LOCAL PTAS FOR INDEPENDENT SET AND VERTEX COVER IN LOCATION AWARE UNIT DISK GRAPHS

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

Abstract. We present the first local approximation schemes for maximum independent set and minimum vertex cover in unit disk graphs. In the graph model we assume that each node knows its geographic coordinates in the plane (location aware nodes). Our algorithms are local in the sense that the status of each node v (whether or not v is in the computed set) depends only on the vertices which are a constant number of hops away from v. This constant is independent of the size of the network. We give upper bounds for the constant depending on the desired approximation ratio. We show that the processing time which is necessary in order to compute the status of a single vertex is bounded by a polynomial in the number of vertices which are at most a constant number of vertices away from it. Our algorithms give the best possible approximation ratios for this setting.

The technique which we use to obtain the algorithm for vertex cover can also be employed for constructing the first global PTAS for this problem in unit disk graph which does not need the embedding of the graph as part of the input.

1. Introduction

Locality plays an important role in wireless and ad-hoc-networks. In such networks, there is often no global entity to organize the network traffic. So the nodes have to negotiate their coordination in a distributed manner. As in most cases the entire network is much too large to be explored by a single node, we are interested in local algorithms. These are algorithms, in which the status of a node v (e.g., whether or not it is in the independent set or vertex cover) depends only on the nodes which are a constant number of hops away from v. This constant has to be independent of the size of the overall network. The locality concept is also advantageous for disaster recovery and dynamically changing network. If only parts of the network are changed or lost, using a local algorithm only fractions of the solution need to be recomputed and so we do not need to redo the entire calculation.

We model the wireless network with location aware Unit Disk Graphs (UDGs). This represents the situation where all nodes have an identical transmission range and know about their geographic position in the plane. Whether communication

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).

between two nodes is possible depends only on their Euclidean distance. This is due to the fact that wireless devices naturally have a limited transmission range. Since positioning devices, like GPS, become more and more common the concept of location awareness becomes relevant.

In wireless networks, clustering is an important aspect of organizing network traffic. A maximal independent set can be used for clustering by defining its nodes to be clusterheads. A cluster consists of the nodes which are adjacent to a single clusterhead. The latter are responsible for the communication of nodes within their cluster with other nodes. The maximum independent set and minimum vertex cover problems are closely related since the inverse set of an independent set is a vertex cover. However, approximation ratios for the above problems are not preserved by this operation, so we need to design approximation algorithms for both problems.

1.1. **Related Work.** Maximum independent set and minimum vertex cover are both NP-hard in general graphs [6]. For independent set it is even impossible to approximate the problem in polynomial time with a better factor than $|V|^{1/2-\epsilon}$, unless $P = NP$ [7]. (Observe that finding a maximum independent set in a graph is the same as finding a maximum clique in its complementary graph.) However, for vertex cover there are several polynomial time approximation algorithms which achieve an approximation factor of 2, e.g., in [1]. But it is NP-hard to find an approximation better than $10\sqrt{5} - 21 \approx 1,3607$ [5], so there can be no PTAS, unless $P = NP$.

When restricting the problem to unit disk graphs, both problems are still NP-hard [3]. But there are constant ratio algorithms known [11] (in the case of vertex cover with a ratio of 3/2 which is better than in the general case). If the embedding of the graph is part of the input there are PTASs known due to Hunt III et al. [8]. Note however that finding an embedding for a unit disk graph is NP-hard [2] (since the recognition of a unit disk graph is NP-hard). Even finding an approximation for the embedding is NP-hard [9]. For the case that the embedding of the graph is unknown, Nieberg et al. found a PTAS for the weighted independent set problem in UDGs [13]. Kuhn et al. proposed a local approximation scheme for independent set [10] for growth-bounded graphs (this class includes UDGs). However, their definition of locality is not the same as assumed in this paper. In their algorithm, whether a vertex v is in the independent set depends on the vertices which are up to $O(\log^* n)$ hops away from v which is not constant as it depends on the size of the graph.

1.2. **Main Result.** We present local approximation algorithms for maximum independent set and minimum vertex cover with approximation ratios $1-\epsilon$ and $1+\epsilon$ respectively. They are the first local approximation schemes for these problems in unit disk graphs. Their technique is very similar to the one of the local dominating set algorithm presented in [14]. We give upper bounds for the locality distance (the maximum k such that whether or not a vertex is in the computed set depends only on the vertices which are at most k hops away from it) depending on the desired approximation ratio. In order to be able to guarantee the approximation factor for vertex cover, we prove that the set of all vertices of a unit disk graph forms a factor 12 approximation for minimum vertex cover (assuming that there is at least one edge to cover in the graph). The technique used in this proof can be expanded

to show that in every $K_{1,m}$–free graph the set of all vertices forms a factor $2m$ approximation for vertex cover.

Our technique to derive the PTAS for minimum vertex cover has applications in other settings as well. We explain how it can be used to construct the first global PTAS for vertex cover in unit disk graph which does not need the embedding of the graph as part of the input.

1.3. **Organization of the Paper.** The remainder of this paper is organized as follows: In Section 2 we introduce some basic concepts and definitions including a tiling of the plane in hexagons which our algorithms are using. Our $1-\epsilon$ approximation algorithm for independent set is presented in Section 3. After that in Section 4 we prove our bound for vertex cover and present our $1+\epsilon$ approximation algorithm for this problem. There we also discuss how this technique can be used to derive a global PTAS for vertex cover which does not rely on the embedding of the graph. Finally in Section 5 we summarize our results and discuss open problems.

2. Preliminaries

We give some basic definitions and explain a tiling of the plain which our algorithms are using. Then we introduce the concept of 1-separated collections which will enable us to establish a lower bound for the maximum independent set problem.

2.1. **Definitions.** 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 our algorithms is a unit disk graph.

A set $I \subseteq V$ is an *independent set* if for every pair of vertices v, v' with $v \in I$ and $v' \in I$ it holds that $\{v,v'\} \notin E$. A set $VC \subseteq V$ is called a *vertex cover* if for every edge $e=\{u,v\}$ it holds that either $u \in VC$ or $v \in VC$. Equivalently, a set VC is a vertex cover if and only if $V \setminus VC$ is an independent set.

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 an independent set or vertex cover) 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.

2.2. **Tiling of the Plane.** This method of tiling the plane is taken from [4] and [14]. The plane is split into hexagons and a class number is assigned to each hexagon. The tiling has the following properties:

- Each vertex is in exactly one hexagon.
- Two vertices in the same hexagon are connected by an edge.
- Each hexagon has a class number.

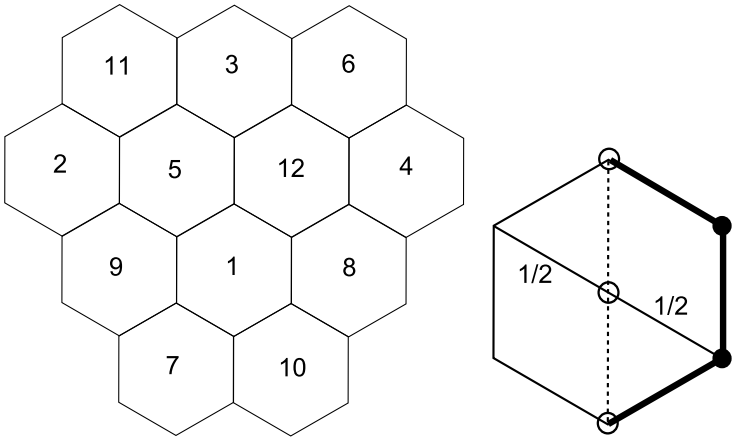

FIGURE 1. (a) A tile divided into 12 hexagons. Having 12 hexagons in one tile achieves a minimum Euclidean distance between to hexagons of the same class of 2. (b) One hexagon of the tiling. The bold lines indicate the parts of its border that belong to this hexagon

- The distance between two vertices in different hexagons with the same class number is at least a certain constant positiv integer.
- 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 $(2c+1)^2 < \left(\frac{1}{1-\epsilon}\right)^c$. 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 Figures 1 and 2). 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 1(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. **1-Separated Collections.** We present the concept of 1-separated collections. We use it in order to establish an upper bound for an optimal independent set. Let $G=(V,E)$ be a unit disk graph.

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 *1-separated collection* if for any two vertices $s \in S_h$ and $s' \in S_{h'}$ with $h \neq h'$ it holds $d(s,s') > 1$.

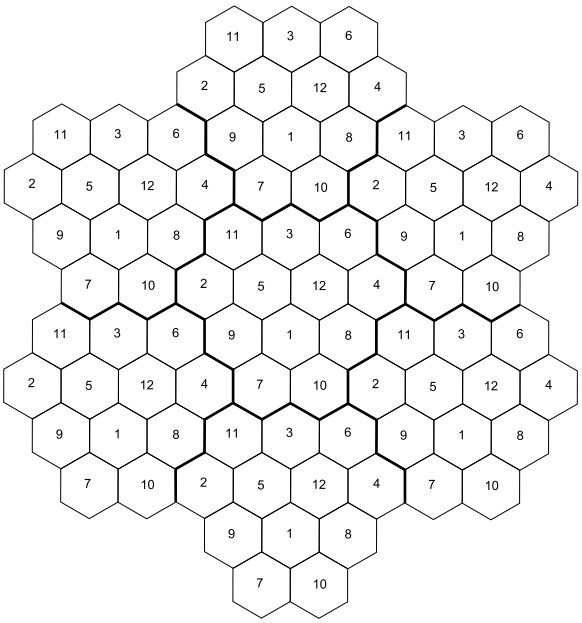

FIGURE 2. Several tiles glued together

Let $I : \mathcal{P}(V) \to \mathcal{P}(V)$ be an operation returning an *independent set of maximum cardinality* for the subset of vertices given as argument to it. Later in the algorithm we will construct sets S_h and will surround them by sets T_h such that $S_h \subseteq T_h \subseteq N(S_h)$. Now we establish our upper bound for a maximum independent set for G.

Lemma 1. *Let the sets T_h, $h \in H$ be sets such that $V = \bigcup_{h\in H} T_h$. We have that*

$$|I(V)| \leq \sum_{h\in H} |I(T_h)|$$

Proof. Since $I(T_h)$ is a maximum independent set for T_h it holds that $|I(V) \cap T_h| \leq |I(T_h)|$. As $V = \bigcup_{h\in H} T_h$ every vertex $v \in I(V)$ is contained in at least one set T_h. So it follows that $|I(V)| \leq \sum_{h\in H} |I(T_h)|$. □

So we see that the cardinality of the union $\bigcup_{h\in H} I(T_h)$ is an upper bound for the cardinality of a maximum independent set. The idea is now to construct a 1-separated collection S_h and a surrounding set T_h for each set S_h with $T_h \subseteq N(S_h)$. We want to do it in a way such that $|I(T_h)|$ is not much larger than $|I(S_h)|$. If then $\bigcup_{h\in H} T_h = V$ we can show that $\bigcup_{h\in H} I(S_h)$ is not much smaller than an optimal independent set for G.

Lemma 2. *Let $\mathcal{S} = \bigcup_{h\in H} S_h$ be a 1-separated collection in G and let T_h,$h \in H$ be subsets of V with $S_h \subseteq T_h$ for all $h \in H$. If there exists an $\epsilon > 0$ such that*

$$(1 - \epsilon) \cdot |I(T_h)| \leq |I(S_h)|$$

holds for all $h \in H$ and if $V = \bigcup_{h\in H} T_h$ then the set $\bigcup_{h\in H} I(S_h)$ is a $(1 - \epsilon)$-approximation of a maximum independent set in G.

Proof. Let I_{OPT} be an optimal independent set for G. It holds that

$$\left|\bigcup_{h\in H} I\left(S_h\right)\right| = \sum_{h\in H} \left|I\left(S_h\right)\right| \geq (1-\epsilon)\cdot \sum_{h\in H} \left|I\left(T_h\right)\right| \geq (1-\epsilon)\cdot I_{OPT}$$

□

3. Independent Set

We present a local $1-\epsilon$ approximation algorithm for maximum independent set. Its technique is very similar to the one of the local algorithm for minimum dominating set presented in [14].

3.1. **The Algorithm.** Before giving the formal algorithm we present the main structure and provide an intuitive description of the algorithm. For some hexagons h we construct a set S_h and a set T_h with $S_h \subseteq T_h \subseteq N\left(S_h\right)$. The sets T_h contain all vertices in their respective hexagon h and the vertices in a certain surrounding area. They 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 S_h a maximum independent set $I\left(S_h\right)$. We output $I := \bigcup_h I\left(S_h\right)$.

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 sets $S_h = T_h = \emptyset$. 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 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 all $r \leq c$ the r-neighborhoods $N^r\left(v_h\right)$ and compute the maximum independent sets $I\left(N^r\left(v_h\right)\right)$. We determine the smallest value of r with $r \leq c-1$ such that

$$(1-\epsilon)\cdot\left|I\left(N^{r+1}(v_h)\setminus C(i)\right)\right| \leq \left|I\left(N^r(v_h)\setminus C(i)\right)\right| \tag{1}$$

holds and denote it by $\bar{r}$. Later we will prove that there is at least one value for r with $r \leq c-1$ such that Inequality 1 does indeed hold (see Lemma 3). Now mark all vertices in $T_h := N^{\bar{r}+1}\left(v_h\right) \setminus C(i)$ as *covered*. We define $S_h := N^{\bar{r}}\left(v_h\right) \setminus C(i)$. In Lemma 4 we will prove that the sets S_h (for various hexagons h) form a 1-separated collection. We assign all vertices in $I\left(S_h\right)$ to the independent set. We do this procedure for all 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 $I := \bigcup_{h\in H} I\left(S_h\right)$.

The previous discussion is presented in Algorithm 1.

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

```
// Algorithm is executed independently by each node v;
inSet:=false;
if ∃ hexagon h with v ∈ S_h then
    compute I(S_h);
    if v ∈ I(S_h) then inSet:=true
end
if inSet=true then Become part of the independent set I else Do not become part of I
```

3.2. **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 I is an independent set for G.*
(2) *Let I_{OPT} be an optimal independent set. It holds that $|I| \geq (1-\epsilon) \cdot |I_{OPT}|$.*
(3) *Whether or not a vertex v is in I 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. It is very similar to the proof given in [14] for the correctness of the minimum dominating set algorithm presented there.

3.2.1. *Correctness.* We want to prove that the set I is indeed an independent set for G. As mentioned above we first prove that it is sufficient to examine values for r with $r \leq c-1$ while computing $I\left(N^r(v)\right)$, employing an argument used in [13].

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*

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

are bounded from above by $c-1$.

Proof. Assume on the contrary that Inequality 2 is false for all $r \in \{0, 1, ..., c-1\}$, i.e. for these values of r it holds that

$$\left|I\left(N^{r+1}(v) \setminus C(i)\right)\right| > \left(\frac{1}{1-\epsilon}\right) \cdot \left|I\left(N^r(v) \setminus C(i)\right)\right|.$$

By Corollary 3 in [12] the number of vertices in a maximum independent set for a neighborhood $N^c(v)$ is bounded by $(2c+1)^2$. It holds that $\left|I\left(N^0(v)\right)\right| = |I(\{v\})| = 1$. So we have that

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

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

Lemma 4. *The sets S_h, $h \in H$ form a 1-separated collection.*

Proof. Let $S_1, S_2, ..., S_m$ be the sets in the order in which they were computed by the algorithm (as mentioned above, the order in which the hexagons of one class are being processed does not matter). We prove the claim by induction on k.

We begin with $k = 1$. As $N(S_1) = T_1$ and by construction $S_j \cap T_1 = \emptyset$ for all sets S_j it follows that the distance between S_1 and any set S_j is strictly larger than 1.

Now assume the claim is true for all sets S_k with $k \leq i-1$. From the construction of S_i it follows that $T_i = N(S_i)\setminus\bigcup_{j=1}^{j<i} T_j$. By construction it follows that $S_j \cap T_i = \emptyset$ for all sets S_j with $j > i-1$. This completes the proof. □

Proof. (of part 1 of Theorem 1): Each set $I(S_h)$ is an independent set. From Lemma 4 it follows that the sets S_h form a 1-separated collection. So no two vertices $v \in S_h$, $v' \in S_{h'}$ with $h \neq h'$ are connected by an edge. So the union $\bigcup_{h\in H} I(S_h) = I$ is an independent set. □

3.2.2. *Approximation Ratio.* We prove that the size of I is by at most a factor $1-\epsilon$ smaller than the size of a maximum independent set.

Lemma 5. *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 $\bar{r}+1 \geq 1$) it follows that v is contained in T_h and therefore covered by T_h which is a contradiction. □

Proof. (of part 2 of theorem 1): From the construction we can see that for every pair S_h, T_h it holds that $(1-\epsilon)\cdot|I(T_h)| \leq |I(S_h)|$. From Lemma 5 it follows that $\bigcup_{h\in H} T_h = V$. So the conditions of Lemma 2 are satisfied and it holds that $|I| = \left|\bigcup_{h\in H} I(S_h)\right| \geq (1-\epsilon)\cdot I_{OPT}$ □

3.2.3. *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 I 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 introduce two technical lemmas before we can prove part 3 of Theorem 1.

Lemma 6. *Algorithm 1 satisfies the following three locality properties:*

(1) *Let v_h be the coordinator vertex of a hexagon h of class k. What vertices are in T_h and S_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 $T_{h'}$ or a set $S_{h'}$ 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 a set $T_{h'}$ (or $S_{h'}$) then what other vertices are in $T_{h'}$ (or $S_{h'}$ respectively) depends only on the vertices which are at most $2c \cdot k$ 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 and S_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 $T_{h'}$ or a set S_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 $T_{h'}$ (or S_h) then what vertices are in $T_{h'}$ (or S_h respectively) 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 and S_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 $T_{h'}$ or $S_{h'}$. 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 $v' \in T_{h'}$ or $v' \in S_{h'}$. 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 $S_{h'}$ and to find out whether $v' \in T_{h'}$ or $v' \in S_{h'}$. If this is the case, we immediately know the sets $T_{h'}$ and $S_{h'}$ 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 and S_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 and S_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''.

Now let v' be a vertex. We want to find out whether v' is covered by a set $T_{h'}$ or $S_{h'}$ 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 $T_{h'}$ or $S_{h'}$ 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 $T_{h'}$ and $S_{h'}$. Then we can check if $v' \in T_{h'}$ or $v' \in S_{h'}$. If this is the case, we immediately know the sets $T_{h'}$ and $S_{h'}$ 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 7. *Let v be vertex. Whether or not v is in I depends only on the vertices which are at most $1 + 2c \cdot (b-1) + c$ hops away from v.*

Proof. First we prove the following claim: Let v be a vertex in a hexagon h with $class(h) = k$ and let v be contained in a set $T_{h'}$. Let $S_{h'}$ be the set of the 1-separated collection which is contained in $T_{h'}$. We claim that what vertices (other than v) are in $T_{h'}$ and what vertices are in $S_{h'}$ 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 6. 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 6). If there is such a set $T_{h'}$ with $v \in T_{h'}$ then $T_{h'}$, $S_{h'}$ and whether $v \in S_{h'}$ depends only on the vertices which are at most b_{k-1} hops away from v' as well (see Lemma 6).

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 sets $T_{h'}$ and $S_{h'}$ (which are then in fact T_h and S_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 $T_{h'}$ and $S_{h'}$. Altogether we conclude that for computing $T_{h'}$ and $S_{h'}$ 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 a vertex v computes whether it is in the set I it determines whether it is included in a set S_h. If this is the case, it computes the maximum independent

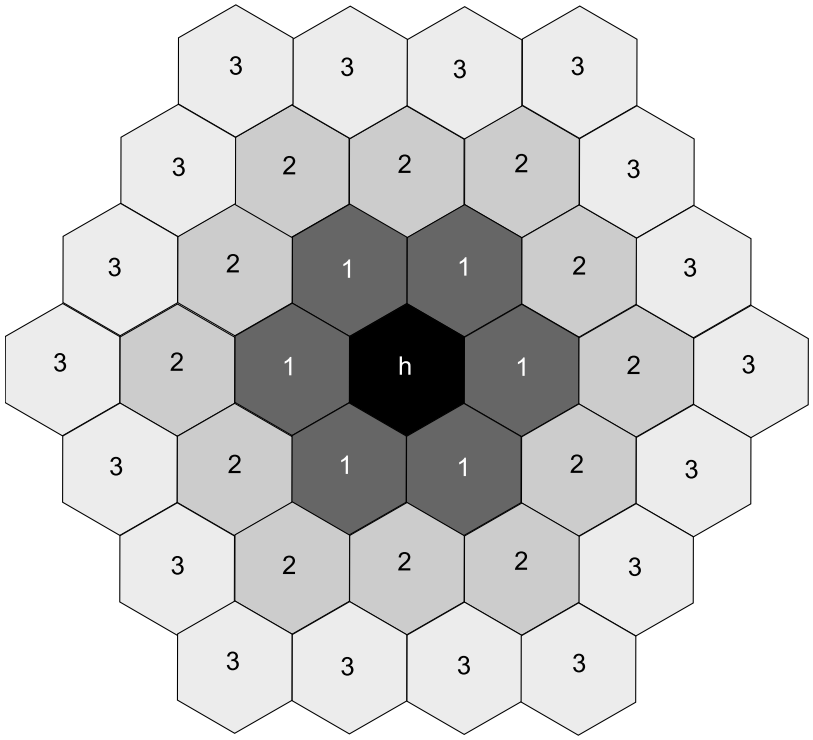

FIGURE 3. A hexagon with three circles of hexagons around it. Hexagons with the same number belong to the same circle.

set $I(S_h)$. Then v is part of the independent set if it is contained in $I(S_h)$. So we need only to explore the the vertices which are at most $1 + 2c \cdot (k-1) + c$ hops away from v. □

Lemma 8. *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 3 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.

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

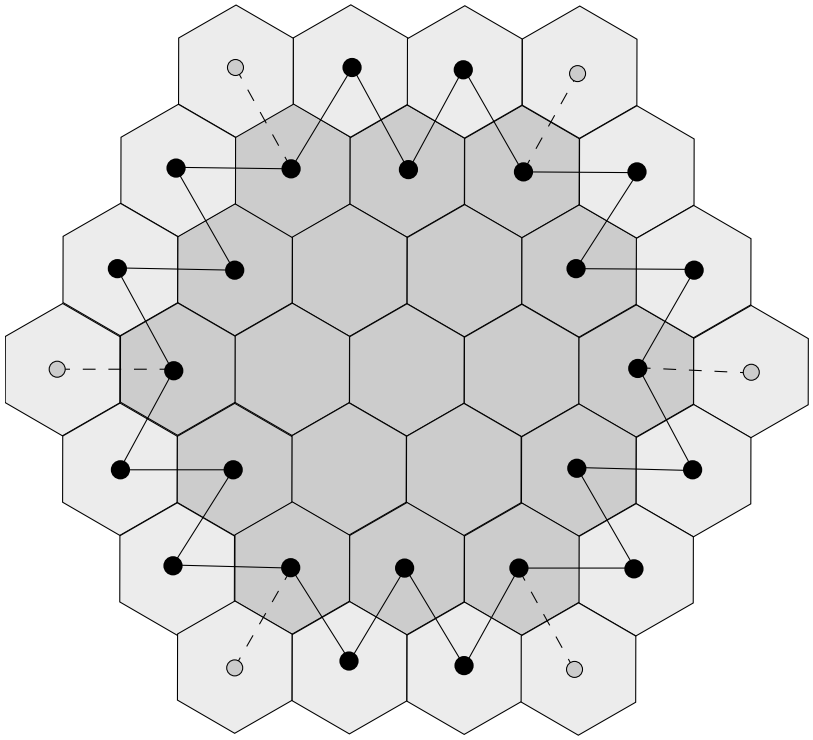

FIGURE 4. Adding a third circle around h

$v, v' \in V_B$ are connected by an edge if and only if their corresponding hexagons are adjacent to each other. Figure 4 shows this construction for adding the third circle around h. Then there will be exactly six vertices of degree one (they are the gray vertices in Figure 4 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. □

Proof. (of part 3 of Theorem 1): We want to show that whether or not a vertex v is in I 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 $c \leq \frac{1}{\epsilon^2} + 1$ i.e. $c \in O\left(\frac{1}{\epsilon^2}\right)$. By definition c is the smallest even integer such that $(2c+1)^2 < \left(\frac{1}{1-\epsilon}\right)^c$. We calculate that

$$(2c+1)^2 < \left(\frac{1}{1-\epsilon}\right)^c \quad (3)$$

$$\Leftrightarrow 2\ln(2c+1) < c \cdot \ln\left(\frac{1}{1-\epsilon}\right). \quad (4)$$

From taking derivatives we get that

$$\ln\left(\frac{1}{1-\epsilon}\right) \geq \ln(1+\epsilon) \geq \frac{1}{2}\epsilon \quad (5)$$

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

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

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

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

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 $c < \frac{1}{\epsilon^2}+1$. We observe that $\left(2\frac{1}{\epsilon_0^2}+1\right)^2 < \left(\frac{1}{1-\epsilon_0}\right)^{\frac{1}{\epsilon_0^2}} < \left(\frac{1}{1-\epsilon}\right)^{\frac{1}{\epsilon_0^2}}$ for $\epsilon > \epsilon_0$. So for all these values of ϵ we can find a value for c such that $c < \frac{1}{\epsilon_0^2}+1$. 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 7 we know that we need to explore the vertices at most $1+2c\cdot(b-1)+c$ hops away from v. From the definition of b, the above lemmas and Lemma 8 we get

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

□

3.2.4. *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 independent set. 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 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). When executing the algorithm for a single vertex v, maximum independent 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 [12] the number of vertices in a maximum independent 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 maximum independent sets $I(N^r(v'))$ for each $r \in \{0, 1, ..., c\}$. 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 $n_\alpha(v)^{O\left(1/\epsilon^4\right)}$. □

4. Vertex Cover

In this section we show how the local $1-\epsilon$ approximation algorithm for independent set presented in Section 3 can be used for locally computing a $1+\epsilon$ approximation of the minimum vertex cover problem. First we show that taking all vertices of a unit disk graph leads to a 12 approximation for vertex cover (if the graph contains edges which are to be covered by the vertex cover). Then we present our algorithm and prove its correctness.

4.1. Factor 12 Upper Bound.

We consider a connected unit disk graph with at least two vertices. We prove an upper bound of 12 for the number of all vertices in comparison with the number of vertices in a minimum vertex cover.

Theorem 2. *Let $G = (V, E)$ be a connected unit disk graph with $|V| \geq 2$ and let VC_{OPT} be a minimum vertex cover. It holds that*

$$|V| \leq 12 \cdot |VC_{OPT}|$$

Proof. We partition the vertices V into two sets V_1 and V_2 such that $V_1 \cap V_2 = \emptyset$ and $V = V_1 \cup V_2$. We prove that $|V_1| \leq 2 \cdot |V_{OPT}|$ and $|V_2| \leq 10 \cdot |VC_{OPT}|$. As $|V| = |V_1| + |V_2|$ it follows then that $|V| \leq 12 \cdot |VC_{OPT}|$.

First we define the set V_1. Let $M \subseteq E$ be a maximal matching (i.e. a matching which cannot be extended by adding another edge to M). We define $V_1 := \{u, v | \{u, v\} \in M\}$. As M is a matching it follows that $|V_1| \leq 2 \cdot |VC_{OPT}|$.

Now we define $V_2 := V \setminus V_1$. Since M is a maximal matching it follows that V_2 does not contain any adjacent vertices. Since G is a unit disk graph and therefore does not contain a $K_{1,6}$ it follows that every vertex $v \in V_1$ is adjacent to at most 5 vertices in V_2. Since $|V| \geq 2$ and G is connected it follows that $|V_1| \geq 1$ and $|V_2| \leq 5 \cdot |V_1| \leq 10 \cdot |VC_{OPT}|$. So we conclude $|V| = |V_1| + |V_2| \leq 2 \cdot |VC_{OPT}| + 10 \cdot |VC_{OPT}| \leq 12 \cdot |VC_{OPT}|$. □

Corollary 1. *For every $K_{1,m}$ free graph $G = (V, E)$ it holds that*

$$|V| \leq 2m \cdot |VC_{OPT}|$$

4.2. Local $1+\epsilon$ Approximation Algorithm for Minimum Vertex Cover.

Let $G = (V, E)$ be a connected unit disk graph. The main idea of our algorithm is the following: We compute an approximate solution I for maximum independent set. Then we define $VC := V \setminus I$ as our vertex cover.

Let $1+\epsilon$ be the desired approximation factor for minimum vertex cover. We define $\epsilon' := \min\left(\frac{1}{11}\epsilon, \frac{1}{2}\right)$. Using Algorithm 1 we locally compute a $1-\epsilon'$ approximation for maximum independent set (note that $1-\epsilon' > 0$ since $\epsilon' \leq 1/2$). Denote by I the computed set. We define the vertex cover VC by $VC := V \setminus I$. We output VC. This description is presented in Algorithm 2.

Algorithm 2: Local algorithm for finding a vertex cover in a unit disk graph

// Algorithm is executed independently by each node v;
define $\epsilon' := \min\left(\frac{1}{11}\epsilon, \frac{1}{2}\right)$;
Run Algorithm 1 with approximation ratio $1 - \epsilon'$;
// Denote by I the computed independent set;
if $v \in I$ **then** do NOT become part of the vertex cover VC **else** become part of VC

Output: Vertex cover VC

4.3. **Proof of Correctness.** We prove the correctness of Algorithm 2, its approximation factor, its locality and its processing time in Theorem 3.

Theorem 3. *Let G be a unit disk graph and let $\epsilon > 0$. Algorithm 2 has the following properties:*

(1) *The computed set VC is a vertex cover for G.*
(2) *Let VC_{OPT} be an optimal vertex cover. It holds that $|VC| \leq (1+\epsilon) \cdot |VC_{OPT}|$.*
(3) *Whether or not a vertex v is in VC 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 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.

4.3.1. *Correctness.* We prove that the set VC is a vertex cover for G.

Proof. (of part 1 of Theorem 3): The set I is an independent set for G (see Theorem 1). Assume on the contrary that there is an edge $e = (u, v)$ with $u \notin VC$ and $v \notin VC$. As $VC = V \setminus I$ it follows that $u \in I$ and $v \in I$. This is a contradiction since I is an independent set. So VC is a vertex cover for G. □

4.3.2. *Approximation Ratio.* We prove that for an optimal vertex cover VC_{OPT} it holds that $|VC| \leq (1+\epsilon) \cdot |VC_{OPT}|$.

Proof. (of part 2 of Theorem 3): Let I_{OPT} be an optimal independent set. First we discuss the case where G has no edges and therefore consist of a single vertex v (since we assume that G is connected). In this case the maximum independent set is $\{v\}$. Since $0 < (1 - \epsilon')$ we have that $0 < (1-\epsilon') \cdot |I_{OPT}| \leq |I|$ and so for the computed independent set I it holds that $I = \{v\}$. So then $V \setminus I = VC = \emptyset$ which is the optimal vertex cover.

Now assume that $|V| \geq 2$. So we can apply Theorem 2 and conclude that $|V| \leq 12 \cdot |VC_{OPT}|$. We know that $(1-\epsilon') \cdot |I_{OPT}| \leq |I|$. From $|V| - |I_{OPT}| = |VC_{OPT}|$ we have $\frac{|I_{OPT}|}{|VC_{OPT}|} = \frac{|V|}{|VC_{OPT}|} - 1$. We compute that

$$\begin{aligned}
\frac{|VC|}{|VC_{OPT}|} &= \frac{|V|-|I|}{|VC_{OPT}|} \\
&= \frac{|V|}{|VC_{OPT}|} - \frac{|I|}{|VC_{OPT}|} \\
&\leq \frac{|V|}{|VC_{OPT}|} - (1-\epsilon')\frac{|I_{OPT}|}{|VC_{OPT}|} \\
&= \frac{|V|}{|VC_{OPT}|} - (1-\epsilon')(\frac{|V|}{|VC_{OPT}|} - 1) \\
&= 1 + \epsilon'(\frac{|V|}{|VC_{OPT}|} - 1) \\
&\leq 1 + \epsilon'(12-1) \\
&= 1 + 11\epsilon' \\
&\leq 1 + \epsilon
\end{aligned}$$

So it follows that $|VC| \leq (1+\epsilon) \cdot |VC_{OPT}|$. □

4.3.3. *Locality.* Now we want to prove that Algorithm 2 is local (part 3 of Theorem 3).

Proof. (of part 3 of Theorem 3): According to Theorem 1 whether or not a vertex v belongs to I depends only on the vertices which are at most $O\left(\frac{1}{\epsilon'^6}\right) = O\left(\frac{1}{\epsilon^6}\right)$ hops away from v. Computing the set $VC = V \setminus I$ does not affect the locality of our algorithms. So whether a vertex v belongs to VC depends only on the vertex which are at most $O\left(\frac{1}{\epsilon^6}\right)$ hops away from v. □

4.3.4. *Processing Time.* We prove that the processing time of Algorithm 2 is bounded by a polynomial. Again, we measure the processing time of a vertex v in $n_\alpha(v)$, that is the number of vertices which are at most αhops away from v, where α denotes the locality distance of Algorithm 2.

Proof. The processing time of Algorithm 2 is dominated by the processing time of Algorithm 1. So it is bounded by $n_\alpha(v)^{O\left(1/\epsilon'^4\right)}$ and therefore bounded by $n_\alpha(v)^{O\left(1/\epsilon^4\right)}$. □

4.4. **Global PTAS for Vertex Cover Without Embedding of the Graph.** Combining the PTAS for independent set and the upper bound for vertex cover in unit disk graphs (the set of all vertices giving a factor 12 approximation) we derived a PTAS for the vertex cover problem. This technique can be used in every setting where a PTAS for independent set is known and there is a constant upper bound for the number of vertices in a graph in comparison with the size of a minimum vertex cover.

In particular, Nieberg et al. [13] presented a global PTAS for maximum independent set in unit disk graphs which does not need the embedding of the graph as part of the input. Using our result for the upper bound for vertex cover in unit disk graphs their algorithm can be extended to the first global PTAS for vertex cover in unit disk graphs which does not rely on the embedding of the graph. This can be done in the same way as when employing our local PTAS for independent set in order to obtain our local $1+\epsilon$ approximation algorithm for vertex cover: We

run the algorithm for independent set with an approximation ratio of $1-\epsilon'$ (with $\epsilon' = \min\left(\frac{1}{11}\epsilon, \frac{1}{2}\right)$) and output the inverse of the computed independent set. The proof that this gives an approximation ratio of $1+\epsilon$ is the same as in Theorem 3.

5. Conclusion

We presented the first local $1-\epsilon$ and $1+\epsilon$ approximation algorithms for maximum independent set and minimum vertex cover, respectively. Local algorithms cannot compute optimal solutions for these problems (note that this holds no matter if $P = NP$ or $P \neq NP$). So our algorithms give the best possible approximation ratios for these problems in our setting. Despite the locality constraint, our algorithms achieve the same approximation factors which can be guaranteed by the best known global polynomial time algorithms for the discussed problems.

We estimated the locality distances of our algorithms depending on ϵ. It is evident that further improvements are desirable. It remains open to design local PTASs for the considered problems with lower locality distances. In order to guarantee the $1+\epsilon$ approximation factor for minimum vertex cover we proved that all vertices of a unit disk graph form a factor 12 approximation for vertex cover (assuming that there are edges in the graph which are supposed to be covered). With the same method it can be proven for any $K_{1,m}$-free graph that the set of all vertices forms a factor $2m$ approximation for vertex cover. It remains open to improve this bound or to show that it is tight. If the bound for unit disk graphs could be improved, this would immediately prove a lower locality distance of our local PTAS for vertex cover. Additionally, due to the design of our algorithms a better locality distance for minimum independent set would imply an improved locality for vertex cover as well.

Also of interest would be to generalize the concepts used in our algorithms to related types of graphs like quasi-Unit Disk Graphs or graphs defined by other geometric shapes.

References

[1] R. Bar-Yehuda and S. Even. A local-ratio theorem for approximating the weighted vertex cover problem. *Annals of Discrete Mathematics*, 25:27–45, 1985.

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

[4] 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. In *Proceedings of LATIN 2008*. LNCS, 2008. to appear.

[5] I. Dinur and S. Safra. The importance of being biased. In *Proceedings of the 34th Annual ACM Symposium on Theory of Computing (STOC-02)*, pages 33–42, New York, May 19–21 2002. ACM Press.

[6] M. R. Garey and D. S. Johnson. Computers and intractability: A guide to the theory of NP-completeness. 1979.

[7] J. Håstad. Clique is hard to approximate within $n^{1-\epsilon}$. *Electronic Colloquium on Computational Complexity (ECCC)*, 4(38), 1997.

[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] 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.

[10] 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.

[11] 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.

[12] 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.

[13] T. Nieberg, J. L. Hurink, and W. Kern. A robust PTAS for maximum weight independent sets in unit disk graphs. In J. Hromkovič, M. Nagel, and B. Westfechtel, editors, *Graph-Theoretic Concepts in Computer Science: 30th International Workshop, WG 2004, Bad Honnef, Germany*, volume 3353 of *Lecture Notes in Computer Science*, pages 214–221, Berlin, 2004. Springer-Verlag.

[14] A. Wiese and E. Kranakis. Local PTAS for Dominating and Connected Dominating Set in Location Aware UDGs. *to appear*, 2007.