Algorithm and Hardness Results on Liar's Dominating Set and k-tuple Dominating Set

Sandip Banerjee[1] and Sujoy Bhore[2](✉)

[1] Department of Computer Science, Hebrew University of Jerusalem, Jerusalem, Israel
sandip.ndp@gmail.com

[2] Algorithms and Complexity Group, TU Wien, Vienna, Austria
sujoy@ac.tuwien.ac.at

Abstract. Given a graph $G = (V, E)$, the dominating set problem asks for a minimum subset of vertices $D \subseteq V$ such that every vertex $u \in V \setminus D$ is adjacent to at least one vertex $v \in D$. That is, the set D satisfies the condition that $|N[v] \cap D| \geq 1$ for each $v \in V$, where $N[v]$ is the closed neighborhood of v. In this paper, we study two variants of the classical dominating set problem: k-tuple dominating set (k-DS) problem and Liar's dominating set (LDS) problem, and obtain several algorithmic and hardness results. On the algorithmic side, we present a constant factor ($\frac{11}{2}$)-approximation algorithm for the Liar's dominating set problem on unit disk graphs. Then, we design a polynomial time approximation scheme (PTAS) for the k-tuple dominating set problem on unit disk graphs. On the hardness side, we show a $\Omega(n^2)$ bits lower bound for the space complexity of any (randomized) streaming algorithm for Liar's dominating set problem as well as for the k-tuple dominating set problem. Furthermore, we prove that the Liar's dominating set problem on bipartite graphs is W[2]-hard.

1 Introduction

The dominating set problem is regarded as one of the fundamental problems in theoretical computer science which finds its applications in various fields of science and engineering [3,8]. A *dominating set* of a graph $G = (V, E)$ is a subset D of V such that every vertex in $V \setminus D$ is adjacent to at least one vertex in D. The *domination number*, denoted as $\gamma(G)$, is the minimum cardinality of a dominating set of G. Garey and Johnson [6] showed that deciding whether a given graph has domination number at most some given integer k is NP-complete. For a vertex $v \in V$, the open neighborhood of the vertex v denoted as $N_G(v)$ is defined as $N_G(v) = \{u | (u, v) \in E\}$ and the closed neighborhood of the vertex v denoted as $N_G[v]$ is defined as $N_G[v] = N_G(v) \cup \{v\}$.

S. Bhore—Supported by the Austrian Science Fund (FWF) under project number P31119.

C. J. Colbourn et al. (Eds.): IWOCA 2019, LNCS 11638, pp. 48–60, 2019.
https://doi.org/10.1007/978-3-030-25005-8_5

k-tuple Dominating Set (k-DS): Fink and Jacobson [5] generalized the concept of dominating sets as follows.

> **k-tuple Dominating Set (k-*DS*) Problem**
> **Input:** A graph $G = (V, E)$ and a non-negative integer k.
> **Goal:** Choose a minimum cardinality subset of vertices $D \subseteq V$ such that, for every vertex $v \in V$, $|N_G[v] \cap D| \geq k$.

The k-*tuple domination number* $\gamma_k(G)$ is the minimum cardinality of a k-DS of G. A good survey on k-DS can be found in [4,12]. Note that, 1-tuple dominating set is the usual dominating set, and 2-tuple and 3-tuple dominating set are known as *double dominating set* [7] and *triple dominating set* [15], respectively. Further note that, for a graph $G = (V, E)$, $\gamma_k(G) = \infty$, if there exists no k-DS of G. Klasing et al. [10] studied the k-DS problem from hardness and approximation point of view. They gave a $(\log|V| + 1)$-approximation algorithm for the k-tuple domination problem in general graphs, and showed that it cannot be approximated within a ratio of $(1 - \epsilon)\log|V|$, for any $\epsilon > 0$.

Liar's Dominating Set (LDS): Slater [17] introduced a variant of the dominating set problem called Liar's dominating set problem. Given a graph $G = (V, E)$, in this problem the objective is to choose minimum number of vertices $D \subseteq V$ such that each vertex $v \in V$ is double dominated and for every two vertices $u, v \in V$ there are at least three vertices in D from the union of their neighborhood set. The LDS problem is an important theoretical model for the following real-world problem. Consider a large computer network where a virus (generated elsewhere in the Internet) can attack any of the processors in the network. The network can be viewed as an unweighted graph. For each node $v \in V$, an anti-virus can: (1) detect the virus at v as well as in its closed neighborhood $N[v]$, and (2) find and report the vertex $u \in N[v]$ at which the virus is located. Notice that, one can make network G virus free by deploying the anti-virus at the vertices $v \in D$, where D is the minimum size *dominating set*. However, in certain situations the anti-viruses may fail. Hence, to make the system virus free it is likely to double-guard the nodes of the network, which is indeed the 2-tuple DS. However, despite of the double-guarding, the anti-viruses may fail to cure the system properly due to some software error or corrupted circumstances. Therefore, for every pair of nodes, it is be important to introduce a guard that sees both of them. This leads us to the Liar's dominating set problem. We define the problem formally below.

> **Liar's Dominating Set (*LDS*) Problem**
> **Input:** A graph $G = (V, E)$ and a non-negative integer k.
> **Goal:** Choose a subset of vertices $L \subseteq V$ of minimum cardinality such that for every vertex $v \in V$, $|N_G[v] \cap L| \geq 2$, and for every pair of vertices $u, v \in V$ of distinct vertices $|(N_G[u] \cup N_G[v]) \cap L| \geq 3$.

1.1 Our Results

In this paper, we obtain seveal algorithmic and hardness results for LDS and k-DS problems on various graph families. On the algorithmic side in Sect. 2, we present a constant factor ($\frac{11}{2}$)-approximation algorithm for the Liar's dominating set (LDS) problem on unit disk graphs. Then, we design a polynomial time approximation scheme (PTAS) for the k-tuple dominating set (k-DS) problem on unit disk graphs. On the hardness side in Sect. 3, we show a $\Omega(n^2)$ bits lower bound for the space complexity of any (randomized) streaming algorithm for Liar's dominating set problem as well as for the k-tuple dominating set problem. Furthermore, we prove that the Liar's dominating set problem on bipartite graphs is W[2]-hard.

2 Algorithmic Results

2.1 Approximation Algorithm for LDS on Unit Disk Graphs

Unit disk graphs are widely used to model of wireless sensor networks. A unit disk graph (UDG) is an intersection graph of a family of unit radius disks in the plane. Formally, given a collection $\mathcal{C} = \{C_1, C_2, \ldots, C_n\}$ of n unit disks in the plane, a UDG is defined as a graph $G = (V, E)$, where each vertex $u \in V$ corresponds to a disk $C_i \in \mathcal{C}$ and there is an edge $(u, v) \in E$ between two vertices u and v if and only if their corresponding disks C_u and C_v contain v and u, respectively. Here, we study the LDS problem on UDG.

Liar's Dominating Set on UDG (*LDS-UDG*) Problem
Input: A unit disk graph $G = (\mathcal{P}, E)$, where $\mathcal{P}$ is a set of n disk centers.
Output: A minimum size subset $D \subseteq \mathcal{P}$ such that for each point $p_i \in \mathcal{P}$, $|N[p_i] \cap D| \geq 2$, and for each pair of points $p_i, p_j \in \mathcal{P}$, $|(N[p_i] \cup N[p_j]) \cap D| \geq 3$.

Jallu et al. [9] studied the LDS problem on unit disk graphs, and proved that this problem is NP-complete. Furthermore, given an unit disk graph $G = (V, E)$ and an $\epsilon > 0$, they have designed a $(1+\epsilon)$-factor approximation algorithm to find an LDS in G with running time $n^{O(c^2)}$, where $c = O(\frac{1}{\epsilon} \log \frac{1}{\epsilon})$. In this section, we design a $\frac{11}{2}$-factor approximation algorithm that runs in sub-quadratic time.

For a point $p \in \mathcal{P}$, let $C(p)$ denote the disk centered at the point p. For any two points $p, q \in \mathcal{P}$, if $q \in C(p)$, then we say that q is a neighbor of p (sometimes we say q is covered by p) and vice versa. Since for any Liar's dominating set D, $|(N[p_i] \cup N[p_j]) \cap D| \geq 3$ $(\forall p_i, p_j \in \mathcal{P})$ holds, we assume that $|\mathcal{P}| \geq 3$ and for all the points $p \in \mathcal{P}$, $|N[p]| \geq 2$. For a point $p_i \in \mathcal{P}$, let $p_i(x)$ and $p_i(y)$ denote the x and y coordinates of p_i, respectively. Let $\mathrm{Cov}_{\frac{1}{2}}(C(p_i))$, $\mathrm{Cov}_1(C(p_i))$, $\mathrm{Cov}_{\frac{3}{2}}(C(p_i))$ denote the set of points of $\mathcal{P}$ that are inside the circle centered at p_i and of radius $\frac{1}{2}$, 1 and $\frac{3}{2}$ unit, respectively.

The basic idea of our algorithm is as follows. Initially, we sort the points of $\mathcal{P}$ based on their x-coordinates. Now, consider the leftmost point (say p_i).

We compute the sets $\mathrm{Cov}_{\frac{1}{2}}(C(p_i))$, $\mathrm{Cov}_1(C(p_i))$ and $\mathrm{Cov}_{\frac{3}{2}}(C(p_i))$. Next, we compute the set $Q = \mathrm{Cov}_{\frac{3}{2}}(C(p_i)) \setminus \mathrm{Cov}_1(C(p_i))$. Further, for each point $q_i \in Q$, we compute the set $S(q_i) = \mathrm{Cov}_1(C(q_i)) \cap \mathrm{Cov}_{\frac{1}{2}}(C(p_i))$. Finally, we compute the set $S = \bigcup S(q_i)$. Moreover, our algorithm is divided into two cases.

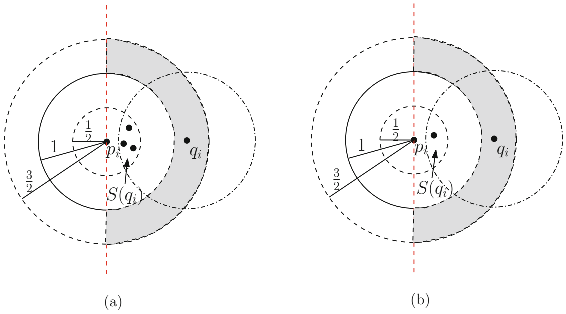

Fig. 1. An illustration of Case 1; (a) $|S(q_i)| \geq 3$, (b) $|S(q_i)| \leq 2$.

Case 1 ($S \neq \emptyset$): For each point $q_i \in Q$ such that $S(q_i) \neq \emptyset$, we further distinguish between the following cases.

1. If $|S(q_i)| \geq 3$: we pick two arbitrary points from the set $S(q_i)$, and include them in the output set D (see Fig. 1(a)).
2. If $|S(q_i)| \leq 2$: in this case, we select one point p_a and a possible second point p_b in the output set D (see Fig. 1(b)).

Once these points are selected, we remove the remaining points from $\mathrm{Cov}_1(C(q_i))$ at this step from the set Q. Notice that the points that lie in $\mathrm{Cov}_1(C(q_i))$ are already 1-dominated. Later, we can pick those points if required. However, observe that we may choose a point p_i from $\mathrm{Cov}_{\frac{1}{2}}(C(p_i))$ while in Case 1.2. That would not constitute a LDS. So we maintain a counter t in Case 1. This counter keeps track of how many points we are picking from the set $S(q_i)$ in total, for each point $q_i \in Q$. If t is at least 2, we simply add p_i to the output set and do not enter into Case 2. Otherwise, we proceed to Case 2.

Case 2 ($S = \emptyset$ or $t < 2$): here, we further distinguish between the following cases.

1. If $|\mathrm{Cov}_{\frac{1}{2}}(C(p_i))| \geq 3$: then we choose 2 points arbitrarily in the output set D (see Fig. 2(a)).
2. If $|\mathrm{Cov}_{\frac{1}{2}}(C(p_i))| = 2$: let $p_i, p_x \in \mathrm{Cov}_{\frac{1}{2}}(C(p_i))$ be these points. We include both of them in the output set D. This settles the first condition of LDS for them. However, in order to fulfill the second condition of LDS for p_i and p_x, we must include at least one extra point here. First, we check the cardinality of $X = (\mathrm{Cov}_1(C(p_i)) \cap \mathrm{Cov}_1(C(p_x))) \setminus \{p_i, p_x\}$. If $|X| \neq \emptyset$, then we pick an

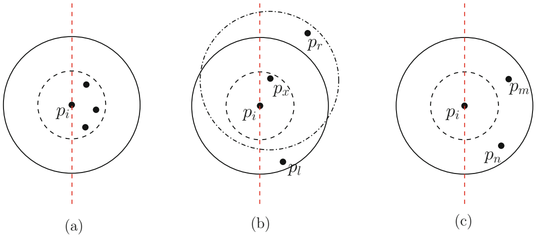

Fig. 2. An illustration of Case 2; (a) $|\mathrm{Cov}_{\frac{1}{2}}(C(p_i))| \geq 3$, (b) $|\mathrm{Cov}_{\frac{1}{2}}(C(p_i))| = 2$, (c) $|\mathrm{Cov}_{\frac{1}{2}}(C(p_i))| = 1$.

arbitrary point p_m from X, and include p_m in D. Otherwise, we include two points $p_l \in \mathrm{Cov}_1(C(p_i))$ and $p_r \in \mathrm{Cov}_1(C(p_x))$, and include them in D (see Fig. 2(b)). Note that, in this case, we know that p_l and p_r exist due to the input constraint of an LDS problem.

3. If $|\mathrm{Cov}_{\frac{1}{2}}(C(p_i))| = 1$: then we check $\mathrm{Cov}_1(C(p_i))$ and include two points p_m and p_n arbitrarily from $\mathrm{Cov}_1(C(p_i)) \setminus \mathrm{Cov}_{\frac{1}{2}}(C(p_i))$ (see Fig. 2(c)).

This fulfills the criteria of LDS of points in $\mathrm{Cov}_{\frac{1}{2}}(C(p_i))$. Then we delete the remaining points of $\mathrm{Cov}_{\frac{1}{2}}(C(p_i))$ from $\mathcal{P}$. Next we select the left-most point from the remaining and repeat the same procedure until $\mathcal{P}$ is empty. The pseudo-code of the algorithm is given in the full version of the paper (see [1]).

Lemma 1. *[$\star$][1] The set D obtained from our algorithm, is a LDS of the unit disk graph defined on the points of $\mathcal{P}$.*

Lemma 2. *[$\star$] Our algorithm outputs a LDS $D \subseteq \mathcal{P}$ of the unit disk graph defined on the points of $\mathcal{P}$ with approximation ratio $\frac{11}{2}$.*

The algorithm runs in polynomial time (to be precise in sub-quadratic time). Thus, from Lemmas 1 and 2 we conclude the following theorem.

Theorem 1. *The algorithm computes a LDS of the unit disk graph defined on the points of $\mathcal{P}$ in sub-quadratic time with approximation factor $\frac{11}{2}$.*

2.2 PTAS for k-DS on Unit Disk Graphs

In this section we give a PTAS for the k-tuple dominating set on unit disk graphs with a similar approach used by Nieberg and Hurink [13]. It might be possible to design a PTAS by using local search or shifting strategy for the k-tuple dominating set problem on unit disk graphs. However, the time complexity of these algorithms would be high. Thus we use the approach of Nieberg and Hurink [13], that gurantees a better running time.

---

[1] Proof of results labeled with [$\star$] have been deferred to the full version [1] due to space constraint.

Let $G = (V, E)$ be an unit disk graph in the plane. For a vertex $v \in V$, let $N^r(v)$ and $N^r[v] = N^r(v) \cup \{v\}$ be the r-th neighborhood and r-th closed neighborhood of v, respectively. For any two vertices $u, v \in V$, let $\delta(u, v)$ be the distance between u and v in G, that is the number of edges of a shortest path between u and v in G. Let $D_k(V)$ be the minimum k-tuple dominating set of G. For a subset $W \subseteq V$, let $D_k(W)$ be the minimum k-tuple dominating set of the induced subgraph on W. We prove the following theorem.

Theorem 2. *There exists a PTAS for the k-tuple dominating set problem on unit disk graphs.*

Proof. The 2-separated collection of subsets is defined as follows: Given a graph $G = (V, E)$, let $S = \{S_1, \ldots, S_m\}$ be a collection of subsets of vertices $S_i \subset V$, for $i = 1, \ldots, m$, such that for any two vertices $u \in S_i$ and $v \in S_j$ with $i \neq j$, $\delta(u, v) > 2$. In the following lemma we prove that the sum of the cardinalities of the minimum k-tuple dominating sets $D_k(S_i)$ for the subsets $S_i \in S$ of a 2-separated collection is a lower bound on the cardinality of $D_k(V)$.

Lemma 3. [$\star$] *Given a graph $G = (V, E)$, let $S = \{S_1, \ldots, S_m\}$ be a 2-separated collection of subsets of V then, $|D_k(V)| \geq \sum_{i=1}^{m} |D_k(S_i)|$.*

From Lemma 3, we get the lower bound of the minimum k-tuple dominating set of G. If we can enlarge each of the subset S_i to a subset T_i such that the k-tuple dominating set of S_i (that is $D_k(S_i)$) is locally bounded to the k-tuple dominating set of T_i (that is $D_k(T_i)$), then by taking the union of them we get an approximation of the k-tuple dominating set of G. For each subset S_i, let there is a subset T_i (where $S_i \subset T_i$), and let there exists a bound $(1 + \epsilon)$ $(0 < \epsilon < 1)$ such that $|D_k(T_i)| \leq (1 + \epsilon) \cdot |D_k(S_i)|$. Then, if we take the union of the k-tuple dominating sets of all T_i, this is a $(1 + \epsilon)$-approximation of the k-tuple dominating sets of the union of subsets S_i (for $i = 1, \ldots, m$). Now, we describe the algorithm. Let $V_0 = V$. Consider an arbitrary vertex $v \in V_0$, and begin computing the k-DS of $N^r[v]$, until $D_k(N^{r+2}[v]) > \rho \cdot D_k(N^r[v])$ (for a constant ρ). We iteratively process the remaining graph induced by $V_{i+1} = V_i \setminus N^{\hat{r_i}+2}[v_i]$ (where $\hat{r_i}$ is the first point at ith iteration when the condition is violated).

Lemma 4. *Let $\{N_1, \ldots, N_\ell\}$ be the set of neighborhoods created by the above algorithm (for $\ell < n$). The union $\bigcup_{i=1}^{\ell}(D_k(N_i))$ forms a k-tuple dominating set of G.*

Proof. Consider the set $V_{i+1} = V_i \setminus N_i$, and we know $N_i \subset V_i$. Thus, $V_{i+1} = V_i \cup N_i$. The algorithm stops while $V_{\ell+1} = \emptyset$, which means $V_\ell = N_\ell$. Besides, $\bigcup_{i=1}^{\ell}(N_i) = V$. Thus, if we compute the k-tuple dominating set $D_k(N_i)$ of each N_i, their union clearly is the k-tuple dominating set of the entire graph. $\square$

These subsets $N^{\hat{r_i}}[v_i]$, for $i = 1, \ldots, \ell$, created by the algorithm form a 2-separated collection $\{N^{\hat{r_1}}[v_1], \ldots, N^{\hat{r_\ell}}[v_\ell]\}$ in G. Consider any two neighborhoods N_i, N_{i+1}. We have computed N_{i+1} on graph induced by $V \setminus V_i$. So, for any two vertices $u \in N_i$ and $v \in N_{i+1}$, the distance is greater than 2. Thus we have the following corollary.

Corollary 1. *The algorithm returns a k-tuple dominating set* $\bigcup_{i=1}^{\ell}(D_k(N_i))$ *of cardinality no more than* ρ *the size of a dominating set* $D_k(V)$*, where* $\rho = (1+\epsilon)$*.*

It needs to be shown that this algorithm has a polynomial running time. The number of iterations ℓ is clearly bounded by $|V| = n$. It is important to show that for each iteration we can compute the minimum k-tuple dominating set $D_k(N^r[v])$ in polynomial time for r being constant or polynomially bounded. Consider the r-th neighborhood of a vertex v, $N^r[v]$. Let I^r be the maximal independent set of the graph induced by $N^r[v]$. From [13], we have $I^r \leq (2r+1)^2 = O(r^2)$. The cardinality of a minimum dominating set in $N^r[v]$ is bounded from above by the cardinality of a maximal independent set in $N^r[v]$. Hence, $|D(N^r[v]| \leq (2r+1)^2 = O(r^2)$. Now, we prove the following lemma.

Lemma 5. $|D_k(N^r[v])| \leq O(k^2 \cdot r^2)$.

Proof. Let I_1^r be the first maximal independent set of $N^r[v]$. We know $|D(N^r[v])| \leq |I_1^r| \leq (2r+1)^2$. Now, we take the next maximal independent set I_2^r from $N^r[v] \setminus I_1^r$, and take the union of them $(I_1^r \cup I_2^r)$. Notice that every vertex $v \in (N^r[v] \setminus (I_1^r \cup I_2^r))$ has 2 neighbors in $(I_1^r \cup I_2^r)$, so they can be 2-tuple dominated by choosing vertices from $(I_1^r \cup I_2^r)$. Also, every vertex $v \in I_2^r$ can be 2-tuple dominated by choosing vertices from $(I_1^r \cup I_2^r)$, since v itself can be one and the other one can be picked from I_1^r. Additionally, for each vertex $u \in I_1^r$, we take a vertex z from the neighborhood of u in $(N^r[v] \setminus (I_1^r \cup I_2^r))$. Let $W(I_1^r)$ be the union of these vertices. Now, every vertex $v \in N^r[v]$ can be 2-tuple dominated by choosing vertices from $(I_1^r \cup I_2^r \cup W(I_1^r))$. So, $|D_2(N^r[v])| \leq |(I_1^r \cup I_2^r \cup W(I_1^r))|$. $|(I_1^r \cup I_2^r \cup W(I_1^r))| \leq 3 \cdot (2r+1)^2$. Hence, $|D_2(N^r[v])| \leq 3 \cdot (2r+1)^2$. We continue this process k times.

After k steps, we get the union of the maximal independent sets $A = \{I_1^r \cup \ldots \cup I_k^r\}$. Additionally, we get the unions of $B = \{W(I_1^r) \cup W(I_1^r \cup I_2^r) \cup \ldots \cup W(I_1^r \cup \ldots \cup I_{k-1}^r)\}$. Notice that every vertex $v \in N^r[v]$ can be k-tuple dominated by choosing vertices from $(A \cup B)$. The cardinality of $(A \cup B)$ is at most $(2r+1)^2 \cdot (1+3+\ldots+(2k-1))$, which is $(2r+1)^2 \cdot k^2$. We also know $|D_k(N^r[v])|$ is upper bounded by $(A \cup B)$. Thus, $|D_k(N^r[v])| \leq (2r+1)^2 \cdot k^2 \leq O(k^2 \cdot r^2)$.

Nieberg and Hurink [13] showed that for a unit disk graph, there exists a bound on $\hat{r_1}$ (the first value of r that violates the property $D(N^{r+2}[v]) > \rho \cdot D(N^r[v])$). This bound depends on the approximation ρ not on the size of the of the unit disk graph $G = (V, E)$ given as input. Precisely, they have proved that there exists a constant $c = c(\rho)$ such that $\hat{r_1} \leq c$, that is, the largest neighborhood to be considered during the iteration of the algorithm is bounded by a constant. Thereby, putting everything together, we conclude the proof. □

3 Hardness Results

3.1 Streaming Lower Bound for LDS

In this section, we consider the streaming model: the edges arrive one-by-one in some order, and at each time-stamp we need to decide if we either *store* the edge or *forget* about it. We now show that any streaming algorithm that solves the LDS problem must essentially store all the edges.

Theorem 3. *Any randomized*[2] *streaming algorithm for LDS problem on n-vertex graphs requires $\Omega(n^2)$ space.*

Proof. We will reduce from the INDEX problem in communication complexity:

> **Index Problem**
> **Input:** Alice has a string $X \in \{0,1\}^N$ given by $x_1 x_2 \ldots x_N$. Bob has an index $\iota \in [N]$.
> **Question:** Bob wants to find x_ι, i.e., the ι^{th} bit of X.

It is well-known that there is a lower bound of $\Omega(N)$ bits in the one-way randomized communication model for Bob to compute x_i [11]. We assume an instance of the INDEX problem where N is a perfect square, and let $r = \sqrt{N}$. Fix any bijection from $[N] \to [r] \times [r]$. Consequently we can interpret the bit string as an adjacency matrix for a bipartite graph with r vertices on each side. Let the two sides of the bipartition be $V = \{v_1, v_2, \ldots, v_r\}$ and $W = \{w_1, w_2, \ldots, w_r\}$.

From the instance of INDEX, we construct an instance G_X of the LDS. Assume that Alice has an algorithm that solves the k-tuple dominating set problem using $f(r)$ bits. First, we insert the edges corresponding to the edge interpretation of X between nodes v_i and w_j: for each $i, j \in [k]$, Alice adds the edge (v_i, w_j) if the corresponding entry in X is 1. Alice then sends the memory contents of her algorithm to Bob, using $f(r)$ bits.

Bob has the index $\iota \in [N]$, which he interprets as (I, J) under the same bijection $\phi : [N] \to [r] \times [r]$. He receives the memory contents of the algorithm, and proceeds to do the following:

- Add two vertices a and b, and an edge $a - b$
- Add an edge from each vertex of $V \setminus v_I$ to a
- Add an edge from each vertex of $W \setminus w_J$ to a
- Add five vertices $\{u, y, u', y', z\}$ and edges $u - u', y - y', u - z$ and $y - z$.
- Add an edge from each vertex of $V \cup W \cup \{a, b\}$ to each vertex from $\{u, y\}$

Let D be a minimum LDS of G_X. Note that D has to be a double dominating set of G_X. Since u' has only 2 neighbors in G_X, it follows that $\{u, u'\} \subseteq D$. Similarly $\{y, y'\} \subseteq D$. Note that z also has only two neighbors in G_X. Hence, we have that $N[z] \cup N[b] = \{u, y, a, b\}$. Since we must have $|(N[z] \cup N[b]) \cap D| \geq 3$, it follows that at least one of a or b must belong in D. Since $N[b] \subseteq N[a]$, without loss of generality we can assume that $a \in D$. Therefore, so far we have concluded that $\{u, u', y, y', a\} \subseteq D$.

The next two lemmas show that finding the minimum value of a LDS of G_X allows us to solve the corresponding instance X of INDEX.

[2] By randomized algorithm we mean that the algorithm should succeed with probability $\geq \frac{2}{3}$.

Lemma 6. *$x_\iota = 1$ implies that the minimum size of a LDS of G_X is 6.*

Proof. Suppose that $x_\iota = 1$, i.e., $v_I - w_J$ is an edge in G_X. We now claim that $D := \{u, u', y, y', a\} \cup v_I$ is a LDS of G_X.

First we check that D is indeed a double dominating set of G_X

- For each vertex in $\lambda \in G_X \setminus \{u, u', y, y', z\}$ we have $(N[\lambda] \cap D) \supseteq \{u, y\}$
- $(N[u] \cap D) \supseteq \{u, u'\}$
- $(N[y] \cap D) \supseteq \{y, y'\}$
- $(N[z] \cap D) = \{u, y\}$
- $(N[u'] \cap D) = \{u, u'\}$
- $(N[y'] \cap D) = \{y, y'\}$

We now check the second condition. Let $T = G_X \setminus \{u, u', y, y', z\}$, and $T' = G_X \setminus T$

- For each $\lambda \in T \setminus \{v_I, w_J\}$ and each $\delta \in T'$ we have $(N[\lambda] \cup N[\delta]) \cap D \supseteq \{a, u, y\}$
- For each $\delta \in T'$ we have $(N[v_I] \cup N[\delta]) \cap D \supseteq \{v_I, u, y\}$
- For each $\delta \in T'$ we have $(N[w_J] \cup N[\delta]) \cap D \supseteq \{v_I, u, y\}$
- Now we consider pairs where both vertices are from T'. By symmetry, we only have to consider following choices
  - $(N[u'] \cup N[u]) \cap D = \{u, u', y\}$
  - $(N[u'] \cup N[z]) \cap D = \{u, u', y\}$
  - $(N[u'] \cup N[y]) \cap D = \{u, u', y, y'\}$
  - $(N[u'] \cup N[y']) \cap D = \{u, u', y, y'\}$
- Now we consider pairs where both vertices are from T. By symmetry, we only have to consider following choices
  - For each $\lambda \in V \setminus v_I \cup W \setminus w_J$ we have $(N[\lambda] \cup N[b]) \cap D \supseteq \{u, y, a\}$ and $(N[\lambda] \cup N[a]) \cap D \supseteq \{u, y, a\}$
  - $(N[v_I] \cup N[b]) \cap D \supseteq \{u, y, a\}$
  - $(N[v_I] \cup N[a]) \cap D \supseteq \{u, y, a\}$
  - $(N[w_J] \cup N[b]) \cap D \supseteq \{u, y, a\}$
  - $(N[w_J] \cup N[a]) \cap D \supseteq \{u, y, a\}$
  - $(N[w_J] \cup N[v_I]) \cap D \supseteq \{v_I, y, a\}$
  - For each $\gamma \in V \setminus v_I$ we have $(N[w_J] \cup N[\gamma]) \cap D \supseteq \{v_I, y, a, u\}$
  - For each $\gamma \in W \setminus w_J$ we have $(N[v_I] \cup N[\gamma]) \cap D \supseteq \{v_I, y, a, u\}$
  - For each $\gamma \in V \setminus v_I$ and $\gamma' \in W \setminus w_J$ we have $(N[\gamma] \cup N[\gamma']) \cap D \supseteq \{y, a, u\}$

Hence, it follows that D is indeed a LDS of G_X of size 6.

Lemma 7. *$x_\iota = 0$ implies that the minimum size of a LDS of G_X is ≥ 7.*

Proof. Now suppose that $x_\iota = 0$, i.e., v_I and w_J do not have an edge between them in G_X. Let D' be a minimum LDS of G_X. We have already seen above that $\{u, u', y, y', a\} \subseteq D$.

Consider the pair (v_I, z). Currently, we have that $(N[v_I] \cup N[z]) \cap \{u, u', y, y', a\} = \{u, y\}$. Hence, D' must contain a vertex, say $\mu \in N[v_I] \setminus \{u, y\}$. Consider the pair (w_J, z). Currently, we have that $(N[w_J] \cup N[z]) \cap$

$\{u, u', y, y', a\} = \{u, y\}$. Hence, D' must contain a vertex, say $\mu' \in N[w_J] \setminus \{u, y\}$. Since v_I and w_J do not form an edge, we have that $\mu \neq \mu'$. Hence, $|D'| \geq 5+2 = 7$.

Thus, by checking whether the value of a minimum LDS on the instance G_X is 6 or 7, Bob can determine the index x_ι. The total communication between Alice and Bob was $O(f(r))$ bits, and hence we can solve the INDEX problem in $f(r)$ bits. Recall that the lower bound for the INDEX problem is $\Omega(N) = \Omega(r^2)$. Note that $|G_X| = n = 2r + 5 = O(r)$, and hence $\Omega(r^2) = \Omega(n^2)$. □

Corollary 2. *Let $\epsilon > 0$ be a constant. Any (randomized) streaming algorithm that achieves a $(\frac{7}{6} - \epsilon)$-approximation for a LDS requires $\Omega(n^2)$ space.*

Proof. Theorem 3 shows that distinguishing between whether the minimum value of the LDS is 6 or 7 requires $\Omega(n^2)$ bits. The claim follows since $6 \cdot (\frac{7}{6} - \epsilon) < 7$. □

3.2 Streaming Lower Bounds for *k*-DS

Theorem 4. *For any $k = O(1)$, any randomized (See footnote 2) streaming algorithm for the k-tuple dominating set problem on n-vertex graphs requires $\Omega(n^2)$ space.*

Proof. We reduce from the INDEX problem in communication complexity. We assume an instance of the INDEX problem where N is a perfect square, and let $r = \sqrt{N}$. Fix any bijection from $[N] \to [r] \times [r]$. Consequently we can interpret the bit string as an adjacency matrix for a bipartite graph with r vertices on each side. Let the two sides of the bipartition be $V = \{v_1, v_2, \ldots, v_r\}$ and $W = \{w_1, w_2, \ldots, w_r\}$. From the instance of INDEX, we construct an instance G_X of the k-tuple dominating set. Assume that Alice has an algorithm that solves the k-tuple dominating set problem using $f(r)$ bits. First, we insert the edges corresponding to the edge interpretation of X between nodes v_i and w_j: for each $i, j \in [k]$, Alice adds the edge (v_i, w_j) if the corresponding entry in X is 1. Alice then sends the memory contents of her algorithm to Bob, using $f(r)$ bits.

Bob has the index $\iota \in [N]$, which he interprets as (I, J) under the same bijection $\phi : [N] \to [r] \times [r]$. He receives the memory contents of the algorithm, and proceeds to do the following:

- Add (k+1) vertices $A = \{a_1, a_2, \ldots, a_k\}$ and b.
- Add edges $\{a_i - b \ : \ 1 \leq i \leq k\}$.
- Add an edge from each vertex of $V \setminus (v_I)$ to each vertex of A.
- Add an edge from each vertex of $W \setminus (w_J)$ to each vertex of A.
- Add an edge v_I to each vertex of $A \setminus a_k$.
- Add an edge from w_J to each vertex of $A \setminus a_k$.

The next lemma shows that finding the minimum value of a k-tuple dominating set of G_X allows us to solve the corresponding instance X of INDEX.

Lemma 8. *[$\star$] The minimum size of a k-tuple dominating set of G_X is $k+1$ if and only if $x_\iota = 1$.*

Thus, by checking whether the value of minimum k-tuple dominating set on the instance G_X is $k+1$ or $k+2$, Bob can determine the index x_ι. The total communication between Alice and Bob was $O(f(r))$ bits, and hence we can solve the INDEX problem in $f(r)$ bits. Recall that the lower bound for the INDEX problem is $\Omega(N) = \Omega(r^2)$. Note that $|G_X| = n = 2r+k+1 = O(r)$ since $k = O(1)$, and hence $\Omega(r^2) = \Omega(n^2)$.

Corollary 3. *Let $1 > \epsilon > 0$ be any constant. Any (randomized) streaming algorithm that approximates a k-tuple dominating set within a relative error of ϵ requires $\Omega(n^2)$ space.*

Proof. Choose $\epsilon = \frac{1}{k}$. Theorem 8 shows that the relative error is at most $\frac{1}{k+2}$, which is less than ϵ. Hence finding an approximation within ϵ relative error amounts to finding the exact value of the k-tuple dominating set. Hence, the claim follows from the lower bound of $\Omega(n^2)$ of Theorem 8.

3.3 W-Hardness Results for LDS

The LDS problem was is NP-complete on general graphs [17], and NP-complete on bipartite graphs, split graphs (see, e.g., [14,16]). Bishnu et al. [2] showed that LDS problem on planar graphs admits a linear kernel and W[2]-hard on general graphs. We prove that LDS problem is W[2]-hard on bipartite graphs.

Theorem 5. *Liar's dominating set on bipartite graphs is W[2]-hard.*

Proof. We prove this by giving a parameterized reduction from the dominating set problem in general undirected graphs. Let $(G = (V, E), k)$ be an instance of the dominating set, where k denotes the size of the dominating set. We construct a bipartite graph $G' = (V', E')$ from $G = (V, E)$. First, we create two copies of V, namely $V_1 = \{u_1 | u \in V\}$ and $V_2 = \{u_2 | u \in V\}$. Next, we introduce two extra vertices z_1, z_2 in V_1, and two extra vertices z'_1, z'_2 in V_2. Furthermore, we introduce two special vertices $s_{z'_1}, s_{z'_2}$ in V_1 and two special vertices s_{z_1}, s_{z_2} in V_2. The entire vertex set V' is $V_1 \cup V_2$, where $V_1 = \{u_1 | u \in V\} \cup \{z_1, z_2, s_{z'_1}, s_{z'_2}\}$ and $V_2 = \{u_2 | u \in V\} \cup \{z'_1, z'_2, s_{z_1}, s_{z_2}\}$. Now, if there is an edge $(u, v) \in E$, then we draw the edges (u_1, v_2) and (v_1, u_2). We draw the edges of the form (u_1, u_2) in G' for every vertex $u \in V$. Then, we add edges from every vertex in $V_1 \setminus \{z_1, z_2, s_{z'_1}, s_{z'_2}\}$ to z'_1, z'_2, and the edges from every vertex in $V_1 \setminus \{z'_1, z'_2, s_{z_1}, s_{z_2}\}$ to z_1, z_2. Finally we add the edges $(z_1, z'_1), (z_2, z'_2)$ and $(z_1, s_{z_1}), (z_2, s_{z_2}), (z'_1, s_{z_1}), (z'_2, s_{z'_2})$. This completes the construction (see Fig. 3).

We show that G has a dominating set of size k if and only if G' has a LDS of size $k+8$. Let D denote the dominating set of the given graph G. We claim that $D' = \{u_1 | u \in D\} \cup \{z_1, z_2, s_{z'_1}, s_{z'_2}\} \cup \{z'_1, z'_2, s_{z_1}, s_{z_2}\}$ is a LDS of G'. Note that for any vertex $v \in V'$, $|N_{G'}[v] \cap D'| \geq 2$, since $\{z_1, z_2, z'_1, z'_2\}$ is in D'. This fulfills the first condition of the LDS. Now, for every pair of vertices, $u, v \in V'$, we show that $|(N_{G'}[u] \cup N_{G'}[v]) \cap D'| \geq 3$. If $u, v \in V_1$, we know $|(N_{G'}[u] \cup N_{G'}[v]) \cap D'| \geq 2$ due to z'_1, z'_2. Now, in the dominating set at least one additional vertex dominates

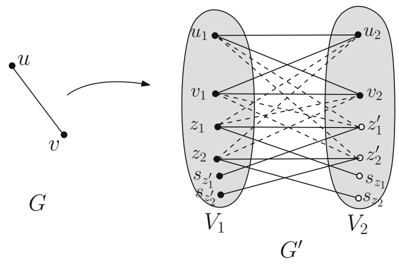

Fig. 3. Construction of G' from G (Illustration of Theorem 5).

them. Thus, $|(N_{G'}[u] \cup N_{G'}[v]) \cap D'| \geq 3$. Similarly, when $u, v \in V_2$ or $u \in V_1$ and $v \in V_2$. This fulfills the second condition of the LDS.

Conversely, let D' be a LDS in G'. Note that, $\{z_1, z_2, z'_1, z'_2\}$ are always part of D', since z_1, z_2 are the only neighbors of s_{z_1}, s_{z_2} and z'_1, z'_2 are the only neighbors of $s_{z'_1}, s_{z'_2}$. These special vertices are taken in the construction to enforce $\{z_1, z_2, z'_1, z'_2\}$ to be in D'. Now we know, for any pair of vertices p, q, $|(N_{G'}[p] \cup N_{G'}[q]) \cap D'| \geq 3$. This implies p, q is dominated by at least one vertex or one of them is picked, except $\{z_1, z_2, z'_1, z'_2\}$. Otherwise, $|(N_{G'}[p] \cup N_{G'}[q]) \cap D'| < 3$. This violates the second condition of LDS. Now, when p, q are both part of the same edge in G (say $u_2, v_2 \in V_2$; see Fig. 3), we need at least one vertex from $\{u_1, v_1, u_2, v_2\}$ in D'. This means that for every vertex $v \in V$, $|N_G[v] \cap D| \geq 1$. Thus, D is a dominating set of G where the cardinality of D is at most k. $\square$

Acknowledgements. The authors would like to thank Rajesh Chitnis and M. S. Ramanujan for interesting discussions during various stages of the research.

References

1. Banerjee, S., Bhore, S.: Algorithm and hardness results on liar's dominating set and k-tuple dominating set. CoRR, abs/1902.11149 (2019)
2. Bishnu, A., Ghosh, A., Paul, S.: Linear kernels for k-tuple and liar's domination in bounded genus graphs. Discrete Appl. Math. **231**, 67–77 (2017)
3. Chang, G.J.: Algorithmic aspects of domination in graphs. In: Du, D.Z., Pardalos, P.M. (eds.) Handbook of Combinatorial Optimization, pp. 1811–1877. Springer, Boston (1998). https://doi.org/10.1007/978-1-4613-0303-9_28
4. Chellali, M., Favaron, O., Hansberg, A., Volkmann, L.: k-domination and k-independence in graphs: a survey. Graphs Comb. **28**(1), 1–55 (2012)
5. Fink, J.F., Jacobson, M.S.: n-domination in graphs. In: Graph theory with Applications to Algorithms and Computer Science, pp. 283–300. Wiley, Hoboken (1985)
6. Garey, M.R., Johnson, D.S.: Computers and Intractability, vol. 29. W. H. Freeman, New York (2002)
7. Harary, F., Haynes, T.W.: Double domination in graphs. Ars Comb. **55**, 201–214 (2000)
8. Haynes, T.W., Hedetniemi, S., Slater, P.: Fundamentals of Domination in Graphs. CRC Press, Boca Raton (1998)

9. Jallu, R.K., Jena, S.K., Das, G.K.: Liar's dominating set in unit disk graphs. In: Wang, L., Zhu, D. (eds.) COCOON 2018. LNCS, vol. 10976, pp. 516–528. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-94776-1_43
10. Klasing, R., Laforest, C.: Hardness results and approximation algorithms of k-tuple domination in graphs. Inf. Process. Lett. **89**(2), 75–83 (2004)
11. Kushilevitz, E., Nisan, N.: Communication Complexity. Cambridge University Press, Cambridge (1997)
12. Liao, C.S., Chang, G.J.: k-tuple domination in graphs. Inf. Process. Lett. **87**(1), 45–50 (2003)
13. 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
14. Panda, B.S., Paul, S.: Liar's domination in graphs: complexity and algorithm. Discrete Appl. Math. **161**(7–8), 1085–1092 (2013)
15. Rautenbach, D., Volkmann, L.: New bounds on the k-domination number and the k-tuple domination number. Appl. Math. Lett. **20**(1), 98–102 (2007)
16. Roden, M.L., Slater, P.J.: Liar's domination in graphs. Discrete Math. **309**(19), 5884–5890 (2009)
17. Slater, P.J.: Liar's domination. Networks **54**(2), 70–74 (2009)