Improved Algorithm for Maximum Independent Set on Unit Disk Graph

Ramesh K. Jallu and Guatam K. Das(✉)

Department of Mathematics Indian Institute of Technology Guwahati, Guwahati, India
{j.ramesh,gkd}@iitg.ernet.in

Abstract. In this paper, we present a 2-factor approximation algorithm for the maximum independent set problem on a unit disk graph, where the geometric representation of the graph has been given. We use dynamic programming and farthest point Voronoi diagram concept to achieve the desired approximation factor. Our algorithm runs in $O(n^2 \log n)$ time and $O(n^2)$ space, where n is the input size. We also propose a polynomial time approximation scheme (PTAS) for the same problem. Given a positive integer k, it can produce a solution of size $\frac{1}{(1+\frac{1}{k})^2}|OPT|$ in $n^{O(k)}$ time, where $|OPT|$ is the optimum size of the solution. The best known algorithm available in the literature runs in (i) $O(n^3)$ time and $O(n^2)$ space for 2-factor approximation, and (ii) $n^{O(k \log k)}$ time for PTAS [Das, G.K., De, M., Kolay, S., Nandy, S.C., Sur-Kolay, S.: Approximation algorithms for maximum independent set of a unit disk graph. Information Processing Letters 115(3), 439–446 (2015)].

Keywords: Maximum independent set · Unit disk graph · Approximation algorithm

1 Introduction

An *intersection graph* of objects is a graph, where the vertex set is the set of objects and there is an edge between two objects if their intersection is non empty. A *unit disk graph* (UDG) is an intersection graph of disks of equal radii in the plane. Given a set $C = \{C_1, C_2, \ldots, C_n\}$ of n circular disks in the plane, each having diameter 1, the corresponding UDG $G = (V, E)$ is defined as follows: each vertex $v_i \in V$ corresponds to a disk $C_i \in C$, and there is an edge between two vertices if and only if the Euclidean distance between the corresponding disk centers is at most 1.

An *independent set* of a graph $G = (V, E)$ is a set of vertices $V' \subseteq V$ such that no two vertices in V' are adjacent in G. The objective of the *independent set problem* for a given graph G is to find an independent set of maximum cardinality, which is called as *maximum independent set* (MIS) or *Largest independent set* of G.

S. Govindarajan and A. Maheshwari (Eds.): CALDAM 2016, LNCS 9602, pp. 212–223, 2016.
DOI: 10.1007/978-3-319-29221-2_18

The weighted version of the independent set problem is known as *maximum weighted independent set* (MWIS) problem, where each vertex $v \in V$ is assigned a positive weight w_v. The objective is to find an independent set of maximum total weight.

In this paper we consider the problem of finding a MIS on a given UDG, where the coordinates of the disk centers have been given. We call this problem as MIS problem on UDG. Some of the applications of MIS are in map labeling, clustering in wireless ad-hoc networks, coding theory, etc.

The remainder of the paper is organized as follows. Next section we discuss existing work available in the literature. Section 3 discusses preliminaries and introduces some notations that are necessary to understand the 2-factor approximation algorithm for MIS on UDG proposed in Sect. 4. We propose a PTAS in Sect. 5. Finally we conclude the paper in Sect. 6.

2 Related Work

The MIS problem on UDG is known to be NP-hard [7]. A simple 5-factor approximation algorithm is proposed in [11] and by taking the advantage of the structure of the given UDG, the authors proposed a heuristic algorithm which provides a performance guarantee 3. Both the algorithms do not require geometric representation (i.e., coordinates of the disk centers) of the UDG. If the geometric representation is given, the later algorithm runs in $O(n^2)$ time. For a given $(k+1)$-claw free graph $(k \geq 4)$ and for every $\epsilon > 0$, Halldórsson [8] proposed a $(\frac{k}{2}+\epsilon)$- factor approximation algorithm (that does not require the geometric representation of disks) in time $O(n^{\log_k \frac{1}{\epsilon}})$ using local improvement search technique for the MIS problem. Therefore there exists a $(\frac{5}{2}+\epsilon)$-factor for UDGs as they are 6-claw free. Most of the work in the literature assume that the geometric representation of the UDG is given, this assumption allows us to partition the plane into grids and solve each grid. Matsui [12] considered the MIS problem on UDG defined on a slab (i.e., all the disk centers lie between two parallel lines) of fixed width k, and proposed an algorithm that finds an independent set of maximum cardinality in $O(n^{4\lceil\frac{2k}{\sqrt{3}}\rceil})$ time, where n denotes the number of vertices in the UDG. The author also proposed a $(1-\frac{1}{r})$-factor approximation algorithm for the MIS problem on a UDG, which runs in $O(rn^{4\lceil\frac{2(r-1)}{\sqrt{3}}\rceil})$ time and uses $O(n^{2r})$ space, for any integer $r \geq 2$. The algorithm can also be extended to the weighted version of the MIS problem. For a given set R of rectangles of fixed size, Agarwal et al. [1] proposed a 2-factor approximation algorithm for the MIS problem that runs in $O(n \log n)$ time. The authors also proposed a PTAS that computes an independent set of rectangles of size at least $\frac{\gamma}{(1+\frac{1}{k})}$, for any $k \geq 1$, where γ is the size of a maximum independent set of R. For a given set of arbitrary rectangles of bounded aspect ratio in $\mathbb{R}^d$, Chan [2] proposed a PTAS that runs in $O(n^{\frac{1}{\epsilon^{d-1}}})$ time and space, where $0 < \epsilon \leq 1$. Chan et al. [3] considered the same problem for pseudo disks in the plane. Their algorithm produces a solution of size $(1-\epsilon)|OPT|$, where $|OPT|$ is the cardinality of the MIS. Recently Das et al. [4] proposed a 2-factor approximation algorithm for the MIS problem with time and space complexities $O(n^3)$ and $O(n^2)$ respectively. Their approach is, (i) split the region into a set of disjoint strips of unit width and compute a MIS for each non empty strip independently with the aid of dynamic programming, (ii) find the union of the solutions for odd and even strips separately and consider the one with maximum cardinality. The authors also proposed a PTAS with the aid of two level shifting strategy of Hochbaum and Maass [9]. For any given positive integer $k > 1$ the PTAS produces a solution of size $\frac{1}{(1+\frac{1}{k})^2}|OPT|$ in $O(k^4 n^{\sigma_k \log k} + n \log n)$ time and $O(n + k \log k)$ space, where OPT is an optimum solution and $\sigma_k \leq \frac{7k}{3} + 2$. For the MIS problem on UDG, van Leeuwen [10] proposed a fixed parameter tractable algorithm which runs in $O(t^2 2^{2t} n)$ time, where the parameter t is called the *thickness* of the UDG. A UDG is said to have thickness t, if each strip in the slab decomposition (of width 1) of the UDG contains at most t disk centers.

Nieberg et al. [13] proposed a PTAS for the MWIS problem on UDG for the case geometric representation is not given. Erlebach et al. [6] also proposed a PTAS for finding a MWIS in an intersection graph of arbitrary radii disks, based on dynamic programming and the shifting strategy proposed by Hochbaum and Maass. Their approach can be extended for other geometric objects such as squares, regular polygons, and rectangles which are approximately squares.

2.1 Our Contribution

In this paper we present a 2-factor approximation algorithm for the MIS problem on a given UDG under the assumption that the geometric representation of the UDG is given. Our algorithm runs in $O(n^2 \log n)$ time using $O(n^2)$ space. We also propose a polynomial time approximation scheme (PTAS) for the same problem. Given a positive integer k, it can produce a solution of size $\frac{1}{(1+\frac{1}{k})^2}|OPT|$ in $n^{O(k)}$ time, where $|OPT|$ is the optimum size of the solution. The best known algorithm available in the literature runs in (i) $O(n^3)$ time and $O(n^2)$ space for 2-factor approximation, and (ii) $n^{O(k \log k)}$ time for PTAS [4]. Hence, our 2-factor approximation algorithm as well as PTAS are much faster than the best known 2-factor approximation algorithm and PTAS for the MIS problem on unit disk graphs.

3 Preliminaries

Let $\mathcal{P}$ be the set of points (disk centers) corresponding to the given UDG and the cardinality of $\mathcal{P}$, denoted by $|\mathcal{P}|$ is n. From now on we deal with the point set $\mathcal{P}$ instead of the given UDG. We use $x(p_i), y(p_i)$ to represent the x and y coordinates respectively for the point $p_i \in \mathcal{P}$ and $d(p_i, p_j)$ to denote the Euclidean distance between two points p_i and p_j. We say that two points p_i and p_j in $\mathcal{P}$ are independent (some times we say that p_i is an independent point of p_j and vice versa in the rest of the paper) if $d(p_i, p_j) > 1$. Our objective is to find a maximum size subset $\mathcal{P}'$ of $\mathcal{P}$ such that all the points in $\mathcal{P}'$ are mutually independent. Without loss of generality, we assume that no two points in $\mathcal{P}$ have the same x-coordinate. A *horizontal strip* H is the region in the plane bounded by two horizontal parallel lines. Let $\mathcal{Q} = \{p_1, p_2, \ldots, p_m\}$ be the set of points lying in a horizontal strip H in increasing order of their x-coordinates.

Lemma 1. *[4] Let p_1, p_2, p_3, and p_4 be four points of $\mathcal{P}$ lying inside a horizontal strip H of width 1 such that $x(p_1) < x(p_2) < x(p_3) < x(p_4)$. If p_1, p_2, p_3 are pairwise independent and p_2, p_3, p_4 are also pairwise independent, then p_1 and p_4 must be independent.*

We define the set $S_{i,j}$ is as follows: (i) all the points in $S_{i,j}$ are mutually independent, (ii) $S_{i,j}$ is a maximum cardinality subset of $\mathcal{Q}$, and (iii) p_j and p_i are two right most points in $S_{i,j}$ with $j < i$. Let $n(S_{i,j})$ denote the number of points in $S_{i,j}$. We use $S_i = \{S_{i,j} \mid 1 \leq j < i\}$ to denote the collections of sets $S_{i,j}$ for fixed i. We say that two points p_u and p_v in $S_{i,j}$ are *consecutive* if $x(p_u) < x(p_v)$ and there is no other point p_w of $S_{i,j}$ such that $x(p_u) < x(p_w) < x(p_v)$. For simplicity, the set $S_{i,j}$ can be viewed as a chain $C_{i,j}$. In general, a *chain* is a series of connected line segments. In our context, the chain $C_{i,j}$ corresponding to the set $S_{i,j}$ is defined by joining consecutive points using line segments from left to right. Therefore, S_i can be viewed as a collection of chains ending at p_i (see Fig. 1). Note that these chains may or may not have a common point(s) except p_i. For a given horizontal strip H we first compute a MIS of the set $\mathcal{Q} = \{p_1, p_2, \ldots, p_m\}$ of points lying inside the strip. The basic idea of our algorithm is to extend the length of chains as long as possible while processing the points from left to right iteratively. We find a largest possible independent subset of $\{p_1, p_2, \ldots, p_i\}$ in the i^{th} iteration for $1 \leq i \leq m$. Finally, we obtain a MIS which is a longest chain (a chain of maximum length) after processing all the points in the strip H.

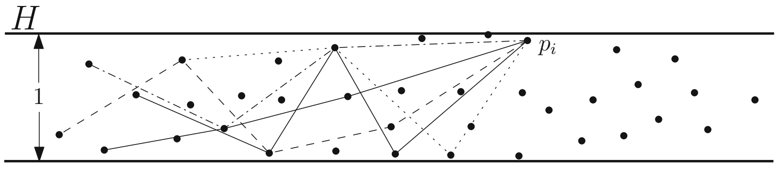

Fig. 1. Pictorial representation of the collection S_i in the form of chains (not all are drawn).

Let a variable n_i be associated with each point p_i in the strip which is used to store the size of largest independent subset of $\{p_1, p_2, \ldots, p_i\}$, i.e., $n_i = \max\{n(s_{i,j}) \mid j < i\}$. In other words, the length of the longest chain ending at p_i is n_i. Initially we set $n_i = 0$ for every point p_i in the strip, indicating that the maximum length of a chain ending at p_i is zero. The value of n_i gets updated while the point p_i is being processed. Therefore, we have a largest independent subset of $\{p_1, p_2, \ldots, p_i\}$ by the time p_i is processed. We define the following sets: $S_i^\alpha = \{p_j \in \mathcal{Q} \mid p_j \in S_{i,j}$ with $j < i$ and $|S_{i,j}| = n_i - \alpha\}$ for $\alpha = 0, 1, 2, \cdots$. These sets play crucial role in the proposed algorithm. We use $FPVD(S)$ to denote the farthest point Voronoi diagram (FPVD) [5] of a point set S.

4 2-Factor Approximation Algorithm

In this section we propose a dynamic programming based algorithm with the help of FPVD to compute a MIS of the points lying in a horizontal strip H of width 1. We partition the region containing the points in $\mathcal{P}$ into disjoint strips $H_1, H_2, \ldots, H_\nu$ of width 1 using the horizontal lines at y-coordinates $h_1, h_2, \ldots, h_\nu + 1$ such that no point in $\mathcal{P}$ lies on any horizontal line. The i^{th} strip H_i contains the points $P_i = \{p \in \mathcal{P} \mid h_i < y(p) < h_{i+1}\}$. We compute a MIS for each non empty strip separately.

Description of our algorithm to find a MIS for a given set $\{p_1, p_2, \ldots, p_m\}$ of points lying in a strip H of width 1 is as follows: let $\mu > 1$ be a predefined sufficiently large constant. For the points $p_1, p_2, \ldots, p_\mu$, we find a maximum independent subset in a naive way. We process the points one by one from left to right. For every point p_i in the strip we maintain a collection of sets $S_i = \{S_{i,j}\}$, where $j < i$. We compute these sets for $1 < i \leq \mu$ in brute force manner (because, before processing the point $p_{\mu+1}$ we should have the sets $S_{i,j}$s in hand for every $1 < i \leq \mu$ and $j < i$) and for $i > \mu$ on the fly while p_i is being processed. We define $S_{i,j} = \emptyset$ if p_i and p_j are not independent. Without loss of generality we assume that the points in the sets are stored in increasing order of their x-coordinate.

We also maintain the three sets S_i^0, S_i^1, S_i^2 (defined in the previous section) and their corresponding FPVDs separately for each point p_i in the strip. By definition, each set $S_i^\alpha (0 \leq \alpha \leq 2)$ contains the last but one points in the chains having length $n_i - \alpha$ ending at p_i. These sets and their FPVDs of every point up to p_i should be in hand before proceeding to process the point p_{i+1} in the strip.

Lemma 2. *Let p_ℓ be the farthest independent point of p_{i+1} in S_i^α (for some $0 \leq \alpha \leq 2$). If p_ℓ, p_i, and p_{i+1} are mutually independent then p_{i+1} is independent with all the points in the chain $C_{i,\ell}$.*

Proof. Let p_u be a point lying left to p_ℓ (i.e., $x(p_u) < x(p_\ell)$) in the chain $C_{i,\ell}$. By the definition of $C_{i,\ell}$, the points p_u and p_i are independent. Therefore, p_u, p_ℓ, p_i are pairwise independent. Hence by Lemma 1, p_u and p_{i+1} are independent as p_ℓ, p_i, and p_{i+1} are mutually independent. □

Lemma 3. *Let p_i and p_{i+1} be independent. Also, let p_u be the farthest independent point of p_{i+1} in S_i^α (for some $0 \leq \alpha \leq 2$). If p_u, p_i, and p_{i+1} are mutually independent then the cardinality of a MIS M' of $\{p_1, p_2, \ldots, p_{i+1}\}$ having p_u, p_i, and p_{i+1} as right most three points is greater than or equal to the cardinality of a*

MIS M'' of $\{p_1, p_2, \ldots, p_{i+1}\}$ having p_v, p_i, and p_{i+1} as right most three points, where p_v is the farthest point of p_{i+1} in S_i^β, for $\beta \geq \alpha$.

Proof. $|M'| \geq |M''|$, follows from the definition of S_i^α and S_i^β and $\beta \geq \alpha$. Now we have to prove that p_v, p_i, and p_{i+1} are mutually independent. The case $\beta = \alpha$ is trivial. Let us consider the case $\beta > \alpha$, i.e., $\beta - \alpha = 1$ or 2. By the statement of the lemma p_i and p_{i+1} are independent, and p_v and p_i are independent due to definition of S_i^β. Now consider the chain $C_{i,u}$, since p_u is the farthest point of p_{i+1} in S_i^α, the length of $C_{i,u}$ is $n_i - \alpha$. Let $\{s_1, s_2, \ldots, s_{n_i-\alpha-2}, s_{n_i-\alpha-1}(= p_u), s_{n_i-\alpha}(= p_i)\}$ be the set of points in the chain $C_{i,u}$ from left to right. Consider the following two cases:

Case A: $\beta - \alpha = 1$
Case B: $\beta - \alpha = 2$

In Case A, $s_{n_i-\alpha-2} \in S_i^\beta$ and in Case B, $s_{n_i-\alpha-3} \in S_i^\beta$.
Since $s_{n_i-\alpha-1}$ and p_{i+1} are independent and by Lemma 1, (i) $s_{n_i-\alpha-2}$ and p_{i+1}, and (ii) $s_{n_i-\alpha-3}$ and p_{i+1} are independent. Since in Case A $s_{n_i-\alpha-2} \in S_i^\beta$ and in Case B $s_{n_i-\alpha-3} \in S_i^\beta$, then p_v and p_{i+1} are independent as p_v is the farthest point of p_{i+1} in S_i^β. Thus the lemma. □

Lemma 4. *Let p_i and p_{i+1} be independent. If p_u is the farthest point of p_{i+1} in S_i^2, then p_u and p_{i+1} are independent.*

Proof. Note that $S_i^\alpha = \{p_j \in \mathcal{Q} \mid p_j \in S_{i,j}$ with $j < i$ and $|S_{i,j}| = n_i - \alpha\}$ for $\alpha = 0, 1, 2$. Consider a chain $C_{i,j}$ corresponding to $S_{i,j}$ such that $n(S_{i,j}) = n_i$. Let the members of the chain $C_{i,j}$ be $s_1, s_2, \ldots, s_{n_i-4}, s_{n_i-3}, s_{n_i-2}, s_{n_i-1}(= p_j), s_{n_i}(= p_i)$ from left to right. Now consider a chain containing the points $s_1, s_2, \ldots, s_{n_i-4}, s_{n_i-3}, s_{n_i}(= p_i)$ of length $n_i - 2$. Therefore, $s_{n_i-3} \in S_i^2$. Observe that s_{n_i-3} is independent with p_{i+1}, since $s_{n_i-3}, s_{n_i-2}, s_{n_i-1}, s_{n_i}(= p_i)$ are pairwise independent and $x(s_{n_i-3}) < x(s_{n_i-2}) < x(s_{n_i-1}) < x(s_{n_i})$. Therefore, $x(s_{n_i}) - x(s_{n_i-3}) > 1$ (by Lemma 1). Again $x(p_{i+1}) > x(p_i)$ implies s_{n_i-3} and p_{i+1} are independent. Now, since s_{n_i-3} is independent with p_{i+1} then p_u is also independent with p_{i+1} as (i) $p_u \in S_i^2$, (ii) $s_{n_i-3} \in S_i^2$, and (iii) p_u is the farthest point of p_{i+1} in S_i^2. Thus the lemma. □

Lemma 5. *Let p_i and p_{i+1} be independent. If p_u, p_v, and p_w are the farthest points of p_{i+1} in S_i^0, S_i^1, and S_i^2 respectively, then either (i) p_u, p_{i+1}, or (ii) p_v, p_{i+1}, or (iii) p_w, p_{i+1} are independent.*

Proof. Follows form Lemma 4 as p_w and p_{i+1} are independent. □

4.1 Algorithm

Here, we assume that the set of points $\{p_1, p_2, \ldots, p_i\}$ are already processed one by one from left to right. Now we describe the method of processing the point p_{i+1}. Let $S_{i+1}^0 = S_{i+1}^1 = S_{i+1}^2 = \emptyset$. Note that at the time of processing p_{i+1}, we have (i) the collection $\{S_{u,v}\}$ and $n(S_{u,v})$ such that $1 \leq v < u \leq i$, and (ii) the sets S_u^0, S_u^1, S_u^2 and their FPVDs for every $u \leq i$. The steps involved in processing the point p_{i+1} are as follows. If $d(p_i, p_{i+1}) > 1$, then we find a point $p_\ell \in S_i^0$, which is farthest from p_{i+1}. If $d(p_{i+1}, p_\ell) > 1$, then $S_{i+1,i} = \{p_{i+1}\} \cup S_{i,\ell}$ (i.e., we extend the chain ending at p_i corresponding to $S_{i,\ell}$ up to p_{i+1}) and $n(S_{i+1,i}) = n(S_{i,\ell}) + 1$. Note that we are not storing $S_{i+1,i}$ explicitly. We can use a matrix M of size $m \times m$ and store p_ℓ in the $(i+1,i)^{th}$ entry of M. If $d(p_{i+1}, p_\ell) \leq 1$, then we repeat the same process with S_i^1 and S_i^2 in order. We repeat the entire process for $p_{i-1}, p_{i-2}, \ldots, p_1$. Calculate $n_{i+1} = \max\{n(S_{i+1,j}) \mid j < i+1\}$. To find the sets S_{i+1}^0, S_{i+1}^1, and S_{i+1}^2 we repeat the above process again. If $n(S_{i+1,i}) = n_{i+1} - \alpha$ then $S_{i+1}^\alpha = S_{i+1}^\alpha \cup \{p_i\}$ for $0 \leq \alpha \leq 2$. Next, we store FPVDs of S_{i+1}^0, S_{i+1}^1, and S_{i+1}^2 to process the remaining points in the horizontal strip H. The pseudo code of the algorithm for processing the point p_{i+1} is given in Algorithm 1. In the algorithm flag variables $flag1$ and $flag2$ are used to handle the cases $S_i^\alpha = \emptyset$ for any $\alpha = 0, 1, 2$ and there is no independent point left to p_{i+1} respectively.

4.2 Correctness of the Algorithm

Let the current point being processed is p_{i+1}. If $d(p_{i+1}, p_i) > 1$ we check for the independence of p_{i+1} with the farthest point in S_i^0, S_i^1, and S_i^2 in order. The farthest independent point, say p_ℓ, encountered first is considered to be in the solution. The existence of p_ℓ is guaranteed by Lemma 5. By Lemma 2, p_{i+1} is independent with all the points in the chain $C_{i,\ell}$. Therefore, the points in the chain together with p_{i+1} forms an independent set of $\{p_1, p_2, \ldots, p_{i+1}\}$ and that is the possible maximum independent set having p_ℓ, p_i, and p_{i+1} as right most three points (see Lemma 3). Hence, we can safely extend the chain $C_{i,\ell}$ up to p_{i+1}. We considered all the points $p_i, p_{i-1}, \ldots, p_1$ (see line number 3 in Algorithm 1) and hence we are considering all possible chains ending at p_{i+1}.

Lemma 6. *Algorithm 1 processes the point p_{i+1} correctly in $O(i \log i)$ time and uses $O(m^2)$ space.*

Proof. Correctness of Algorithm 1 follows from the discussion in SubSect. 4.2. The worst case time complexity of lines 4–21 is $O(\log i)$ due to planar point location in line number 9. Therefore, time complexity of lines 3–22 is $O(i \log i)$. Again, time complexity of lines 27–33 is $O(i \log i)$. Computing FPVDs in line number 34 can be done in $O(i \log i)$. Thus the total time complexity of Algorithm 1 is $O(i \log i)$.

The space complexity follows from (i) size of the matrix M, (ii) collection $\{S_{i,j}\}$, (iii) counters $n(S_{i,j})$, (iv) sets S_i^0, S_i^1, S_i^2, and (v) storing FPVDs of the sets S_i^0, S_i^1, S_i^2 for $1 \leq i \leq m$. □

We now describe the algorithm for computing a MIS for the set of points $\{p_1, p_2, \ldots, p_m\}$ within a strip H of width 1. For a given predefined constant μ we execute Algorithm 1 for each point $p_{\mu+1}, p_{\mu+2}, \ldots, p_m$ in the strip and report the largest set $S_{i,j}$ for $1 \leq j < i \leq m$. Hence the size of a MIS for a given strip of width 1 is equal to $\max\{n_1, n_2, \ldots, n_m\}$. The pseudo code of the algorithm is available in Algorithm 2.

Algorithm 1. Processing the point p_{i+1}

```
Input: (i) $S_{u,v}$ (in the form of matrix $M$) and $n(S_{u,v})$ for $1 \leq v < u \leq i$, and (ii)
    $S_u^0, S_u^1$, and $S_u^2$ and their FPVDs for $u \leq i$.
Output: (i) $S_{i+1,j}$ (in the form of matrix $M$) and $n(S_{i+1,j})$ for $j < i+1$, and (ii)
    $S_{i+1}^0, S_{i+1}^1, S_{i+1}^2$ and their FPVDs.
 1: Let $M$ be a matrix of size $m \times m$ and $M[i,j] \leftarrow \phi$ for $1 \leq i,j \leq m$
 2: $flag1 = 0, flag2 = 0$
 3: for ($w = i, i-1, \ldots, 1$) do
 4:     if ($d(p_w, p_{i+1}) > 1$) then
 5:         $flag2 = 1$
 6:         for ($\alpha = 0, 1, 2$) do
 7:             if ($S_w^\alpha \neq \emptyset$) then
 8:                 $flag1 = 1$
 9:                 Find the farthest point $p_\ell$ of $p_{i+1}$ in $FPVD(S_w^\alpha)$ using planar point
    location algorithm [14].
10:                 if ($d(p_\ell, p_{i+1}) > 1$) then
11:                     $M[i+1, w] \leftarrow p_\ell$
12:                     $n(S_{i+1,w}) = n(S_{w,\ell}) + 1$
13:                     break /* break the for loop for ($\alpha = 0, 1, 2$) */
14:                 end if
15:             end if
16:         end for
17:         if ($flag1 = 0$) then
18:             $M[i+1, w] \leftarrow p_w$
19:             $n(S_{i+1,w}) = 2$
20:         end if
21:     end if
22: end for
23: $S_{i+1}^0 \leftarrow \emptyset, S_{i+1}^1 \leftarrow \emptyset, S_{i+1}^2 \leftarrow \emptyset$
24: if $flag2 = 0$ then
25:     $n_{i+1} = 1$
26: else
27:     $n_{i+1} = \max\{n(S_{i+1,j}) \mid j < i+1\}$
28:     Repeat line numbers 3 - 22 by replacing lines 11 - 12 by lines 29 - 33.
29:     for ($\alpha = 0, 1, 2$) do
30:         if ($n(S_{i+1,w}) = n_{i+1} - \alpha$) then
31:             $S_{i+1}^\alpha = S_{i+1}^\alpha \cup \{p_w\}$
32:         end if
33:     end for
34:     Compute and store $FPVD(S_{i+1}^0), FPVD(S_{i+1}^1)$, and $FPVD(S_{i+1}^2)$.
35: end if
```

Theorem 1. *Algorithm 2 correctly computes a MIS for the set* $Q = \{p_1, p_2, \ldots, p_m\}$ *inside a strip* H *of width 1 in* $O(m^2 \log m)$ *time using* $O(m^2)$ *space.*

Algorithm 2. *MIS_STRIP*

```
Input: The set $\mathcal{Q} = \{p_1, p_2, \ldots, p_m\}$ of $m$ points lying in the strip $H$ of width 1 and a constant $\mu$.
Output: A maximum cardinality subset $\mathcal{Q}'$ of $\mathcal{Q}$ such that the points in $\mathcal{Q}'$ are mutually independent.
1: For the points $\{p_1, p_2, \ldots, p_\mu\}$ compute $\{S_{i,j}\} (j < i)$ in brute force manner.
2: for ($i = \mu + 1$ to $m$) do
3:     Process the point $p_i$ by calling Algorithm 1.
4: end for
5: Return a set with maximum cardinality among $\{S_{i,j}\}$ for $1 \le j < i \le m$.
```

Proof. Correctness of the algorithm follows from Lemma 6. Time complexity of Algorithm 2 is $\sum_{i=1}^{m} O(i \log i)$ (see **for** loop in line number 2 in Algorithm 2, where it calls Algorithm 1 $O(m)$ times). Therefore, total time complexity of Algorithm 2 is $O(m^2 \log m)$ in worst case.

Space complexity of Algorithm 2 follows from Lemma 6 as we can reuse the matrix M for every call to Algorithm 1. □

Now, we describe an algorithm to find a MIS for the point set $\mathcal{P}$. Let $MIS_1, MIS_2, \ldots, MIS_\nu$ be the largest possible independent sets corresponding to the points in $\mathcal{P} \cap H_1, \mathcal{P} \cap H_2, \ldots, \mathcal{P} \cap H_\nu$ respectively. We execute Algorithm 2 for every strip H_i for $1 \le i \le \nu$. Let MIS_{odd} and MIS_{even} be the union of maximum independent sets in odd and even strips respectively. We report MIS_{odd} if $|MIS_{odd}| \ge |MIS_{even}|$, otherwise we report MIS_{even}. The pseudo code of the algorithm is given in Algorithm 3

Theorem 2. *Given a set $\mathcal{P}$ of n points (disk centers) corresponding to a given UDG, a subset of at least $\frac{1}{2}|OPT|$ mutually independent points (disks) can be computed in $O(n^2 \log n)$ time and using $O(n^2)$ space using Algorithm 3, where $|OPT|$ is the cardinality of a largest independent set for the point set $\mathcal{P}$.*

Proof. Let χ be the solution obtained by Algorithm 3. Observe that both MIS_{odd} and MIS_{even} are independent as all strips are of width 1 unit and two points in $\mathcal{P}$ are independent if the Euclidean distance between them is greater than 1. Also, observe that the points in any two even strips (resp. odd) are independent. We have to prove that $|\chi| > \frac{1}{2}|OPT|$, where OPT is a MIS of $\mathcal{P}$. Since MIS_{odd} and MIS_{even} are union of solutions in odd and even strips respectively, hence,

$$|MIS_{odd}| + |MIS_{even}| \ge |OPT| \tag{1}$$

$$|\chi| + |\chi| \ge |MIS_{odd}| + |MIS_{even}| \tag{2}$$

From inequalities 1 and 2, $|\chi| \ge \frac{1}{2}|OPT|$.

The time and space complexities follow as we can execute Algorithm 3 for every strip independently. Thus the theorem. □

Algorithm 3. $MIS\_\mathcal{P}$

```
Input: The set $\mathcal{P}$ of $n$ points and strips $H_1, H_2, \ldots, H_\nu$.
Output: An independent subset of $\mathcal{P}$.
 1: Compute $MIS_1, MIS_2, \ldots, MIS_\nu$ for the points lying in strips $H_1, H_2, \ldots, H_\nu$ by
    calling Algorithm 2 for each strip separately.
 2: if ($\nu = 2u$) then
 3:     $MIS_{odd} = \bigcup_{i=0}^{u-1} MIS_{2i+1}$ and $MIS_{even} = \bigcup_{i=1}^{u} MIS_{2i}$
 4: else
 5:     if ($\nu = 2u + 1$) then
 6:         $MIS_{odd} = \bigcup_{i=0}^{u} MIS_{2i+1}$ and $MIS_{even} = \bigcup_{i=1}^{u} MIS_{2i}$
 7:     end if
 8: end if
 9: if $|MIS_{odd}| \geq |MIS_{even}|$ then
10:     return $MIS_{odd}$
11: else
12:     return $MIS_{even}$
13: end if
```

5 Polynomial Time Approximation Scheme

We design a polynomial time approximation scheme (PTAS) for the maximum independent set problem on a given UDG, where the geometric representation of the graph has been given i.e., the center of the unit disks are given. We assume that $\mathcal{P}$ be the set of centers of the unit disks associated with the UDG. Also assume that $\mathcal{R}$ be an enclosing rectangle of the point set $\mathcal{P}$. To design a PTAS we use two level shifting strategy, proposed by Hauchbaum and Maass [9]. In the first level of shifting strategy we execute $k + 1$ iterations as follows: in the i-th iteration ($0 \leq i \leq k$), we partition the region $\mathcal{R}$ into disjoint vertical slabs such that (i) the first slab is of width i starting from left, (ii) width of each even slab is 1, and (iii) width of other slab is k (note that width of last slab may be less than k). Therefore, solution of different slabs of width k are non-intersecting [1].

In an iteration of the first level, we consider only those vertical slabs containing at least one point in $\mathcal{P}$, and compute maximum independent set by applying second level shifting strategy by considering horizontal partition of each vertical slab, add up the solutions of all slabs to get the solution of that iteration. The iteration producing maximum size solution is reported.

Lemma 7. *[4] If n_k is the maximum number of mutually non-overlapping unit disks whose centers lie in a strip of width $k > 1$ and intersected by a vertical line ℓ, then $n_k \leq \frac{7k}{3} + 2$.*

5.1 Computing MIS for Unit Disks Centered in a $k \times K$ Square

Let $Q \subseteq \mathcal{P}$ be the set of points inside a cell χ of size $k \times k$. Consider a vertical line ℓ_v and a horizontal line ℓ_h that partition χ into four sub-cells each of size

$\frac{k}{2} \times \frac{k}{2}$. Let $Q(\ell_v, \ell_h) \subseteq Q$ be the set of points whose distance from ℓ_v or ℓ_h is at most $\frac{1}{2}$, and $Q_1, Q_2, Q_3, Q_4 \subseteq Q$ be the set of points in the four quadrants whose distance from ℓ_v and ℓ_h is greater than $\frac{1}{2}$. To compute a MIS for the set of points in Q, we use the following divide and conquer technique.

Consider all possible subsets $Q' \subseteq Q(\ell_v, \ell_h)$ of size at most $2 \times n_k$, where $n_k = \frac{7k}{3} + 2$ (since $2 \times n_k$ is the maximum possible size of the point set in $Q(\ell_v, \ell_h)$ that can appear in an optimal solution due to Lemma 7). For each of Q', we do the following in each quadrant: delete all the points in Q_i ($i = 1, 2, 3, 4$) which are not independent with Q'. Let $Q'_i \subseteq Q_i$ be the remaining set of points. Now compute the optimum solution for Q'_i recursively using the same procedure. If $T(m, k)$ is the time complexity for finding MIS in χ, then $T(m, k) = 4 * T(m, \frac{k}{2}) \times m^{2n_k} = m^{O(k)}$. Thus, we have the following result:

Theorem 3. *Given a set P of n points in the plane and an integer $k > 1$, the proposed algorithm computes an independent set of size at least $\frac{1}{(1+\frac{1}{k})^2}|OPT|$ in $n^{O(k)}$ time, where $|OPT|$ is the optimum size of the solution.*

6 Conclusion

In this paper we proposed a 2-factor approximation algorithm for the MIS problem on UDG, where the geometric representation of the UDG is given. Our algorithm runs in $O(n^2 \log n)$ time and $O(n^2)$ space, outperforming the existing algorithms in the literature with respect to time complexity by a factor of $\frac{n}{\log n}$ [4]. We also proposed a PTAS for the same problem. The running time of our proposed PTAS is $n^{O(k)}$. The previous best known PTAS runs in $n^{O(k \log k)}$ time [4].

References

1. Agarwal, P., van Kreveld, M., Suri, S.: Label placement by maximum independent set in rectangles. Comput. Geom. **11**(3), 209–218 (1998)
2. Chan, T.M.: Polynomial-time approximation schemes for packing and piercing fat objects. J. Algorithms **46**(2), 178–189 (2003)
3. Chan, T.M., Har-Peled, S.: Approximation algorithms for maximum independent set of pseudo-disks. Discrete Comput. Geom. **48**(2), 373–392 (2012)
4. Das, G.K., De, M., Kolay, S., Nandy, S.C., Sur-Kolay, S.: Approximation algorithms for maximum independent set of a unit disk graph. Inf. Process. Lett. **115**(3), 439–446 (2015)
5. De Berg, M., Van Kreveld, M., Overmars, M., Schwarzkopf, O.C.: Computational Geometry. Springer, Heidelberg (2000)
6. Erlebach, T., Jansen, K., Seidel, E.: Polynomial-time approximation schemes for geometric intersection graphs. SIAM J. Comput. **34**(6), 1302–1323 (2005)
7. Garey, M., Johnson, D.: Computers and intractability: a guide to the theory of NP-completeness. Freeman, New York (1979)
8. Halldórsson, M.M.: Approximating discrete collections via local improvements. In: Proceedings of the Sixth Annual ACM-SIAM Symposium on Discrete Algorithms, pp. 160–169. Society for Industrial and Applied Mathematics (1995)

9. Hochbaum, D.S., Maass, W.: Approximation schemes for covering and packing problems in image processing and VLSI. J. ACM (JACM) **32**(1), 130–136 (1985)
10. van Leeuwen, E.J.: Approximation algorithms for unit disk graphs. In: Kratsch, D. (ed.) WG 2005. LNCS, vol. 3787, pp. 351–361. Springer, Heidelberg (2005)
11. Marathe, M.V., Breu, H., Hunt, H.B., Ravi, S.S., Rosenkrantz, D.J.: Simple heuristics for unit disk graphs. Networks **25**(2), 59–68 (1995)
12. Matsui, T.: Approximation algorithms for maximum independent set problems and fractional coloring problems on unit disk graphs. In: Akiyama, J., Kano, M., Urabe, M. (eds.) JCDCG 1998. LNCS, vol. 1763, pp. 194–200. Springer, Heidelberg (2000)
13. Nieberg, T., Hurink, J.L., Kern, W.: A robust PTAS for maximum weight independent sets in unit disk graphs. In: Hromkovič, J., Nagl, M., Westfechtel, B. (eds.) WG 2004. LNCS, vol. 3353, pp. 214–221. Springer, Heidelberg (2004)
14. Preparata, F.P., Shamos, M.: Computational Geometry: An Introduction. Springer Science & Business Media, New York (2012)