LOCAL PTAS FOR DOMINATING AND CONNECTED DOMINATING SET IN LOCATION AWARE UDGS

ANDREAS WIESE[*,§] AND EVANGELOS KRANAKIS[**,§§]

Abstract. We present local $1+\epsilon$ approximation algorithms for the minimum dominating and the connected dominating set problems in location aware Unit Disk Graphs (UDGs). Our algorithms are local in the sense that the status of a vertex v (i.e. whether or not v is part of the set to be computed) depends only on the vertices which are a constant number of edges (hops) away from v. This constant is independent of the size of the network. In our graph model we assume that each vertex knows its geographic coordinates in the plane (location aware nodes). Our algorithms give the best approximation ratios known for this setting. Moreover, the processing time that each vertex needs to determine whether or not it is part of the computed set is bounded by a polynomial in the number of vertices which are a constant number of hops away from it.

1. Introduction

Locality is a particularly important issue in wireless ad hoc networks since in such networks the wireless devices do not have knowledge about the entire network and it is often not practical or even impossible to explore the whole network completely. Especially in the case of dynamically changing networks the attempt to examine the entire system would require too much time. Therefore we are interested in local algorithms where the status of a node v (whether or not v is in the dominating set, connected dominating set etc.) depends only on the nodes that are at most a constant number of edges (hops) away from v. This ensures that when computing a solution in a network messages do not propagate uncontrollably far. It is also advantageous for disaster recovery since in this context one does not need to recompute the entire solution but only the parts that have been affected by the incident. Also in dynamic networks (especially but not limited to the case of ad hoc wireless networks) we want that local changes in the graph only affect the solution locally so that we do not have to recompute the entire solution if only small local changes occur.

Maintaining Topology Control (i.e., network stability, power conservation, interference reduction etc.) is an important issue in ad hoc networks. In order to deal

Research conducted while the authors were visiting the School of Computing Science at Simon Fraser University, Vancouver.

[*] Technische Universität Berlin, Institut für Mathematik, Germany.

[§] Research supported by a scholarship from DAAD (German Academic Exchange Service).

[**] School of Computer Science, Carleton University, 1125 Colonel By Drive, Ottawa, Ontario, Canada K1S 5B6.

[§§] Research supported in part by NSERC (Natural Science and Engineering Research Council of Canada). Research supported in part by MITACS (Mathematics of Information Technology and Complex Systems).

with this vertices are often organized in clusters where one vertex in each cluster takes the role of a clusterhead that the other vertices in the cluster are assigned to. So the clusterheads form a dominating set. They are responsible for the communication of the members of the cluster with other nodes. For being able to send messages from one cluster to another one wants the clusterheads to form a connected graph which results in a connected dominating set. For efficiency reasons we want the dominating and connected dominating sets to be as small as possible.

We model the wireless network as a Unit Disk Graph (UDG) consisting of nodes with identical transmission range in which each node knows about its geographic coordinates but does not have any other information that distinguishes it from the other nodes. This models the case of identical wireless devices that have a constant transmission range and know about their geographic position e.g. from a GPS receiver or from virtual coordinates assigned by another source. In wireless and sensor networks each device has a limited transmission range and whether communication between two nodes is possible depends on their Euclidean distance. With the advent of GPS devices our assumption regarding positional knowledge in each node seems to be relevant.

1.1. **Related work.** For general graphs $G = (V, E)$ with n nodes computing minimum dominating and minimum connected dominating sets is NP-hard. Dominating set cannot be approximated with a ratio better than $c \cdot \log |V|$ for some $c > 0$ unless every problem in NP can be solved deterministically in $O(n^{poly \log n})$ [16], but it can be approximated within a bound of $O(\log n)$ [9].

For the restricted case of unit disk graphs the situation is a bit different. Dominating set and connected dominating set remain NP-hard [5]. However, there are constant ratio approximations known [1, 14] and polynomial time approximation schemes. The first polynomial time approximation scheme (PTAS) was proposed by Hunt III et al. [8] and needs the geometric embedding of the graph as part of the input. Later Nieberg and Hurink [15] found a PTAS that computes an approximation for the dominating set without the geometric embedding of the graph. This is an advantage since the recognition of a unit disk graph (given a graph determining if it is a unit disk graph or not) is NP-hard [2] and therefore finding an embedding for a given unit disk graph is NP-hard as well. Even finding an approximation for an embedding of a unit disk graph is NP-hard [10]. For the connected dominating set problem Cheng et al. proposed a global PTAS [4] that needs the embedding of the graph as part of the input. Gfeller and Vicari presented a distributed (but not local) approximation scheme for connected dominating set in growth-bounded graphs [7] (this includes unit disk graphs) which does not rely on the geometric embedding of the graph.

When looking for local algorithms, Kuhn et al. [12] proposed local approximation schemes for maximum independent set and dominating set for growth-bounded-graphs. This class of graphs includes UDGs. However, in their definition of locality the status of a vertex v depends on the vertices up to $O(\log^* n)$ hops from v. This bound depends on the size of the graph and is therefore not constant. In the graph model they use the fact that each vertex has a unique ID to distinguish itself from the other vertices. This is the graph model that was mostly discussed in the literature in terms of algorithms and lower bounds, see e.g. [13, 11].

However, in this paper we assume a different model in which every vertex knows its coordinates in the plane and can make use of them for computing a dominating

or a connected dominating set. The first work on such algorithms using this model was [6]. In this paper, Czyzowicz et al. present a factor 5 approximation for dominating set and a $7.453 + \epsilon$ approximation for connected dominating set.

1.2. **Main result.** Our main results are a local $1 + \epsilon$ approximation algorithm for dominating set, a local $3 + \epsilon$ approximation algorithm for connected dominating set and a local $1 + \epsilon$ approximation algorithm for connected dominating set. The latter is an improved version of the $3 + \epsilon$ approximation algorithm mentioned before. In terms of the approximation ratio our algorithms outperform the formerly known algorithms for our setting and achieve the same performance bounds that are possible for global polynomial time algorithms assuming $P \neq NP$. The locality distance (i.e. the radius of the area that needs to be explored around a vertex) of our algorithms for connected dominating set is better than the locality distance of the previously known local algorithm [6] for this problem. However, our dominating set algorithm needs a much larger locality distance than the algorithm presented in [6] for the same problem. For each problem we give upper bounds for the respective locality distance.

The remainder of the paper is organized as follows: in Section 2 we present our local $1 + \epsilon$ approximation algorithm for minimum dominating set. In Section 3 we show how this can be extended to a local $3 + \epsilon$ approximation for connected dominating set. In Section 4 we present our local $1+\epsilon$ approximation algorithm for the minimum connected dominating set problem. Finally in Section 5 we summarize our algorithms, discuss what parameters would be worthwhile to be improved and what remains open.

2. Local Algorithm for Dominating Set

In this section we present a local $1 + \epsilon$ approximation algorithm for minimum dominating set in unit disk graphs, prove its correctness and give an upper bound for its locality distance. We show that the latter depends only on ϵ.

First we introduce some definitions and preliminaries including the concept of 2-separated collections which enables us to guarantee a $1+\epsilon$ approximation factor. This concept is the same as the one presented in [15] for giving that approximation guarantee. Then we present a tiling of the plane in hexagons with certain properties which is essentially the same tiling as the one given in [6]. After that we present our algorithm and prove its correctness.

2.1. **Preliminaries.** An undirected graph $G = (V, E)$ is a *unit disk graph* if there is an embedding in the plane for G such that two vertices u and v are connected by an edge if and only if the Euclidean distance between them is at most 1. The graph G we consider for all our algorithms is a connected unit disk graph.

A set $D \subseteq V$ *dominates* a vertex v if there is a vertex $d \in D$ and an edge $\{v, d\}$. The set D is called a *dominating set* for G if it dominates every vertex in G. A set D is called a *minimum dominating set* for G if it is a dominating set and for all other dominating sets $D' \subseteq V$ it holds that $|D| \leq |D'|$.

Definition 1. For two vertices u and v let $d(u, v)$ be the hop-distance between u and v, that is the number of edges on a shortest path between these two vertices.

The hop-distance is not necessarily the geometric distance between two vertices. Denote by $N^r(v) = \{u \in V \,|\, d(u, v) \leq r\}$ the r-neighborhood of a vertex v. For ease

of notation we set $N^0(v) := \{v\}$, $N(v) := N^1(v)$ and for a set $V' \subseteq V$ we define $N(V') = \bigcup_{v' \in V'} N(v')$. Note that $v \in N(v)$. We define the diameter of a set of vertices $V' \subseteq V$ as $diam(V') := \max_{u,v \in V'} d(u,v)$.

Denote by the *locality distance* (or short the *locality*) of an algorithm the minimum α such that the status of a vertex v (e.g. whether or not v is in a dominating or connected dominating set) depends only on the vertices in $N^\alpha(v)$. In all algorithms presented in this paper we will prove that α depends only on the desired approximation factor for the respective problem.

Now we introduce the concept of a 2-separated collection which will give us a lower bound for the optimal dominating set.

Definition 2. Let H be an index set and let the sets S_h with $h \in H$ be subsets of V. The sets S_h are called a *2-separated collection* if for any two vertices $s \in S_h$ and $s' \in S_{h'}$ with $h \neq h'$ it holds $d(s,s') > 2$.

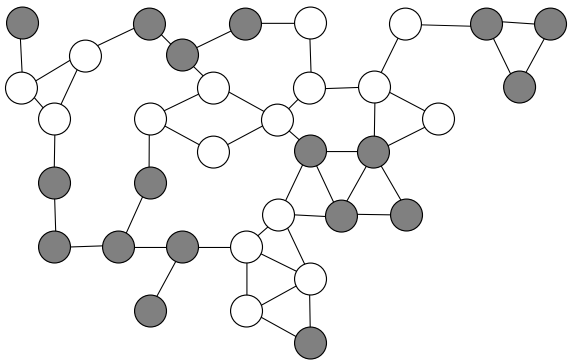

FIGURE 1. The grey vertices form a 2-separated collection

Figure 1 shows an example for a 2-separated collection.

Definition 3. Let $D : \mathcal{P}(V) \to \mathcal{P}(V)$ be an operation returning a *dominating set of minimum cardinality* for the subset of vertices given as argument to it.

Note that the set $D(V')$ may contain any vertices from V, not only vertices from V'. Now we establish our lower bound for the minimum dominating set for G.

Lemma 1. *For a 2-separated collection of sets S_h with $h \in H$ we have*

$$|D(V)| \geq \sum_{h \in H} |D(S_h)|$$

Proof. In Section 3 in [15] the concept of 2-separated collections is introduced and this lemma is proved. □

So we see that the cardinality of the union $\bigcup_{h \in H} D(S_h)$ is a lower bound for the cardinality of the minimum dominating set. The idea is now to surround each set S_h from the 2-separated collection with a suitable set T_h such that $|D(T_h)|$ is not much larger than $|D(S_h)|$. If then the union $D := \bigcup_{h \in H} D(T_h)$ is a dominating set for G, we can show that D is not much larger than an optimal dominating set for G.

Lemma 2. *Let $\mathcal{S} = \bigcup_{h \in H} S_h$ be a 2-separated collection in G and let T_h,$h \in H$ be subsets of V with $S_h \subset T_h$ for all $h \in H$. If there exists an $\epsilon > 0$ such that*

$$|D(T_h)| \leq (1+\epsilon) \cdot |D(S_h)|$$

holds for all $h \in H$ and if $\bigcup_{h \in H} D(T_h)$ forms a dominating set in G, the set $\bigcup_{h \in H} D(T_h)$ is a $(1+\epsilon)$-approximation of a minimum dominating set in G.

Proof. This is proved as Corollary 1 in [15]. □

2.2. **Tiling of the plane.** The plane is split into hexagons and each hexagon gets a class number. The tiling has the following properties:

- Each vertex is in exactly one hexagon.
- Every two vertices in a hexagon are connected by an edge.
- Each hexagon has a class number.
- The distance between two vertices in different hexagons with the same class number is at least a certain constant.
- The number of hexagonal classes is bounded by a constant.

We achieve these properties as follows: First we define the constant c to be the smallest even integer such that $(2c+1)^2 < \left(\sqrt{1+\epsilon}\right)^c$. We consider a tiling of the plane with tiles. Each tile consists of hexagons of diameter one that are being assigned different class numbers (see Figure 2 and Figure 3). Denote by H the set of all hexagons containing vertices of G (only these hexagons are relevant for us) and by b the number of hexagons in one tile. Ambiguities caused by vertices at the border of hexagons are resolved as shown in Figure 2 (b): The right borders excluding the upper and lower apexes belong to a hexagon, the rest of the border does not. We assume that the tiling starts with the coordinates (0,0) being in the center of a tile of class 1. We choose the number of hexagons per tile in such a way that two hexagons of the same class have an Euclidean distance of at least $2c+1$. Note that this implies that two vertices in different hexagons of the same class number are at least $2c+1$ hops away from each other. Later we will show that we need at most $12c^2+18c+7$ hexagons per tile to ensure this, i.e. $12c^2+18c+7 \leq b$. Let $class(h)$ be the class number of a hexagon h.

2.3. **The algorithm.** Before giving the formal algorithm we present the main concepts and provide an intuitive description of the algorithm. For some hexagons h we construct a set T_h that contains all vertices in h and the vertices in a certain surrounding area. These sets T_h are disjoint and have certain properties that ensure the desired approximation ratio of $1+\epsilon$. We call vertices contained in a set T_h *covered*. The construction of the sets T_h is done by iterating over the class numbers of the hexagons. First we cover hexagons of class 1 by computing sets T_h for all hexagons h of class 1. Assume that all hexagons of class i have already been covered. We proceed to cover all hexagons of class $i+1$ whose vertices have not been completely covered so far by computing sets T_h for those hexagons. We stop when all vertices in all hexagons have been covered. Moreover, the number of iterations does not exceed the total number of classes. Finally we compute for all sets T_h the minimum dominating set $D(T_h)$. We output $D := \bigcup_h D(T_h)$.

Now we present the algorithm in detail. Fix $\epsilon > 0$ and let b the number of hexagonal classes. For all $h \in H$ we initialize the set $T_h := \emptyset$. If all vertices of a hexagon have been covered, call this hexagon *covered*. For $i = 1, ..., b$ do the following: Consider a hexagon $h \in H$ of class i which is not covered. Define

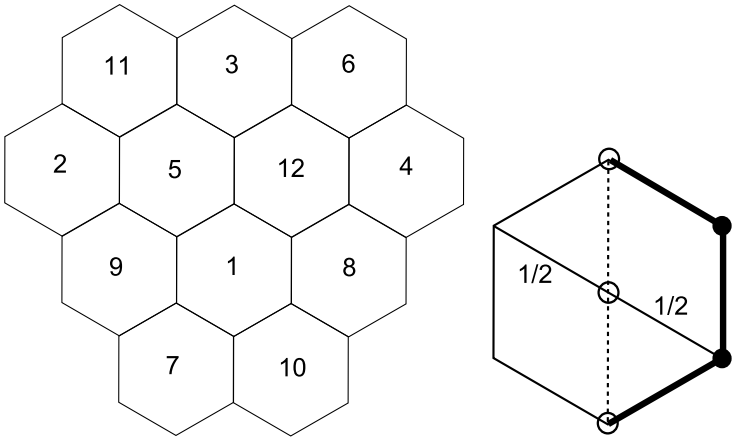

FIGURE 2. (a) A tile divided into 12 hexagons. Having 12 hexagons in one tile achieves a minimum Euclidean distance between to hexagons of the same class of 2. (b) One hexagon of the tiling. The bold lines indicate the parts of its border that belong to this hexagon

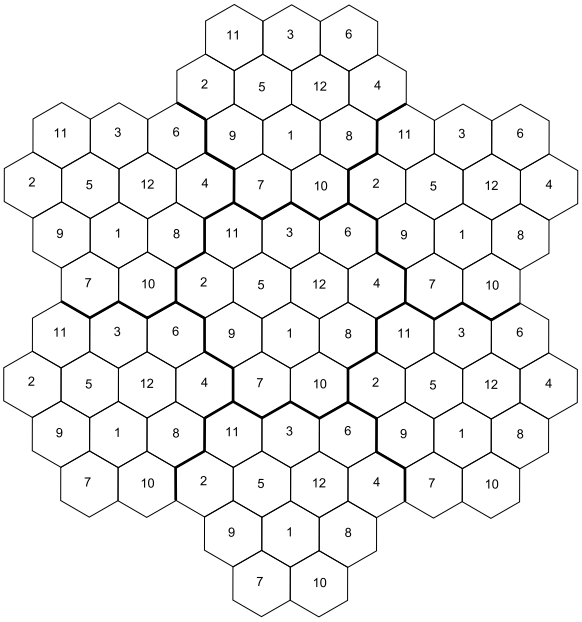

FIGURE 3. Several tiles glued together

the vertex v_h which is closest to the center of h and which is not covered yet to be the *coordinator vertex* of h. Ambiguities are resolved by choosing the vertex with the smallest x-coordinate among vertices with the least distance to the center of h. Denote by $C(i)$ all vertices which are covered in previous iterations where

hexagons of classes $i' < i$ were considered. Compute for even $r \in \{0, 2, 4, ..., c\}$ the r-neighborhoods $N^r(v_h)$ and compute the minimum dominating sets $D(N^r(v_h))$. We determine the smallest value of r with $r \leq c-2$ such that

$$\left|D\left(N^{r+2}(v_h) \setminus C(i)\right)\right| \leq (1+\epsilon) \cdot \left|D\left(N^r(v_h) \setminus C(i)\right)\right| \quad (1)$$

holds and denote it by $\bar{r}$. Later we will prove that there is at least one value for r with $r \leq c-2$ such that Inequality 1 does indeed hold (see Lemma 3). Now mark all vertices in $T_h := N^{\bar{r}+2}(v_h) \setminus C(i)$ as *covered*. In the sequel we will use the notation $S_h := N^{\bar{r}}(v_h) \setminus C(i)$ and we will prove in Lemma 5 that the sets S_h (for various hexagons h) form a 2-separated collection. We assign all vertices in $D(T_h)$ to the dominating set. We do this procedure for hexagons of class i which are not covered yet. As two vertices in different hexagons of the same class number are at least $2c+1$ hops away from each other the order in which the hexagons are processed does not matter. We output $D := \bigcup_{h \in H} D(T_h)$.

The previous discussion is presented in Algorithm 1.

Algorithm 1: Local algorithm for finding a dominating set in a unit disk graph

```
// Algorithm is executed independently by each node v;
dominator:=false;
Find all vertices in N(v);
forall v' ∈ N(v) do
    find the hexagon h' such that v' ∈ T_{h'};
    compute D(T_{h'});
    if v ∈ D(T_{h'}) then
        dominator:=true
    end
end
if dominator=true then Become part of the dominating set D else Do not become part of D
```

2.4. **Proof of correctness.** We prove the correctness of Algorithm 1, its approximation factor, its locality and its processing time in Theorem 1.

Theorem 1. *Let G be a unit disk graph and let $\epsilon > 0$. Algorithm 1 has the following properties:*

(1) *The computed set D is a dominating set for G.*
(2) *Let D_{OPT} be an optimal dominating set. It holds that $|D| \leq (1+\epsilon) \cdot |D_{OPT}|$.*
(3) *Whether or not a vertex v is in D depends only on the vertices at most $O\left(\frac{1}{\epsilon^6}\right)$ hops away from v, i.e. Algorithm 1 is local.*
(4) *The processing time for a vertex v is bounded by a polynomial in the number of vertices at most $O\left(\frac{1}{\epsilon^6}\right)$ hops away from v.*

We will prove the four parts of this theorem in four steps. In each step we first give some lemmas which are required to understand the proof of the theorem.

2.4.1. *Correctness.* We want to prove that the set D is indeed a dominating set for G. As mentioned above we first prove that it is sufficient to examine values for r with $r \leq c-2$ while computing the $D\left(N^r(v)\right)$, employing an argument used in [15].

Lemma 3. *Let v be a coordinator vertex. While computing its neighborhood $N^r(v)$ the values of r that need to be considered to find a value $\bar{r}$ such that*

$$\left|D\left(N^{\bar{r}+2}(v) \setminus C(i)\right)\right| \leq (1+\epsilon) \cdot \left|D\left(N^{\bar{r}}(v) \setminus C(i)\right)\right| \tag{2}$$

are bounded by $c-2$.

Proof. Assume on the contrary that Inequality 2 is false for all $r \in \{0, 2, ..., c-2\}$, i.e. for these values of r it holds that

$$\left|D\left(N^{r+2}(v) \setminus C(i)\right)\right| > (1+\epsilon) \cdot \left|D\left(N^{r}(v) \setminus C(i)\right)\right|$$

By Corollary 3 in [15] the number of vertices in a minimum dominating set for a neighborhood $N^c(v)$ is bounded by $(2c+1)^2$. It holds that $\left|D\left(N^0(v)\right)\right| = |D(\{v\})| = 1$. So we have that

$$\begin{aligned}
(2c+1)^2 &\geq \left|D\left(N^c(v) \setminus C(i)\right)\right| \\
&> (1+\epsilon) \cdot \left|D\left(N^{c-2}(v) \setminus C(i)\right)\right| \\
&> (1+\epsilon)^2 \cdot \left|D\left(N^{c-4}(v) \setminus C(i)\right)\right| \\
&> ... \\
&> (1+\epsilon)^{\frac{c}{2}} \cdot \left|D\left(N^0(v) \setminus C(i)\right)\right| \\
&\geq \left(\sqrt{1+\epsilon}\right)^c
\end{aligned}$$

But from the definition of c we know that $(2c+1)^2 < \left(\sqrt{1+\epsilon}\right)^c$ which is a contradiction. So at least for one value of $r \in \{0, 2, ..., c-2\}$ it holds that $\left|D\left(N^{r+2}(v)\right)\right| \leq (1+\epsilon) \cdot \left|D\left(N^r(v)\right)\right|$. □

Lemma 4. *The sets T_h cover all vertices of the graph.*

Proof. Assume on the contrary that there is a vertex v which is not covered by any $T_h, h \in H$. Let h be the hexagon to which v belongs and let i be its class number. At some point in the algorithm, the hexagons of class i were considered. Then there were vertices in h which were not covered yet (at least v). So the coordinator vertex of h must have marked a set T_h as covered. However, as the hexagons have a diameter of 1 (and $\bar{r}+2 \geq 1$) it follows that v is contained in T_h and therefore covered by T_h which is a contradiction. □

Proof. (of part 1 of Theorem 1): Let $v \in V$ be a vertex. By Lemma 12, v is covered by a set T_h. Then the set $D\left(T_h\right)$ dominates v. As Algorithm 1 outputs $D = \bigcup_h D\left(T_h\right)$, it holds that $D(T_h) \subseteq D$ and therefore D dominates v. □

2.4.2. *Approximation ratio.* We prove that the size of D is only a factor $1+\epsilon$ greater than the size of a minimum dominating set.

Lemma 5. *The subsets S_h, $h \in H$ form a 2-separated collection.*

Proof. The claim follows from Lemma 3 in [15]. □

Proof. (of part 2 of theorem 1): From the construction we can see that for every pair S_h, T_h it holds that $|D\left(T_h\right)| \leq (1+\epsilon) \cdot |D\left(S_h\right)|$. So the conditions of Lemma 2 are satisfied. □

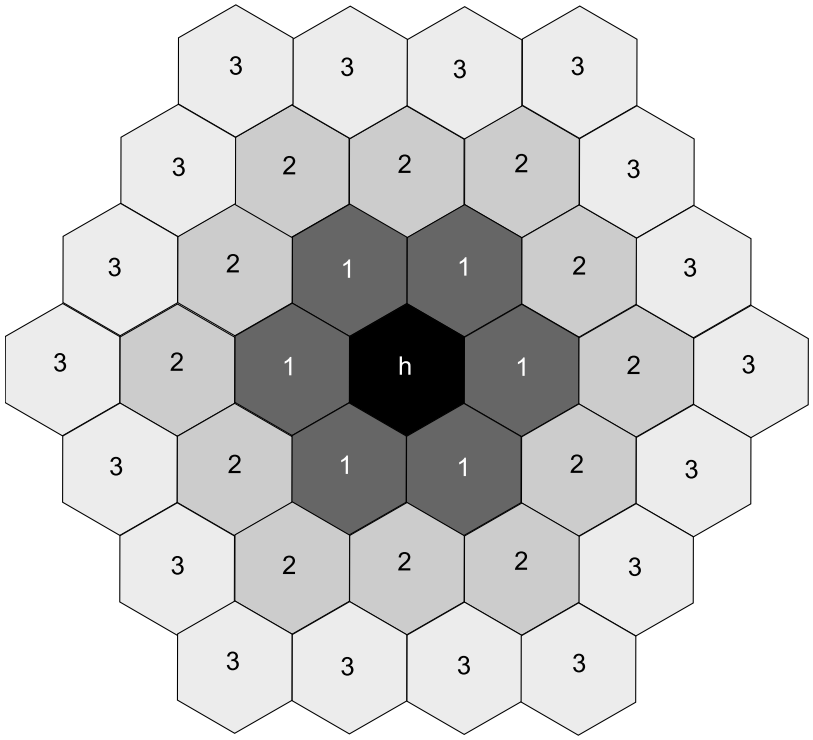

FIGURE 4. A hexagon h with three circles of hexagons around it. Hexagons with the same number belong to the same circle.

2.4.3. *Locality.* Now we want to prove that Algorithm 1 is local (part 3 of Theorem 1). We prove that whether or not a vertex v belongs to the computed set D depends only on the vertices at most a constant α hops away from v. This constant depends only on ϵ. We give an upper bound for α in terms of ϵ.

First we study our tiling of the plane and prove an upper bound for the number of hexagons per tile. Then we present two technical lemmas before we can prove part 3 of Theorem 1.

Lemma 6. *For ensuring a minimum Euclidean distance of d between two hexagons of the same class number, we need at most $3d^2 + 3d + 1$ hexagons per tile. So for a minimum distance of $2c + 1$ we need at most $12c^2 + 18c + 7$ hexagons per tile.*

Proof. We build a tile according to the following construction: Consider one hexagon h in the center and several circles of hexagons around it. A circle around a set of hexagons H is a set of hexagons H' placed around H with minimum cardinality such that no hexagon of H lies at the edge of the resulting set $H \cup H'$. Figure 4 shows a hexagon h with three circles of hexagons around it. Since the length of an edge of a hexagon is $1/2$, adding one circle of hexagons around such a tile increases the minimum distance between h and the border by at least $1/2$. So it is sufficient to add $\frac{d}{2} \cdot \frac{1}{1/2} = d$ circles of hexagons around h to ensure a minimum distance of $d/2$ between any point in h and the border of the tile.

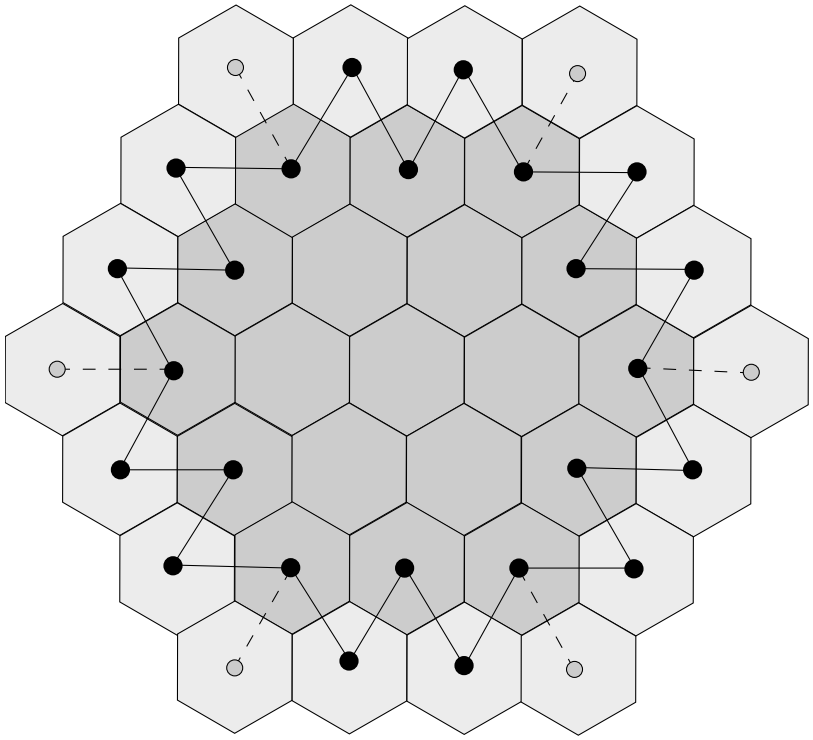

FIGURE 5. Adding a third circle around h

Claim: Our construction with d circles around h consists of $3d^2+3d+1$ hexagons, including h. We show this by proving that each new circle has exactly six hexagons more than the circle before. Assume a circle of hexagons C_1 has $|C_1|$ hexagons and another circle of hexagons C_2 is placed around it. We define a bipartite graph $G_B = (V_B, E_B)$ with one vertex for each hexagon in C_1 and C_2. Two vertices $v, v' \in V_B$ are connected by an edge if and only if their corresponding hexagons are adjacent to each other. Figure 5 shows this construction for adding the third circle around h. Then there will be exactly six vertices of degree one (they are the grey vertices in Figure 5 which have a dashed adjacent edge). Denote these vertices by V'_B. Then $G''_B = (V_B \setminus V_{B'}, E_B)$ will be one single cycle. This proves that $|C_2| = 6 + |C_1|$. So the number of hexagons will be $1 + \sum_{i=1}^{d} 6i = 3d^2 + 3d + 1$.

This implies that we need $3d^2+3d+1$ hexagons to achieve a minimum Euclidean distance of $d/2$ between h and the edge of a tile. We assign the class number of the hexagons in such a way that corresponding hexagons get the same class number in each tile. So in a tiling with such tiles there will be a minimum Euclidean distance of d between two hexagons of the same class as h. By symmetry this follows for the hexagons of the other class numbers as well.

Substituting $2c+1$ for d shows that $12c^2+18c+7$ hexagons per tile are sufficient to ensure a minimum Euclidean distance of $2c + 1$ between two hexagons of the same class number. □

Lemma 7. *We show three locality related properties of our algorithm:*

(1) *Let v_h be the coordinator vertex of a hexagon h of class k. What vertices are in T_h depends only on the vertices which are at most $2c \cdot (k-1) + c$ hops away from v_h.*
(2) *Let v' be any vertex. Whether v' is contained in a set $T_{h'}$ with $class(h') \leq k$ depends only on the vertices which are at most $2c \cdot k$ hops away from v'. If v' is contained in such a set $T_{h'}$ then what vertices are in $T_{h'}$ depends only on the vertices which are at most b_k hops away from v'.*
(3) *Let v'' be any vertex in a hexagon h'' of class k. Whether or not v'' is the coordinator vertex of h'' depends only on the vertices which are at most $1 + 2c \cdot (k-1)$ hops away from v''.*

Proof. For ease of notation we introduce the sequences a_k, b_k and c_k. Let a_k be the smallest integer such that what vertices are in T_h depends only on the vertices which are at most a_k hops away from v_h. So in order to prove property 1 we want to show that $a_k \leq 2c \cdot (k-1) + c$. Let b_k be the smallest integer such that whether v' is contained in a set $T_{h'}$ with $class(h') \leq k$ depends only on the vertices which are at most b_k hops away from v' and if v' is contained in such a set $T_{h'}$ then what vertices are in $T_{h'}$ depends only on the vertices which are at most b_k hops away from v'. For proving property 2 we need to show that $b_k \leq 2c \cdot k$. Let c_k be the smallest integer such that whether or not v'' is the coordinator vertex of h'' depends only on the vertices which are at most c_k hops away from v''. So for proving property 3 we need to show that $c_k \leq 1 + 2c \cdot (k-1)$.

Proof by induction. We begin with $k = 1$. As we need to explore the vertices at most c hops away from v_h in order to compute T_h, we conclude that $a_1 \leq c$.

Let v'' be a vertex in a class 1 hexagon h''. To find out whether v'' is the coordinator vertex of h'', we need to explore the vertices which are at most 1 hop away from v''. So $c_1 \leq 1$.

Let v' be a vertex. We want to find out whether there is a hexagon h' with $class(h') \leq 1$ such that v' is contained in the set $T_{h'}$. If yes, the coordinator vertex $v_{h'}$ of h' can be at most c hops away from v'. So we need to explore all vertices which are at most c hops away from v' to find all vertices in class 1 hexagons because only they could possibly be coordinator vertices for their hexagon h' such that $v' \in T_{h'}$. To find out if any of them is the coordinator vertex of their respective hexagon h' we need to explore the area $c_1 \leq 1$ hop around them. If one of them is a coordinator vertex, we need to explore the vertices at most $a_1 \leq c$ from it in order to compute $T_{h'}$ and to find out whether $v' \in T_{h'}$. If this is the case, we immediately know the set $T_{h'}$ as well. So we only need to explore the vertices which are at most $b_1 \leq c + \max(a_1, c_1) \leq 2c$ hops away from v' in order to compute this task.

Assume that the claims in the lemma hold for all $k \leq i-1$. Let v_h be the coordinator vertex of a hexagon h of class i. In order to compute T_h we need to explore the vertices which are at most c hops away from v_h and therefore need to find out for each vertex in $N^c(v_h)$ whether it has been covered by a set $T_{h'}$ with $class(h') < i$. So for computing T_h we need to explore the vertices which are $a_i \leq c + b_{i-1}$ hops away from v_h.

Let v'' be a vertex in a hexagon h'' of class i. To find out whether v'' is the coordinator vertex for h'' we need to explore all other vertices in h'' and find out if they have been covered by a set $T_{h'}$ with $class(h') < i$. For this we need to explore the vertices which are at most $c_i \leq 1 + b_{i-1}$ hops away from v''.

Let now v' be a vertex. We want to find out whether v' is covered by a set $T_{h'}$ with $class(h') \leq i$. So first we need to explore all vertices at most c hops away from v'. This is the set $N^c(v')$. Only vertices in this set can possibly be coordinator vertices for a hexagon h' such that $T_{h'}$ contains v'. To check if a vertex in $N^c(v')$ is a coordinator vertex, we need to explore all vertices which are at most c_i hops away from it. If a vertex $v_{h'}$ in $N^c(v')$ is the coordinator vertex for its hexagon h', we need to explore the vertices which are at most a_i hops away from $v_{h'}$ in order to compute $T_{h'}$. Then we can check if $v' \in T_{h'}$. If this is the case, we immediately know the set $T_{h'}$ as well. This gives us $b_i \leq c + \max(a_i, c_i) \leq c + \max(c + b_{i-1}, 1 + b_{i-1}) \leq c + a_i$.

So we have shown that $a_1 \leq c$, $b_1 \leq 2c$, $c_1 \leq 1$, $c_i \leq 1 + b_{i-1}$, $b_i \leq c + a_i$ and $a_i \leq c + b_{i-1}$. This implies $a_i \leq c + b_{i-1} \leq c + c + a_{i-1} \Rightarrow a_i \leq 2c \cdot (i-1) + c$, $b_i \leq 2c \cdot i$ and $c_i \leq 1 + 2c \cdot (i-1)$. □

Lemma 8. *Let v be vertex. Whether or not v is in D depends only on the vertices which are at most $2 + 2c \cdot (b-1) + c$ hops away from v.*

Proof. Before we prove the lemma we show the following claim: Let v' be a vertex in a hexagon h with $class(h) = k$ and let v' be contained in a set $T_{h'}$. We claim that what vertices (other than v') are in $T_{h'}$ depends only on the vertices which are at most $1 + 2c \cdot (k-1) + c$ hops away from v'.

Proof of the claim: We use the notation a_k, b_k and c_k as introduced in the proof of Lemma 7. For computing the set $T_{h'}$ we need to check whether v' is covered by a set $T_{h'}$ with $class(h') < class(h)$. This depends only on the vertices at most $b_{k-1} \leq 2c \cdot (k-1)$ hops away from v' (see Lemma 7). If there is such a set $T_{h'}$ with $v' \in T_{h'}$ then $T_{h'}$ depends only on the vertices which are at most b_{k-1} hops away from v' as well (see Lemma 7).

If there is no set $T_{h'}$ such that $class(h') < class(h)$ and $v' \in T_{h'}$ the algorithm has to find out whether there is another vertex $v'' \neq v'$ in h that is the coordinator vertex for h. In order to do this, we need to explore the vertices at most $c_k \leq 1 + 2c \cdot (k-1)$ hops away from v'. If there is such a vertex v'' then we need to explore the vertices which are at most $1 + a_k \leq 1 + 2c \cdot (k-1) + c$ hops away from v' in order to compute the set $T_{h'}$. If not, then v' is the coordinator vertex for h. Then we need to explore the vertices which are at most $a_k \leq 2c \cdot (k-1) + c$ hops away from v' in order to compute $T_{h'}$. Altogether we conclude that for computing $T_{h'}$ we need to know only about the vertices at most $1 + 2c \cdot (k-1) + c$ hops away from v'. This proves the claim.

Now we prove the lemma. When a vertex v computes whether it is in D it does the following: For each $v' \in N(v)$ it computes the set T_h that covers v'. Then v is part of the dominating set if and only if it is part of any of the $D(T_h)$. So we only need to explore the the vertices which are at most $2 + 2c \cdot (k-1) + c$ hops away from v. □

Proof. (of part 3 of Theorem 1): We want to show that whether or not a vertex v is in D depends only on the vertices at most $O\left(\frac{1}{\epsilon^6}\right)$ away from v.

First we want to show that there is an ϵ_0 such that for all $\epsilon < \epsilon_0$ it holds that $c \leq \frac{1}{\epsilon^2} + 2$ i.e. $c \in O\left(\frac{1}{\epsilon^2}\right)$. By definition c is the smallest even integer such that $(2c+1)^2 < \left(\sqrt{1+\epsilon}\right)^c$. We calculate that

$$\begin{aligned} (2c+1)^2 &< \left(\sqrt{1+\epsilon}\right)^c && (3)\\ \Leftrightarrow 4\ln(2c+1) &< c\cdot\ln(1+\epsilon) && (4)\end{aligned}$$

From taking derivatives we get that

$$\ln(1+\epsilon) \geq \frac{1}{2}\epsilon \tag{5}$$

holds for all $\epsilon \leq 1$. From Inequalities 5 and 4 we conclude that it is sufficient to show that there is an ϵ_0 such that

$$4\ln(2c+1) < \frac{1}{2}\epsilon\cdot c$$

holds for all for $\epsilon < \epsilon_0 \leq 1$. We set $c := \frac{1}{\epsilon^2}$ and get

$$\begin{aligned} 4\ln\left(\frac{2}{\epsilon^2}+1\right) &< \frac{1}{2}\cdot\frac{1}{\epsilon}\\ \Leftrightarrow 8\ln\left(\frac{2}{\epsilon^2}+1\right) &< \frac{1}{\epsilon}\end{aligned}$$

Since $\ln(2n^2+1) \in o(n)$ there is an $\epsilon_0 \leq 1$ such that the above holds for all $\epsilon \leq \epsilon_0$. So for $\epsilon \leq \epsilon_0$ we can find a value for c with $c < \frac{1}{\epsilon^2}+2$ (remember that c has to be an even integer). We observe that $\left(2\frac{1}{\epsilon_0^2}+1\right)^2 < \left(\sqrt{1+\epsilon_0}\right)^{\frac{1}{\epsilon_0^2}} < \left(\sqrt{1+\epsilon}\right)^{\frac{1}{\epsilon_0^2}}$ for $\epsilon > \epsilon_0$. So for all these values of ϵ we can find a value for c such that $c < \frac{1}{\epsilon_0^2}+2$. This proves $c \in O\left(\frac{1}{\epsilon^2}\right)$.

Denote by $\alpha(\epsilon)$ the locality distance of Algorithm 1 when run with a performance guarantee of $1+\epsilon$. From Lemma 8 we know that we need to explore the vertices at most $2+2c\cdot(b-1)+c$ hops away from v. From the definition of b and the above lemmas we get

$$\begin{aligned} \alpha(\epsilon) &\leq 2+2c\cdot(b-1)+c\\ &\leq 2+2c\cdot(12c^2+18c+7-1)+c\\ &= 24c^3+36c^2+13c+2\\ &\in O\left(\frac{1}{\epsilon^6}\right)\end{aligned}$$

□

Table 1 displays trade-offs between approximation ratios and locality distances which are attained by our algorithm. It shows that for a 5 approximation we need to explore the vertices which are at most 10,089 hops away from a given vertex v. In comparison, the local 5 approximation algorithm presented in [6] only needs the information of the vertices which are at most 11 hops away from a given vertex.

2.4.4. *Processing time.* The processing time is the time that a single vertex needs in order to compute whether or not it is part of the dominating set. As our algorithm is local it is not very suitable to quantify the processing time in comparison to the total number of vertices in G. Instead we measure it with respect to the number of vertices which are at most α hops away from a vertex v since these are all vertices

Dominating Set		
Approximation Ratio	Upper Bound for Locality Distance of	
	Algorithm 1	Algorithm in [6]
1.5	8,056,511	-
2	273,264	-
3	46,814	-
4	20,531	-
5	10,089	11

TABLE 1. Locality distances in hops of Dominating Set algorithms for several approximation factors. The second column shows the upper bounds of the locality distance of our algorithm. The third column shows an upper bound for the locality of the local 5 approximation algorithm presented in [6].

that a vertex v needs to explore when computing its status. We denote this number by $n_\alpha(v)$ (i.e. $n_\alpha(v) = |N^\alpha(v)|$). We show that the processing time is bounded by a polynomial in $n_\alpha(v)$.

Proof. (of part 4 of Theorem 1). In the algorithm, minimum dominating sets for the sets $N^r(v')$ with $r \in \{0, 1, ..., c\}$ and $v' \in N^\alpha(v)$ must be computed. First we show that this can be done in polynomial time. By Corollary 3 in [15] the number of vertices in a minimum dominating set for a neighborhood $N^r(v')$ is bounded by $(2r+1)^2$. So the computation of such a set can be done in $O\left(n_\alpha(v)^{(2c+1)^2}\right)$, e.g. by enumeration. For each vertex $v' \in N^\alpha(v)$ we might have to compute minimum dominating sets $D(N^r(v'))$ for each $r \in \{0, 2, ..., c\}$. As this dominates the processing time of the algorithm, we find that it is in $O\left(n_\alpha(v)^{(2c+1)^2} \cdot n_\alpha(v) \cdot c\right)$ and therefore bounded by $n_\alpha(v)^{O\left(1/\epsilon^4\right)}$. □

3. Local $3+\epsilon$ Approximation for Connected Dominating Set

In this section we present a local $3+\epsilon$ approximation algorithm for connected dominating set in unit disk graphs, prove its correctness and give an upper bound for its locality distance. We prove that its locality distance depends only on ϵ.

We use our local $1+\epsilon$ algorithm for dominating set as stated in Section 2 as a subroutine, compute a dominating set with a certain size guarantee and add vertices (we refer to them as bridges) in order to turn it into a connected dominating set. For the latter we use the same technique as used in [6] but employ a slightly different analysis. This leads to a $3+\epsilon$ approximation for connected dominating set.

3.1. **Algorithm.** Before we go into details we give an overview of the algorithm. Let $G = (V, E)$ be a connected unit disk graph and let $3+\epsilon$ the desired approximation factor. First we define a certain constant $\bar{\epsilon}$ and compute a $1+\bar{\epsilon}$ approximation for minimum dominating set using the local algorithm stated in Section 2. The computed set D dominates the graph but might have several connected components. We consider a similar tiling of the plane as used in Section 2 and choose one vertex of each hexagon h to be the head vertex. As long as D is not connected there is a head vertex $\bar{v}_h$ such that the restriction of D to a certain neighborhood of $\bar{v}_h$

will be disconnected. We add vertices to D until its restriction to the neighborhood of each head vertex is connected. This leads to a $3+\epsilon$ approximation of minimum connected dominating set.

Now we present the algorithm in detail. We fix an integer d to be the smallest integer such that $d > \frac{6}{\epsilon} + 3$ and define $\bar{\epsilon} := \frac{(d-3)}{(d-1)} \cdot \frac{(3+\epsilon)}{3} - 1$ (note that $\bar{\epsilon} > 0 \Leftrightarrow d > \frac{6}{\epsilon} + 3$). First we locally

compute a dominating set with size at most $(1+\bar{\epsilon}) \cdot |D_{OPT}|$ (with D_{OPT} being an optimal dominating set for G) by using the local algorithm stated in Section 2. Denote the computed dominating set by D. Now consider a tiling of the plane that ensures a minimum Euclidean distance of $d+2$ between two different hexagons of the same class (this implies a minimum hop-distance of $d+2$ between two vertices in different hexagons of the same class). Denote by b the number of different hexagonal classes needed for such a tiling. We will prove that $b \leq 3d^2 + 15d + 19$. Denote by H the set of all hexagons that contain vertices of G (only these hexagons are relevant for us). In each hexagon h we declare the vertex that is closest to the center of h to be the *head vertex* $\bar{v}_h$ (note that this is not the same definition as for the coordinator vertex for a hexagon in Section 2). Ambiguities here are resolved by choosing the vertex with the lowest x-coordinate among the vertices with the same distance to the center. Our algorithm runs in b rounds, one for each class number. Denote by D_i the computed set after the ith round and define $D_0 := D$. In each round i, $1 \leq i \leq b$ the head vertex $\bar{v}_h$ of each class i hexagon h computes $N := D_{i-1} \cap N^d(\bar{v}_h)$. If there are two connected components in N which could be connected by one vertex $v \in h$ we assign this vertex v to the computed set. If there are two connected components in N which could be connected by two adjacent vertices $v \in h$ and $v' \in h'$ with $h' \in H$ we assign these vertices v and v' to the computed set (note that v' is not necessarily in h). Perform both operations until they are no longer applicable. As two vertices in different hexagons with the same class number are at least $d+2$ hops away from each other the order in which the hexagons of one class number are considered does not matter. This will result in a set $CD := D_b$. We output CD. The previous discussion is presented in Algorithm 2.

Algorithm 2: Algorithm for finding a connected dominating set in a unit disk graph $G = (V, E)$

```
// The algorithm is executed independently in each vertex v
dominator:=false;
Compute whether v is in D using Algorithm 1;
if v ∈ D then dominator:=true;
Find out which vertices in N^2(v) are head vertices;
for i:=1 to b do
    if there is a head vertex of class i in N^2(v) that assigns v to D_i then
        dominator:=true
    end
end
if dominator=true then Become part of the connected dominating set CD
else Do not become part of CD
```

This is essentially the same algorithm as Algorithm 2 in [6][1].

3.2. **Proof of correctness.** We prove the correctness of Algorithm 2, its approximation factor, its locality and its processing time in Theorem 2.

Theorem 2. *Let G be a unit disk graph and let $\epsilon > 0$. Algorithm 2 has the following properties:*

1. *The computed set CD is a connected dominating set for G.*
2. *Let CD_{OPT} be an optimal dominating set. It holds that $|CD| \leq (3+\epsilon) \cdot |CD_{OPT}|$.*
3. *Whether or not a vertex v is in CD depends only on the vertices at most $O\left(\frac{1}{\epsilon^6}\right)$ hops away from v, i.e. Algorithm 2 is local.*
4. *The processing time for a vertex $v \in V$ is bounded by a polynomial in the number of vertices at most $O\left(\frac{1}{\epsilon^6}\right)$ hops away from v.*

We will prove the four parts of this theorem in four steps.

3.2.1. *Correctness.* We want to prove that the computed set CD is a connected dominating set for G. For the proof we need a technical lemma about dominating sets.

Lemma 9. *Let D be a dominating set for G. Then there is a family of sets $\mathcal{B} = \{B_1, B_2, ..., B_m\}$ such that for all $B_i \in \mathcal{B}$ it holds that $1 \leq |B_i| \leq 2$ and $D \cup B_i$ has at least one connected component less than D and that $D \cup \bigcup_{i=1}^{m} B_i$ is connected.*

Proof. Define a graph $G_B = (V_B, E_B)$ with

$$\begin{aligned} V_B &:= \{C \subseteq V | C \text{ is a connected component in } D\} \\ E_B &:= \{E \subseteq V | |E| \leq 2 \text{ and } E \text{ connects two connected components in } D\} \end{aligned}$$

Since G is connected and D is a dominating set for G, it follows that G_B is connected and therefore E_B defines the set $\mathcal{B}$.

□

Proof. (of part 1 of Theorem 2): For the correctness of the first part of the algorithm (computation of D) see Theorem 1. Since by construction $D \subseteq CD$ and D is a dominating set, it remains to show that CD is connected.

Assume on the contrary that there are at least two connected components in CD. As D is a dominating set, there is a set $\mathcal{B} = \{B_1, B_2, ..., B_m\}$ such that for all $B_i \in \mathcal{B}$ it holds that $1 \leq |B_i| \leq 2$ and the union $D \cup B_i$ has one connected component less than D and that $D \cup \bigcup_{i=1}^{m} B_i$ is connected (see Lemma 9).

In other words, each B_i connects two connected components in D. As CD is not connected and we never delete a vertex from the computed set, it follows that D is not connected and for the set $\mathcal{B}$ stated above above it holds that $\mathcal{B} \neq \emptyset$. As CD is not connected there must be at least one such set $B_i \in \mathcal{B}$ such that $B_i \nsubseteq CD$. Suppose $|B_i| = 2$, let $B_i = \{u, v\}$ and w.l.o.g. let $v \in B_i \setminus CD$. Assume v is in the hexagon h of class c with the coordinator vertex v_h. But then in round c the algorithm would have found two connected components in $N^d(v_h) \cap D_{c-1}$ that could be connected by adding u and v. This is a contradiction. The case where $|B_i| = 1$ is proved similarly. □

[1]However, the algorithm that is presented in [6] has two phases, one in which only single-vertex bridges are added and one where only two-vertex bridges are added. In order to get a smaller locality distance for our algorithm we only have one phase. This might in practice result in bigger connected dominating sets, but does not affect the performance guarantee that we can give.

3.2.2. *Approximation ratio.* We want to show that Algorithm 2 achieves an approximation ratio of $3+\epsilon$. A proof using similar techniques was given in [6].

Proof. (of part 2 of Theorem 2) We can view CD as obtained from D by sequentially adding vertices (or pairs of vertices):

(1) the vertices are added in order according to the class number of the hexagon whose head vertex assigned them to CD,
(2) the vertices assigned to CD by the same head vertex are added in the order they were considered by the algorithm, and
(3) the order of the vertices added by different head vertices in hexagons of the same class number does not matter

Consequently, we have a sequence $D = S_0, S_1, ..., S_r = CD$. For each S_i we can define VG_i as the graph obtained by applying the Local Minimal Spanning Tree construction of Chávez et al. [3] to the UDG of S_i with choosing d as the locality distance for the Local Minimal Spanning Tree (this is denoted by k in [3]). This construction ensures that each spanner VG_i is planar. Since different hexagons of the same class have an Euclidean distance of at least d and each head vertex v_h explores the vertices which are at most d hops away from it and connects connected components in $D_i \cap N^d(v_h)$, it follows that each face of VG_i is of size at least $2d$ (we denote by the size of a face the number of adjacent edges to it where edges in the interior of a face are counted twice). We will denote by VG the graph obtained by considering the final S_r.

We say that an addition of a node (or a pair of nodes) to obtain S_i from S_{i-1} is *idle*, if the number of connected components in VG_i is equal to the number of components in VG_{i-1}. Let us denote by V_0 the size of D. Let NI_1 and NI_2 be the numbers of additions of single nodes and pairs of nodes, respectively, which are not idle. Similarly, let I_1 and I_2 be the number of idle single and double-node additions, respectively. Let V denote the number of nodes of VG (i.e. $V = |CD|$), E denote the number of edges and F the number of faces. From Euler's formula we have that

$$F + V = E + 2 \tag{6}$$

Therefore, from the way CD was constructed, we have

$$V = V_0 + NI_1 + I_1 + 2(NI_2 + I_2) \tag{7}$$

Note that our algorithm adds a vertex (or pair of vertices) only when they connect at least two connected components (when looking up to a distance d). Therefore, only two or three edges are added to VG_{i-1} in order to get VG_i (depending on whether a single vertex bridge or a two-vertex bridge has been added). Combining this observation with the definition of idle additions, we have

$$E \geq V - 1 + I_1 + I_2 \tag{8}$$

As initially there were at most V_0 components and at the end VG is connected, from the definition of NI_1 and NI_2 we have

$$NI_1 + NI_2 \leq V_0 - 1$$

As each face is of size at least $2d$ and the sum of face sizes is exactly $2E$ we get

$$dF \leq E \tag{9}$$

Substituting for F into Euler's formula 6 according to the above Inequality 9 we get $E/d + V \geq E + 2$ and therefore $V - 2 \geq E(1 - 1/d)$. After substituting for E according to Inequality 8 and using some calculus we get

$$\frac{V}{d-1} - 1 \geq I_1 + I_2$$

Altogether we get

$$\begin{aligned} V &\leq V_0 + 2(NI_1 + NI_2) + 2(I_1 + I_2) \\ &\leq V_0 + 2(V_0 - 1) + 2\left(\frac{V}{d-1} - 1\right) \end{aligned}$$

Denote by D_{OPT} an optimal dominating set and by CD_{OPT} an optimal connected dominating set for G. As $V_0 \leq (1 + \bar{\epsilon}) \cdot |D_{OPT}| \leq (1 + \bar{\epsilon}) \cdot |CD_{OPT}|$ and $\bar{\epsilon} := \frac{(d-3)}{(d-1)} \cdot \frac{(3+\epsilon)}{3} - 1$ we get

$$\begin{aligned} V &\leq \frac{(d-1)}{(d-3)}(3V_0 - 4) \\ &\leq \frac{(d-1)}{(d-3)} 3\left((1 + \bar{\epsilon})\,|CD_{OPT}|\right) \\ &= \frac{(d-1)}{(d-3)}(3 + 3\bar{\epsilon})|CD_{OPT}| \\ &= (3 + \epsilon) \cdot |CD_{OPT}| \end{aligned}$$

□

3.2.3. *Locality.* We denote by β the locality distance of Algorithm 2. We give an upper bound for β and prove that it depends only on the desired approximation ratio $1 + \epsilon$.

Lemma 10. *Let v be a vertex. The assignment of v to the computed set by the head vertex of a class k hexagon depends only on v testing whether or not the vertices at most $d_k := k(2 + d)$ hops away from it are in D.*

Proof. Proof by induction. Let $k = 1$. Whether v was added by the head vertex v_h of a class 1 hexagon h depends only on the vertices at most d from v_h. The distance between v and v_h is a most 2. This gives us $d_1 = 2 + d$.

Assume the claim is true for all $k \leq i-1$. Whether v was added by a head vertex v_h of a class i hexagon h depends on the vertices at most d from v_h and whether they were added by a head vertex of a class $i - 1$ hexagon. As v_h is at most 2 from v we get $d_i = d_{i-1} + 2 + d$.

Altogether we get $d_k = k(2 + d)$. □

Proof. (of part 3 of Theorem 2): We denote by $\beta(\epsilon)$ the locality distance of Algorithm 2 when run with a performance guarantee of $1 + \epsilon$ and by $\alpha(\bar{\epsilon})$ the locality distance of Algorithm 1 when it is run with a performance guarantee of $1 + \bar{\epsilon}$ (see Section 2). As d is the smallest integer such that $d > \frac{6}{\epsilon} + 3$ we get that $d \leq \frac{6}{\epsilon} + 4$. The constant b defines the numbers of different class numbers which are needed in order to ensure a minimum Euclidean distance of $d+2$ between two hexagons of the same class. From Lemma 6 we know that $b \leq 3(d+2)^2+3(d+2)+1 = 3d^2+15d+19$.

Now we want to show that $\frac{1}{\bar{\epsilon}} \in O\left(\frac{1}{\epsilon}\right)$. We observe that $\frac{6}{\epsilon} + 3 < d < \frac{6}{\epsilon} + 4$. So it follows that

$$\begin{aligned}
\bar{\epsilon} &= \frac{(d-3)}{(d-1)} \cdot \frac{(3+\epsilon)}{3} - 1 \\
&\geq \frac{(\frac{6}{\epsilon}+3-3)}{(\frac{6}{\epsilon}+4-1)} \cdot \frac{(3+\epsilon)}{3} - 1 \\
&\geq \frac{\frac{6}{\epsilon}}{(\frac{6}{\epsilon}+3)} \cdot \frac{(3+\epsilon)}{3} - 1 \\
&= \frac{27\epsilon}{18+9\epsilon} \\
&\geq \frac{3}{2}\epsilon \\
\Rightarrow \frac{1}{\bar{\epsilon}} &\in O\left(\frac{1}{\epsilon}\right)
\end{aligned}$$

Using Lemma 10 we conclude that it is sufficient to explore the vertices that are at most $b \cdot (2+d) + \alpha(\bar{\epsilon})$ from a given vertex v. This gives an upper bound for $\beta(\epsilon)$. We calculate

$$\begin{aligned}
\beta(\epsilon) &\leq b \cdot (2+d) + \alpha(\bar{\epsilon}) \\
&\leq (3(d+2)^2 + 3(d+2) + 1) \cdot (2+d) + \alpha(\frac{(d-3)}{(d-1)} \cdot \frac{(3+\epsilon)}{3} - 1) \\
&\leq (3(\frac{6}{\epsilon}+4+2)^2 + (\frac{6}{\epsilon}+4+2) + 1) \cdot (2+\frac{6}{\epsilon}+4) + \alpha(\frac{(d-3)}{(d-1)} \cdot \frac{(3+\epsilon)}{3} - 1) \\
&\in O\left(\frac{1}{\epsilon^6}\right)
\end{aligned}$$

□

Table 2 in Section 4 displays trade-offs between approximation ratios and locality distances which are attained by our algorithm and the algorithm presented in [6].

3.2.4. *Processing time.* Recall that β is the locality distance of Algorithm 2. We want to prove that the time Algorithm 2 needs to compute whether a vertex v is in CD is bounded by a polynomial in the number of vertices which are at most β hops away from v. We denote this number by $n_\beta(v)$ (i.e. $n_\beta(v) = \left|N^\beta(v)\right|$). The vertices in $N^\beta(v)$ are all vertices that v explores when computing whether or not it is part of CD.

Proof. For some vertices in $N^\beta(v)$ we need to compute whether they are in D. This can be done in $n_\beta(v)^{O\left(1/\bar{\epsilon}^4\right)}$ processing time. For each class $i \in \{1, ..., b\}$ we need to compute the d-neighborhood of head vertices of class i and find out whether adding one or two vertices with one of them being in a hexagon of class i could connect two connected components. This can be done in $O(n_\beta(v))$. As $b \in O\left(d^2\right) \subseteq O\left(\frac{1}{\epsilon^2}\right)$ and $\frac{1}{\bar{\epsilon}} \in O\left(\frac{1}{\epsilon}\right)$ (see the proof of part 3 of Theorem 2) our processing time is bounded by $n_\beta(v)^{O\left(1/\epsilon^4\right)}$.

□

4. Local $1+\epsilon$ approximation for connected dominating set

In Section 3 we presented a local $3+\epsilon$ approximation algorithm for Connected Dominating Set. In this section we will generalize its methodology and design a local $1+\epsilon$ approximation algorithm for connected dominating set.

4.1. The algorithm.

Let $G=(V,E)$ be a connected unit disk graph. First we recapitulate Algorithm 2 and explain how we will generalize it. Then we present our $1+\epsilon$ approximation algorithm in detail.

4.1.1. *Main concept.* Let us first recall the main steps in Algorithm 2. We first computed a dominating set D for G with a certain performance ratio (compared to an optimal dominating set) using Algorithm 1. We used a 2-separated collection for establishing a lower bound for the performance ratio of D. Each connected component in D had obviously at least one vertex. We connected these components by adding bridges of vertices to the computed set.

Before we outline the new algorithm we introduce the concept of a $2k$-separated collection which generalizes the concept of a 2-separated collection as presented in Section 2.

Definition 4. Let H be an index set and let sets S_h with $h \in H$ be subsets of V. The sets S_h are called a *$2k$-separated collection* if for any two vertices $s \in S_h$ and $s' \in S_{h'}$ with $h \neq h'$ it holds $d(s,s') > 2k$.

So for $k=1$ a $2k$-separated collection is the same as a 2-separated collection as defined in Section 2.

In the new algorithm we first compute a dominating set D for G with some properties that achieves a certain performance ratio compared to an optimal *connected* dominating set. For establishing a lower bound for the performance ratio of D we will define sets which will form a $2k$-separated collection. Similarly to the computation of D in Algorithm 1 we will compute disjoint set T_h for some hexagons h such that each vertex is contained in exactly one set T_h. In contrast to Algorithm 1 we will ensure that each connected component in D has at least k vertices (we will define the constant k later). Analogously to Algorithm 2 we will connect these components by adding bridges of vertices.

4.1.2. *Description of the algorithm.* Now we present the algorithm in detail. Let $1+\epsilon$ be the desired approximation factor. We fix d to be the smallest integer such that $d \leq \sqrt[3]{1+\epsilon}$ and fix k to be the smallest integer such that $(1+\frac{2}{k}) \leq \sqrt[3]{1+\epsilon}$. We define $\bar{\epsilon} := \sqrt[3]{1+\epsilon} - 1$ and c to be the smallest integer such that $(2c+1)^2 < \left(\sqrt[k]{1+\bar{\epsilon}}\right)^{c/2}$ with $c \equiv 0 \bmod 2k$. We tile the plane with tiles as described in Section 2 such that two hexagons of the same class have a minimum Euclidean distance of $2c+1$. Denote by b the number of different class numbers. As proven in Lemma 6 the constant b is bounded by $12c^2+18c+7$. For all $h \in H$ we initialize the set $T_h := \emptyset$.

First we check whether $diam(G) < k+2$. If this is the case we explore the entire graph G, compute a minimum connected dominating set for G and stop. From now on assume that $diam(G) \geq k+2$ (later we will exploit the fact that this implies that a minimum connected dominating set for G has at least k vertices).

With the definitions of the constants as above (including the definition of k) we start a loop with one iteration for each class number. For $i = 1, ..., b$ do the

following: Denote by $C(i)$ all vertices that are contained in sets T_h with $class(h) < i$ and call these vertices *covered.* If all vertices in a hexagon have been covered, call this hexagon *covered*. Consider a hexagon h of class i that is not covered yet. Define the vertex v_h which is closest to the center of h and not covered to be the *coordinator vertex* of h. Ambiguities are resolved by choosing the vertex with the smallest x-coordinate among vertices with the least distance to the center of h. For a set $V' \subseteq V$ denote by $D_k(V')$ a dominating set for V' in $N^k(V')$ of minimum cardinality such that each connected component in $D_k(V')$ has at least k vertices (note that $D_1(V') = D(V')$). Consider for all integers r with $r \equiv 0 \bmod 2k$ and $0 \leq r \leq c$ the r-neighborhoods $N^r(v_h)$ and compute the sets $D_k\left(N^r(v_h)\right)$. Find a value for r with $r \leq c - 2k$ such that

$$\left|D_k\left(N^{r+2k}(v_h) \setminus C(i)\right)\right| \leq (1+\bar{\epsilon}) \cdot \left|D_k\left(N^r(v_h) \setminus C(i)\right)\right| \tag{10}$$

Denote by $\bar{r}$ the smallest value for r such that Inequality 10 holds. We will prove later in Lemma 11 that there is always a value for r that satisfies this. Mark all vertices in $T_h := N^{r+2k}(v_h) \setminus C(i)$ as covered and define $S_h := N^r(v_h) \setminus C(i)$. In the analysis we will prove that the sets S_h form a $2k$-separated collection for G (see Lemma 13). Do this for all hexagons of class i that are not covered yet. As two vertices in different hexagons of the same class number are at least $2c+1$ hops away from each other the order in which the hexagons are processed does not matter. We define $D := \bigcup_h D_k(T_h)$.

Then we connect all connected components in D using the local algorithm for connecting a dominating set as it is given in Section 3. We employ it with the constant d defined as above. Note that in the original algorithm in Section 3 it was defined differently. Also note that for this subroutine another hexagonal tiling of the plane will be employed, namely a tiling in which two hexagons of the same class have an Euclidean distance of at least $d+2$. Denote the resulting set by CD. We will prove that $|CD| \leq (1+\epsilon)\,|CD_{OPT}|$ with CD_{OPT} being an optimal connected dominating set.

The previous discussion is presented in Algorithm 3.

4.2. **Proof of correctness.** We prove the correctness of Algorithm 3, its approximation factor, its locality and the processing time in Theorem 3.

Theorem 3. *Let $G = (V, E)$ be a unit disk graph and let $\epsilon > 0$. Algorithm 3 has the following properties:*

1. *The computed set CD is a connected dominating set for G.*
2. *Let CD_{OPT} be an optimal dominating set. It holds that $|CD| \leq (1+\epsilon)|CD_{OPT}|$.*
3. *Whether or not a vertex v is in CD depends only on the vertices which are at most $O\left(\frac{1}{\epsilon^9}\right)$ hops away from v, i.e. Algorithm 3 is local.*
4. *The processing time for a vertex v is bounded by a polynomial in the number of vertices which are at most $O\left(\frac{1}{\epsilon^9}\right)$ hops away from v.*

We will prove the four parts of this theorem in four steps. In each step we first give some lemmas which are required to understand the proof of the theorem.

4.2.1. *Correctness.* We want to prove that the computed set CD is indeed a connected dominating set for G. As mentioned above we first prove that it is sufficient to examine values for r with $r \leq c - 2k$ while computing the sets $D_k\left(N^r(v)\right)$. This is very similar to Lemma 3.

Algorithm 3: Algorithm for finding a connected dominating set in a unit disk graph $G = (V, E)$

```
// Algorithm is executed independently by each node v;
if diam(G) < k + 2 then
    Explore entire graph G;
    Compute minimum connected dominating set directly;
    Stop;
end
// From here on we assume diam(G) ≥ k + 2
dominator:=false;
Find all vertices in N^k(v);
forall v' ∈ N^k(v) do
    find the hexagon h' such that v' ∈ T_{h'};
    compute D_k(T_{h'});
    if v ∈ D_k(T_{h'}) then dominator:=true
end
Find out which vertices in N^2(v) are head vertices;
for i:=1 to b do
    if there is a head vertex of class i in N^2(v) that assigns v to D_i then
        dominator:=true
    end
end
if dominator=true then Become part of the dominating set CD else Do not
become part of CD
```

Lemma 11. *Let v be a coordinator vertex of a class i hexagon. While computing its neighborhood $N^r(v)$ the values of r that need to be considered to find a value $\bar{r}$ such that*

$$\left|D_k\left(N^{\bar{r}+2k}(v) \setminus C(i)\right)\right| \leq (1+\bar{\epsilon}) \cdot \left|D_k\left(N^{\bar{r}}(v) \setminus C(i)\right)\right| \tag{11}$$

are bounded by $c - 2k$.

Proof. Assume on the contrary that Inequality 11 is false for all $r \in \{0, 2k, 4k, ..., c-2k\}$, i.e. for these values of r it holds that

$$\left|D_k\left(N^{r+2k}(v) \setminus C(i)\right)\right| > (1+\bar{\epsilon}) \cdot \left|D_k\left(N^{r}(v) \setminus C(i)\right)\right|.$$

By Corollary 3 in [15] the number of vertices in a minimum dominating set for a neighborhood $N^c(v)$ is bounded by $(2c+1)^2$. So the number of vertices needed for a dominating set for a neighborhood $N^c(v)$ such that each connected component has at least k vertices is bounded by $k \cdot (2c+1)^2$.

It always holds that $\left|D_k\left(N^0(v) \setminus C(i)\right)\right| = |D_k(\{v\})| = k$. So we have that

$$\begin{aligned} k \cdot (2c+1)^2 &\geq |D_k\left(N^c(v) \setminus C(i)\right)| \\ &> (1+\bar{\epsilon}) \cdot \left|D_k\left(N^{c-2k}(v) \setminus C(i)\right)\right| \\ &> (1+\bar{\epsilon})^2 \cdot \left|D_k\left(N^{c-4k}(v) \setminus C(i)\right)\right| \\ &> ... \\ &> (1+\bar{\epsilon})^{\frac{c}{2k}} \cdot \left|D_k\left(N^0(v) \setminus C(i)\right)\right| \\ &\geq k \cdot \left(\sqrt[k]{1+\bar{\epsilon}}\right)^{c/2} \end{aligned}$$

But from the definition of c we know that $(2c+1)^2 < \left(\sqrt[k]{1+\bar{\epsilon}}\right)^{c/2}$ and therefore $k \cdot (2c+1)^2 < k \cdot \left(\sqrt[k]{1+\bar{\epsilon}}\right)^{c/2}$ which is a contradiction. So at least for one value of $r \in \{0, 2k, 4k, ..., c-2k\}$ it holds that $\left|D_k\left(N^{r+2k}(v)\right)\right| \leq (1+\bar{\epsilon}) \cdot |D_k\left(N^r(v)\right)|$. □

Lemma 12. *The sets T_h cover all vertices of the graph.*

Proof. Assume on the contrary that there is a vertex v which is not covered by any $T_h, h \in H$. Let h be the hexagon to which v belongs and let i be its class number. At some point in the algorithm, the hexagons of class i were considered. Then there were vertices in h which were not covered yet (at least v). So the coordinator vertex of h must have marked a set T_h as covered. As the hexagons have a diameter of 1 (and $\bar{r} + k \geq 1$) it follows that v is contained in T_h and therefore covered by T_h which is a contradiction. □

Proof. (of part 1 of Theorem 3): When $diam(G) < k+2$ we compute a connected dominating set directly and there is nothing to prove. So from now on we assume that $diam(G) \geq k+2$.

Let v be a vertex. Lemma 12 shows that v is covered by a set T_h. Then the set $D_k\left(T_h\right)$ dominates v. As $D = \bigcup_h D_k\left(T_h\right)$ it holds that $D_k(T_h) \subseteq D$ and therefore D dominates v. For the correctness of the subroutine for connecting D to a connected dominating set we refer to the proof of Theorem 2. □

4.2.2. *Approximation ratio.* We want to show that Algorithm 3 achieves an approximation ratio of $1+\epsilon$. First we show that the sets S_h form a $2k$-separated collection. Then we prove that the sets S_h establish a lower bound for an optimal connected dominating set CD_{OPT}, i.e. $\left|\bigcup_{h \in H} S_h\right| \leq |CD_{OPT}|$. After this we prove that $|CD_{OPT}| \leq (1+\bar{\epsilon}) \cdot |D|$ and finally that $|CD_{OPT}| \leq (1+\epsilon) \cdot |CD|$

Lemma 13. *The sets S_h form a $2k$-separated collection.*

Proof. The claim is proven similarly as Lemma 3 in [15] (just with a $2k$-separated collection instead of a 2-separated collection).

Consider the sets S_h in the order in which they were defined by the algorithm and denote them by $S_1, S_2, ..., S_m$ (as mentioned in the description of the algorithm the order in which the sets S_h were defined during one iteration of the loop does not matter). Define $V_1 := V$ and recursively $V_i := V_{i-1} \setminus S_i$. Since $V_2 = V_1 \setminus N^{2k}(S_1)$ the collection $\{S_1, V_2\}$ is a $2k$-separated collection in G. By induction, suppose that $\{S_1, ..., S_{i-1}, V_i\}$ is a $2k$-separated collection in G. So every vertex in V_i has a distance of more than $2k$ to any other vertex in $S_1, ..., S_{i-1}$. Considering $V_{i+1} = V_i \setminus N^{2k}\left(S_i\right)$ we see that the minimum distance between any two vertices v and

v' with $v \in V_{i+1}$ and $v' \in S_i$ is at least $2k$. Therefore, $\{S_1, ..., S_i, V_{i+1}\}$ is again a $2k$-separated collection. □

Now we prove the lower bound which is established by the S_h.

Lemma 14. *Assume that $diam(G) \geq k+2$. For a $2k$-separated collection of sets S_h with $h \in H$ in G, we have*

$$|CD_{OPT}| \geq \sum_{h \in H} |D_k(S_h)|$$

Proof. Define $CD_{OPT}(S_h) := CD_{OPT} \cap N^k(S_h)$. As the S_h form a $2k$-separated collection it holds that $N^k(S_h) \cap N^k(S_{h'}) = \emptyset$ for $h \neq h'$ and therefore $CD_{OPT}(S_h) \cap CD_{OPT}(S_{h'}) = \emptyset$ for $h \neq h'$.

Claim: Each connected component in $CD_{OPT}(S_h)$ which dominates at least one vertex in S_h has at least k vertices.

Proof of the claim: Assume on the contrary there is a connected component $C \subseteq CD_{OPT}(S_h)$ with $|C| < k$ such that C dominates at least one vertex in S_h. As C dominates at least one vertex in S_h, it follows that $C \cap N(S_h) \neq \emptyset$. From $|C| < k$ follows that $N^k(S_h) \setminus N^{k-1}(S_h) = \emptyset$. So C is "trapped" in $N^{k-1}(S_h)$. As CD_{OPT} is connected it follows that C is the only connected component in CD_{OPT}. But from $diam(G) \geq k+2$ it follows that $CD_{OPT} \geq k$ which is a contradiction. This completes the proof of the claim.

We continue with the proof of the lemma. As $D_k(S_h)$ is the set with minimum cardinality such that $D_k(S_h)$ dominates S_h and that each connected component in $D_k(S_h)$ has at least k vertices, it follows that $|CD_{OPT}(S_h)| \geq |D_k(S_h)|$. So we conclude that

$$|CD_{OPT}| = \sum_{h \in H} |CD_{OPT}(S_h)| \geq \sum_{h \in H} |D_k(S_h)|$$

□

Having proved this lemma we can give an upper bound on the size of D.

Lemma 15. *The set D satisfies the following upper bound:*

$$|D| \leq (1+\bar{\epsilon}) \cdot |CD_{OPT}|$$

Proof. From the construction of the sets S_h and T_h we see that for every pair S_h, T_h it holds that $|D_k(T_h)| \leq (1+\bar{\epsilon}) \cdot |D_k(S_h)|$. So we get

$$\begin{aligned} |D| &= |\bigcup_{h \in H} D_k(T_h)| \\ &\leq \sum_{h \in H} |D_k(T_h)| \\ &\leq (1+\bar{\epsilon}) \cdot \sum_{h \in H} |D_k(S_h)| \\ &\leq (1+\bar{\epsilon}) \cdot |CD_{OPT}| \text{ (by Lemma 14)} \end{aligned}$$

□

Proof. (of part 2 of Theorem 3): When $diam(G) < k+2$ we compute an optimal connected dominating set directly and there is nothing to prove. So from now on we assume that $diam(G) \geq k+2$.

We use the same construction of the sets S_i and their planar spanners VG_i as in the proof of the approximation factor in Theorem 2. We also use the same notation for $NI_1, NI_2, I_1, I_2, V_0, VG$ and V. Let us recall that in the proof of Theorem 2 it was shown that $I_1 + I_2 \leq \frac{V}{d-1} - 1$.

As each connected component in $D_k(T_h)$ has at least k vertices, we conclude that the size of each connected component in $D = \bigcup_h D_k(T_h)$ is at least k. So we have at most V_0/k connected components in D. As VG is connected we get from the definition of NI_1 and NI_2

$$NI_1 + NI_2 \leq V_0/k - 1$$

This leads us to

$$\begin{aligned} V &\leq V_0 + 2(NI_1 + NI_2) + 2(I_1 + I_2) \\ &\leq V_0 + 2(V_0/k - 1) + 2\left(\frac{V}{d-1} - 1\right) \end{aligned}$$

From the definitions of $d, \bar{\epsilon}$ and k it follows that $\left(\frac{d-1}{d-3}\right) \leq \sqrt[3]{1+\epsilon}$, $(1+\bar{\epsilon}) = \sqrt[3]{1+\epsilon}$ and $\left(1+\frac{2}{k}\right) \leq \sqrt[3]{1+\epsilon}$. As Lemma 15 shows that $V_0 \leq |CD_{OPT}|(1+\bar{\epsilon})$ we conclude easily that:

$$\begin{aligned} V &\leq \left(\frac{d-1}{d-3}\right) \cdot \left(V_0\left(1+\frac{2}{k}\right) - 4\right) \\ &\leq \left(\frac{d-1}{d-3}\right) \cdot (1+\bar{\epsilon}) \cdot \left(1+\frac{2}{k}\right) \cdot |CD_{OPT}| \\ &\leq \sqrt[3]{1+\epsilon} \cdot \sqrt[3]{1+\epsilon} \cdot \sqrt[3]{1+\epsilon} \cdot |CD_{OPT}| \\ &= (1+\epsilon) \cdot |CD_{OPT}| \end{aligned}$$

□

4.2.3. *Locality.* We want to prove that Algorithm 3 is local (part 3 of Theorem 3). We denote by γ its locality and show that it is constant for a fixed ϵ. We give an upper bound for γ in terms of ϵ. Our proof is very similar to the locality proofs in Theorems 1 and 2.

In Section 2 we presented Lemma 7 which holds for Algorithm 3 exactly as it does for Algorithm 1 which it was originally stated for. For the sake of completeness we repeat it here as Lemma 16.

Lemma 16. *We show three locality related properties of our algorithm:*

(1) *Let v_h be the coordinator vertex of a hexagon h of class k. What vertices are in T_h depends only on the vertices which are at most $2c \cdot (k-1) + c$ hops away from v_h.*
(2) *Let v' be any vertex. Whether v' is contained in a set $T_{h'}$ with $class(h') \leq k$ depends only on the vertices which are at most $2c \cdot k$ hops away from v'. If v' is contained in such a set $T_{h'}$ then what vertices are in $T_{h'}$ depends only on the vertices which are at most b_k hops away from v'.*

(3) *Let v'' be any vertex in a hexagon h'' of class k. Whether or not v'' is the coordinator vertex of h'' depends only on the vertices which are at most $1 + 2c \cdot (k-1)$ hops away from v''.*

Proof. For ease of notation we introduce the sequences a_k, b_k and c_k. Let a_k be the smallest integer such that what vertices are in T_h depends only on the vertices which are at most a_k hops away from v_h. So in order to prove property 1 we want to show that $a_k \leq 2c \cdot (k-1) + c$. Let b_k be the smallest integer such that whether v' is contained in a set $T_{h'}$ with $class(h') \leq k$ depends only on the vertices which are at most b_k hops away from v' and if v' is contained in such a set $T_{h'}$ then what vertices are in $T_{h'}$ depends only on the vertices which are at most b_k hops away from v'. For proving property 2 we need to show that $b_k \leq 2c \cdot k$. Let c_k be the smallest integer such that whether or not v'' is the coordinator vertex of h'' depends only on the vertices which are at most c_k hops away from v''. So for proving property 3 we need to show that $c_k \leq 1 + 2c \cdot (k-1)$.

Proof by induction. We begin with $k = 1$. As we need to explore the vertices at most c hops away from v_h in order to compute T_h, we conclude that $a_1 \leq c$.

Let v'' be a vertex in a class 1 hexagon h''. To find out whether v'' is the coordinator vertex of h'', we need to explore the vertices which are at most 1 hop away from v''. So $c_1 \leq 1$.

Let v' be a vertex. We want to find out whether there is a hexagon h' with $class(h') \leq 1$ such that v' is contained in the set $T_{h'}$. If yes, the coordinator vertex $v_{h'}$ of h' can be at most c hops away from v'. So we need to explore all vertices which are at most c hops away from v' to find all vertices (in class 1 hexagons) which could possibly be coordinator vertices for their hexagon h' such that $v' \in T_{h'}$ and $class(h') \leq 1$. To find out if any of them is the coordinator vertex of their respective hexagon h' we need to explore the vertices which are at most $c_1 \leq 1$ hops away from them. If one of them is a coordinator vertex, we need to explore the vertices at most $a_1 \leq c$ from it in order to compute $T_{h'}$ and to find out whether $v' \in T_{h'}$. If this is the case, we immediately know the set $T_{h'}$ as well. So we only need to explore the vertices which are at most $b_1 \leq c + \max(a_1, c_1) \leq 2c$ hops away from v' in order to compute this task.

Assume that the claims in the lemma hold for all $k \leq i-1$. Let v_h be the coordinator vertex of a hexagon h of class i. In order to compute T_h we need to explore the vertices which are at most c hops away from v_h and therefore need to find out for each vertex in $N^c(v_h)$ whether it has been covered by a set $T_{h'}$ with $class(h') < i$. So for computing T_h we need to explore the vertices which are $a_i \leq c + b_{i-1}$ hops away from v_h.

Let v'' be a vertex in a hexagon h'' of class i. To find out whether v'' is the coordinator vertex for h'' we need to explore all other vertices in h'' and find out if they have been covered by a set $T_{h'}$ with $class(h') < i$. For this we need to explore the vertices which are at most $c_i \leq 1 + b_{i-1}$ hops away from v''.

Let now v' be a vertex. We want to find out whether v' is covered by a set $T_{h'}$ with $class(h') \leq i$. So first we need to explore all vertices at most c hops away from v'. This is the set $N^c(v')$. Only vertices in this set can possibly be coordinator vertices for a hexagon h' such that $T_{h'}$ contains v'. To check if a vertex in $N^c(v')$ is a coordinator vertex, we need to explore all vertices which are at most c_i hops away from it. If a vertex $v_{h'}$ in $N^c(v')$ is the coordinator vertex for its hexagon h', we need to explore the vertices which are at most a_i hops away from $v_{h'}$ in order to compute

$T_{h'}$. Then we can check if $v' \in T_{h'}$. If this is the case, we immediately know the set $T_{h'}$ as well. This gives us $b_i \leq c + \max(a_i, c_i) \leq c + \max(c + b_{i-1}, 1 + b_{i-1}) \leq c + a_i$.

So we have shown that $a_1 \leq c$, $b_1 \leq 2c$, $c_1 \leq 1$, $c_i \leq 1 + b_{i-1}$, $b_i \leq c + a_i$ and $a_i \leq c + b_{i-1}$. This implies $a_i \leq c + b_{i-1} \leq c + c + a_{i-1} \Rightarrow a_i \leq 2c \cdot (i-1) + c$, $b_i \leq 2c \cdot i$ and $c_i \leq 1 + 2c \cdot (i-1)$. □

Lemma 17. *Let v be a vertex. Whether $v \in CD$ depends only on the vertices which are at most $k + 1 + 2c \cdot (b+1) + c + b \cdot (2+d)$ hops away from v.*

Proof. Let v' be a vertex. Exactly as in the proof of Lemma 8 we claim that what vertices (other than v') are in $T_{h'}$ depends only on the vertices which are at most $1 + 2c \cdot (k-1) + c$ hops away from v'.

Proof of the claim: We use the notation a_k, b_k and c_k as introduced in the proof of Lemma 16. For computing the set $T_{h'}$ we need to check whether v' is covered by a set $T_{h'}$ with $class(h') < class(h)$. This depends only on the vertices at most $b_{k-1} \leq 2c \cdot (k-1)$ hops away from v' (see Lemma 16). If there is such a set $T_{h'}$ with $v' \in T_{h'}$ then $T_{h'}$ depends only on the vertices which are at most b_{k-1} hops away from v' as well (see Lemma 16).

If there is no set $T_{h'}$ such that $class(h') < class(h)$ and $v' \in T_{h'}$ the algorithm has to find out whether there is another vertex $v'' \neq v'$ in h that is the coordinator vertex for h. In order to do this, we need to explore the vertices at most $c_k \leq 1 + 2c \cdot (k-1)$ hops away from v'. If there is such a vertex v'' then we need to explore the vertices which are at most $1 + a_k \leq 1 + 2c \cdot (k-1) + c$ hops away from v' in order to compute the set $T_{h'}$. If not, then v' is the coordinator vertex for h. Then we need to explore the vertices which are at most $a_k \leq 2c \cdot (k-1) + c$ hops away from v' in order to compute $T_{h'}$. Altogether we conclude that for computing $T_{h'}$ we need to know only about the vertices at most $1 + 2c \cdot (k-1) + c$ hops away from v'. This proves the claim.

When the vertex v computes whether it is in D it does the following: For each $v' \in N^k(v)$ it computes the set T_h that covers v'. Then v is part of the dominating set if and only if it is part of any of the $D_k(T_h)$. So if v is in D depends only on the vertices which are at most $1 + k + 2c \cdot (k-1) + c$ hops away from v.

Now we consider phase 2 where the different connected components are linked by bridges. If v is assigned to CD in phase 2 depends only if the vertices at most $b(2+d)$ from v are in D (see Lemma 10).

So whether v is in CD depends on the vertices which are at most $k + 1 + 2c \cdot (b-1) + c + b \cdot (2+d)$ hops away from v. □

Proof. (of part 3 of Theorem 3): We want to show that whether or not a vertex v is in D depends only on the vertices at most $O\left(\frac{1}{\epsilon^9}\right)$ away from v. We need the following properties of the constants that we defined:

$$\begin{aligned} d &\in O\left(\frac{1}{\epsilon}\right) \\ \frac{1}{\bar{\epsilon}} &\in O\left(\frac{1}{\epsilon}\right) \\ k &\in O\left(\frac{1}{\epsilon}\right) \\ c &\in O\left(\frac{1}{\epsilon^3}\right) \\ b &\in O\left(c^2\right) \end{aligned}$$

After having proved this we will substitute the derived upper bounds in the term presented in Lemma 17 and prove part 3 of Theorem 3.

First we want to show that $d \in O\left(\frac{1}{\epsilon}\right)$. We calculate that

$$\begin{aligned} \sqrt[3]{1+\epsilon} &\geq \frac{d-1}{d-3} \\ \Leftrightarrow \sqrt[3]{1+\epsilon} \cdot (d-3) &\geq d-1 \\ \Leftrightarrow d \cdot \left(\sqrt[3]{1+\epsilon} - 1\right) &\geq 3 \cdot \sqrt[3]{1+\epsilon} - 1 \\ \Leftrightarrow d &\geq \frac{3 \cdot \sqrt[3]{1+\epsilon} - 1}{\sqrt[3]{1+\epsilon} - 1} \end{aligned}$$

and get that

$$\begin{aligned} d &\geq \frac{3 \cdot \sqrt[3]{1+\epsilon} - 1}{\sqrt[3]{1+\epsilon} - 1} \\ &\geq 3 + \frac{2}{\epsilon} \end{aligned}$$

It follows that $d \leq 4 + \frac{2}{\epsilon}$ and therefore $d \in O\left(\frac{1}{\epsilon}\right)$. Now we want to show that $\frac{1}{\bar{\epsilon}} \in O\left(\frac{1}{\epsilon}\right)$. For $\epsilon \leq 1$ it holds that

$$\begin{aligned} 1 + \bar{\epsilon} &\leq \sqrt[3]{1+\epsilon} \\ \Leftrightarrow (1+\bar{\epsilon})^3 &\leq 1+\epsilon \\ \Leftrightarrow 1 + 3\bar{\epsilon} + 3\bar{\epsilon}^2 + \bar{\epsilon}^3 &\leq 1+\epsilon \\ \Leftrightarrow 3\bar{\epsilon} + 3\bar{\epsilon}^2 + \bar{\epsilon}^3 &\leq \epsilon \\ \Rightarrow \bar{\epsilon} &\geq \frac{\epsilon}{7} \end{aligned}$$

and therefore $\frac{1}{\bar{\epsilon}} \in O\left(\frac{1}{\epsilon}\right)$.

Next we show that $k \in O\left(\frac{1}{\epsilon}\right)$. Recall that k is defined as the smallest integer such that $\left(1 + \frac{2}{k}\right) \leq \sqrt[3]{1+\epsilon}$. Since this leads to the same calculation as above we substitute $\frac{2}{k}$ for $\bar{\epsilon}$ and get

$$\begin{aligned} \frac{\epsilon}{7} &\leq \frac{2}{k} && (12)\\ \Leftrightarrow k &\leq \frac{14}{\epsilon} && (13)\\ \Rightarrow k &\in O\left(\frac{1}{\epsilon}\right) && (14) \end{aligned}$$

Now we want to show that there is an ϵ_0 such that for all $\epsilon < \epsilon_0$ it holds that $c \leq \frac{1}{\epsilon^3} + k$ i.e. $c \in O\left(\frac{1}{\epsilon^3}\right)$. By definition c is the smallest integer such that $(2c+1)^2 < \left(\sqrt[k]{1+\epsilon}\right)^c$ and $c \equiv 0 \bmod k$. We calculate that

$$\begin{aligned} (2c+1)^2 &< \left(\sqrt[k]{1+\epsilon}\right)^{c/2} && (15)\\ \Leftrightarrow 4k\ln(2c+1) &< c\cdot\ln(1+\epsilon) && (16) \end{aligned}$$

From taking derivatives we get that

$$\ln(1+\epsilon) \geq \frac{1}{2}\epsilon \quad (17)$$

holds for all $\epsilon \leq 1$. From Inequalities 13, 16 and 17 and we conclude that it is sufficient to show that there is an ϵ_0 such that

$$4\left(\frac{14}{\epsilon}\right)\ln(2c+1) < \frac{1}{2}\epsilon\cdot c$$

holds for all for $\epsilon < \epsilon_0 \leq 1$. We set $c := \frac{1}{\epsilon^3}$ and get

$$\begin{aligned} 56\cdot\frac{1}{\epsilon}\cdot\ln\left(\frac{2}{\epsilon^3}+1\right) &< \frac{1}{2}\cdot\frac{1}{\epsilon^2}\\ \Leftrightarrow 112\cdot\ln\left(\frac{2}{\epsilon^3}+1\right) &< \frac{1}{\epsilon} \end{aligned}$$

Since $\ln(2n^3+1) \in o(n)$ there is an $\epsilon_0 \leq 1$ such that the above holds for all $\epsilon \leq \epsilon_0$. So for $\epsilon \leq \epsilon_0$ we can find a value for c with $c < \frac{1}{\epsilon^3} + k < \frac{1}{\epsilon^3} + \frac{14}{\epsilon}$ (remember that c has to be an integer such that $c \equiv 0 \bmod k$). Denote by k_0 the value of k calculated for ϵ_0. We observe that $\left(2\frac{1}{\epsilon_0^2}+1\right)^2 < \left(\sqrt[k_0]{1+\epsilon_0}\right)^{\frac{1}{\epsilon_0^3}} < \left(\sqrt[k]{1+\epsilon}\right)^{\frac{1}{\epsilon_0^3}}$ for $\epsilon > \epsilon_0$. So for all these values of ϵ we can find a value for c such that $c < \frac{1}{\epsilon_0^3} + \frac{14}{\epsilon_0}$. This proves $c \in O\left(\frac{1}{\epsilon^3}\right)$. The claim $b \in O\left(c^2\right)$ follows from Lemma 6.

Denote by $\gamma(\epsilon)$ the locality distance of Algorithm 3 when run with a performance guarantee of $1+\epsilon$.

For checking whether $diam(G) < k+2$ we only need to explore the vertices which are at most $k+2$ hops away from a given vertex v: If $N^{k+2}(v) \neq N^{k+1}(v)$ we know for sure that $diam(G) \geq k+2$ and if $N^{k+2}(v) = N^{k+1}(v)$ we know that $N^{k+2}(v) = N^{k+2}(v) = V$ and we can check whether $diam(G) < k+2$ based on this information. So when $diam(G) < k+2$ then $\gamma(\epsilon)$ is at most $k+2$ and therefore bounded by $O\left(\frac{1}{\epsilon}\right) \subseteq O\left(\frac{1}{\epsilon^9}\right)$. From now on assume that $diam(G) \geq k+2$. From Lemma 17 we know that we need to explore the vertices at most $k+1+2c\cdot(b-1)+c+b\cdot(2+d)$ hops away from a given vertex v. From the definition of b and the above lemmas we get

Connected Dominating Set			
Approximation Ratio	Upper Bound for Locality Distance of		
	Algorithm 3	Algorithm 2	Algorithm in [6]
1.1	$8.47805 \cdot 10^{14}$	-	-
1.2	$9.95049 \cdot 10^{12}$	-	-
1.5	$2.65059 \cdot 10^{10}$	-	-
2	$3.42895 \cdot 10^{8}$	-	-
3	$5.75726 \cdot 10^{6}$	-	-
3.5	$1.21160 \cdot 10^{6}$	$9.32608 \cdot 10^{6}$	-
4	$3.72278 \cdot 10^{5}$	$8.06049 \cdot 10^{6}$	-
5	$9.3987 \cdot 10^{4}$	$8.0745 \cdot 10^{4}$	-
6	$1.5859 \cdot 10^{4}$	$1.6150 \cdot 10^{4}$	-
7.5	$1.0938 \cdot 10^{4}$	$8.155 \cdot 10^{3}$	$2.03759 \cdot 10^{8}$
10	$1.002 \cdot 10^{3}$	$4.604 \cdot 10^{3}$	$5.702 \cdot 10^{3}$

TABLE 2. Locality distances of Connected Dominating Set Algorithms for several approximation factors. The second column shows upper bounds for the $1+\epsilon$ approximation algorithm and the third column the bounds for the $3+\epsilon$ approximation algorithm presented in this paper. The fourth column shows upper bounds for the locality of the local $7.453+\epsilon$ algorithm for connected dominating set presented in [6].

$$\begin{aligned}\gamma(\epsilon) &\leq k+1+2c\cdot(b-1)+c+b\cdot(2+d)\\ &\leq k+1+2c\cdot\left(12c^2+18c+7-1\right)+c+\left(12c^2+18c+7\right)\cdot(2+d)\\ &\in O\left(\frac{1}{\epsilon^9}\right)\end{aligned}$$

□

Table 2 displays trade-offs between approximation ratios and locality distances which are attained by the algorithms for connected dominating set presented in this paper and the algorithm presented in [6].

4.2.4. *Processing time.* Recall that γ is the locality distance of Algorithm 3. Similar to Algorithms 1 and 2 we measure the runtime of Algorithm 3 when run by a vertex v in the number of vertices in $N^{\gamma}(v)$. Denote this number by $n_{\gamma}(v)$. We want to prove that the time Algorithm 3 needs to compute whether v is in CD is bounded by a polynomial in $n_{\gamma}(v)$. The vertices in $N^{\gamma}(v)$ are all vertices that v needs to explore when computing whether or not it is part of CD.

Proof. (of part 4 of Theorem 3): First we examine the case where $diam(G) < k+2$. By Corollary 3 in [15] the number of vertices in a minimum dominating set for a neighborhood $N^r(v)$ of a vertex v is bounded by $(2r+1)^2$ and therefore the size of a minimum connected dominating set for $N^r(v)$ is bounded by $3\cdot(2r+1)^2$. So the processing time to find a minimum connected dominating set for G is in $O\left(n_{\gamma}(v)^{3\cdot(2k+3)^2}\right)$ and therefore bounded by $n_{\gamma}(v)^{O\left(1/\epsilon^2\right)}$.

Now assume that $diam(G) \geq k+2$. In the first part of the algorithm where the set D is computed the sets $D_k\left(N^r(v')\right)$ for $r \in \{0, 2k, 4k, ..., c\}$ and $v' \in N^\gamma(v)$ need to be determined. First we show that this can be done in polynomial time. By Corollary 3 in [15] the number of vertices in a minimum dominating set for a neighborhood $N^r(v')$ is bounded by $(2r+1)^2$. It follows that the number of vertices in $D_k\left(N^r(v')\right)$ is bounded by $k \cdot (2r+1)^2$. So the computation of such a set can be done in $O\left(n_\gamma(v)^{k\cdot(2c+1)^2}\right)$, e.g. by enumeration. For each vertex $v' \in N^\gamma(v)$ we might have to compute the sets $D_k\left(N^r(v')\right)$ for each $r \in \{0, 2k, 4k, ..., c\}$. So the processing time for computing D is in $O\left(n_\gamma(v)^{k\cdot(2c+1)^2} \cdot n_\gamma(v) \cdot c\right)$ and therefore bounded by $n_\gamma(v)^{O\left(1/\epsilon^7\right)}$.

Now consider the the second part of the algorithm where we connect the connected components in D to the set CD. For each class $i \in \{1, ..., b\}$ we need to compute the d-neighborhood of head vertices of class i and find out whether adding one or two vertices with one of them being in a hexagon of class i could connect two connected components. For one head vertex this can be done in $O(n_\gamma(v))$ and therefore the runtime of the second part of our algorithm is bounded by $O(n_\gamma(v)^2)$. So our overall processing time is bounded by $n_\gamma(v)^{O\left(1/\epsilon^7\right)}$. □

5. Conclusion

In this paper we designed local approximation algorithms for dominating and connected dominating set in the setting of location aware nodes. We presented a local $1+\epsilon$ algorithm for dominating set, a local $3+\epsilon$ algorithm for connected dominating set and improved the latter to a local $1+\epsilon$ algorithm. Our algorithms give better performance ratios than the formerly known local algorithms with constant locality distances [6]. Our approximation ratios are optimal since no local algorithm based on location aware nodes can compute an optimal solution for every given graph (note that this does not depend on whether $P \neq NP$ or $P = NP$). If $P \neq NP$ then for the problems which we addressed our local algorithms achieve exactly the same approximation ratios as global polynomial time algorithms.

We estimated the locality distances of our algorithms and it is evident that further improvements would be desirable. In particular an interesting question to study would be how much they can be improved for a given $1+\epsilon$ approximation factor. Also of interest are lower bounds for the possible approximation ratio for a given locality distance. From that we could conclude how far we are away from the best possible local approximation for a given locality distance. Another open question is how much of the geometric information of a node is really needed to be able to obtain the above results.

References

[1] K. Alzoubi, P. Wan, and O. Frieder. Message-optimal connected dominating sets in mobile ad hoc networks. In *MobiHoc '02: Proceedings of the 3rd ACM international symposium on Mobile Ad Hoc Networking & Computing*, pages 157–164, New York, NY, USA, 2002. ACM Press.

[2] H. Breu and D. G. Kirkpatrick. Unit disk graph recognition is NP-hard. *Computational Geometry. Theory and Applications*, 9(1-2):3–24, January 1998.

[3] E. Chávez, S. Dobrev, E. Kranakis, J. Opatrny, L. Stacho, and J. Urrutia. Local construction of planar spanners in unit disk graphs with irregular transmission ranges. In José R. Correa,

Alejandro Hevia, and Marcos A. Kiwi, editors, *LATIN*, volume 3887 of *Lecture Notes in Computer Science*, pages 286–297. Springer, 2006.

[4] X. Cheng, X. Huang, D. Li, W. Wu, and D.-Z. Du. A polynomial-time approximation scheme for the minimum-connected dominating set in ad hoc wireless networks. *Networks*, 42(4):202–208, 2003.

[5] B. N. Clark, C. J. Colbourn, and D. S. Johnson. Unit disk graphs. *Discrete Math.*, 86(1-3):165–177, 1990.

[6] J. Czyzowicz, S. Dobrev, T. Fevens, H. González-Aguilar, E. Kranakis, J. Opatrny, and J. Urrutia. Local algorithms for dominating and connected dominating sets of unit disc graphs with location aware nodes. *to appear*, 2007.

[7] B. Gfeller and E. Vicari. A Faster Distributed Approximation Scheme for the Connected Dominating Set Problem for Growth-Bounded Graphs. In Evangelos Kranakis and Jaroslav Opatrny, editors, *ADHOC-NOW*, volume 4686 of *Lecture Notes in Computer Science*, pages 59–73. Springer, 2007.

[8] H. B. Hunt III, M. V. Marathe, V. Radhakrishnan, S. S. Ravi, D. J. Rosenkrantz, and R. E. Stearns. NC-approximation schemes for NP- and PSPACE-hard problems for geometric graphs. *J. Algorithms*, 26(2):238–274, 1998.

[9] D. S. Johnson. Approximation algorithms for combinatorial problems. In *Proc. 5th Ann. ACM Symp. Theory Computing*, pages 38–49, NY, 1973. ACM. also in J. Comput. Syst. Sci. 9 3 (Dec. 1974), 256-278.

[10] F. Kuhn, T. Moscibroda, and R. Wattenhofer. Unit disk graph approximation. In *DIALM-POMC '04: Proceedings of the 2004 joint workshop on Foundations of mobile computing*, pages 17–23, New York, NY, USA, 2004. ACM Press.

[11] F. Kuhn, T. Moscibroda, and R. Wattenhofer. The price of being near-sighted. In *SODA '06: Proceedings of the seventeenth annual ACM-SIAM Symposium on Discrete Algorithms*, pages 980–989, New York, NY, USA, 2006. ACM Press.

[12] F. Kuhn, T. Nieberg, T. Moscibroda, and R. Wattenhofer. Local approximation schemes for ad hoc and sensor networks. In *DIALM-POMC '05: Proceedings of the 2005 joint workshop on Foundations of mobile computing*, pages 97–103, New York, NY, USA, 2005. ACM Press.

[13] N. Linial. Locality in distributed graph algorithms. *SIAM J. Comput.*, 21(1):193–201, 1992, February.

[14] M. V. Marathe, H. Breu, H. B. Hunt III, S. S. Ravi, and D. J. Rosenkrantz. Simple heuristics for unit disk graphs. *Networks*, 25(1):59–68, 1995.

[15] T. Nieberg and J. L. Hurink. A PTAS for the minimum dominating set problem in unit disk graphs. In T. Erlebach and G. Persiano, editors, *3rd International Workshop on Approximation and Online Algorithms, WAOA 2005, Palma de Mallorca, Spain*, volume 3879 of *Lecture Notes in Computer Science*, pages 296–306, Heidelberg, Germany, 2006. Springer-Verlag.

[16] R. Raz and S. Safra. A sub-constant error-probability low-degree test, and a sub-constant error-probability PCP characterization of NP. In *Proceedings of the 29th Annual ACM Symposium on the Theory of Computing (STOC '97)*, pages 475–484, New York, May 1997. Association for Computing Machinery.