Parallel Algorithm for Minimum Partial Dominating Set in Unit Disk Graph

Weizhi Hong, Zhao Zhang(✉), and Yingli Ran(✉)

College of Mathematics and Computer Sciences, Zhejiang Normal University, Jinhua 321004, Zhejiang, China
hxhzz@sina.com, ranyingli@zjnu.edu.cn

Abstract. In this paper, we consider the minimum partial dominating set problem in unit disk graphs (MinPDS-UD). Given a set of points V on the plane with $|V| = n$, two points in V are adjacent in the unit disk graph if their Euclidean distance is no larger than one unit length. A point dominates itself and all its neighboring points. For an integer $k \leq n$, the goal of MinPDS-UD is to find a minimum subset of points $D \subseteq V$ such that at least k points are dominated by D. We present the first parallel algorithm for MinPDS-UD. It runs in $O(\log n)$ rounds on $O(n)$ machines, and achieves a constant approximation ratio.

Keywords: Partial dominating set · Unit disk graph · Parallel algorithm · Approximation ratio

1 Introduction

For a graph $G = (V, E)$ with vertex set V and edge set E, a vertex v is *dominated* by a vertex set $D \subseteq V$ if either $v \in D$ or v has a neighbor in D. A *dominating set* (DS) of G is a subset $D \subseteq V$ which dominates every vertex of G. The *minimum dominating set problem* (MinDS) is to compute a dominating set of the smallest size. It is widely used in many fields such as wireless networks [23]. MinDS is a well-known NP-hard problem [11] and there are extensive studies on its approximation algorithms [7]. This paper considers the partial version which requires D to dominate at least k vertices instead of all vertices. This is called the *minimum partial dominating set problem* (MinPDS).

A *unit disk graph* is a graph in which every vertex corresponds to a point on the plane and there is an edge between two vertices of the graph if the Euclidean distance between the two corresponding points is no greater than one unit. It is a widely adopted topology in a homogeneous wireless sensor network [23]. This paper studies the MinPDS problem on a unit disk graph (MinPDS-UD).

MinPDS is a special case of the *minimum partial set cover problem* (MinPSC). Given a ground set U with n elements, a collection $\mathcal{S}$ of subsets of U and an

This research work is supported in part by NSFC (11901533, U20A2068, 11771013), and ZJNSFC (LD19A010001).

D.-Z. Du et al. (Eds.): COCOA 2021, LNCS 13135, pp. 527–537, 2021.
https://doi.org/10.1007/978-3-030-92681-6_41

integer $k \leq n$, the goal of MinPSC is to find a minimum size sub-collection of $\mathcal{S}$ that covers at least k elements of U. The *minimum set cover problem* (MinSC) is a special case of MinPSC with $k = n$. A MinPDS instance can be viewed as a MinPSC instance by setting $U = V$ and $\mathcal{S} = \{S_v : v \in V\}$, where S_v is the set of neighbors of v including v itself. There are a lot of studies on parallel algorithms for MinSC [2,18,21]. For the partial version MinPSC, Ran *et al.* [21] presented a parallel algorithm with approximation ratio at most $\frac{f}{1-2\varepsilon}$ in $O(\frac{1}{\varepsilon}\log\frac{mn}{\varepsilon})$ rounds, where f is the maximum number of sets containing a common element, $0 < \varepsilon < \frac{1}{2}$ is a constant, and m is the number of the sets. This is the only paper we know on parallel algorithm for the partial cover problem with a theoretical guarantee of performance. Note that f may be as big as $\Theta(n)$, which is very large. The question is, using the speciality of the dominating set problem and the geometric structure of a unit disk graph can we design a parallel algorithm for MinPDS-UD with a constant approximation ratio?

1.1 Related Works

For MinDS, Bar-Yehuda and Moran proved that MinDS on general graphs is polynomially equivalent to MinSC [1]. Thus no polynomial time algorithm can achieve an approximation ratio within $(1-\varepsilon)\log n$ for any real number $\varepsilon > 0$ unless $P = NP$ [6,8], where n is the number of vertices. Clark *et al.* [4] proved that MinDS is NP-hard even on unit disk graphs. Extensive studies on approximation algorithms for MinDS can be found in the monograph [7]. In particular, MinDS-UD has much better approximation due to the geometric structure of unit disk graphs. Hunt *et al.* [15] presented a PTAS for MinDS-UD using partition and shifting technique, the running time is $n^{O(\frac{1}{\varepsilon^2})}$. Nieberg and Hurink [20] presented a PTAS for MinDS-UD without requiring a geometric representation of the unit disk graph, the running time is $n^{O(\frac{1}{\varepsilon}\log\frac{1}{\varepsilon})}$. For the partial verstion, Joachim *et al.* [17] presented an exact algorithm for MinPDS-UD whose running time is $O(n(16+\varepsilon)^k)$.

As will be seen from Sect. 3, the MinDS-UD problem is related with the *minimum unit disk cover problem* (MinUDC), the goal of which is to find the minimum number unit disks (that is, disks of the same size) to cover all points. The *continuous version* of MinUDC (in which unit disks can be placed anywhere on the plane) is NP-hard [9] and has been known to admit PTAS [13,14]. The *discrete version* of MinUDC is much more tricky. Das *et al.* [5] presented an 18-approximation algorithm for the discrete MinUDC problem. The ratio was improved to $(9+\varepsilon)$ by Raslimisnata *et al.* [22]. For the weighted version of MinUDC, constant approximation ratio is known [3] based on LP rounding, and Li and Jin [19] found a PTAS using a complicated guessing and dynamic programming technique.

Most of the above algorithms are sequential, which have very high running time, especially for the LP-based methods and dynamic programming methods. Although divide and conquer technique has some parallel mechanism, the above algorithms using this technique run in time at least $n^{O(\frac{1}{\varepsilon}\log\frac{1}{\varepsilon})}$, and thus are not parallel algorithms in the real sense.

There are a lot of studies on parallel algorithms for MinSC. Khuller *et al.* [18] presented a parallel $(f+\varepsilon)$-approximation algorithm in $O(f \log n \log \frac{1}{\varepsilon})$ rounds, where f is the maximum number of sets containing a common element. For the MinPSC problem, Gandhi *et al.* [10] designed a parallel algorithm with approximation ratio $\frac{f}{1-\varepsilon}$ in $(1+f\log\frac{1}{\varepsilon})(1+\log n)$ rounds. Since f might be as large as $\Theta(n)$, the number of rounds is not log-polynomial in the input size. Recently, Ran *et al.* [21] improved the result to approximation ratio at most $\frac{f}{1-2\varepsilon}$ in $O(\frac{1}{\varepsilon}\log\frac{mn}{\varepsilon})$ rounds. As we have noted before, these ratios for the general problem might be too large for a geometric setting. This motivates us to find a better parallel algorithm for MinPDS-UD.

1.2 Our Contributions

In this paper, we design a parallel algorithm for MinPDS-UD. Although MinPDS is a special case of MinPSC, compared with [21], which has approximation ratio $\frac{f}{1-2\varepsilon}$ for MinPSC in $O(\frac{1}{\varepsilon}\log\frac{mn}{\varepsilon})$ rounds, special structure indeed brings new benefit: a constant approximation ratio can be achieved in $O(\log n)$ rounds for MinDS-UD, where n is the number of vertices of the unit-disk graph. This is the first parallel constant approximation algorithm for MinPDS-UD.

In Sect. 2, we present an algorithm for MinPDS-UD by utilizing a relation between maximal independent set and dominating set, obtaining approximation ratio at most 80 in $O(\log n)$ rounds on $O(n)$ machines. Then in Sect. 3, we propose another algorithm by exploring a relation between unit disk cover and dominating set, improving the ratio to 14 in $O(\log n)$ rounds on $O(n)$ machines. A big challenge brought by the "partial" consideration is to determine which points are to be dominated. We employ a greedy idea in a parallelized manner.

2 A Constant Approximation Algorithm for MinPDS-UD

In this section, we make use of maximal independent set to design a parallel algorithm for MinPDS-UD. An *independent set* (IS) in a graph is a set of mutually nonadjacent vertices. A maximal independent set (MIS) is an IS such that adding any vertex is no longer independent. Note that an MIS is also a DS.

A unit disk graph G can also be viewed as an intersection graph of unit disks, that is, every vertex corresponds to a point on the plane and a disk of diameter 1 centered at this point. Two vertices are adjacent in G if and only if their corresponding unit disks have nonempty intersection. From such a point of view, a DS of a unit disk graph G is a set of unit disks D such that every other unit disk of $V(G) \setminus D$ has a nonempty intersection with some unit disk in D, and an IS of G is a set of disjoint unit disks.

For a MinPDS-UD instance on unit disk graph G with a geometric representation on the plane, suppose the n points are contained in a square Q. Partition Q into blocks of side-length 2×2, yielding a partition P. If block b in P contains no point, then b is called an empty block. Otherwise, b is called a nonempty block. Let $block(P)$ be the set of all nonempty blocks in

partition P, i.e. $block(P) = \{b$: there exists at least one point in b, $b \in P\}$. For a $b \in block(P)$, denote by $V_P(b)$ the set of points contained in b with respect to partition P, sort all nonempty blocks as $b_1, b_2, \cdots, b_q$ such that $|V_P(b_1)| \geq |V_P(b_2)| \geq \cdots \geq |V_P(b_q)|$. Since $|V| = n$ and $q \leq n$, there exists an index n_P such that

$$\sum_{i=1}^{n_P} |V_P(b_i)| \geq k \text{ and } \sum_{i=1}^{n_P-1} |V_P(b_i)| < k. \tag{1}$$

The algorithm is described in Algorithm 1. It returns the better one between two solutions A_{P_1} and A_{P_2} with respect to two partition P_1 and P_2. Assume, without loss of generality, that both P_1 and P_2 contain all points. For each partition P, sort all nonempty blocks b in decreasing order of $|V_P(b)|$ and find the index n_P satisfying inequality (1). Our next step is to find, for each $b \in block(P)$, a dominating set dominating all the points in b. Note that a point p in b might be dominated by a vertex v whose center is outside of b. Since two vertices are adjacent if and only if their corresponding points have distance no larger than 1, such a v must have its center in the *extended block* b' of b, which is obtained from b through extending its four boundaries by 1 (see Fig. 1). Compute a maximal independent set $\mathcal{I}(b)$ in b' to serve as a DS of b, using a parallel algorithm for MIS such as the one described in [12]. By the choice of n_P, $\bigcup_{b \in block(P)} \mathcal{I}(b)$ covers at least k points.

Algorithm 1. Algorithm for MinPDS-UD by MIS

Input: A geometric representation of a unit disk graph G.
Output: A set A of vertices dominating at least k vertices of G.

$P_1 \leftarrow$ a partition of the area containing all points into blocks of side-length 2×2
$P_2 \leftarrow$ a shifting of P_1 to north-east by 1 unit up and 1 unit right
$j \leftarrow 0$
for $j = 1$ to 2 **do**
    Sort blocks in $block(P_j)$ as $b_1, \ldots, b_q$ such that $|V_{P_j}(b_1)| \geq \cdots \geq |V_{P_j}(b_q)|$.
    $n_{P_j} \leftarrow \arg\min_{j'} \{j' : |\bigcup_{i=1}^{j'} V_{P_j}(b_i)| \geq k\}$
    for any b_i with $i \leq n_{P_j}$ in parallel **do**
        $\mathcal{I}(b_i) \leftarrow$ a maximal independent set in b_i'
    end for
    $A_{P_j} \leftarrow \bigcup_{i=1}^{n_{P_j}} \mathcal{I}(b_i)$
end for
if $|A_{P_1}| \leq |A_{P_2}|$ **then**
    $A \leftarrow A_{P_1}$
else
    $A \leftarrow A_{P_2}$
end if
return A

The next lemma evaluates the size of an MIS in an extended block (Fig. 1).

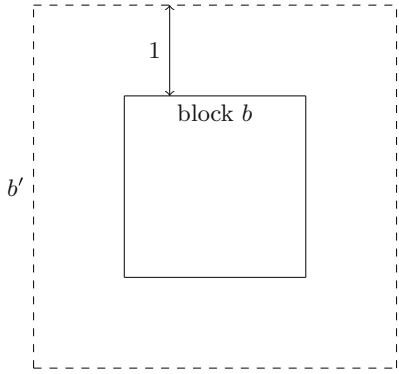

Fig. 1. Block b and its extended block b'.

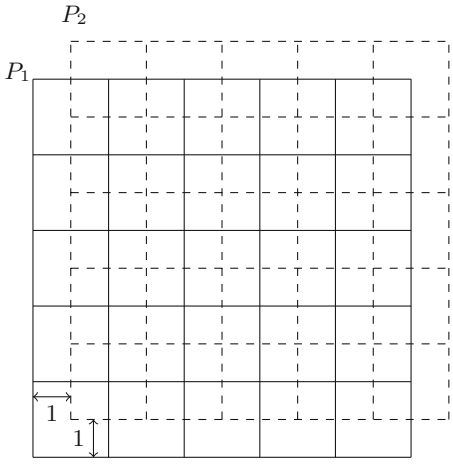

Fig. 2. Partition P_1 and partition P_2.

Lemma 1. *The size of a maximal independent set $\mathcal{I}(b)$ computed in line 8 of Algorithm 1 is at most 32.*

Proof. Let b' be the extended block of b and b'' be the block extending the boundaries of b by $\frac{3}{2}$ (see Fig. 3). Since every unit disk in $\mathcal{I}(b)$ has its center located in b', it must be completely contained in b''. Combining this observation with the fact that an independent set corresponds to a set of mutually disjoint unit disks, so $|\mathcal{I}(b)|$ is upper bound by $\lceil \frac{(2+3)^2}{\pi/4} \rceil \leq 32$.

Next we estimate the approximation ratio of Algorithm 1.

Theorem 1. *Algorithm 1 achieves approximation ratio at most 80 and runs in $O(\log n)$ rounds on $O(n)$ machines.*

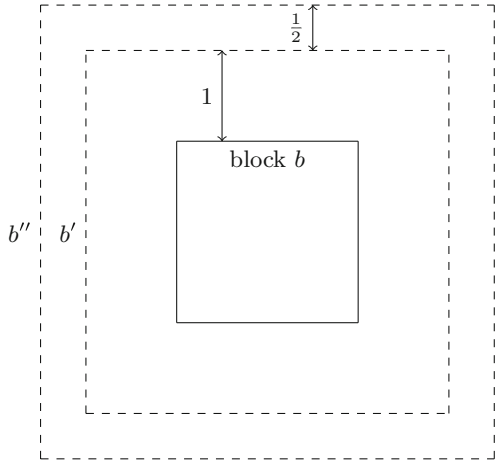

Fig. 3. Extending the boundaries of the block by 1 and $\frac{3}{2}$

Proof. Let OPT be an optimal solution and opt be the value of the OPT. For a partition P, denote by n_{o_P} the number of blocks in $block(P)$ intersecting the unit disks in OPT. For a block $b \in block(P)$, denote $OPT_P(b) = \{d \in OPT\colon d \cap b \neq \emptyset\}$ the set of unit disks of OPT intersecting b. Then

$$n_{o_P} \leq \sum_{b \in block(P)} |OPT_P(b)|. \tag{2}$$

Denote by H_P and Y_P the set of unit disks in OPT that intersect two horizontal strips and two vertical strips of P, respectively. Note that if a disk intersects more than two blocks in P, it must belong to both H_P and Y_P. Furthermore, a unit disk can intersect at most four blocks of P. Therefore,

$$\sum_{b \in block(P)} |OPT_P(b)| \leq opt + |H_P| + 2|Y_P|. \tag{3}$$

Note that a unit disk cannot belong to both H_{P_1} and H_{P_2}. Therefore, $H_{P_1} \cap H_{P_2} = \emptyset$ and thus

$$|H_{P_1}| + |H_{P_2}| \leq opt. \tag{4}$$

Similarly

$$|Y_{P_1}| + |Y_{P_2}| \leq opt. \tag{5}$$

By the greedy method in line 6 of Algorithm 1, we have

$$n_P \leq n_{o_P}. \tag{6}$$

Combining Lemma 1 with inequalities (2), (3) and (6), for any partition P_j,

$$|A_P| \leq 32 n_P \leq 32 n_{o_P} \leq 32(opt + |H_P| + 2|Y_P|). \tag{7}$$

Combining inequalities (4), (5) and (7),

$$|A_{P_1}| + |A_{P_2}| \leq 32(2opt + |H_{P_1}| + 2|Y_{P_1}| + |H_{P_2}| + 2|Y_{P_2}|) \leq 160opt.$$

Since Algorithm 1 chooses the minimum of $|A_{P_1}|$ and $|A_{P_2}|$, we have $|A| \leq 80opt$.

Next we estimate the number of rounds and the number of machines needed by Algorithm 1. Line 5 and 6 can be done in $O(\log n)$ rounds on $O(n)$ machine by a parallel sorting method a parallel selecting method in [16], Using the algorithm in [12] to compute an MIS in parallel needs $O(\log n)$ rounds on $O(n)$ machines. The other operations can be done in $O(1)$ rounds on $O(n)$ machines. So the adaptive complexity follows.

3 An Improved Approximation Algorithm for MinPDS-UD

In this section, we propose another parallel algorithm for MinPDS-UD, which improves the approximation ratio as well as the adaptive complexity.

The algorithm makes use of a relation between MinPDS-UD and a restricted version of the partial unit disk cover problem. Given a set V of n points and a set $\mathcal{D}$ of disks of the same size on the plane, the goal of the *Minimum Partial Unit Disk Cover problem* (MinPUDC) is to find a minimum number of disks to cover at least k points. The meaning of "restricted" is that the centers of those disks in $\mathcal{D}$ coincide with the points in V. In fact, for a MinPDS-UD instance on unit disk graph G, let the points corresponding to the vertices of G to be the points to be covered, as well as the centers of disks of diameter 2. Note that a disk of diameter 2 centered at point u covers point v if and only if the Euclidean distance between u and v is at most 1, that is, v is dominated by u in G. Hence we may focus on such a restricted MinPUDC problem. It should be emphasized that in the following, a unit disk refers to a disk of diameter 2, not 1.

The algorithm is described in Algorithm 2. It differs from Algorithm 1 in three aspects. First, instead of taking the better one between two solutions, it only computes a solution for one partition. Second, the size of each block is $\frac{\sqrt{2}}{2} \times \frac{\sqrt{2}}{2}$ (not 2×2). Third, a DS in block $b \in block(P)$ is no longer approximated by an MIS, but is computed by selecting an arbitrary point in b; denote the unit disk (of diameter 2) centered at this point as $D(b)$. The other steps are the same as Algorithm 1. Note that $D(b)$ covers all the points in block b (see Fig. 4 for an illustration). By the choice of n_P in line 3 of Algorithm 2, $\bigcup_{i=1}^{n_P} D(b)$ covers at least k points.

Lemma 2. *Every $D(b)$ can intersect at most 14 blocks.*

Proof. Note that a disk of diameter 2 is completely contained in a square of side-length $2\sqrt{2} \times 2\sqrt{2}$, and thus it can intersect at most $4 \times 4 = 16$ blocks. To further reduce the number, we divide block b into four cells of side-length $\frac{\sqrt{2}}{4} \times \frac{\sqrt{2}}{4}$, and denote them as $e_1, \ldots, e_4$ (see Fig. 5 for an illustration). Without loss of generality, we assume that the selected point v is located in cell e_1 (see

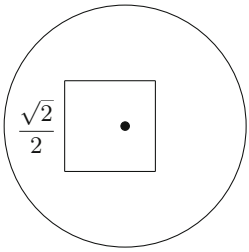

Fig. 4. A disk centered in a $\frac{\sqrt{2}}{2} \times \frac{\sqrt{2}}{2}$ block can cover this block.

Algorithm 2. Algorithm for restricted MinPUDC

Input: Area Q containing all points in V.
Output: A disks set A, which covers at least k points of V.

1: $P \leftarrow$ a partition of Q into blocks of side-length $\frac{\sqrt{2}}{2} \times \frac{\sqrt{2}}{2}$
2: Sort nonempty blocks as $b_1, \cdots, b_q$ such that $|V_P(b_1)| \geq |V_P(b_2)| \geq \cdots \geq |V_P(b_q)|$.
3: $n_P \leftarrow \arg\min_{j'}\{j' : \bigcup_{i=1}^{j'} |V_P(b_i)| \geq k\}$
4: **for** any block b_i with $i \leq n_P$ in parallel **do**
5: $\quad D(b_i) \leftarrow$ the disk centered at an arbitrarily selected point in b_i
6: **end for**
7: **return** $A \leftarrow \bigcup_{i=1}^{n_P} D(b_i)$

Fig. 6). The following two facts can be observed. First, $D(b)$ does not interest b_{11}. Otherwise, the radius of the disk is larger than 1 since the distance between any point in b_{11} and v is larger than 1, contradicting the fact that the diameter of a disk is 2. Second, $D(b)$ does not simultaneously interest both b_{14} and b_{41}, since the distance between any point in b_{41} and any point in b_{14} is larger than 2. It follows that $D(b)$ can intersect at most 14 blocks.

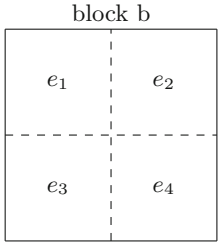

Fig. 5. Divide a block into four cells

Next we evaluate $|A|$ in Algorithm 2.

Theorem 2. *The approximation ratio of Algorithm 2 is at most 14 and Algorithm 2 runs $O(\log n)$ rounds on $O(n)$ machines.*

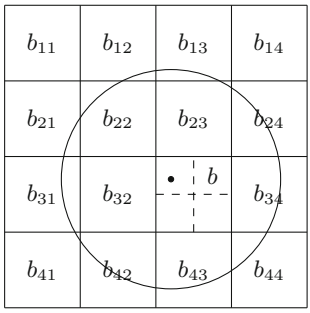

Fig. 6. $D(b)$ can intersect at most 14 blocks

Proof. Since we only selects *one* point in each of the n_P nonempty blocks,

$$|A| = n_P. \tag{8}$$

Using the same notation n_{O_P} as in the proof of Lemma 1, we also have inequality (6). Combining this with Lemma 2, we have

$$n_P \leq n_{O_P} \leq 14opt. \tag{9}$$

Combining (8) and (9), approximation ratio 14 is proved.

Since line 2 and line 3 of Algorithm 2 can be done in $O(\log n)$ rounds on $O(n)$ machines (see [16]), and line 5 can be done in constant time on each machine, the adaptive complexity follows.

4 Conclusion

This paper presented two parallel approximation algorithms for MinPDS-UD. The first one makes use of a relation between a maximal independent set and a dominating set, achieving approximation ratio 80 in $O(\log n)$ rounds on $O(n)$ machines. The second one transforms the MinPDS-UD problem into a restricted partial unit disk cover problem, achieving approximation ratio 14 in $O(\log n)$ rounds on $O(n)$ machines. These are the first parallel algorithms for MinPDS-UD achieving constant approximation ratio in log-polynomial rounds.

The first method is much more complicated, while its approximation ratio is worse. The reason might be: using inequality (2) to bridge the computed solution and an optimal solution might be too loose, it is tight only when every block intersects very few disks from the optimal solution, but the estimation for the number of vertices in an MIS only depends on the size of the block. In fact, the larger the block size is, the looser inequality (2) will be, and the larger the number of disks in a MIS will be. We believe that the first method might yield a better solution if a more delicate relation between MIS and PDS can be found. One difficulty lies in the fact that in a partial cover problem, one does not know which points should be covered.

Note that our method in Sect. 3 can be applied to the *minimum partial continuous unit disk cover problem*, the goal of which is to find the minimum number of disks with diameter 2, that *can be located anywhere* on the plane, to cover at least k points. Similar method yields a parallel algorithm with approximation ratio at most 9 in $O(\log n)$ rounds on $O(n)$ machines.

Note that our method can only be used for the cardinality case. New techniques have to be further explored for the weighted version.

References

1. Bar-Yehuda, R., Moran, S.: On approximation problems related to the independent set and vertex cover problem. Disc. Appl. Math. **9**(1), 1–10 (1984)
2. Berger, B., Rompel, J., Shor, P.W.: Efficient NC algorithms for set cover with applications to learning and geometry. J. Comput. Syst. Sci. **49**(3), 454–477 (1994)
3. Chan, T.M., Grant, E.: Weighted capacitated, priority, and geometric set cover via improved quasi-uniform sampling. In: ACM-SIAM Symposium on Discrete Algorithms, pp. 1576–1585. SIAM (2012)
4. Clark, B.N., Colbourn, C.J., Johnson, D.S.: Unit disk graphs. Disc. Math. **86**(1–3), 165–177 (1990)
5. Das, G.K., Fraser, R., Lopez-ortiz, A., Nickerson, B.G.: On the discrete unit disk cover problem. Int. J. Comput. Geometry Appl. **22**(5), 407–419 (2012)
6. Dinur, I., Steurer, D.: Analytical approach to parallel repetition. In: 46th Annual ACM Symposium on Theory of Computing, pp. 624–633, New York (2014)
7. Du, D.Z., Wang, P.J.: Connected Dominating Set: Theory and Applications. Springer, New York (2013). https://doi.org/10.1007/978-1-4614-5242-3
8. Feige, U.: A threshold of ln n for approximating set cover. In: 28th International Proceedings on ACM Symposium on Theory of Computing, pp. 314–318. ACM, New York (1996)
9. Fowler, R.J., Paterson, M.S., Tanimoto, S.L.: Optimal packing and covering in the plane are NP-complete. Inf. Process. Lett. **12**(3), 133–137 (1981)
10. Gandhi, R., Khuller, S., Srinivasan, A.: Approximation algorithms for partial covering problems. J. Algorithms **53**(1), 55–84 (2004)
11. Garey, M.R., Johnson, D.S.: Computers and Intractability: A Guide to the Theory of NP-Completeness. W. H. Freeman & Co.Subs. of Scientific American, Inc. 41 Madison Avenue, 37th Fl. New York, NY, United States (1979)
12. Ghaffari, M., Haeupler, B.: A Time-optimal randomized parallel algorithm for MIS. In: Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms, pp. 2892–2903. SIAM (2021)
13. Gonzalez, T.F.: Covering a set of points in multidimensional space. Inf. Process. Lett. **40**(4), 181–188 (1991)
14. Hochbaum, D.S., Maass, W.: Approximation schemes for covering and packing problems in image processing and VLSI. J. ACM **32**(1), 130–136 (1985)
15. Hunt, H.B., III., Marathe, M.V., Radhakrishnan, V., Ravi, S.S., Rosenkrantz, D.J., Stearns, R.E.: NC-approximation schemes for NP- and PSPACE-hard problems for geometric graphs. J. Algorithms **26**(2), 238–274 (1998)
16. JáJá J.: An Introduction to Parallel Algorithms. Addison Wesley Longman Publishing Co., Inc. 350 Bridge Pkwy suite 208 Redwood City, CA, United State (1992)

17. Joachim, K., Daniel, M., Peter, R.: Partial vs. complete domination: t-dominating set. In: International Conference on Current Trends in Theory and Practice of Computer Science, vol. 4362(1), pp. 367–376 (2007)
18. Khuller, S., Vishkin, U., Young, N.: A primal-dual parallel approximation technique applied to weighted set and vertex covers. J. Algorithms **17**(2), 280–289 (1994)
19. Li, J., Jin, Y.: A PTAS for the weighted unit disk cover problem. In: Halldórsson, M.M., Iwama, K., Kobayashi, N., Speckmann, B. (eds.) ICALP 2015. LNCS, vol. 9134, pp. 898–909. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-47672-7_73
20. Nieberg, T., Hurink, J.: A PTAS for the minimum dominating set problem in unit disk graphs. In: Erlebach, T., Persinao, G. (eds.) WAOA 2005. LNCS, vol. 3879, pp. 296–306. Springer, Heidelberg (2006). https://doi.org/10.1007/11671411_23
21. Ran, Y.L., Zhang, Y., Zhang, Z.: Parallel approximation for partial set cover. Appl. Math. Comput. **408**, 126358 (2021)
22. Rashmisnata, A., Manjanna, B., Gautam, K.D.: Unit disk cover problem in 2D. J. Discrete Algorithms **33**, 193–201 (2015)
23. Wu, J., Dai, F., Gao, M., Stojmenovic, I.: On calculating power-aware connected dominating sets for efficient routing in Ad Hoc wireless networks. J. Commun. **4**(1), 59–70 (2002)