Parameterized Complexity of Paired Domination

Nikita Andreev[1], Ivan Bliznets[2], Madhumita Kundu[4], Saket Saurabh[3,4], Vikash Tripathi[3](✉), and Shaily Verma[5]

[1] Belgrade, Serbia
[2] University of Groningen, Groningen, The Netherlands
i.bliznets@rug.nl
[3] The Institute of Mathematical Sciences, Chennai, India
{saket,vikasht}@imsc.res.in
[4] University of Bergen, Bergen, Norway
Madhumita.Kundu@uib.no
[5] Hasso Plattner Institute, University of Potsdam, Potsdam, Germany
Shaily.Verma@hpi.de

Abstract. The Paired Domination problem is one of the well-studied variants of the classical Dominating Set problem. In a graph G on n vertices, a dominating set D (set of vertices such that $N[D] = V(G)$) is called a *paired dominating set* of G, if $G[D]$ has perfect matching. In the Paired Domination problem, given a graph G and a positive integer k, the task is to check whether G has a paired dominating set of size at most k. The problem is a variant of the Dominating Set problem, and hence inherits most of the hardness of the Dominating Set problem; however, the same cannot be said about the algorithmic results. In this paper, we study the problem from the perspective of parameterized complexity, both from solution and structural parameterization, and obtain the following results.

1. We design an (non-trivial) exact exponential algorithm running in time $\mathcal{O}(1.7159^n)$.
2. It admits Strong Exponential Time Hypothesis (SETH) optimal algorithm parameterized by the treewidth (tw) of the graph G. The algorithm runs in time $4^{\mathsf{tw}} n^{\mathcal{O}(1)}$; and unless SETH fails, there is no algorithm running in time $(4-\epsilon)^{\mathsf{tw}} n^{O(1)}$ for any $\epsilon > 0$.
3. We design an $4^d n^{O(1)}$ algorithm parameterized by the distance to cluster graphs. We complement this result by proving that the problem does not admit a polynomial kernel under this parameterization and under parameterization by vertex cover number.
4. Paired Domination admits a polynomial kernel on graphs that exclude a biclique $K_{i,j}$.
5. We also prove that one of the counting versions of Paired Domination parameterized by cliquewidth admits $n^{2\mathsf{cw}} n^{O(1)}$ time algorithm parameterized by cliquewidth (cw). However, it does not admit an FPT algorithm unless #SETH is false.

N. Andreev—Independent Researcher, Dubai, UAE.

A. A. Rescigno and U. Vaccaro (Eds.): IWOCA 2024, LNCS 14764, pp. 523–536, 2024.
https://doi.org/10.1007/978-3-031-63021-7_40

Keywords: Paired Domination · Parameterized Complexity · Exact Algorithms · Kernelization

1 Introduction

Given a graph G, a set $D \subseteq V(G)$ is called a *dominating* set if every vertex not in D has a neighbor in D. Given a graph G and an integer k, the Dominating Set problem asks if there exists a dominant set of size at most k. The combinational and algorithmic aspects of the problem are extensively studied in the literature [16,17]. To get an idea of the variants and generalizations of domination, refer to the books [14,15].

The Dominating Set problem generally models a guarding problem in which one needs to use the smallest number of guards to protect a certain object. This object is represented by a graph. Guards can be placed only in vertices, and if a guard occupies vertex v, then he is controlling vertex v and all its neighbors. Under this model, it is natural to require an additional property: each guard has a partner with whom they back up each other. This model was introduced by Haynes and Slater [18]. In this case, the problem is called Paired Domination instead of Dominating Set.

Given a graph G, a dominating set $D \subseteq V$ is called a *paired dominating set* (PD-set for short), if the induced subgraph $G[D]$ has a perfect matching. Let M be a perfect matching in $G[D]$, then for an edge $uv \in M$, we say that u is *paired* with v. Given a graph G and an integer k, the Paired Domination problem asks if G has a PD-set of size at most k. It is important to note that the problem Paired Domination does not generalize Dominating Set. The problem is known to be NP-complete even for restricted graph classes such as split graphs [5], bipartite graphs [5], and planar graphs [23]. For results on Paired Domination, we refer to a survey paper [8] and the book [14]. Although there have been several findings on complexity issues, the exploration of algorithmic approaches to address NP-hardness has not been as extensive. In this paper we fill this gap by studying the problem within the framework of parameterized complexity. We note that Paired Domination problem almost was not studied from the parameterized complexity point of view. The only result, known to us, is obtained by Hanaka et al. [13]. They studied an r-Grouped Dominating Set and for $r = 2$ this problem coincides with the Paired Domination problem. Hanaka et al. [13] presented an $3^{\mathsf{vc}} n^{O(1)}$ algorithm for Paired Domination.

As stated above, the problem is a close variant of the Dominating Set problem. Therefore, it is natural to find similarities and dissimilarities between Dominating Set and Paired Domination. The Dominating Set problem is well studied in the context of parameterized complexity. The problem is known to be a canonical W[2]-hard when parameterized by the solution size [10]. The problem remains W[2]-hard even in some restricted graph classes, including bipartite and split graphs [21], where the parameter is the solution size. On the positive side, the problem is FPT on restricted graph classes such as planar graphs [1,4], nowhere dense graphs [7], d-degenerate graphs [2], $K_{i,j}$-free graphs [20] (a complete bipartite graph with one side of size i and the other of size j), and graphs of bounded genus [12]. The problem is also known to be FPT with respect to structural parameters, such as treewidth [19,22], cliquewidth, and vertex cover number of a graph.

Our Results

We design the first non-trivial exact exponential time algorithm running in time $\mathcal{O}(1.7159^n)$ to compute a minimum cardinality PD-set of a graph. The algorithm first enumerates all the minimal dominating sets of a given graph G in time $\mathcal{O}(1.7159^n)$ and then for each of them uses a subroutine based on maximum matching algorithm to complete it to a PD-set of a graph. Finally, we output the one with a minimum cardinality. Designing an exact algorithm whose execution time is better than the number of minimal dominating sets of a graph G ($\mathcal{O}(1.7159^n)$) is an interesting question.

Next, we consider structural parameters, treewidth (tw), vertex cover number (vc) of a graph, and distance to cluster graphs. Recall that distance to cluster graphs does not exceed vertex cover number. It is known that Dominating Set admits an algorithm with running time $3^{\mathsf{tw}}n^{\mathcal{O}(1)}$; and unless SETH fails, there is no algorithm running in time $(3-\epsilon)^{\mathsf{tw}}n^{O(1)}$ for any $\epsilon > 0$ [19]. We show a similar result for Paired Domination. That is, we give an algorithm that runs in time $4^{\mathsf{tw}}n^{\mathcal{O}(1)}$; and unless SETH fails, there is no algorithm running in time $(4-\epsilon)^{\mathsf{tw}}n^{O(1)}$ for any $\epsilon > 0$. The algorithm is a classical dynamic programming algorithm over graphs of bounded treewidth with a subset convolution trick applied to join node to speed up the computation. Then we design a $4^{\mathsf{d}}n^{\mathcal{O}(1)}$ algorithm where d is the distance to cluster graphs. We complement this result by proving that the problem does not admit a polynomial kernel when parameterized by the vertex cover number. As $\mathsf{d} \leq \mathsf{vc}$ we conclude that the Paired Domination problem does not admit a polynomial kernel when parameterized by the distance to cluster graphs.

One of the largest family of graphs where Dominating Set is known to admit a FPT algorithm parameterized by the solution size k is $K_{i,j}$-free graphs. We show that Paired Domination behaves similarly to Dominating Set on biclique-free graphs. Towards this, we design a polynomial kernel for Paired Domination on graphs that exclude a biclique $K_{i,j}$. The kernelization algorithm is inspired by the kernelization algorithm designed by Philip et al. [20] for the Dominating Set problem.

All graphs considered in this paper are simple and finite. For the basic graph theoretic notations and definitions refer to the book [9], and for the notations and definitions related to parameterized complexity refer to the book [6]. For $k \geq 1$, an integer, we use $[k]$ to denote the set of integers $\{1, 2, \ldots, k\}$.

2 Exact Exponential Time Algorithm

In this section, we design an exact exponential-time algorithm for Paired Domination. Note that if a graph G contains isolated vertices, then there is no PD-set in G; that is, G is a No-instance. Therefore, we assume that the graph G has no isolated vertex. We first design an algorithm that, given a graph G and a dominating set D, constructs a PD-set D' such that $D \subseteq D'$.

```
Algorithm 1. Dom-set to PD-set
Input: A graph $G$ without an isolated vertex and a dominating set $D$ of $G$.
Initialize: $D' = D$;
begin
    Compute a maximum matching $M$ of induced subgraph $G[D']$;
    Let $A = D' \setminus V(M)$;
    while ($A \neq \emptyset$) do
        Pick a vertex $v \in D'$;
        if ($N_G(v) \subseteq D'$) then
            $D' = D' \setminus \{v\}$;
            $A = A \setminus \{v\}$;
        else
            Let $u \in N_G(v) \setminus D'$;
            $D' = D' \cup \{u\}$;
            $A = A \setminus \{v\}$;
    return $D'$
```

Lemma 1 (♣[1]). *Let G be a graph, D be a PD-set of G, and $D' \subseteq D$ be a minimal dominating set of G. If S is a PD-set of G constructed from D' using Algorithm 1, then $|S| \leq |D|$.*

The above lemma concludes that if D is a minimum size PD-set of a graph G and D' is a minimal dominating set of G contained in D, then we can obtain another PD-set S of G containing D' such that $|S| \leq |D|$. More specifically, as D is a minimum-sized PD-set of G, $|S| = |D|$. Thus, if for a graph G, $D_1, D_2, \ldots, D_r$ are all possible minimal dominating sets of G, and $S_1, S_2, \ldots, S_r$ are the corresponding PD-sets such that $D_i \subseteq S_i$, and D is a minimum size PD-set of G, then $D = S_i$, where S_i has minimum size among the sets $S_1, S_2, \ldots, S_r$. Fomin et al. [11] propose an algorithm that enumerates all minimal dominating sets of a graph in time $\mathcal{O}(1.7159^n)$. Formally the result is stated below.

Theorem 1. [11] *For a graph G on n vertices, all the minimal dominating sets of G can be enumerated in time $\mathcal{O}(1.7159^n)$.*

Using the above results, we obtain the following theorem.

Theorem 2. *For a graph G on n vertices,* Paired Domination *can be solved in time $\mathcal{O}(1.7159^n)$.*

[1] Proofs of the results marked with (♣) are omitted due to space constraint.

3 Parameterization by Treewidth and Pathwidth

In this section, we give a dynamic programming-based FPT algorithm for Paired Domination parameterized by the treewidth of the input graph G and prove matching lower bound assuming SETH. To compute a minimum cardinality PD-set of a graph G, we use a nice tree decomposition of G with an edge introduce node. For the definition and properties of tree decomposition, see [6]. We use the following notations in our algorithm. Consider two nonempty sets, A and B. Then $A \setminus B = \{x \in A \mid x \notin B\}$. Let $f : A \mapsto B$ be a function. Then for $a \in A$, the function $f_{a \mapsto b} : A \mapsto B$ is defined as $f_{a \mapsto b}(x) = f(x)$ for all $x \in A \setminus \{a\}$ and $f_{a \mapsto b}(a) = b$, and the function $f_{a_1 \mapsto b_1, a_2 \mapsto b_2} : A \mapsto B$ is defined as $f_{a_1 \mapsto b_1, a_2 \mapsto b_2}(x) = f(x)$ for all $x \in A \setminus \{a_1, a_2\}$ and $f_{a_1 \mapsto b_1, a_2 \mapsto b_2}(a_1) = b_1$ and $f_{a_1 \mapsto b_1, a_2 \mapsto b_2}(a_2) = b_2$. For a set $A' \subseteq A$, the *restriction* of f denoted by $f_{A'}$ is defined as $f_{A'}(a) = f(a)$ for each $a \in A'$. In this case, the function f is called an *extension* of $f_{A'}$.

Let $(T, \{X_t\}_{t \in V(T)})$ be a tree decomposition of the given graph G with width tw. For a node $t \in V(T)$, let T_t denotes the subtree of T rooted at t and $G_t = (V_t, E_t)$ be the graph associated with T_t, where $V_t = \bigcup_{t \in V(T_t)} X_t$ and $E_t = \{e \mid e \text{ is introduced in the subtree rooted at } t\})$. Note that $V_r = V(G)$ and $G_r = G$, where r is the root node of T. We define a function $f : X_t \mapsto \{1, \hat{1}, 0, \hat{0}\}$ that labels the vertices in bag X_t. Each label in the set $\{1, \hat{1}, 0, \hat{0}\}$ has the following definition:

1 – vertices that are mapped to 1 are part of the partial solution $D \subseteq V_t$ and are paired with a vertex in D.
$\hat{1}$ – vertices that are mapped to $\hat{1}$ are part of the partial solution $D \subseteq V_t$ but are not paired in D.
0 – vertices that are mapped to 0 are not part of the partial solution $D \subseteq V_t$ but must be dominated by D.
$\hat{0}$ – the vertices mapped to $\hat{0}$ are not part of the partial solution $D \subseteq V_t$ and do not need to be dominated by D.

Let t be a node in the tree T, and X_t be the corresponding bag. There are $4^{|X_t|}$ possible labelling of X_t. A set $D \subseteq V_t$ *respects* a function $f : X_t \mapsto \{1, \hat{1}, 0, \hat{0}\}$ on X_t, if D satisfies the following properties:

1. $D \cap X_t = f^{-1}(1) \cup f^{-1}(\hat{1})$.
2. D dominates all the vertices in the set $V_t \setminus f^{-1}(\hat{0})$.
3. There is a perfect matching M in $G[D \setminus f^{-1}(\hat{1})]$, that is, matching M saturates all the vertices in $D \setminus f^{-1}(\hat{1})$. In this case, we say that vertices in $D \setminus f^{-1}(\hat{1})$ are paired.

For a node $t \in V(T)$ and a function $f : X_t \mapsto \{1, \hat{1}, 0, \hat{0}\}$, we define $\text{PD}[t, f]$ to be the minimum cardinality of a set D such that D respects f on X_t. If for a node t and labeling f of X_t, no such minimum cardinality set D exists, then we set $\text{PD}[t, f] = +\infty$. Depending on the node type, we now compute the value of $\text{PD}[t, f]$ for each node $t \in V(T)$ and $f : X_t \mapsto \{1, \hat{1}, 0, \hat{0}\}$, traversing the tree T in a bottom-up manner.

1. **If t is a leaf node**: By the definition of nice tree decomposition $X_t = \emptyset$. It implies that each color class is an empty set in any labeling f of X_t. Moreover, $\text{PD}[t, f] = 0$, as $V_t = \emptyset$.
2. **If t is a vertex introduce node**: Since t is a vertex introduce node, t has exactly one child t' such that $X_t = X_{t'} \cup \{v\}$, where $v \notin X_{t'}$. We observe that $v \notin V_{t'}$ by the definition of nice tree decomposition. Since there is no edge incident on v introduced at the current node t, v cannot be dominated or paired in G_t. Consider a labelling $f : X_t \mapsto \{1, \hat{1}, 0, \hat{0}\}$ of X_t, then:

$$\text{PD}[t, f] = \begin{cases} +\infty & \text{if } f(v) = 1; \\ 1 + \text{PD}[t', f_{X_{t'}}] & \text{if } f(v) = \hat{1}; \\ +\infty & \text{if } f(v) = 0; \\ \text{PD}[t', f_{X_{t'}}] & \text{if } f(v) = \hat{0}. \end{cases}$$

3. **If t is an edge introduce node**: Since t is an edge introduce node, t has exactly one child, say t', such that $X_t = X_{t'}$ and there is an edge $uv \in E(G)$ that is introduced at node t. That is, $uv \in G_t$ but $uv \notin G_{t'}$. For a labeling $f : X_t \mapsto \{1, \hat{1}, 0, \hat{0}\}$, we derive the following recurrence based on the labeling of vertices u and v.

$$\text{PD}[t, f] = \begin{cases} \min\{\text{PD}[t', f_{u \mapsto \hat{1}, v \mapsto \hat{1}}], \text{PD}[t', f]\} & \text{if } (f(u), f(v)) = (1, 1); \\ \text{PD}[t', f_{v \mapsto \hat{0}}] & \text{if } (f(u), f(v)) \in \{(1, 0), (\hat{1}, 0)\}; \\ \text{PD}[t', f_{u \mapsto \hat{0}}] & \text{if } (f(u), f(v)) \in \{(0, 1), (0, \hat{1})\}; \\ \text{PD}[t', f] & \text{otherwise.} \end{cases}$$

4. **If t is a forget node**: By the definition of nice tree decomposition, t has exactly one child t' such that $X_t = X_{t'} \setminus \{v\}$ for some vertex $v \in X_{t'}$. Note that v does not appear in any bag above X_t in T as $v \notin X_t$. Therefore, v should be dominated (or belongs to the dominating set) in $G_{t'}$. We give the following recurrence: $\text{PD}[t, f] = \min\{\text{PD}[t', f_{v \mapsto 1}], \text{PD}[t', f_{v \mapsto 0}]\}$.
   Note that every path decomposition of a graph G is also a tree decomposition with no join node. Moreover, for introduce node, introduce edge node, and forget node we considering $4^{|X_t|}$ labellings of X_t and for each labelling f, $\text{PD}[t, f]$ can be computed in time $|V(G)|^{\mathcal{O}(1)}$. From this we can infer that Paired Domination can be solved in time $\mathcal{O}(4^{\mathsf{pw}} |V(G)|^{\mathcal{O}(1)})$.
5. **If t is a join node**: By the definition of nice tree decomposition, t has exactly two children t_1 and t_2 such that $X_t = X_{t_1} = X_{t_2}$. Let f_1, f_2 and f be labellings of X_{t_1}, X_{t_2}, and X_t, respectively. We say that f_1 and f_2 are *compatible* with f, if the following conditions hold, for all $v \in X_t$:
   (a) $f(v) = 1$ if and only if $(f_1(v), f_2(v)) \in \{(1, \hat{1}), (\hat{1}, 1)\}$,
   (b) $f(v) = \hat{1}$ if and only if $(f_1(v), f_2(v)) = (\hat{1}, \hat{1})$,
   (c) $f(v) = 0$ if and only if $(f_1(v), f_2(v)) \in \{(0, \hat{0}), (\hat{0}, 0)\}$,
   (d) $f(v) = \hat{0}$ if and only if $(f_1(v), f_2(v)) = (\hat{0}, \hat{0})$.

Observe that there are 6 choices for a compatible triplet $(f(v), f_1(v), f_2(v))$ and so 6^{tw} choices for all the vertices in bag X_t. We use subset convolution for faster computation of the compatible functions. For this, we rewrite the above four conditions for compatible functions as follows:

(a) $f^{-1}(1) = f_1^{-1}(1) \cup f_2^{-1}(1)$,
(b) $f_1^{-1}(1) \cap f_2^{-1}(1) = \emptyset$,
(c) $f^{-1}(0) = f_1^{-1}(0) \cup f_2^{-1}(0)$,
(d) $f_1^{-1}(0) \cap f_2^{-1}(0) = \emptyset$.

We would fix the set of vertices mapped to 1 or $\hat{1}$ and apply the subset convolution to compute such functions. Let us fix $R \subseteq X_t$ and let $\mathcal{F}(R)$ be the set of all functions f such that $f^{-1}(1) \cup f^{-1}(\hat{1}) = R$. Recall that we want to compute the value of $\text{PD}[t, f]$ for all $f \in \mathcal{F}(R)$. We can represent each function in $\mathcal{F}(R)$ as two sets $S \subseteq X_t \setminus R$ and $Q \subseteq R$ such that S is a pre-image of 0 and Q is a pre-image of 1. Thus, a function f represented by S and Q can be defined as:

$$\Phi_S^Q(x) = \begin{cases} 1 & \text{if } x \in Q; \\ \hat{1} & \text{if } x \in R \setminus Q; \\ 0 & \text{if } x \in S; \\ \hat{0} & \text{if } x \in X_t \setminus (R \cup S). \end{cases}$$

Therefore, for every labeling $f \in \mathcal{F}(R)$ we have:

$$\text{PD}[t, f] = \min_{Q \subseteq R} \left\{ \min_{\substack{A \cup B = f^{-1}(0) \\ A \cap B = \emptyset}} \left\{ \text{PD}[t_1, \Phi_A^Q] + \text{PD}[t_2, \Phi_B^Q] - |R| \right\} \right\}.$$

We show that using subset convolution computations at join node can be done in $4^{\mathsf{tw}} n^{\mathcal{O}(1)}$ time. The correctness proofs of each recurrence and running time analysis are omitted due to space constraints.

Theorem 3 (♣). *Given a graph G of treewidth* tw*,* Paired Domination *can be solved in $\mathcal{O}(4^{\mathsf{tw}} \cdot |V(G)|^{\mathcal{O}(1)})$ time.*

Now we prove a matching lower bound.

Theorem 4. *Unless SETH fails, there is no algorithm for* Paired Domination *with running time $\mathcal{O}^*((4-\epsilon)^{pw})$ for any $\epsilon > 0$, where pw is the pathwidth of the input graph.*

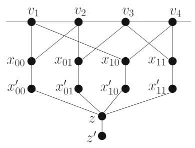

Fig. 1. Gadget G'

Proof. From SETH, it follows that there is no $\mathcal{O}^*\left((2-\varepsilon)^n\right)$ running time algorithm for SAT, where n is the number of variables in the input formula. To establish our lower bound, we transform a given CNF formula F with n variables and m clauses into a graph G with a path decomposition of width at most $k = \frac{n}{2} + C$ for some constant C. Moreover, F is satisfiable if and only if the graph G has a PD-set of a special size, which we will determine later. Therefore, if we can solve PD-set in $\mathcal{O}^*\left(4^{\lambda k}\right)$ time for $\lambda < 1$, then we would have a $\mathcal{O}^*\left(4^{\lambda(n/2+C)}\right) = \mathcal{O}^*\left((2-\varepsilon)^n\right)$ running time algorithm for SAT, which leads to a contradiction. Now we proceed with the details of the transformation.

We assume that $F = C_1 \wedge C_2 \wedge \cdots \wedge C_m$. Without loss of generality, we assume that F depends on variables $x_1, x_2, \ldots, x_{n-1}, x_n$, where $n = 2p$. For each $j \in [p]$ we consider a pair of variables x_{2j-1}, x_{2j}. For each such pair of variables, we assign a gadget G' shown in Fig. 1. Consider all four possible assignments for these pairs of variables: $00, 01, 10, 11$. We associate these assignments with the following pairs of vertices $\{v_3, v_4\}$, $\{v_1, v_4\}$, $\{v_2, v_3\}$, $\{v_1, v_2\}$ in the corresponding gadget. We note that if an assignment ab corresponds to a pair $\{v_i, v_j\}$, then the vertex x_{ab} is not connected to vertices v_i, v_j in the gadget. For each pair of vertices x_{2j-1}, x_{2j} we create a chain of gadgets G' such that the vertex v_1 of one gadget is connected to the vertex v_4 of the preceding gadget. Each such chain contains m gadgets, and the i-th gadget corresponds to a clause C_i from F. For each $i \in [m]$, we create a vertex named C_i for the clause C_i. We connect the vertex x'_{ab} from the i-th gadget of a the pair x_{2j-1}, x_{2j} with the vertex C_i if the assignment $x_{2j-1} = a, x_{2j} = b$ satisfies clause C_i. We denote obtained construction by H. H is schematically shown in the left square of Fig. 2. We repeat this construction, H, exactly $\frac{3}{2}n + 1$ times, and connect these parts in a chain fashion, as shown in Fig. 2. Additionally, we add vertices l, l', r, r', and edges (l, r), (l, l'), (r, r'). The first vertices of the leftmost H (the left blue rectangle in Fig. 2) are combined into a clique and are connected respectively to l. Similarly, the last vertices in the right-most H (the right blue rectangle in Fig. 2) generate a clique and are connected to the vertex r. This completes the construction. The entire construction is depicted schematically in Fig. 2. In Fig. 2, each block of four consecutive vertices corresponds to the vertices v_1, v_2, v_3, v_4 of the gadget G'. The remaining vertices from the gadget are not shown in the figure to avoid clutter.

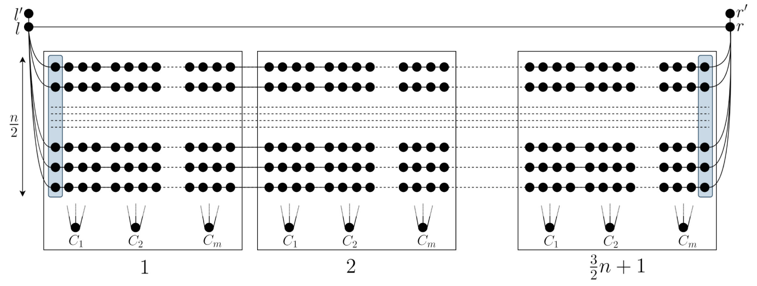

Fig. 2. General scheme of the constructed graph

Now, we are ready to prove the following lemma.

Lemma 2 (♣). *Formula F has a satisfying assignment if and only if the constructed graph G has a PD-set of size at most* $2nm\left(\frac{3}{2}n + 1\right) + 2$.

We have demonstrated that the constructed instance of PD-set is equivalent to the original formula F. Now, we need to verify that the constructed graph indeed has a pathwidth of at most $\frac{n}{2} + C$.

Lemma 3 (♣). *For each formula F, the constructed graph G (see Fig. 2) has a pathwidth of at most $\frac{n}{2} + C$ for some constant C.*

From Lemmas 2 and 3 we deduce the statement of Theorem 4. □

4 Parameterization by Distance to Cluster Graphs

A graph is called a cluster graph if all its connected components are cliques (i.e., complete graphs). We say that a graph G has a distance d to a cluster graph if there exists a subset $X \subseteq V(G)$ such that $|X| \leq \mathsf{d}$, and $G - X$ is a cluster graph.

It is trivial to find a PD-set in cluster graphs in polynomial time if there exists one. Hence, it is natural to consider Paired Domination parameterized by the distance to cluster graphs.

Theorem 5. Paired Domination *can be solved in $\mathcal{O}^*(4^{\mathsf{d}})$ time, where d is the distance of the input graph to a cluster graph.*

Proof. Recall that a graph is a cluster graph if and only if it does not contain a P_3 (a path of length three) as an induced subgraph. Consequently, a minimum-size modulator to cluster graphs can be found in $\mathcal{O}^*(3^{\mathsf{d}})$ time, where d is the distance to cluster graphs. From now on we assume that we are given a modulator to cluster graphs, denoted as $X \subset V(G)$, such that $G \setminus X$ is a cluster graph and $|X| = \mathsf{d}$. Let us denote $G - X$ by C. Without loss of generality, we assume that $C = C_1 \sqcup C_2 \sqcup \cdots \sqcup C_k$ where for each $i \in [k]$ graph C_i is a clique in $G - X$.

Our goal is to find a set of vertices D such that D is a dominating set, and there exists a perfect matching in $G[D]$. We represent D as $D_1 \cup D_2$, where $D_1 = D \cap X$, $D_2 = D \cap C$. Since we do not know the actual value of D or D_1, we consider 2^{d} potential values for $D_1 \subseteq X$. For each such set D_1, we find the smallest set $D_2' \subseteq C$ such that $D_1 \cup D_2'$ forms a PD-set. Subsequently, we output the smallest among these PD-sets. Note that $D_1 \cup D_2'$ is a PD-set if (i) D_2' dominates the set $X \setminus N[D_1]$, (ii) D_2' contains at least one vertex in a clique C_i if $C_i \not\subseteq N_G(D_1)$, (iii) $G[D_1 \cup D_2']$ contains a perfect matching.

In order to find such D_2' for each potential value of D_1, we employ a dynamic programming. We order the vertices of the cluster graph C such that firstly we list all vertices of clique C_1, then vertices of C_2 and so on up to C_k. Let $v_1, v_2, \ldots, v_{|C|}$ be the ordering. By $C[i]$, we denote a maximal clique in C that contains the vertex v_i, i.e., $C[i] = C_j$, if the vertex v_i is contained in C_j. We consider a function PDS with arguments i, x, y, S', D', where $i \in [|C|]$, $x, y \in \{0, 1\}$, $S' \subset X \setminus N_G[D_1]$, $D' \subseteq D_1$. The function $PDS(i, x, y, S', D')$ denotes any subset $Y \subseteq \{v_1, v_2, \ldots, v_i\}$ of minimum size such that:

- Let $C[i] = C_j$ i.e., the vertex v_i is contained in clique C_j then for any $k < j$ we have $Y \cap C_k \neq \emptyset$ or $C_k \subseteq N_G[D_1]$;
- if $x = 1$, then $C[i] \cap Y \neq \emptyset$;
- if $y = 0$, then $G[D' \cup Y]$ has a perfect matching;
- if $y = 1$ then $G[D' \cup Y]$ has a matching that covers all vertices except one vertex from $C_j \cap Y$ (recall that $(C[i] = C_j)$);
- The set Y dominates the set S'.

If such Y does not exist, then $PDS(i, x, y, S', D') = null$. It is easy to see that the required D'_2 is given by $PDS(|C|, x, 0, X \setminus N_G[D_1], D_1)$, where $x = 1$ if $C[|C|] \not\subseteq N_G[D_1]$ (i.e., the last clique is not fully dominated by D_1), otherwise $x = 0$. Now we show how to calculate the values $PDS(i, x, y, S', D')$ using dynamic programming.

Base Cases: If $S' = \emptyset, x = 0, y = 0$ and there exists a perfect matching in D' then $PDS(0, x, y, S', D') = \emptyset$; otherwise, $PDS(0, x, y, S', D') = null$.

To compute the value in our dynamic programming table for a cell (i, x, y, S', D') we consider two cases: 1) v_i is not the first vertex in the clique $C[i]$, 2) v_i is the first vertex in the clique $C[i]$.

Case 1: Vertex v_i is not the first vertex in the clique $C[i]$.

If vertex v_i does not belong to Y, then we have $Y = PDS(i-1, x, y, S', D')$. Otherwise, $v_i \in Y$ and we explore several potential scenarios:

1. The vertex v_i has a pair u among its neighbors in D' (i.e., edge $v_i u$ is part of the matching in a paired dominating set). In this case, $Y = \{v_i\} \cup PDS(i-1, 0, y, S' \setminus N_G[v_i], D' \setminus \{u\})$.
2. The vertex v_i matches with some vertex $v_j \in C[i]$, where $j < i$. If $y = 0$, then $v_j \in (Y \cap C[i]) \setminus \{v_i\}$. Hence, $Y = \{v_i\} \cup PDS(i-1, 0, 1, S' \setminus N_G[v_i], D')$. If $y = 1$, then there is a vertex v_k such that $v_k \in Y \cap C[i]$, $k \neq i, k \neq j)$, and v_k belongs to the paired dominating set but does not have a pair yet, i.e. its pair is a vertex v_q. Therefore, in $(D \cap C[i]) \setminus \{v_i\}$, there are two unpaired vertices, v_j and v_k. In this situation, we have edges $v_j v_i$ and $v_k v_q$ as edges in the matching within the paired dominating set. However, we can replace these edges in a matching within paired dominating set with edges $v_j v_k$, $v_i v_q$. Thus, in this case, the following recurrence relation holds: $Y = \{v_i\} \cup PDS(i-1, 0, 0, S' \setminus N_G[v_i], D')$. Hence, cases $y = 0$ and $y = 1$ can be combined by the following formula: $Y = \{v_i\} \cup PDS(i-1, 0, 1-y, S' \setminus N_G[v_i], D')$.
3. The vertex v_i is the only unpaired vertex in the clique. In that case, $y = 1$. Similarly to the previous case, we get the following formula: $Y = \{v_i\} \cup PDS(i-1, 0, 1-y, S' \setminus N_G[v_i], D')$.

Combining all the subcases in this case, we obtain the following recurrence:

$$\begin{aligned} PDS(i, x, y, S', D') = \min\{&PDS(i-1, x, y, S', D'), \\ &\{v_i\} \cup PDS(i-1, 0, 1-y, S' \setminus N_G[v_i], D'), \\ \min_{u \in N_G[v_i] \cap D'} &\{\{v_i\} \cup PDS(i-1, 0, y, S' \setminus N_G[v_i], D' \setminus u)\}.\} \end{aligned}$$

Case 2: The vertex v_i is the first vertex in its clique, i.e., $C[i-1] \neq C[i]$. We introduce a new variable x_1. If $i > 1$ and $C[i-1] \not\subseteq N_G[D_1]$ we set $x_1 = 1$; otherwise, we set $x_1 = 0$.

Depending on the values of x and y, we show how to compute $PDS(i, x, y, S', D')$. In the case when $x = y = 0$, we either can take v_i or not. If we do not take v_i then we have $PDS(i,0,0,S',D') = PDS(i-1,x_1,0,S',D')$ since $v_i \notin Y$. If we take v_i, then v_i must have a pair inside D'. Hence, in this case $PDS(i,0,0,S',D') = \min_{u \in N_G[v_i] \cap D'}\{v_i\} \cup PDS(i-1,x_1,0,S' \setminus N_G[v_i], D' \setminus u)$. Combining two cases we have $PDS(i,0,0,S',D') = \min\{PDS(i-1,x_1,0,S',D'), \min_{u \in N_G[v_i] \cap D'}\{v_i\} \cup PDS(i-1,x_1,0,S' \setminus N_G[v_i], D' \setminus u)\}$. If $x=1$ and $y=0$, then $v_i \in Y$. Hence, for similar reasons as for the case $x=y=0$ we have $PDS(i,1,0,S',D') = \min_{u \in N_G[v_i] \cap D'}\{PDS(i-1,x_1,0,S' \setminus N_G[v_i], D' \setminus u)\}$. If $y=1$, then $v_i \in Y$. Therefore, we have $PDS(i,0,1,S',D') = PDS(i,1,1,S',D') = \{v_i\} \cup PDS(i-1,x_1,0,S' \setminus N_G[v_i], D')$.

Summarizing, we have $PDS(i,x,y,S',D') =$

$$\begin{cases} \min\{PDS(i-1,x_1,0,S',D'), & \\ \quad \min_{u \in N_G[v_i] \cap D'} \{v_i\} \cup PDS(i-1,x_1,0,S' \setminus N_G[v_i], D' \setminus u)\}, & \text{if } x=y=0; \\ \min_{u \in N_G[v_i] \cap D'}\{PDS(i-1,x_1,0,S' \setminus N_G[v_i], D' \setminus u)\}, & \text{if } x=1, y=0; \\ \{v_i\} \cup PDS(i-1,x_1,0,S' \setminus N_G[v_i], D'), & \text{otherwise.} \end{cases}$$

In all cases we showed how to compute $PDS(i,x,y,S',D')$ based on previous values. Now we proceed with a running time analysis. Note that for a fixed $D_1 \subseteq X$, the size of the dynamic programming table is at most $|C| \cdot 2 \cdot 2 \cdot 2^{|X \setminus N_G[D_1]|} \cdot 2^{|D_1|} = \mathcal{O}^*(2^{|X|}) = \mathcal{O}^*(2^d)$. Therefore, the total running time of the algorithm is at most $\mathcal{O}^*(4^{\mathsf{d}})$. □

To complement the previous result, we show that Paired Domination does not admit a polynomial kernel parameterized by the distance to cluster graphs (d). Actually, we prove a stronger result using another parameter, that is, the vertex cover size of the input graph as $\mathsf{d} \leq \mathsf{vc}$. We derive a *polynomial parameter transformation* from the Red-Blue Dominating set problem to accomplish this task. Given a bipartite graph $G=(R,B,E)$ and an integer k, where R and B are the set of red vertices and blue vertices, respectively, the Red-Blue Dominating set problem asks to if there exists a set $R' \subseteq R$ (called a red-blue dominating set) of size at most k such that R' dominates all the vertices in set B, that is, $N(R') = B$. It is known that the Red-Blue Dominating set problem does not admit a polynomial kernel parameterized by the cardinality of R unless NP $\subseteq$ co-NP/poly, see [6]. For the definitions and terminology of polynomial parameter transformation, please refer to the book [6]. Now, we are ready to prove the following result:

Theorem 6. *The* Paired Domination *does not have a polynomial kernel when parameterized by vertex cover size, unless* NP $\subseteq$co-NP/poly.

Proof. Given an instance $(G=(R,B,E),k)$ of Red-Blue Dominating set parameterized by $|R|$, we construct an instance (G',k',ℓ) of Paired Domination parameterized by ℓ, where ℓ is the size of vertex cover as follows:

Take two copies of the set R, say R_1 and R_2, and two copies of B, say B_1 and B_2. For a vertex $r \in R$, let r_1 and r_2 be the copies of the vertex r in R_1 and R_2, respectively. Similarly, for a vertex $b \in B$, let b_1 and b_2 be the copies of the vertex b in B_1 and B_2, respectively. If $rb \in E(G)$ for some $r \in R$ and $b \in B$, we add edges r_1b_1 and r_2b_2 in G'. Furthermore, for each $r \in R$, we make r_1 and r_2 adjacent in G'. Next, we take a path on three vertices a, b, and c and make c adjacent to all vertices in $R_1 \cup R_2$. It is easy to observe that the size of a vertex cover of G' is at most $2|R| + 2$ that is $\ell \leq 2|R| + 2$. Let $k' = 2k + 2$.

Next, we prove that the graph G has a red-blue dominating set of size at most k if and only if G' has a PD-set of size at most $2k + 2$.

For the forward direction, let S be a red-blue dominating set of G' of size at most k. It is easy to see that the set $D = \{r_1, r_2 \mid r \in S\} \cup \{b, c\}$ is a PD-set of G' such that $|D| \leq 2k + 2$.

For the reverse direction, consider a PD-set D of G' of size at most $2k + 2$. To dominate the vertex a, either a or b must be in D. Furthermore, a vertex in $\{a, b\}$ can only be paired with another vertex in $\{a, b, c\}$. Thus, we have $|D \cap \{a, b, c\}| \geq 2$. Therefore, if $a, b \in D$, we can safely replace the vertex a with c in D. Thus, assume that $c \in D$. Note that $|D \setminus \{a, b, c\}| \leq 2k$. Let $D_1 = D \cap (R_1 \cup B_1)$ and $D_2 = D \cap (R_2 \cup B_2)$. Then either $|D_1| \leq k$ or $|D_2| \leq k$. Without loss of generality, let us assume that $|D_1| \leq k$. Note that a vertex in B_1 can only be dominated by itself or a neighbor in R_1, and all the vertices in R_1 are already dominated by c, as $c \in D$. Thus, D_1 dominates the set B_1. Observe that if $b_1 \in B_1 \cap D_1$, then for some $r_1 \in N_{G'}(b_1)$, the updated set $D_1 = D_1 \setminus \{b_1\} \cup \{r_1\}$ also dominates all the vertices in B_1. Repeating this process ensures that D_1 dominates all vertices in B_1 and contains no vertex from B_1. Since the graph induced in $R_1 \cup B_1$ is an exact copy of G, the set $S = \{r \mid r_1 \in D_1\}$ is a red-blue dominating set of size at most k. Hence, Paired Domination does not admit a polynomial kernel when parameterized by vertex cover. □

5 $K_{i,j}$-Free Graphs, Counting Version Parameterized by Cliquewidth

We show that Paired Domination behave similarly to Dominating Set on $K_{i,j}$-free graphs. It is known that Dominating Set admits a polynomial kernel on $K_{i,j}$-free graphs [20]. We adopt their technique for Paired Domination.

Theorem 7 (♣). *For fixed $j \geq i \geq 1$, Paired Domination admits a polynomial kernel on a graph that excludes $K_{i,j}$ as a subgraph.*

We also consider the counting version of Paired Domination parameterized by cliquewidth. Specifically, we prove the following result.

Theorem 8 (♣). *There exists a constant $c \in \mathbf{N}$ such that the following holds: Assuming #SETH, there is no integer $k \geq 1$ such that we can count the number of minimum paired dominating sets with matchings (where the same paired dominating set is counted several times depending on how many perfect matchings it has) in time $\mathcal{O}(n^{k-c})$ on an n-vertex graph G given together with a k-expression.*

Furthermore, from Theorem 8, it immediately follows that computing the number of minimum paired dominating sets with different matching inside is unlikely to be in FPT under parameterization by cliquewidth. However, it is possible to count the number of minimum dominating sets in $\mathcal{O}^*(4^{cw})$ as was shown in [3].

Theorem 9 (♣). *Given a graph G and a k-expression G, that introduce each edge exactly once, we can compute the number of matching that generate a paired dominating sets of size ℓ in $\mathcal{O}(n^{2k+C})$ time for some constant $C > 0$.*

References

1. Alber, J., Fellows, M.R., Niedermeier, R.: Polynomial-time data reduction for dominating set. J. ACM **51**(3), 363–384 (2004)
2. Alon, N., Gutner, S.: Linear time algorithms for finding a dominating set of fixed size in degenerated graphs. Algorithmica **54**(4), 544–556 (2009)
3. Bodlaender, H.L., van Leeuwen, E.J., van Rooij, J.M.M., Vatshelle, M.: Faster algorithms on branch and clique decompositions. In: Hliněný, P., Kučera, A. (eds.) MFCS 2010. LNCS, vol. 6281, pp. 174–185. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-15155-2_17
4. Chen, J., Fernau, H., Kanj, I.A., Xia, G.: Parametric duality and kernelization: lower bounds and upper bounds on kernel size. In: Diekert, V., Durand, B. (eds.) STACS 2005. LNCS, vol. 3404, pp. 269–280. Springer, Heidelberg (2005). https://doi.org/10.1007/978-3-540-31856-9_22
5. Chen, L., Lu, C., Zeng, Z.: Labelling algorithms for paired-domination problems in block and interval graphs. J. Comb. Optim. **19**(4), 457–470 (2010)
6. Cygan, M., et al.: Parameterized Algorithms. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-21275-3
7. Dawar, A., Kreutzer, S.: Domination problems in nowhere-dense classes of graphs. In: Foundations of Software Technology and Theoretical Computer Science—FSTTCS 2009, LIPIcs. Leibniz International Proceedings in Informatics, vol. 4, pp. 157–168. Schloss Dagstuhl – Leibniz Center for Informatics, Wadern (2009)
8. Desormeaux, W.J., Henning, M.A.: Paired domination in graphs: a survey and recent results. Util. Math. **94**, 101–166 (2014)
9. Diestel, R.: Graph Theory. Graduate Texts in Mathematics, vol. 173, 4th edn. Springer, Cham (2012)
10. Downey, R.G., Fellows, M.R.: Fixed-parameter tractability and completeness I: basic results. SIAM J. Comput. **24**(4), 873–921 (1995)
11. Fomin, F.V., Grandoni, F., Pyatkin, A.V., Stepanov, A.A.: Combinatorial bounds via measure and conquer: bounding minimal dominating sets and applications. ACM Trans. Algorithms **5**(1), 9:1–9:17 (2008)
12. Fomin, F.V., Thilikos, D.M.: Fast parameterized algorithms for graphs on surfaces: linear kernel and exponential speed-up. In: Díaz, J., Karhumäki, J., Lepistö, A., Sannella, D. (eds.) ICALP 2004. LNCS, vol. 3142, pp. 581–592. Springer, Heidelberg (2004). https://doi.org/10.1007/978-3-540-27836-8_50
13. Hanaka, T., Ono, H., Otachi, Y., Uda, S.: Grouped domination parameterized by vertex cover, twin cover, and beyond. In: Mavronicolas, M. (ed.) CIAC 2023. LNCS, vol. 13898, pp. 263–277. Springer, Cham (2023). https://doi.org/10.1007/978-3-031-30448-4_19

14. Haynes, T.W., Hedetniemi, S.T., Henning, M.A. (eds.): Topics in Domination in Graphs. Developments in Mathematics, vol. 64. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-51117-3
15. Haynes, T.W., Hedetniemi, S.T., Henning, M.A. (eds.): Structures of Domination in Graphs. Developments in Mathematics, vol. 66. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-58892-2
16. Haynes, T.W., Hedetniemi, S.T., Slater, P.J. (eds.): Domination in Graphs: Advanced Topics, Monographs and Textbooks in Pure and Applied Mathematics, vol. 209
17. Haynes, T.W., Hedetniemi, S.T., Slater, P.J.: Fundamentals of Domination in Graphs. Monographs and Textbooks in Pure and Applied Mathematics, vol. 208. Marcel Dekker Inc., New York (1998)
18. Haynes, T.W., Slater, P.J.: Paired-domination in graphs. Netw. Int. J. **32**(3), 199–206 (1998)
19. Lokshtanov, D., Marx, D., Saurabh, S.: Known algorithms on graphs of bounded treewidth are probably optimal. ACM Trans. Algorithms **14**(2), Article no. 13, 30 (2018)
20. Philip, G., Raman, V., Sikdar, S.: Polynomial kernels for dominating set in graphs of bounded degeneracy and beyond. ACM Trans. Algorithms **9**(1), 11:1–11:23 (2012)
21. Raman, V., Saurabh, S.: Short cycles make W-hard problems hard: FPT algorithms for W-hard problems in graphs with no short cycles. Algorithmica **52**(2), 203–225 (2008)
22. van Rooij, J.M.M., Bodlaender, H.L., Rossmanith, P.: Dynamic programming on tree decompositions using generalised fast subset convolution. In: Fiat, A., Sanders, P. (eds.) ESA 2009. LNCS, vol. 5757, pp. 566–577. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-04128-0_51
23. Tripathi, V., Kloks, T., Pandey, A., Paul, K., Wang, H.L.: Complexity of paired domination in AT-free and planar graphs. Theor. Comput. Sci. **930**, 53–62 (2022)