithm with a per-

[1] Email: krumke@informatik.uni-wuerzburg.d400.de.

SSDI 0020-0190(95)00141-7

formance guarantee of $(2-\varepsilon)$ for any $\varepsilon > 0$ unless P = NP.

We show that the techniques of Hochbaum and Shmoys can be extended to obtain a simple yet efficient polynomial time approximation algorithm for p-CENTER$^{(\alpha)}$.

Let $G = (V, E)$ be a graph. We will use $\delta(e)$ for the weight of the edge $e \in E$. If the endpoints of e are known, i.e. $e = \{u, v\}$, we will use $\delta(u, v)$ for the edge weight for the sake of a shorter notation. As usual, we say that a nonnegative distance δ on the edges of G satisfies the *triangle inequality*, if $\delta(v, w) \leqslant \delta(v, u) + \delta(u, w)$ for all $v, w, u \in V$. We are now ready to state the problem formally:

Definition 1. (α-Neighbor p-Center Problem (p-CENTER$^{(\alpha)}$)

Input: An undirected complete graph $G = (V, E_c)$ with nonnegative edge weights $\delta(e)$ $(e \in E_c)$ and two integers $2 \leqslant \alpha \leqslant p \leqslant |V|$.

Output: A set $P \subseteq V$ of p nodes such that

$$\mathcal{R}^{(\alpha)}(P) = \max_{v \in V-P} \delta^{(\alpha)}(v, P)$$

is minimized, where

$$\delta^{(\alpha)}(v, P) = \min_{S \subseteq P,\, |S| = \alpha} \max_{s \in S} \delta(s, v).$$

The subset of instances such that the distances obey the triangle inequality will be denoted by p-CENTER$^{(\alpha)}$-TI. Notice that for any subset $P \subseteq V$ of p nodes we have that $\mathcal{R}^{(1)}(P) \leqslant \mathcal{R}^{(2)}(P) \leqslant \cdots \leqslant \mathcal{R}^{(p)}(P)$.

The following definitions are mainly taken from [2]. For a given number Δ, the *bottleneck graph* Bottleneck(G, Δ) of G is defined to be the edge-subgraph containing those edges of the original graph G, which have weight at most Δ. The *t-closure* $G^t = (V, E^t)$ of G contains an edge from u to v if and only if there is a path of length at most t edges in G connecting u and v. For any subset $V' \subseteq V$, we use $G[V']$ to denote the subgraph induced by the nodes in V'.

If $v \in V$ is any node, we let $N_G(v) := \{w \mid \{v, w\} \in E\}$ be the set of neighbors of v in G. Moreover, for any set $S \subseteq V$ we define $N_G(S) := \bigcup_{v \in S} N(v)$.

Recall that a set $U \subseteq V$ is called *independent*, if for any pair u, v of nodes from U there is no edge connecting u and v.

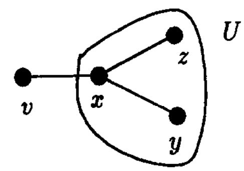

Fig. 1.

Definition 2. A k-independent set is a subset $U \subseteq V$ such that every node $v \in U$ has at most $k-1$ neighbors in U.

Consequently, a 1-independent set is a classical independent set. We use the term *maximal k-independent set* to denote a k-independent set that is maximal with respect to inclusion. Given a graph G, we can always find a maximal k-independent set in polynomial time, simply by choosing a node and then adding nodes repeatedly, until any further addition of a single node would destroy the k-independence of our set.

As the definition of k-independence extends the classical notation of independence, there is a parallel for dominating sets. Recall that a set $D \subseteq V$ is called a *dominating set*, if any node in V either belongs to D or has a neighbor in D.

Definition 3. A k-dominating set is a set $D \subseteq V$ such that each node $v \in V - D$ has at least k neighbors in D.

2. The basic lemmas

It is easy to see that a maximal independent set U is also a dominating set. For $k > 1$, in general it is not true that a maximal k-independent set is also k-dominating; see e.g. the simple example in Fig. 1:

The set $U = \{x, y, z\}$ is maximal 3-independent, because v cannot be added without destroying the 3-independence, but not 3-dominating, for v has only one neighbor in U.

We will now show that, although a maximal k-independent set need not be k-dominating in G, it will be k-dominating in the square graph G^2:

Lemma 4. *Let U be a maximal k-independent set such that $|U| \geqslant k$. Then U is a k-dominating set in G^2.*

Proof. We show that each node $v \in V - U$ has a

```
Procedure test(G)
U ← maximal α-independent set in G^2
if (|U| > p) then return "certificate of failure"
else
begin
  if |U| < p then add nodes arbitrarily to make |U| = p
  return U
end
```

Fig. 2. Test procedure for the α-Neighbor p-Center Problem.

neighbor w_1 in U such that this neighbor is adjacent to $k-1$ nodes $w_2, \ldots, w_k$ in U. It follows that the nodes $w_1, \ldots, w_k$ will be neighbors of v in G^2 and this establishes the claim.

In fact, if there existed a node v that is not adjacent to *any* node in U with $k-1$ neighbors in U, we could add v to U without destroying the property of k-independence contradicting the fact that U is maximal. □

Now we will establish a key relation between the k-dominating sets in G and the k-independent sets in the square G^2:

Proposition 5. *Let V' be a k-dominating set in G. Then $|U| \leqslant |V'|$ for any k-independent set U in G^2.*

Proof. If $U \subseteq V'$ then the claim of the proposition is trivial.

If U is not contained in V', then choose an arbitrary node $u \in U - V'$ and let $S := N_G(u) \cap V'$. Clearly $|S| \geqslant k$, because V' is a k-dominating set. Define $C := N_G(S) \cap (V - V')$ to be the set of vertices in $V - V'$ that are adjacent to the set S. Then any node in $C \cup S$ is adjacent to u in G^2 and thus U can contain at most k vertices from $C \cup S$. On the other hand, we have seen that $|V' \cap (C \cup S)| = |S| \geqslant k$.

Now consider the graph $\widehat{G} := G[V - (C \cup S)]$. We claim that $\widehat{V}' := V' - (C \cup S) = V' - S$ is a k-dominating set in $\widehat{G}$. To see this consider an arbitrary node v from $\widehat{G}$ that is not contained in $\widehat{V}'$. Then $v \in V - V'$. The node v has at least k neighbors in G that are contained in V', since again V' is a k-dominating set in G. None of these neighbors can be contained in S, because otherwise we would have $v \in N_G(S) \cap (V - V') = C$ and thus v were not contained in $\widehat{G}$. Hence $N_G(v) \cap V' \subseteq V' - S = \widehat{V}'$ and all the neighbors of v in S are still present in $\widehat{V}'$.

The set $\widehat{U} := U - (C \cup S)$ is clearly k-independent in $\widehat{G}$. Thus we can repeat the above construction for $V' := \widehat{V}'$ and $U := \widehat{U}$ until we obtain that $\widehat{U} \subseteq \widehat{V}'$. Since in each step we delete at most k nodes from U and at least k nodes from V' it follows that $|U| \leqslant |V'|$. □

3. The algorithm

In this section we will present the algorithm and use the results from Section 2 to analyze its performance guarantee. The techniques that are used, were introduced in [2].

Let $P^* \subseteq V$ be an optimal placement of p nodes and denote the optimal solution value by $\delta^* = \mathcal{R}^{(\alpha)}(P^*)$. The idea behind the algorithm is the following: By definition of the objective function $\mathcal{R}^{(\alpha)}$, the optimal function value δ^* must equal the weight of an edge. We will present a relaxed test procedure test that, given a number Δ either tells us that $\delta^* > \Delta$ or delivers a solution of cost at most 4Δ (2Δ for $\alpha = 1$).

We now sort the edges of G in nondecreasing order, say $\delta(e_1) \leqslant \delta(e_2) \leqslant \cdots \leqslant \delta(e_{\binom{n}{2}})$, and, using the output of the procedure test, perform a binary search to locate the minimum i such that test(Bottleneck($G, \delta(e_i)$)) returns a solution. It follows by the properties of test that $\delta(e_i) \leqslant \delta^*$. The test procedure is shown in Fig. 2, the main procedure is shown in Fig. 3.

First we will establish the following:

Lemma 6. *If the procedure test(G_i) returns a "certificate of failure", then $\delta^* > \delta(e_i)$.*

Proof. Assume that test returns a "certificate of failure", but nonetheless $\delta^* \leqslant \delta(e_i)$. Let $P^* = \{v_1^*, \ldots, v_p^*\}$ be a set of p centers in G such that $\mathcal{R}^{(\alpha)}(P^*) = \delta^*$. By definition of the solution value δ^*, it follows that P^* is an α-dominating set in G_i.

The procedure test can only return a "certificate of failure", if it finds an α-independent set U in G_i^2 that contains more than p elements. But according to Proposition 5 such a set cannot exist in G_i^2. □

Theorem 7. *Let I be any instance of p-*CENTER$^{(\alpha)}$*-TI and denote by $Heur(I)$ the solution value of the solution found by the procedure* Bottleneck-Main. *Then $OPT(I)/Heur(I) \leqslant 4$, where $OPT(I)$ denotes the optimal solution value for I. If $\alpha = 1$ we have the bet-*

```
Procedure Bottleneck-Main(G, δ, p)
Sort the edges of G such that
    δ(e_1) ≤ δ(e_2) ≤ ··· ≤ δ(e_(n choose 2))
low ← k − 1; high ← |V|
while (high − low) > 1 do
  begin
    i ← ⌈(high + low)/2⌉
    G_i ← Bottleneck(G, δ(e_i))
    out.test ← test(G_i)
    if out.test is a "certificate of failure" then low ← i
      else high ← i
  end
output test(Bottleneck(G, δ(e_high)))
```

Fig. 3. Main Bottleneck procedure.

ter following estimate: $OPT(I)/Heur(I) \leqslant 2$.

Proof. Let $OPT(I) = \delta^* = \delta(e_j)$ and consider the call to the test procedure, when $i = j$. By Lemma 6 the procedure test must deliver a solution. Let this solution be P, where by construction P contains a maximal independent set in G_j^2. If $\alpha = 1$, it follows that U is a dominating set in G_j^2 as was remarked at the beginning of the last section.

If $\alpha \geqslant 2$, we can use Lemma 4 to deduce that P is an α-dominating set in $(G_j^2)^2 = G_j^4$.

By definition of the bottleneck graph G_j = Bottleneck$(\delta(e_j))$ each edge weight in G_i is at most $\delta(e_j)$. Consequently, by the triangle inequality G_j^2 and G_j^4 do not contain any edge of weight more than $2\delta(e_j) = 2\delta^*$ or $4\delta(e_j) = 4\delta^*$ respectively. Thus the claimed performance guarantee follows. □

4. Hardness results

Theorem 8. *Unless* P = NP, *for any fixed* $\alpha \leqslant p$ *there can be no polynomial time approximation for p-*CENTER$^{(\alpha)}$ *with a relative performance guarantee of* $O(2^{\text{poly}(|V|)})$. *Moreover, p-*CENTER$^{(\alpha)}$*-TI cannot be approximated in polynomial time within a factor of* $(2-\varepsilon)$ *for any* $\varepsilon > 0$.

Proof. Assume that A is an algorithm with a relative performance guarantee of $O(2^{\text{poly}(|V|)})$. Without loss of generality we can assume that the performance guarantee of A is $M \cdot 2^{q(|V|)}$, where q is a suitable polynomial. Thus given an input of length $\Omega(|V|)$ the function $f(|V|) := M \cdot 2^{q(|V|)}$ is polynomial time computable.

We will show that A can be used to decide DOMINATING SET, a well known NP-complete problem (cf. [1]).

Let I be any instance of DOMINATING SET, given by a graph $G = (V, E)$ and an integer d. We now construct an instance I' of p-CENTER$^{(\alpha)}$ in the following way: We choose $|V|$ pairwise disjoint sets $N_v := \{w_v^{(1)}, \ldots, w_v^{(\alpha-1)}\}$ $(v \in V)$ with $N_v \cap V = \emptyset$. We then let $V' := V \cup \bigcup_{v \in V} N_v$, $p' := d + (\alpha-1)|V|$, $\alpha' := \alpha$ and define $G' = (V', E')$ to be a complete graph on $|V'|$ nodes. The edge-weights $\delta'(e)$ are given by

$$\delta'(u,v) := \begin{cases} 1 & \text{if } u, v \in V \text{ and } \{u,v\} \in E, \\ 1 & \text{if } v \in V \text{ and } u \in N_v \text{ or vice versa,} \\ f(|V|) + \varepsilon' & \text{otherwise,} \end{cases}$$

where we choose $\varepsilon' > 0$ arbitrary. Fig. 4 illustrates the transformation from G to G'. All edges shown have weight 1, the edges not drawn in the figure have weight $f(|V|) + \varepsilon'$.

Observe that by definition of the edge-weights in G', any set P of p' centers has either radius $\mathcal{R}^{(\alpha')}(P) = 1$ or $f(|V|) + \varepsilon'$.

Observe further that in the special case when f is the constant function $f \equiv (2-\varepsilon)$ for some $\varepsilon > 0$, we can choose $\varepsilon' := \varepsilon$ and obtain the distances 1 and 2 in G', as a consequence of which the triangle inequality will be satisfied by the distances defined above.

First, assume that G has a dominating set D of size d. Then $P := D \cup \bigcup_{v \in V} N_v$ is a set of $d + (\alpha - 1) = p'$ centers with solution value $\mathcal{R}^{(\alpha)}(P) = 1$. In that case, because A has a performance guarantee of $f(|V|)$, the radius of the set of centers returned by A must also be 1.

Now suppose that G does not have a dominating set of size d. We claim that in this case any selection of $p' = d + (\alpha-1)|V|$ centers will have radius $f(|V|) + \varepsilon'$. To see this, assume that P is a set of p' centers with solution value 1. First observe that P must include all the nodes from $\bigcup_{v \in V} N_v$.

If $\alpha = 1$, the claim is trivially satisfied, because in that case $N_v = \emptyset$ for all $v \in V$. Hence it suffices to consider the case $\alpha \geqslant 2$. Any node $w \in N_v$ has v as the only neighbor in V', which is within a distance of 1 and, if not included in the set of centers, must be covered by at least α servers within a distance of 1.

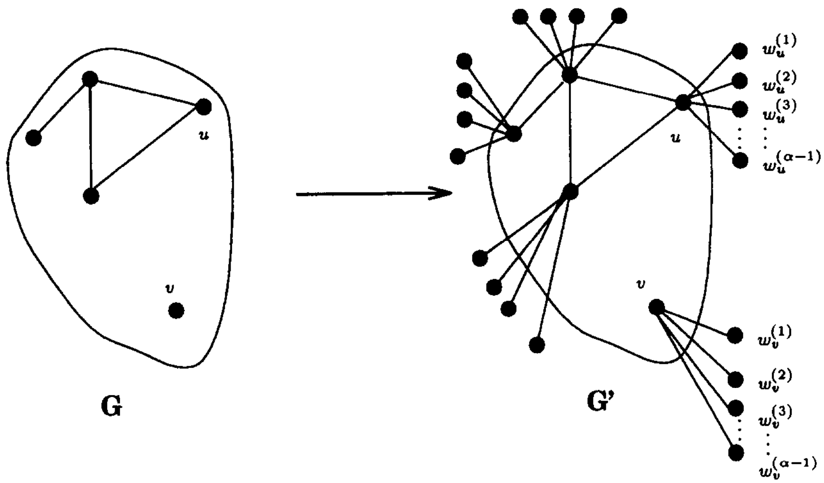

Fig. 4. Transformation of G to G'.

Hence w must be included in the set of centers.

It follows that $P_V := P \cap V \subseteq V$ consists of $|P| - (\alpha - 1)|V| = d$ nodes. We will now show that P_V is a dominating set in V. If we take $v \in V - P_V$, then v must have at least α nodes from P within a distance of 1. Only $\alpha - 1$ from these nodes can be from N_v. Thus by definition of the distances in G', there must be $w \in P_V$ such that $\delta'(v, w) = 1$, i.e. $\{v, w\} \in E$.

Hence P_V is a dominating set of size d in G as a contradiction to the assumption that G does not contain any dominating set of size at most d.

We have seen that A delivers a solution of value 1 if and only if G has a dominating set of size d. Consequently, A can be used to decide the given instance I of DOMINATING SET in polynomial time. □

It should be noted that in the proof of the last theorem $f \in O(2^{\mathrm{poly}(|V|)})$ is the largest we can do in polynomial time, since otherwise the length of the binary representation for $f(|V|)$ is no longer polynomially bounded in the input size.

References

[1] M.R. Garey and D.S. Johnson, *Computers and Intractability* (Freeman, New York, 1979).

[2] D.S. Hochbaum and D.B. Shmoys, A unified approach to approximation algorithms for bottleneck problems, *J. ACM* **33** (3) (1986) 533–550.