Faster Approximation for Maximum Independent Set on Unit Disk Graph

Subhas C. Nandy, Supantha Pandit*, Sasanka Roy

*Corresponding author
Email addresses: nandysc@isical.ac.in (Subhas C. Nandy), pantha.pandit@gmail.com (Supantha Pandit), sasanka@isical.ac.in (Sasanka Roy)

Abstract

Maximum independent set from a given set $D$ of unit disks intersecting a horizontal line can be solved in $O(n^2)$ time and $O(n^2)$ space. As a corollary, we design a factor 2 approximation algorithm for the maximum independent set problem on unit disk graph which takes both time and space of $O(n^2)$. The best known factor 2 approximation algorithm for this problem runs in $O(n^2 \log n)$ time and takes $O(n^2)$ space [1, 2].

Keywords: Maximum independent set, Unit disk graph, Approximation algorithm.

1. Introduction

Intersection graphs of geometric objects have used to model several problems that arise in real scenarios [3]. Two important applications of these graphs are frequency assignment in cellular networks [4, 5] and map labeling [6]. If the geometric objects are disks then the corresponding intersection graph $G(V, E)$ is called the disk graph. Here the vertex set $V$ corresponds to a given set of disks in the plane, and there is an edge between two vertices in $V$ iff the corresponding two disks intersect.

A unit disk graph is an intersection graph where each disk is of diameter 1. Let $G(V, E)$ be a given graph. A set $V' \subseteq V$ is said to be an independent set of $G$ if no two vertices in $V'$ are connected by an edge in $G$. In the maximum
**independent set (MIS)** the goal is to find an independent set $V'$ which has the maximum cardinality. In this paper, we consider the following problem.

**Maximum Independent Set on Unit Disk Graph (MISUDG):**
Given a unit disk graph $G(V, E)$, find an independent set of $G$ whose cardinality is maximum.

To provide an approximation algorithm for MISUDG, we consider the following problem.

**MISUDG-L:** Given a set $D_i$ of $n_i$ unit disks that are intersected by horizontal line $L_i$, find a subset $D' \subseteq D_i$ of maximum cardinality such that no two disks in $D'$ have a common intersection point.

**Related Work:** The MISUDG problem is known to be NP-complete [7, 8, 9]. In Table 1, we demonstrate a comparison study of the progress on MISUDG.

| Reference          | Factor | Time          | Space        |
|--------------------|--------|---------------|--------------|
| Marathe et al. [10]| 3      | $O(n^2)$      | $O(n)$       |
| Das et al. [11]    | 2      | $O(n^3)$      | $O(n^2)$     |
| Jallu and Das [1]  | 2      | $O(n^2 \log n)$ | $O(n^2)$     |
| Das et al. [2]     | 2.16   | $O(n \log^2 n)$ | $O(n \log n)$ |
| Theorem 5          | 2      | $O(n^2)$      | $O(n^2)$     |

| Table 1: Comparison table |

Matsui [12] consider the MISUDG problem. If the disk centers are located inside a strip of fixed height $k$, then this problem can be solved in $O(n^{4\frac{4k}{7k^3}})$ time. Further, for any integer $r \geq 2$, Matsui [12] provided a $(1 - \frac{1}{r})$ factor approximation algorithm for the same problem which takes $O(n^{4\frac{2k-1}{3k^3+1}})$ time and $O(n^{2r})$ space. Das et al. [11], also designed a PTAS for MISUDG problem by using the shifting strategy of Hochbaum and Maass [13]. For a given positive integer $k > 1$, they gave a $(1 + \frac{1}{k})^2$ factor approximation algorithm which runs in $O(k^4n^\sigma_k \log k + n \log n)$ time and $O(n + k \log k)$ space, where $\sigma_k \leq \frac{7k}{8} + 2$.

Recently, Jallu and Das [1], improved the running time of the same problem to
\(O(n^{O(k)})\) by keeping the approximation factor same. A fixed parameter tractable
algorithm for the MISUDG problem was proposed by van Leeuwen [14]. The
running time of that algorithm is \(O(t^22^{2t}n)\), where the parameter \(t\) represents
the thickness of the UDG.

**Our Contributions:**

- We design an exact algorithm for MISUDG-L problem which runs in \(O(n^2)\) time using \(O(n^2)\) space.
- We design a factor 2 approximation algorithm for MISUDG problem which
takes both \(O(n^2)\) time and space. It is an improvement over the best
known result on this problem proposed by Jallu et al. [1]. They gave
a factor 2 approximation algorithm for this problem where the time and
space complexities are \(O(n^2 \log n)\) and \(O(n^2)\) respectively.

**Notations and Definitions:** Let \(D = \{d_1, d_2, \ldots, d_n\}\) be a set of \(n\) unit disks
in the plane. The center of the disk \(d_i \in D\) is \(c_i\). The \(x\)-coordinate of \(c_i\) is
\(x(c_i)\). For a given set \(S\) of disks, \(|S|\) is the cardinality of \(S\). The line segment
connecting two points \(s\) and \(t\) is denoted by \(st\).

**2. \(O(n^2)\) time exact algorithm for MISUDG-L problem**

In this section, we design an exact dynamic programming based algorithm for
MISUDG-L problem. Let \(D_i = \{d_{i1}, d_{i2}, \ldots, d_{in_i}\}\) be a set of \(n_i\) unit disks intersecting
a horizontal line \(L_i\). We partition the set \(D_i\) into two sets \(D^{a}_i\) and \(D^{b}_i\):
where \(D^{a}_i\) is the set of all disks in \(D_i\) whose centers are above the horizontal line
\(L_i\) and \(D^{b}_i\) is the set of all disks in \(D_i\) whose centers are below the horizontal
line \(L_i\). To design the dynamic programming algorithm, we need the following
two lemmas.

**Lemma 1.** Let \(d_1, d_2, d_3 \in D^{a}_i\) be three disks with centers \(c_1, c_2,\) and \(c_3\) respectively. Assume that \(x(c_1) < x(c_2) < x(c_3)\). Now if \(d_1, d_2\) and \(d_2, d_3\) are
non-intersecting, then \(d_1, d_3\) are non-intersecting.

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.
Proof. Suppose on the contrary, we assume that $d_1$ and $d_3$ are intersecting. Then clearly the line segment $c_1c_3$ must be fully covered by $d_1$ and $d_3$. Since $x(c_1) < x(c_2) < x(c_3)$, $c_2$ cannot be above $c_1c_3$. Otherwise, it must intersect $c_1c_3$ and hence intersect either $d_1$ or $d_2$. Further, the perpendicular distance between the horizontal line $L_i$ and any point on $c_1c_3$ is at most 1. Then, if $c_2$ is below $c_1c_3$, it must intersect $c_1c_3$ as the centers are above the horizontal line $L_i$. Therefore, we have arrived at a contradiction that either $d_1$, $d_2$ are intersecting or $d_2$, $d_3$ are intersecting.

Lemma 2. Let $d_1, d_2 \in D^b_i$ and $d_3 \in D^a_i$ be three disks with centers $c_1, c_2,$ and $c_3$ respectively. Assume that $x(c_1) < x(c_2) < x(c_3)$. Now if $d_1, d_2$ and $d_2, d_3$ are non-intersecting, then $d_1, d_3$ are non-intersecting.

Proof. Suppose on the contrary, we assume that $d_1$ and $d_3$ are intersecting. Then clearly $c_1c_3$ is at most 1. Also by the assumption, both $c_1c_2$ and $c_2c_3$ are greater than 1. Let $V_L$ be a vertical line through $c_2$ (see Figure 2). The two lines $L_i$ and $V_L$ intersect at a point $O$ and partition the space into four quadrants: ‘++’, ‘+-’, ‘-+’, and ‘--’. The point $c_3$ is in ‘++’, whereas $c_1$ is in ‘--’. Now consider an unit disk $d^*$ whose center coincides with $O$. Note that, all disks in $D_i$ intersect the line $L_i$. Hence the disk $d_2$ contains the point $O$. Further, since $c_2$ and $c_3$ are non-intersecting, $c_3$ must be outside $d^*$.

Take the segment $c_2c_3$ which intersect $d^*$ at $c'_3$. Further, extend the segment $c_2c_3$ in the direction of $c_2$ such that it intersects another point $c'_2$ on $d^*$. Consider the segment $c_2c'_3$. Now by an easy observation, we say that, the voronoi partition line ($VPL$) of $c'_2$ and $c'_3$ passes through $O$ and intersects the two quadrants ‘+-’ and ‘-+’. Again, consider the segment $c_2c'_3$. Since $c_2$ is on the line through $c'_2$.
and \( c_3' \), the slope of the VPL of \( c_2 \) and \( c_3' \) must be the same as that of \( c_2' \) and \( c_3 \). Further, this VPL is to the right of the VPL of \( c_2' \) and \( c_3 \) and contains the whole ‘−−’ quadrant to its left. Due to similar argument, the VPL of \( c_2, c_3 \) contains the whole ‘−−’ quadrant to its left. Since \( c_1 \) and \( c_2 \) are in ‘−−’ quadrant, clearly the point \( c_1 \) is closer to \( c_2 \) than \( c_3 \). Therefore, \( c_1 c_3 \) is greater than 1, since \( c_1 c_2 \) is greater than 1. This leads to the contradiction that \( c_1 c_3 \) is at most 1.

We now describe the algorithm as follows. Let \( \{d_{a1}, d_{a2}, \ldots, d_{an}^\circ\} \) be the set of disks in \( D_a^\circ \) sorted according to their increasing x-coordinates. Similarly, let \( \{d_{b1}, d_{b2}, \ldots, d_{bn}^\circ\} \) be the set of disks in \( D_b^\circ \) sorted according to their increasing x-coordinates. We add two new disks \( d_{a0}^\circ \) and \( d_{b0}^\circ \) which satisfies the following, (i) \( d_{a0}^\circ \) is to the left of \( d_{a1} \) and \( d_{b0}^\circ \) is to the left of \( d_{b1} \), (ii) both \( d_{a0}^\circ \) and \( d_{b0}^\circ \) are independent with the disks in \( D_i \), and (iii) \( d_{a0}^\circ \) and \( d_{b0}^\circ \) do not intersect each other. For any disk \( d \in D_i \) (\( d \neq \{d_{a0}^\circ, d_{b0}^\circ\} \)), define \( R_i^a(d) \) (resp. \( R_i^b(d) \)) be the rightmost disk in \( D_i^a \) (resp. \( D_i^b \)) which is independent with \( d \) and whose center is to the left of the center of \( d \).

We define a subproblem \( S(k, \ell) \), for \( 0 \leq k \leq n_1 \) and \( 0 \leq \ell \leq n_2 \), to be the set of all disks in \( D_i^a \) which are to the left of the disk \( d_k^a \in D_i^a \) and set of all disks in \( D_i^b \) which are to the left of the disk \( d_\ell^b \in D_i^b \). Let \( I(k, \ell) \) be an optimal set of independent unit disks in \( S(k, \ell) \), and let \( V(k, \ell) \) be the value of this solution.

**Lemma 3.** Let \( D_i^a(k) = \{d_{k1}^a, d_{k2}^a, \ldots, d_{kn}^a\} \) be a set of \( k \) leftmost disks in \( D_i^a \) and \( D_i^b(\ell) = \{d_{\ell1}^b, d_{\ell2}^b, \ldots, d_{\ell\ell}^b\} \) be the set of \( \ell \) leftmost disks in \( D_i^b \). Now,
A. if $x(c_k^a) > x(c_k^b)$, then

- If $d_k^a \in I(k, \ell)$, then $V(k, \ell) = V(RI^a(d_k^a), RI^b(d_k^a)) + 1$
- If $d_k^b \notin I(k, \ell)$, then $V(k, \ell) = V(k - 1, \ell)$

B. if $x(c_k^a) < x(c_k^b)$, then

- If $d_k^b \notin I(k, \ell)$, then $V(k, \ell) = V(RI^a(d_k^b), RI^b(d_k^b)) + 1$
- If $d_k^b \notin I(k, \ell)$, then $V(k, \ell) = V(k, \ell - 1)$

**Proof.** We prove cases 1 and 2. The proof of the cases 3 and 4 are similar. Here we assume that, $x(c_k^a) > x(c_k^b)$, i.e., the disk $d_k^a$ is to the right of the disk $d_k^b$.

Let $T^*$ be a maximum independent set of disks for subproblem $S(k, \ell)$. There are two possibilities, either $d_k^a$ is in the optimal solution or not.

- $d_k^a \in I(k, \ell)$: Let us assume that, $d_k^a = RI^a(d_k^a)$ and $d_k^b = RI^b(d_k^b)$. Since, $d_k^a$ is in the optimal solution, no disk in $D_\ell^a$ (resp. $D_\ell^b$) whose center is in between the centers of $d_k^a$ (resp. $d_k^b$) and $d_k^a$ can be present in any feasible solution. Thus any feasible solution contains disks from $D_\ell^a(\tau)$ and $D_\ell^b(\nu)$.

  Therefore, $T^*$ consists of $d_k^a$, together with the optimal solution to the subproblem $S(\tau, \nu)$.

- $d_k^a \notin I(k, \ell)$: By an argument similar to case 1, we say that, an optimal solution for $D_\ell^a(k - 1)$ and $D_\ell^b(\ell)$ gives an optimal solution for $D_\ell^a$ and $D_\ell^b$.

This completes the proof of the lemma.

Therefore, Lemma 3 suggests the following recurrence relation:

$$V(k, \ell) = \max \begin{cases} 
V(RI^a(d_k^a), RI^b(d_k^a)) + 1, \\
V(k - 1, \ell), \\
V(RI^a(d_k^b), RI^b(d_k^b)) + 1, \\
V(k, \ell - 1), 
\end{cases}$$

for $x(c_k^a) > x(c_k^b)$

for $x(c_k^a) < x(c_k^b)$

**Optimal Solution:** The optimal solution can be found by calling the function $V(n_1, n_2)$ with the base cases $V(k, \ell) = 0$ where both $k, \ell = 2$. Clearly, the final optimal solution contains the disks $d_0^a$ and $d_0^b$. Hence, we reduce the value of the optimal solution by 2 and remove these two disks from the optimal solution.
Running time: Let $T(n_i)$ be the total time taken by an algorithm $Z$ to evaluate $V(n_1, n_2)$. For a particular disk $d \in D_i$, finding either $RI^a(d)$ or $RI^b(d)$ requires $O(n_i)$ time. Hence, in $O(n_i^2)$ time, we find $RI^a(d)$ and $RI^b(d)$ for all $d \in D_i$. During recursive calls, for a particular disk $d$, the disks $RI^a(d)$ and $RI^b(d)$ can be found in $O(1)$ time. Therefore, the running time of $Z$ will be $O(n_i^2)$. Further, this algorithm requires $O(n_i^2)$ space to store the values of $V(k, \ell)$, for $0 \leq k \leq n_1$ and $0 \leq \ell \leq n_2$. Finally, we now have the following theorem.

Theorem 4. $MISUDG-L$ problem can be solved optimally in $O(n_i^2)$ time and $O(n_i^2)$ space.

3. $O(n^2)$ time factor 2 approximation for $MISUDG$ problem

In this section, we design a factor 2 approximation algorithm for $MISUDG$ problem. Let $D = \{d_1, d_2, \ldots, d_n\}$ be a set of $n$ unit disks in the plane. We first place horizontal lines from top to bottom with unit distance between each consecutive pair. Assume that there are $k$ such horizontal lines $\{L_1, L_2, \ldots, L_k\}$.

Let $D_i \subseteq D$ be the set of disks which are intersected by the line $L_i$. Now we have the following observation.

Observation 1. Any two disks, $d \in D_i$ and $d' \in D_j$ are independent (non-intersecting) if $|i - j| > 1$, for $1 \leq i, j \leq k$.

Note that, algorithm $Z$ optimally solves $MISUDG-L$ problem. Run $Z$ on each $D_i$, for $1 \leq i \leq k$ and let $S_i$ be an independent set of unit disks of maximum cardinality in $D_i$, $1 \leq i \leq k$. Let $S_{odd} = \bigcup_{i \text{ is odd}} S_i$ and $S_{even} = \bigcup_{i \text{ is even}} S_i$. We set $S$ as $S_{odd}$ or $S_{even}$ depending on whether $|S_{odd}|$ is greater or less than $|S_{even}|$ and report $S$ as the result of our algorithm. We now have the following theorem.

Theorem 5. The time and space complexities of our algorithm are both $O(n^2)$ and it produces a result with approximation factor 2.

Proof. Let $Opt$ be an optimal solution for $D$. Form Observation 1, we say that the disks in $S_{odd}$ are independent, and so $S_{even}$. Also, we have $|S_{odd}| + |S_{even}| \geq |Opt|$.

Therefore, $2|S| = 2 \max \{|S_{odd}|, |S_{even}|\} \geq |S_{odd}| + |S_{even}| \geq |Opt|$.
Since disks in $S_{odd}$ and $S_{even}$ are mutually independent, the total time required for computing $S_{odd}$ or $S_{even}$ is $O(n^2)$. Hence, the total time for reporting $S$ is $O(n^2)$, as required. For each $D_i$, $Z$ takes $O(n^2)$ space. Hence, the total space complexity is $O(n^2)$.

References

[1] R. K. Jallu, G. K. Das, Improved algorithm for maximum independent set on unit disk graph, in: CALDAM, 2016, pp. 212–223.

[2] G. K. Das, G. D. da Fonseca, R. K. Jallu, Efficient independent set approximation in unit disk graphs, unpublished Manuscript (2016).

[3] F. Roberts, Graph Theory and Its Applications to Problems of Society, Society for Industrial and Applied Mathematics, 1978.

[4] W. K. Hale, Frequency assignment: Theory and applications, Proceedings of the IEEE 68 (12) (1980) 1497–1514.

[5] E. Malesińska, Graph-theoretical models for frequency assignment problems, Ph.D. thesis, Technische Universität Berlin, Berlin, Germany. (1997).

[6] P. K. Agarwal, M. van Kreveld, S. Suri, Label placement by maximum independent set in rectangles, Compu. Geom. 11 (3) (1998) 209 – 218.

[7] D. Wang, Y.-S. Kuo, A study on two geometric location problems, Information Processing Letters 28 (6) (1988) 281 – 286.

[8] M. R. Garey, D. S. Johnson, Computers and Intractability: A Guide to the Theory of NP-Completeness, W. H. Freeman & Co., NY, USA, 1979.

[9] B. N. Clark, C. J. Colbourn, D. S. Johnson, Unit disk graphs, Discrete Mathematics 86 (1) (1990) 165 – 177.

[10] M. V. Marathe, H. Breu, H. B. Hunt, S. S. Ravi, D. J. Rosenkrantz, Simple heuristics for unit disk graphs, Networks 25 (2) (1995) 59–68.

[11] G. K. Das, M. De, S. Kolay, S. C. Nandy, S. Sur-Kolay, Approximation algorithms for maximum independent set of a unit disk graph, Information Processing Letters 115 (3) (2015) 439 – 446.

8
[12] T. Matsui, Approximation algorithms for maximum independent set problems and fractional coloring problems on unit disk graphs, in: JCDCG, 1998, pp. 194–200.

[13] D. S. Hochbaum, W. Maass, Approximation schemes for covering and packing problems in image processing and VLSI, J. ACM 32 (1) (1985) 130–136.

[14] E. J. van Leeuwen, Approximation algorithms for unit disk graphs, in: WG, 2005, pp. 351–361.