Communication Complexity of Pairs of Graph Families with Applications*

Sudeshna Kolay[1], Fahad Panolan[2], and Saket Saurabh[3]

1 **Eindhoven University of Technology, Netherlands**
2 **Department of Informatics, University of Bergen, Norway**
3 **Department of Informatics, University of Bergen, Norway, and The Institute of Mathematical Sciences, HBNI, Chennai, India**

Abstract

Given a graph G and a pair $(\mathcal{F}_1, \mathcal{F}_2)$ of graph families, the function $\mathsf{GDISJ}_{G,\mathcal{F}_1,\mathcal{F}_2}$ takes as input, two induced subgraphs G_1 and G_2 of G, such that $G_1 \in \mathcal{F}_1$ and $G_2 \in \mathcal{F}_2$ and returns 1 if $V(G_1) \cap V(G_2) = \emptyset$ and 0 otherwise. We study the communication complexity of this problem in the two-party model. In particular, we look at pairs of hereditary graph families. We show that the communication complexity of this function, when the two graph families are hereditary, is sublinear if and only if there are finitely many graphs in the intersection of these two families. Then, using concepts from parameterized complexity, we obtain nuanced upper bounds on the communication complexity of $\mathsf{GDISJ}_{G,\mathcal{F}_1,\mathcal{F}_2}$. A concept related to communication protocols is that of a $(\mathcal{F}_1, \mathcal{F}_2)$-separating family of a graph G. A collection $\mathcal{F}$ of subsets of $V(G)$ is called a $(\mathcal{F}_1, \mathcal{F}_2)$-*separating family* for G, if for any two vertex disjoint induced subgraphs $G_1 \in \mathcal{F}_1, G_2 \in \mathcal{F}_2$, there is a set $F \in \mathcal{F}$ with $V(G_1) \subseteq F$ and $V(G_2) \cap F = \emptyset$. Given a graph G on n vertices, for any pair $(\mathcal{F}_1, \mathcal{F}_2)$ of hereditary graph families with sublinear communication complexity for $\mathsf{GDISJ}_{G,\mathcal{F}_1,\mathcal{F}_2}$, we give an enumeration algorithm that finds a subexponential sized $(\mathcal{F}_1, \mathcal{F}_2)$-separating family. In fact, we give an enumeration algorithm that finds a $2^{o(k)}n^{\mathcal{O}(1)}$ sized $(\mathcal{F}_1, \mathcal{F}_2)$-separating family; where k denotes the size of a minimum sized set S of vertices such that $V(G) \setminus S$ has a bipartition (V_1, V_2) with $G[V_1] \in \mathcal{F}_1$ and $G[V_2] \in \mathcal{F}_2$. We exhibit a wide range of applications for these separating families, to obtain combinatorial bounds, enumeration algorithms as well as exact and FPT algorithms for several problems.

1998 ACM Subject Classification F.2 Analysis of Algorithms and Problem Complexity

Keywords and phrases Communication Complexity, Separating Family, FPT algorithms

Digital Object Identifier 10.4230/LIPIcs.MFCS.2017.13

1 Introduction

The two party communication complexity, introduced by Yao [17], is an important research area in theoretical computer science with many applications. This notion of complexity is particularly useful for proving lower bounds for VLSI computation, parallel computation, data structures as well as circuit lower bounds. In this model of communication, there are two players, Alice and Bob, holding inputs $x \in X$ and $y \in Y$ respectively, and they want to compute a given function $f : X \times Y \to \{0, 1\}$, by communicating as few bits as possible. It is assumed that both players have infinite computational power. However, communicating the results to the other player could be very costly. The minimum number of

* The research leading to these results has received funding from the European Research Council (ERC) via grants Rigorous Theory of Preprocessing, reference 267959 and PARAPPROX, reference 306992.

bits communicated, for any pair of inputs (x, y), to compute the function f, is called the (deterministic) communication complexity of f, denoted by $D(f)$. One such communication complexity problem, which has garnered a lot of attention, is the Clique vs Independent Set problem, introduced by Yannakakis [16]. For an n-vertex graph G, the Clique vs Independent Set problem is defined as follows. Alice gets a clique C in G and Bob gets an independent set I in G. Here both Alice and Bob know the graph G and their goal is to decide whether the clique and the independent set intersect in some vertex, by exchanging as few bits as possible. In other words, define the function $\mathsf{CIS}_G(C, I)$ as the cardinality of $V(C) \cap V(I)$ (note that $|V(C) \cap V(I)| \in \{0, 1\}$) and, Alice and Bob want to compute $\mathsf{CIS}_G(C, I)$. It can be shown that $D(\mathsf{CIS}_G) = \mathcal{O}(\log^2 n)$. One can also show that $D(\mathsf{CIS}_G) = \Omega(\log n)$, using the fooling set technique, a method to show communication lower bounds. Closing the gap between the upper and lower bound of CIS_G is a long standing open problem. Very recently, in 2015, Göös et al. [9] showed a near optimal lower bound of $\tilde{\Omega}(\log^2 n)$ for the problem, where $\tilde{\Omega}(m)$ hides divisors poly-logarithmic in m. Later, Göös et al. [8] showed that the same lower bound holds even for randomized communication complexity of the problem. Other versions of two party communication protocols deal with the concepts of nondeterministic and co-nondeterministic protocols. There are many works which study the cost of co-nondeterministic communication protocols of the Clique vs Independent Set problem [11, 1, 15, 7]. For more details on non-deterministic, co-nondeterministic and randomized communication complexities, [13] can be referred.

In this work, we study the communication complexity of graph properties that generalize the function CIS_G. Let $\mathcal{F}_1$ and $\mathcal{F}_2$ be two hereditary graph properties. That is, $\mathcal{F}_1$ and $\mathcal{F}_2$ are two families of graphs such that if $G \in \mathcal{F}_i, i \in \{1, 2\}$, then all induced subgraphs of G are also in $\mathcal{F}_i$. We define a $(\mathcal{F}_1, \mathcal{F}_2)$ communication problem as follows. For any fixed n-vertex graph G, Alice gets an induced subgraph G_1 of G and Bob gets an induced subgraph G_2 of G, such that $G_i \in \mathcal{F}_i, i \in \{1, 2\}$, and their objective is to check whether $V(G_1)$ and $V(G_2)$ intersect, by communicating as few bits as possible. In other words, we define a function $\mathsf{GDISJ}_{G,\mathcal{F}_1,\mathcal{F}_2}$ as $\mathsf{GDISJ}_{G,\mathcal{F}_1,\mathcal{F}_2}(G_1, G_2) = 1$ if $V(G_1)$ and $V(G_2)$ do not intersect and 0 otherwise, where G_1 and G_2 are induced subgraphs of G and $G_i \in \mathcal{F}_i, i \in \{1, 2\}$. Alice and Bob want to find the value of the function $\mathsf{GDISJ}_{G,\mathcal{F}_1,\mathcal{F}_2}$ on (G_1, G_2). Notice that, when $\mathcal{F}_1$ is the family of cliques and $\mathcal{F}_2$ is the family of independent sets, then $\mathsf{GDISJ}_{G,\mathcal{F}_1,\mathcal{F}_2}(C, I) = 1$ if and only if $\mathsf{CIS}_G(C, I) = 0$. A trivial protocol for computing $\mathsf{GDISJ}_{G,\mathcal{F}_1,\mathcal{F}_2}$ is as follows: Alice sends a bit vector of $V(G_1)$ to Bob and Bob checks whether it intersects with the vertex set $V(G_2)$; the number of bits communicated in this protocol is n. One of our main theorems characterizes pairs of graph families for which the trivial protocol is the best one.

▶ **Theorem 1.1.** *For any two hereditary families of graphs $\mathcal{F}_1$ and $\mathcal{F}_2$, for any $n \in \mathbb{N}$, there is an n-vertex graph G such that $D(\mathsf{GDISJ}_{G,\mathcal{F}_1,\mathcal{F}_2}) = \Omega(n)$ if and only if $\mathcal{F}_1 \cap \mathcal{F}_2$ is an infinite family.*

We give a sketch of the proof for this Theorem. We observe that when a pair of hereditary graph families have finitely many graphs in their intersection, they have the following property: In one family, all graphs have their independence number (the maximum size of an independent set) bounded by a constant, while in the other family, all graphs have their clique number (the maximum size of a clique) bounded by some other constant. Thus, we consider the communication complexity for computing $\mathsf{GDISJ}_{G,\mathcal{F}_1,\mathcal{F}_2}$ when $\mathcal{F}_1$ and $\mathcal{F}_2$ are specific families. Let $\mathcal{C}_r$ be the family of graphs such that the independence number is at most r, and $\mathcal{I}_\ell$ be the family of graphs such that the clique number is at most ℓ. Such pairs of families were considered in the study made in [6]. Deriving from Theorem 5 in [10], we show that $D(\mathsf{GDISJ}_{G,\mathcal{C}_r,\mathcal{I}_\ell}) = \mathcal{O}(\log^2 n)$ and, therefore, conclude the hypothesis of Theorem 1.1.

One of our main motivation, to carry out the study done in this article, was to introduce ideas from parameterized complexity in the study of communication complexity and vice versa. Parameterized complexity theory is a framework for a refined analysis of primarily hard (NP-hard) problems. Here, every input instance I of a problem Π is accompanied with an integer parameter k, and the running time is measured in terms of the associated parameter k and the input size. The main idea of parameterized algorithms is to *measure* the running time in terms of both input size as well as a parameter that captures structural properties of the input instance. Using ideas from parameterized complexity, we obtain the following nuanced upper bounds on the communication complexity of the function $\mathsf{GDISJ}_{G,\mathcal{F}_1,\mathcal{F}_2}$. A pair $(\mathcal{F}_1,\mathcal{F}_2)$ of hereditary graph families where $\mathcal{F}_1 \cap \mathcal{F}_2$ is finite, will be referred to as a *good pair of graph families*.

▶ **Theorem 1.2.** *Let G be an n-vertex graph and $(\mathcal{F}_1,\mathcal{F}_2)$ be a good pair of graph families. Let $\mathsf{opt}^G_{\mathcal{F}_1,\mathcal{F}_2}$ be the size of a minimum set S of vertices such that $V(G)\setminus S$ has a bipartition (V_1,V_2) with $G[V_1]\in\mathcal{F}_1$ and $G[V_2]\in\mathcal{F}_2$. Then there is a protocol for $\mathsf{GDISJ}_{G,\mathcal{F}_1,\mathcal{F}_2}$ that has $\mathcal{O}(\log^2(\mathsf{opt}^G_{\mathcal{F}_1,\mathcal{F}_2})+\log n)$ communication complexity.*

For the special case of Clique vs Independent Set problem we get a protocol that has $\mathcal{O}(\log^2(\mathsf{opt}^G_{\mathcal{C}_1,\mathcal{I}_1})+\log n)$ communication complexity. We would like to mention that the protocol used to show that $D(\mathsf{GDISJ}_{G,\mathcal{C}_r,\mathcal{I}_\ell})=\mathcal{O}(\log^2 n)$ uses the full computational power of Alice and Bob. In the protocol, both players are able to compute the communication matrix of the function $\mathsf{GDISJ}_{G,\mathcal{C}_r,\mathcal{I}_\ell}$. In contrast, we design a protocol to study communication complexity in terms of the degeneracy of graphs in the given family, where all the computations of both players are polynomial time operations. In particular, we consider the pair of families $(\mathcal{C}_1,\mathcal{D}_\ell)$, where $\mathcal{D}_\ell$ is the set of all ℓ-degenerate graphs and $\mathcal{C}_1$ is the set of all complete graphs. Note that $\mathcal{D}_0$ is the family of independent sets. Hence, this is still a generalization of the Clique vs Independent Set problem. We prove the following theorem regarding the communication complexity of $\mathsf{GDISJ}_{G,\mathcal{C}_1,\mathcal{D}_\ell}$, with the help of a protocol where both the players only execute polynomial time computations. This will be utilized later.

▶ **Theorem 1.3.** *For any constant $\ell\in\mathbb{N}$ and an n-vertex graph G, there is a deterministic protocol that computes the function $\mathsf{GDISJ}_{G,\mathcal{C}_1,\mathcal{D}_\ell}$ using $\mathcal{O}(\ell\log^2 n)$ bits and where both players have polynomial computational power.*

Separating Families

The main motivation for Yannakakis to introduce the Clique vs Independent Set problem was to study the number of constraints in the linear programming of a vertex packing polytope. As a spin-off of this study, he provided relations between the Clique vs Independent set problem and a CI-separating family (Clique-Independent set separating family): for a graph G, a family $\mathcal{F}$, of subsets of $V(G)$, is called a CI-separating family if for any disjoint clique C and independent set I in G, there is a set $F\in\mathcal{F}$ such that $C\subseteq F$ and $I\cap F=\emptyset$. He showed that the co-nondeterministic communication complexity of CIS_G is $\log q(G)$, where $q(G)$ is the cardinality of a CI-separating family of G. Yannakakis also provided a polynomial sized CI-separating family on comparability graphs and their complements, chordal graphs and their complements, and asked whether there is a polynomial sized family on general graphs, or even on perfect graphs. Lovász [14] extended the work of Yannakakis to t-perfect graphs and gave a polynomial sized CI-separating family on t-perfect graphs. Bousquet et al. [2] proved the existance of polynomial sized CI-separating families for the following class of graphs: random graphs, split-free graphs (here the graph does not have a fixed split graph

as an induced subgraph), graphs with no induced path P_k on k vertices nor its complement (here k is a constant), and graphs with no induced P_5. But, a result of Göös [7], that shows that the co-nondeterministic communication complexity of CIS_G is $\Omega(\log^{1.128} n)$, implies that the cardinality of CI-separating family on general graphs is super polynomial in the number of vertices.

The communication complexity of CIS_G, $D(\mathsf{CIS}_G) = \mathcal{O}(\log^2 n)$ implies that there is a CI-separating family of cardinality $n^{\mathcal{O}(\log n)}$ (See [13]). We would like to remark that the existence of a CI-separating family does not imply that such a family can be enumerated in time polynomial in the size of the family. The best known bound on the cardinality of a enumerable CI-separating family on general graphs is $\mathcal{O}(n^{\frac{\log n}{2}})$, by Hajnal (unpublished, cited in [14]). Cygan et. al. [4] also enumerated a CI-separating family, of cardinality $n^{\mathcal{O}(\log n)}$, in time $n^{\mathcal{O}(\log n)}$. In the special case of finding a CI-separating family, one can use the communication protocol of CIS_G, given in [14], to enumerate such a family in time $n^{\mathcal{O}(\log n)}$. To generalize from the definition of CI-separating families, for a graph G, and a pair of families $\mathcal{F}_1$ and $\mathcal{F}_2$, a notion of $(\mathcal{F}_1, \mathcal{F}_2)$-separating family was introduced. A family $\mathcal{P}$ of vertex subsets of $V(G)$ is called a $(\mathcal{F}_1, \mathcal{F}_2)$-separating family if for any two disjoint vertex subsets V_1 and V_2 with $G[V_1] \in \mathcal{F}_1$ and $G[V_2] \in \mathcal{F}_2$, there is a set $A \in \mathcal{P}$ such that $V_1 \subseteq A$ and $V_2 \cap A = \emptyset$.

From an observation made in [13], it is implied that a non-deterministic protocol for $\mathsf{GDISJ}_{G,\mathcal{F}_1,\mathcal{F}_2}$ corresponds to a $(\mathcal{F}_1, \mathcal{F}_2)$-separating family. This implies that if $D(\mathsf{GDISJ}_{G,\mathcal{F}_1,\mathcal{F}_2}) = c$, then there is a $(\mathcal{F}_1, \mathcal{F}_2)$-separating family of size 2^c. Similar to the case of CI-separating families, describing an enumeration algorithm to find the best separating family is a problem of wide interest. In this paper, we show that a $(\mathcal{C}_r, \mathcal{I}_\ell)$-separating family of size $2^{\mathcal{O}(\log^{r+\ell} n)}$ can be enumerated in time $2^{\mathcal{O}(\log^{r+\ell} n)}$. This, in turn, implies the following theorem.

▶ **Theorem 1.4.** *For any two hereditary families of graphs $\mathcal{F}_1$ and $\mathcal{F}_2$, for each integer $n > 0$, there is an n-vertex graph G such that any $(\mathcal{F}_1, \mathcal{F}_2)$-separating family must be of size $2^{\Omega(n)}$ if and only if $\mathcal{F}_1 \cap \mathcal{F}_2$ is an infinite family.*

Note that although $D(\mathsf{GDISJ}_{G,\mathcal{C}_r,\mathcal{I}_\ell}) = \mathcal{O}(\log^2 n)$, we are not able to find a $(\mathcal{C}_r, \mathcal{I}_\ell)$-separating family of size $2^{\mathcal{O}(\log^2 n)}$ that can be enumerated in time $2^{\mathcal{O}(\log^{r+\ell} n)}$. We also get the following theorem as a "separating family" analogue of Theorem 1.2. This theorem is extremely useful in designing parameterized algorithms.

▶ **Theorem 1.5.** *Let $(\mathcal{F}_1, \mathcal{F}_2)$ be a good pair of graph families and G be an n-vertex graph. Let S be a minimum sized vertex set of G such that $V(G) \setminus S$ has a bipartition (V_1, V_2) with $G[V_1] \in \mathcal{F}_1$ and $G[V_2] \in \mathcal{F}_2$. Let $|S| = \mathsf{opt}^G_{\mathcal{F}_1,\mathcal{F}_2}$. Then a $(\mathcal{F}_1, \mathcal{F}_2)$-separating family, for G, of cardinality $2^{\mathcal{O}(\log^c \mathsf{opt}^G_{\mathcal{F}_1,\mathcal{F}_2})} n^{\mathcal{O}(1)}$ can be enumerated in time $2^{\mathcal{O}(\log^c \mathsf{opt}^G_{\mathcal{F}_1,\mathcal{F}_2})} n^{\mathcal{O}(1)}$, where c is a constant.*

Another pair of graph properties (families of graphs) we consider is the family of complete graphs, $\mathcal{C}_1$ and that of ℓ-degenerate graphs, $\mathcal{D}_\ell$. By Theorem 1.3, we already know that $D(\mathsf{GDISJ}_{G,\mathcal{C}_1,\mathcal{D}_\ell})$ is $\mathcal{O}(\ell \log^2 n)$. We also give an algorithm to enumerate a $(\mathcal{C}_1, \mathcal{D}_\ell)$-separating family for an n-vertex graph, of cardinality $n^{\mathcal{O}(\ell \log n)}$, in time $n^{\mathcal{O}(\ell \log n)}$. In other words, we succeed in efficiently enumerating a $(\mathcal{C}_1, \mathcal{D}_\ell)$-separating family of size $n^{\mathcal{O}(\ell \log n)}$, the existence of which results from $D(\mathsf{GDISJ}_{G,\mathcal{C}_1,\mathcal{D}_\ell}) = \mathcal{O}(\ell \log^2 n)$.

Applications

In 2013, Cygan et al. [4] drew a very interesting relation between the field of enumerating separating families and designing algorithms. As mentioned earlier, a CI-separating family of cardinality $n^{\mathcal{O}(\log n)}$ is enumerated in time $n^{\mathcal{O}(\log n)}$, and this family is used to design fast exact and parameterized algorithms. They showed that Split Vertex Deletion, where we want to delete at most k vertices from a given n-vertex graph to get a split graph, can be solved in time $\mathcal{O}(1.2738^k k^{\mathcal{O}(\log k)} + n^3)$. They also showed that all induced split subgraphs of a given n-vertex graph can be listed in time $\mathcal{O}(3^{n/3} n^{\mathcal{O}(\log n)})$ time. This work motivated the last part of our study: designing exact and FPT algorithms. Not only are the enumeration algorithms for separating families interesting combinatorial questions in their own right, but they also help to design fast FPT and exact exponential time algorithms for a class of problems. A generic class of problems for which a separating family based approach works is as follows. Let $\mathcal{G}$ be a family of graphs. Then $\mathcal{G} + kv$ contains all graphs G such that there is a vertex set $S \subseteq V(G)$, of size at most k, with the property that the graph $G \setminus S \in \mathcal{G}$. Given two graph families $\mathcal{F}_1, \mathcal{F}_2$, we consider the following problem in this paper.

$(\mathcal{F}_1, \mathcal{F}_2)$-p-Partition **Parameter:** k
Input: A graph G and a non-negative integer k
Question: Is there a vertex set $S \subseteq V(G)$, of size at most k, such that there is a partition $V_1 \uplus V_2$ of $V(G) \setminus S$ and $G[V_i] \in \mathcal{F}_i$, $i \in \{1, 2\}$?

The optimization version of $(\mathcal{F}_1, \mathcal{F}_2)$-p-Partition is denoted by $(\mathcal{F}_1, \mathcal{F}_2)$-Partition. Here, the aim is to find the minimum size of a vertex set S such that $V(G) \setminus S$ has a bipartition (V_1, V_2) with $G[V_1] \in \mathcal{F}_1$ and $G[V_2] \in \mathcal{F}_2$. Let $\mathcal{F}_1$ and $\mathcal{F}_2$ be a good pair of graph families. For any positive integer k, let the families $\mathcal{F}_1 + kv$ and $\mathcal{F}_2 + kv$ have FPT recognition algorithms. That is, there are algorithms which take as input a graph G and an integer k, decide whether $G \in \mathcal{F}_i + kv, i \in \{1, 2\}$ and run in time $f(k)|V(G)|^{\mathcal{O}(1)}$. For ease of notation, if $\mathcal{F}_1$ and $\mathcal{F}_2$ be a good pair of graph families, and the families $\mathcal{F}_1 + kv$ and $\mathcal{F}_2 + kv$ have FPT recognition algorithms, then we call $(\mathcal{F}_1, \mathcal{F}_2)$ an FPT-*good pair of families*.

▶ **Theorem 1.6.** *Let $(\mathcal{F}_1, \mathcal{F}_2)$ be an* FPT*-good pair of families. Also, let $\mathcal{A}_1$ and $\mathcal{A}_2$ be the best recognition algorithms for $\mathcal{F}_1 + kv$ and $\mathcal{F}_2 + kv$ respectively. For an n-vertex input graph and non-negative integer k, let the running time of $\mathcal{A}_i, i \in \{1, 2\}$, be $T_i(n, k)$. Then $(\mathcal{F}_1, \mathcal{F}_2)$-*p-Partition *on an instance (G, k) can be solved in time $2^{\mathcal{O}(\log^c k)} n^{\mathcal{O}(1)} \cdot \max\{T_1(n, k), T_2(n, k)\}$.*

One could obtain a result analogous to Theorem 1.6 for $(\mathcal{F}_1, \mathcal{F}_2)$-Partition. Some of the problems for which we get faster FPT and exact algorithms are (Clique,Planar)-p-Partition, (Clique,Triangle-free)-p-Partition, (Clique,Forest)-p-Partition and (Clique, Treewidth-t)-p-Partition.

2 Preliminaries

We use $\mathbb{N}$ to denote the set of natural numbers. For $n \in \mathbb{N}$, we use $[n]$ to denote $\{1, \ldots, n\}$. Through out the paper we use n to denote the number of vertices in the graph used in the context. In this paper, the function log is used to denote the logarithm function with *base* 2. We use standard notations from graph theory [5]. The vertex set and edge set of a graph are denoted as $V(G)$ and $E(G)$ respectively. The complement of the graph G is denoted by $\overline{G}$. The *neighbourhood* of a vertex v is represented as $N_G(v)$, or, when the context of the graph is clear, simply as $N(v)$. The *closed neighbourhood* of a vertex v, denoted by $N[v]$, is

subset $N(v) \cup \{v\}$.The *non-neighbourhood* of a vertex v is denoted by $\overline{N}_G(v)$. The *degree* of a vertex v, or the number of neighbours of v, is denoted by $d_G(v)$. Similarly, the *non-degree* of v, or the number of non-neighbours of v, is denoted by $\overline{d}_G(v)$. An *induced subgraph* of G on the vertex set $V' \subseteq V$ is written as $G[V']$. For a vertex subset $V' \subseteq V$, $G[V \setminus V']$ is also denoted as $G - V'$. We denote by $\omega(G)$ the size of a maximum clique in G. Similarly, $\alpha(G)$ denotes the size of a maximum independent set in G. A subgraph G' of G is denoted as $G' \leq_s G$. A complete graph on n vertices is denoted by K_n. A stable graph on n vertices is a graph G with edge set $\emptyset$, and is denoted by $\overline{K}_n$. An *empty graph* is a graph which does not have any vertices, and therefore no edges as well. Given two subgraphs $G_1, G_2 \leq_s G$, $G_1 \cap G_2$ is the induced subgraph $G[V(G_1) \cap V(G_2)]$. Similarly, $G_1 \cup G_2$ denoted the induced subgraph $G[V(G_1) \cup V(G_2)]$. For any positive integers r, ℓ, we use $R(r, \ell)$ to denote the *Ramsey number*. That is, any graph on at least $R(r, \ell)$ vertices will have either a clique of size r or an independent set of size ℓ. A family $\mathcal{F}$ of graphs is said to be *hereditary* if for any graph $G \in \mathcal{F}$, every induced subgraph of G is also contained in $\mathcal{F}$. Let $\mathcal{G}$ be a family of graphs. Then $\mathcal{G} + kv$ contains all graphs G such that there is a vertex set $S \subseteq V(G)$, of size at most k, with the property that the graph $G - S \in \mathcal{G}$.

Informally, a protocol can be thought of as a communication between two players, Alice and Bob. They have decided on some function f and wish to evaluate $f(x, y)$, for some input $x \in X$ and $y \in Y$. The catch is that x is only known to Alice and y is only known to Bob. Now we give a formal definition of a communication protocol.

▶ **Definition 2.1** ([12]). A protocol Π over a domain $X \times Y$ with range Z is a binary tree where each internal node v is labelled either by a function $a_v : X \to \{0, 1\}$ or by a function $b_v : Y \to \{0, 1\}$, and each leaf is labelled with an element $z \in Z$. The value of the protocol Π on an input (x, y) is the label of the leaf reached by starting at the root, and walking along a path in the tree. At each internal node v labelled by a_v, the walk takes left if $a_v(x) = 0$ and right if $a_v(x) = 1$, and at each internal node labelled by b_v, the walk takes left if $b_v(y) = 0$ and right if $b_v(y) = 1$. The cost of the protocol Π on an input (x, y) is the length of the path taken on the input (x, y). The cost of the protocol Π is the height of the binary tree.

▶ **Definition 2.2** ([12]). For a function $f : X \times Y \to Z$, the deterministic communication complexity of f is the minimum cost of Π, over all protocols Π that compute f. We denote the deterministic communication complexity of f by $D(f)$.

For further reading on Communication Complexity, including the concepts of non-deterministic and co-nondeterministic communication complexity of a function, we refer the reader to [12, 13]. One of the first functions, whose communication complexity was studied, is the Disjointness function. For any $x, y \in \{0, 1\}^n$, the function is defined as,

$$\mathsf{DISJ}_n(x, y) = \begin{cases} 0 & \text{if there exists } i \in [n], x[i] = y[i] = 1 \\ 1 & \text{otherwise} \end{cases}$$

▶ **Proposition 2.3** ([12]). $D(\mathit{DISJ}_n) \geq n$

We study a variant of the DISJ_n function, called the Graphic Disjointness function. Let G be a graph on n vertices and m edges. Let $\mathcal{F}_1$ and $\mathcal{F}_2$ be two hereditary graph families. The following function is defined for the graph G, and the families $\mathcal{F}_1, \mathcal{F}_2$ as follows. For any two vertex subsets V_1 and V_2 such that $G[V_1] \in \mathcal{F}_1$ and $G[V_2] \in \mathcal{F}_2$,

$$\mathsf{GDISJ}_{G,\mathcal{F}_1,\mathcal{F}_2}(V_1, V_2) = \begin{cases} 1 & \text{if } V_1 \cap V_2 = \emptyset \\ 0 & \text{otherwise} \end{cases}$$

A problem, related to that of computing $\mathsf{GDISJ}_{G,\mathcal{F}_1,\mathcal{F}_2}$, is the problem of enumerating separating families for two graph families.

▶ **Definition 2.4.** Let G be a graph on n vertices, $\mathcal{F}_1$ and $\mathcal{F}_2$ be two graph families. Suppose $\mathcal{F}$ is a family of subsets of $V(G)$ with the following property: If we take any two vertex disjoint induced subgraphs $G_1, G_2 \leq_s G$, such that $G_1 \in \mathcal{F}_1$ and $G_2 \in \mathcal{F}_2$, there is a set $F \in \mathcal{F}$ such that $V(G_1) \subseteq F$ and $V(G_2) \cap F = \emptyset$. Then $\mathcal{F}$ is called an $(\mathcal{F}_1, \mathcal{F}_2)$-separating family in G. Such a set F is called a separating set for G_1 and G_2.

▶ **Observation 2.5.** *Let G be an n-vertex graph. Let G_1, G_2 be induced subgraphs of G. Suppose for each $v \in V(G_1)$, $\overline{d}_G(v) < n/2$ and for each $w \in V(G_2)$, $d_G(w) < n/2$. Then, $V(G_1) \cap V(G_2) = \emptyset$ and $\{v \mid v \in V(G), \overline{d}_G(v) < n/2\}$ is a separating set for G_1 and G_2.*

3 Communication protocols for pairs of Hereditary graph families

To prove Theorem 1.1, we first need to prove a sublinear communication complexity bound for a specific pair of graph families. More formally, in this section we consider a pair of hereditary families of graphs, $\mathcal{C}_r = \{H \mid \alpha(H) \leq r\}$ and $\mathcal{I}_\ell = \{H \mid \omega(H) \leq \ell\}$. Here, r and ℓ are two positive integers. In this section, we consider the communication complexity of $\mathsf{GDISJ}_{G,\mathcal{C}_r,\mathcal{I}_\ell}$. Using this, we complete the proof of Theorem 1.1. In the later half of this Section, we give upper bounds on the communication complexity of the function $\mathsf{GDISJ}_{G,\mathcal{F}_1,\mathcal{F}_2}$, in terms of a structural parameter of the graph G. We consider one such structural parameter and design a protocol with the help of this additional parameter.

3.1 Communication Protocol for Families of Sparse and Dense graphs

As a corollary to Theorem 5 of [10], we obtain the following Lemma:

▶ **Lemma 3.1.** *For any $r, \ell \in \mathbb{N}$, $D(GDISJ_{G,\mathcal{C}_r,\mathcal{I}_\ell}) = \mathcal{O}(R(r+1, \ell+1)\log^2 n)$.*

The protocol designed to show that $D(\mathsf{GDISJ}_{G,\mathcal{C}_r,\mathcal{I}_\ell}) = \mathcal{O}(R(r+1,\ell+1)\log^2 n)$ heavily relies on the infinite computational power of both the players (See full version). In this section, we describe a protocol, with much worse communication complexity, but where both players resort to polynomial time computations only. The communication complexity of this protocol is still sublinear in $|V(G)|$. This protocol will be very useful when we design enumeration algorithms for $(\mathcal{C}_r, \mathcal{I}_\ell)$-separating families in Section 4.

We will describe a communication protocol for $\mathsf{GDISJ}_{G,\mathcal{C}_r,\mathcal{I}_\ell}$, with complexity $\mathcal{O}(\log^{r+\ell} n)$, for any $r, \ell \in \mathbb{N}$. Here, Alice receives an induced subgraph G_1 of G such that $G_1 \in \mathcal{C}_r$, and Bob receives an induced subgraph G_2 of G such that $G_2 \in \mathcal{I}_\ell$. They have to determine whether $V(G_1) \cap V(G_2) = \emptyset$ or not. Note that both Alice and Bob receive the graph G.

First, we give a protocol $\Pi_{1,2}$ for $\mathsf{GDISJ}_{G,\mathcal{C}_1,\mathcal{I}_2}$, with a cost of $\mathcal{O}(\log^3 n)$. This protocol uses a protocol $\Pi_{1,1}$, for $\mathsf{GDISJ}_{G,\mathcal{C}_1,\mathcal{I}_1}$, as a sub-protocol. As mentioned earlier, for any pair of induced subgraphs $C, I \in G$, with $C \in \mathcal{C}_1, I \in \mathcal{I}_1$, $\mathsf{GDISJ}_{G,\mathcal{C}_1,\mathcal{I}_1}(C, I) = 1$ if and only if $\mathsf{CIS}_G(C, I) = 0$. The function CIS_G has a deterministic protocol of cost $\mathcal{O}(\log^2 n)$ [16]. Therefore, there is a protocol $\Pi_{1,1}$ of cost $\mathcal{O}(\log^2 n)$ for $\mathsf{GDISJ}_{G,\mathcal{C}_1,\mathcal{I}_1}$. The protocol for the general case $\mathsf{GDISJ}_{G,\mathcal{C}_r,\mathcal{I}_\ell}$ can be designed in a recursive manner that uses protocols of $\mathsf{GDISJ}_{G,\mathcal{C}_r,\mathcal{I}_{\ell-1}}$ and $\mathsf{GDISJ}_{G,\mathcal{C}_{r-1},\mathcal{I}_\ell}$ as subprotocols.

▶ **Lemma 3.2.** *For a graph G, there is a deterministic protocol for computing $GDISJ_{G,\mathcal{C}_1,\mathcal{I}_2}$ using $\mathcal{O}(\log^3 n)$ bits and where both players have polynomial computational power.*

Proof sketch. Let Alice get the induced subgraph G_1 and Bob get the induced subgraph G_2. The following is a protocol $\Pi_{1,2}$ that Alice and Bob will execute. Alice and Bob continue with the protocol till either they detect a vertex in the intersection of $V(G_1)$ and $V(G_2)$,

or one of G, G_1 and G_2 becomes an empty graph. The protocol is executed in top down fashion, i.e., the two players resort to a step only if the previous steps are not applicable.

1. If either G_1 or G_2 is an empty graph, then the players declare that the graphs are disjoint.
2. Alice looks for a vertex $v \in V(G_1)$ with $\overline{d}_G(v) \geq n/2$. She sends the vertex v to Bob. If $v \in V(G_2)$, then Bob lets Alice know and they terminate the protocol. Otherwise, both players delete the vertices of $\overline{N}_G(v) \cup \{v\}$ from the graph G to obtain graph $G' = G - (\overline{N}_G(v) \cup \{v\})$. Alice defines $G'_1 = G_1 - \{v\}$ while Bob defines $G'_2 = G_2 - \overline{N}_G(v)$. Then, they continue the protocol for determining whether $V(G'_1) \cap V(G'_2) = \emptyset$ in G'.
3. Bob looks for a vertex $v \in V(G_2)$ with $d_G(v) \geq n/2$. Bob sends the vertex v to Alice. If $v \in V(G_1)$, then Alice lets Bob know and they terminate the protocol. Otherwise, both players use the protocol $\Pi_{1,1}$ to compute $\mathsf{GDISJ}_{G[N_G(v)],\mathcal{C}_1,\mathcal{I}_1}(G[N_G(v) \cap V(G_1)], G[N_G(v) \cap V(G_2)])$. If the output is 0, then they declare that $V(G_1) \cap V(G_2) \neq \emptyset$ and stop. Otherwise, they delete the vertices of $N_G[v]$ from G to get $G' = G - N_G[v]$. Alice defines $G'_1 = G_1 - N_G[v]$ while Bob defines $G'_2 = G_2 - N_G[v]$. Then, they continue the protocol for determining whether $V(G'_1) \cap V(G'_2) = \emptyset$ in G'.
4. Suppose all the above steps fails, then, both players declare that $V(G_1) \cap V(G_2) = \emptyset$.

The full proof is given in the full version of this paper. ◀

▶ **Corollary 3.3.** *For any graph G, there is a deterministic protocol for $\mathsf{GDISJ}_{G,\mathcal{C}_2,\mathcal{I}_1}$ using $\mathcal{O}(\log^3 n)$ bits, where both players have polynomial computational power.*

We can give a protocol $\Pi_{r,\ell}$, for the problem $\mathsf{GDISJ}_{G,\mathcal{C}_r,\mathcal{I}_\ell}$, of cost $\mathcal{O}(\log^{r+\ell} n)$, using a protocol for $\Pi_{r,\ell-1}$ or $\Pi_{r-1,\ell}$. We use similar arguments as in the protocol $\Pi_{1,2}$, to design the protocol $\Pi_{r,\ell}$. Thus, we can prove the following theorem using induction on $r + \ell$.

▶ **Lemma 3.4.** *For $r, \ell \in \mathbb{N}$ and graph G, there is a deterministic protocol for $\mathsf{GDISJ}_{G,\mathcal{C}_r,\mathcal{I}_\ell}$ using $\mathcal{O}(\log^{r+\ell} n)$ bits and where both players have polynomial computational power.*

3.2 Characterization for Hereditary graph families

We are ready to prove Theorem 1.1. That is, we try to determine $D(\mathsf{GDISJ}_{G,\mathcal{F}_1,\mathcal{F}_2})$ for any given pair of hereditary families $\mathcal{F}_1, \mathcal{F}_2$. If one of $\mathcal{F}_1$ or $\mathcal{F}_2$ is finite, then the number of vertices of each graph in one of the families is bounded by a constant. Thus, a trivial protocol would be for the player, who receives the bounded-sized subgraph, to send the full subgraph over to the other player, using $\mathcal{O}(\log n)$ bits. This implies, $D(\mathsf{GDISJ}_{G,\mathcal{F}_1,\mathcal{F}_2}) = \mathcal{O}(\log n)$. So, the interesting case is when both $\mathcal{F}_1$ and $\mathcal{F}_2$ are infinite. Now we prove Theorem 1.1.

Proof of Theorem 1.1. Without loss of generality we can assume that both $\mathcal{F}_1$ and $\mathcal{F}_2$ are infinite. Suppose the intersection family is finite. This means that there is a constant r such that a complete graph K_r, on r vertices, does not belong to the intersection family, because of finiteness. Since K_r does not belong to $\mathcal{F}_1 \cap \mathcal{F}_2$, it must not belong to at least one of the families. Let this be $\mathcal{F}_1$. Since $\mathcal{F}_1$ is hereditary, no graph in $\mathcal{F}_1$ has K_r as an induced subgraph. This implies that for any graph G in $\mathcal{F}_1$, $\omega(G) \leq r - 1$. Now we show that for any graph G in $\mathcal{F}_2$, $\alpha(G) \leq \ell - 1$ for some constant ℓ. Towards that, we first claim that $\mathcal{F}_1$ contains all stable graphs. Otherwise, since $\mathcal{F}_1$ is a hereditary family, if $\mathcal{F}_1$ does not contain a stable graph on ℓ' vertices, all graphs in $\mathcal{F}_1$ neither have a r-sized clique as an induced subgraph nor an ℓ'-sized independent set as an induced subgraph. However, by Ramsey's theorem, each graph in $\mathcal{F}_1$ has at most $R(r, \ell')$ vertices, thus contradicting the infiniteness of $\mathcal{F}_1$. So far we know that, $\mathcal{F}_1 \cap \mathcal{F}_2$ is finite and $\mathcal{F}_1$ contains all stable graphs. This implies that $\mathcal{F}_2$ does not contain all stable graphs. Let ℓ be an integer such that $\overline{K}_\ell \notin \mathcal{F}_2$. By the

hereditary property of $\mathcal{F}_2$, no graph in $\mathcal{F}_2$ contains $\overline{K}_\ell$ as an induced subgraph. That is, for all graph G in $\mathcal{F}_2$, $\alpha(G) \leq \ell - 1$. Thus, Lemma 3.1 gives us a deterministic protocol for $\mathsf{GDISJ}_{G,\mathcal{F}_1,\mathcal{F}_2}$, with $o(n)$ communication complexity.

For the reverse direction, suppose $\mathcal{F}_1 \cap \mathcal{F}_2$ is an infinite family. To prove $D(\mathsf{GDISJ}_{G,\mathcal{F}_1,\mathcal{F}_2}) = \Omega(n)$ we give a simple reduction from DISJ_n. In DISJ_n, Alice is given $x \in \{0,1\}^n$ and Bob is given $y \in \{0,1\}^n$ and they want to check whether there is an $i \in [n]$ such that $x[i] = y[i] = 1$. Now we create an instance of $\mathsf{GDISJ}_{G,\mathcal{F}_1,\mathcal{F}_2}$ as follows. We fix an n-vertex graph $G \in \mathcal{F}_1 \cap \mathcal{F}_2$ (such a graph exists because of hereditary property), with $V(G) = \{v_1, \ldots, v_n\}$. Let $V_x = \{v_i \in V(G) \mid i \in [n], x[i] = 1\}$ and $V_y = \{v_i \in V(G) \mid i \in [n], y[i] = 1\}$. Since $G \in \mathcal{F}_1 \cap \mathcal{F}_2$, $G[V_x] \in \mathcal{F}_1$ and $G[V_y] \in \mathcal{F}_2$. In the $\mathsf{GDISJ}_{G,\mathcal{F}_1,\mathcal{F}_2}$ problem, Alice gets $G[V_x]$ and Bob gets $G[V_y]$. Clearly $V_x \cap V_y \neq \emptyset$ if and only if there is an $i \in [n]$ such that $x[i] = y[i] = 1$. Hence, by Proposition 2.3, $D(\mathsf{GDISJ}_{G,\mathcal{F}_1,\mathcal{F}_2}) = \Omega(n)$. ◀

For the rest of this paper, a pair $(\mathcal{F}_1, \mathcal{F}_2)$ of hereditary graph families where $\mathcal{F}_1 \cap \mathcal{F}_2$ is a finite graph family, will be referred to as a *good pair of graph families*. The proof of Theorem 1.1 also gives us the following folklore corollary.

▶ **Corollary 3.5.** *Let $\mathcal{F}_1$ and $\mathcal{F}_2$ be a good pair of graph families. Then, there are constants r and ℓ, such that for any graph $G_1 \in \mathcal{F}_1$ and $G_2 \in \mathcal{F}_2$, $\omega(G_1) \leq r$ and $\alpha(G_2) \leq \ell$.*

Corollary 3.5 and Ramsey theorem leads us to another useful corollary.

▶ **Corollary 3.6.** *Let G be a graph. Let $\mathcal{F}_1$ and $\mathcal{F}_2$ be a good pair of graph families. Then there are constant r and ℓ (same as the constants mentioned in Corollary 3.5) such that, for any pair (G_1, G_2) of induced subgraphs of G, if $G_1 \in \mathcal{F}_1$ and $G_2 \in \mathcal{F}_2$, then $|V(G_1) \cap V(G_2)| < R(r+1, \ell+1)$.*

3.3 A Parameterized approach to designing protocols

In Section 3.1, for each pair of constants r, ℓ, we saw a protocol for $\mathsf{GDISJ}_{G,\mathcal{C}_r,\mathcal{I}_\ell}$ with sublinear communication complexity. We also showed that any good pair $(\mathcal{F}_1, \mathcal{F}_2)$ of graph families must be such that there are constants r, ℓ with $\mathcal{F}_1 \subseteq \mathcal{C}_r, \mathcal{F}_2 \subseteq \mathcal{I}_\ell$. In this section, we give an alternate protocol that uses the structure of the input graph G, to obtain a more refined upper bound on the communication complexity of $\mathsf{GDISJ}_{G,\mathcal{F}_1,\mathcal{F}_2}$. For a graph, let $\mathsf{opt}^G_{\mathcal{F}_1,\mathcal{F}_2}$ denote the size of a minimum set S of vertices such that $V(G) \setminus S$ has a bipartition (V_1, V_2) with $G[V_1] \in \mathcal{F}_1$ and $G[V_2] \in \mathcal{F}_2$. In this section, for a graph G and a good pair of graph families $(\mathcal{F}_1, \mathcal{F}_2)$, we give a protocol for $\mathsf{GDISJ}_{G,\mathcal{F}_1,\mathcal{F}_2}$ that has $\mathcal{O}(\log^2(\mathsf{opt}^G_{\mathcal{F}_1,\mathcal{F}_2}) + \log n)$ communication complexity.

Proof of Theorem 1.2. We can assume that Alice and Bob both have a minimum vertex set S such that $V(G) \setminus S$ has a bipartition (V_1, V_2) with $G[V_1] \in \mathcal{F}_1$ and $G[V_2] \in \mathcal{F}_2$. Thus $|S| = \mathsf{opt}^G_{\mathcal{F}_1,\mathcal{F}_2}$. The players also have a bipartition (V_1, V_2) of $V(G) \setminus S$, such that $G[V_1] \in \mathcal{F}_1$ and $G[V_2] \in \mathcal{F}_2$.

Now let Alice receive the induced subgraph $G_1 \in \mathcal{F}_1$ and Bob receive the induced subgraph $G_2 \in \mathcal{F}_2$. The following is a protocol Π that Alice and Bob will execute. The protocol is executed in top down fashion, i.e., the two players resort to a step only if the previous steps are not applicable.

1. If either G_1 or G_2 is an empty graph, then they declare that the graphs are disjoint.
2. Alice and Bob run the protocol $\Pi_{r,\ell}$ to compute $\mathsf{GDISJ}_{G[S],\mathcal{C}_r,\mathcal{I}_\ell}(G_1[S], G_2[S])$. If there is a vertex intersection between $G_1[S]$ and $G_2[S]$, then they declare that the graphs G_1 and G_2 intersect and stop the protocol.

3. Suppose there is no vertex intersection between $G_1[S]$ and $G_2[S]$. Alice sends the vertices of the subgraph $G_1 \cap G[V_2]$ to Bob. If Bob finds that $V(G_2) \cap V(G_1 \cap G[V_2]) \neq \emptyset$, then Bob lets Alice know and they terminate the protocol.
4. Suppose Bob does not find any vertex common to both $V(G_1 \cap G[V_2])$ and $V(G_2)$. Then Bob sends the vertices of the subgraph $G_2 \cap G[V_1]$ to Alice. If Alice finds that $V(G_1) \cap V(G_2 \cap G[V_2]) \neq \emptyset$, then Alice lets Bob know and they terminate the protocol.

Otherwise, they declare that the two graphs G_1 and G_2 do not intersect on any vertex. If $V(G_1[S]) \cap V(G_2[S]) \neq \emptyset$, then the subprotocol $\Pi_{r,\ell}$ correctly detects the intersection in step 2. Otherwise, $V(G_1)$ and $V(G_2)$ can intersect either in V_1 or in V_2 and they detect in step 3 or step 4. If neither of the above cases happen, then $V(G_1) \cap V(G_2) = \emptyset$.

Next, we show the bound on the communication complexity. Following from Lemma 3.1, $\mathsf{GDISJ}_{G[S],\mathcal{C}_r,\mathcal{I}_\ell}(G_1[S], G_2[S])$ can be computed with the communication of $\mathcal{O}(\log^2 \mathsf{opt}^G_{\mathcal{F}_1,\mathcal{F}_2})$ bits. By definition, $G_1 \in \mathcal{F}_1$ and $G[V_2] \in \mathcal{F}_2$. Then, by Corollary 3.6, $|V(G_1) \cap V(G[V_2])| < R(r+1, \ell+1)$. Thus, in step 3, Alice sends at most $R(r+1,\ell+1)\log n$ bits to Bob. By a similar argument, in step 4, Bob sends at most $R(r+1,\ell+1)\log n$ bits to Alice. Therefore, the communication complexity of Π is $\mathcal{O}(\log^{r+\ell}(\mathsf{opt}^G_{\mathcal{F}_1,\mathcal{F}_2}) + \log n)$. ◀

Suppose $(\mathcal{F}_1, \mathcal{F}_2)$ is a pair of hereditary graph families that are not good. By Theorem 1.1, for an n-vertex graph G, any protocol for $\mathsf{GDISJ}_{G,\mathcal{F}_1,\mathcal{F}_2}$ must have communication complexity $\Omega(n)$. This gives us the following corollary.

▶ **Corollary 3.7.** *Let $(\mathcal{F}_1, \mathcal{F}_2)$ be a pair of hereditary graph families that have infinitely many graphs in their intersection. Then, for each integer $n > 0$, there is a graph G such that for any computable function f, there cannot be a protocol for $\mathsf{GDISJ}_{G,\mathcal{F}_1,\mathcal{F}_2}$, that has communication complexity $f(\mathsf{opt}^G_{\mathcal{F}_1,\mathcal{F}_2}) + o(n)$.*

4 Separating families

In this section, we design enumeration algorithms for separating families for a good pair of graph families. It was stated in [13] that a non-deterministic protocol for $\mathsf{GDISJ}_{G,\mathcal{F}_1,\mathcal{F}_2}$ corresponds to a $(\mathcal{F}_1, \mathcal{F}_2)$-separating family. This means that if $\mathsf{GDISJ}_{G,\mathcal{F}_1,\mathcal{F}_2}$ has deterministic, and hence non-deterministic, complexity c, then there exists a $(\mathcal{F}_1, \mathcal{F}_2)$-separating family of size 2^c. From Corollary 3.5 and Lemma 3.1, this means that, for an n-vertex graph, there exists a $(\mathcal{F}_1, \mathcal{F}_2)$-separating family of size $2^{\mathcal{O}(\log^2 n)}$, for any constants r, ℓ. However, since the protocols use players with unlimited power of computation, this does not mean that there is an enumeration algorithm that finds such a separating family in time $2^{\mathcal{O}(\log^2 n)} n^{\mathcal{O}(1)}$. First, for an n-vertex graph G, we design an algorithm to enumerate a $(\mathcal{C}_r, \mathcal{I}_\ell)$-separating family of size $2^{\mathcal{O}(\log^{r+\ell} n)}$, in time $2^{\mathcal{O}(\log^{r+\ell} n)} n^{\mathcal{O}(1)}$. This uses ideas from the protocol given in Lemma 3.4. Then, for a good pair $(\mathcal{F}_1, \mathcal{F}_2)$ of graph families, we utilize the structure of G, for a different approach to design enumeration algorithms for $(\mathcal{F}_1, \mathcal{F}_2)$-separating families.

▶ **Lemma 4.1.** *For any $r, \ell \in \mathbb{N}$, every graph with n vertices has a $(\mathcal{C}_r, \mathcal{I}_\ell)$-separating family of cardinality $2^{\mathcal{O}(\log^{r+\ell} n)}$. Moreover, such a family can be enumerated in time $2^{\mathcal{O}(\log^{r+\ell} n)}$.*

Lemma 4.1 and Corollary 3.5 gives us the following Corollary.

▶ **Corollary 4.2.** *Let $\mathcal{F}_1$ and $\mathcal{F}_2$ be a good pair of graph families. Then, there are constants r and ℓ, such that every n-vertex graph has a $(\mathcal{F}_1, \mathcal{F}_2)$-separating family of cardinality $2^{\mathcal{O}(\log^{r+\ell} n)}$ and it can be enumerated in time $2^{\mathcal{O}(\log^{r+\ell} n)}$.*

In fact, we obtain Theorem 1.4 from Lemma 4.1 and Corollary 3.5.

Separating families and parameterization

We give the proof of Theorem 1.5. We show that the upper bound obtained due to Corollary 4.2 can be improved if we use ideas from Parameterized Complexity, as we did for Theorem 1.2. This is extremely useful for designing FPT algorithms. Again, the ideas from the protocol of Lemma 3.4 comes to more use than the protocol of Lemma 3.1. To show this, we first prove the following lemma.

▶ **Lemma 4.3.** *Let $(\mathcal{F}_1, \mathcal{F}_2)$ be a good pair of graph families. Given, as input, $G, S \subseteq V(G)$ and partition $V_1 \uplus V_2$ of $V(G) \setminus S$ such that $G[V_1] \in \mathcal{F}_1$ and $G[V_2] \in \mathcal{F}_2$, there is an algorithm to enumerate $(\mathcal{F}_1, \mathcal{F}_2)$-separating family $\mathcal{S}$ for G of cardinality $2^{\mathcal{O}(\log^c(|S|))}n^{\mathcal{O}(1)}$ in time $2^{\mathcal{O}(\log^c(|S|))}n^{\mathcal{O}(1)}$, where c is a constant.*

Proof. We know that $G[V_1] \in \mathcal{F}_1$ and $G[V_2] \in \mathcal{F}_2$. Since $(\mathcal{F}_1, \mathcal{F}_2)$ is a good pair of graph families, by Corollary 3.5, we know that there are constants r, ℓ, such that for any $G_1 \in \mathcal{F}_1$ and $G_2 \in \mathcal{F}_2$, $\omega(G_1) \leq r$ and $\alpha(G_2) \leq \ell$. Let us define $c = r + \ell$. By Lemma 4.1, the graph $G[S]$ has a $(\mathcal{F}_1, \mathcal{F}_2)$-separating family $\mathcal{S}'$ of cardinality $2^{\mathcal{O}(\log^{r+\ell}|S|)}$. Moreover, such a family can be enumerated in time $2^{\mathcal{O}(\log^{r+\ell}(|S|))}$. Now consider the family.

$$\mathcal{S} = \left\{A \cup (V_1 \setminus S_1) \cup S_2 \mid A \in \mathcal{S}', S_1 \subseteq V_1, S_2 \subseteq V_2, |S_1| < R(r+1, \ell+1), |S_2| < R(r+1, \ell+1)\right\}$$

The cardinality of $\mathcal{S}$ is bounded by $2^{\mathcal{O}(\log^c(|S|))}n^{\mathcal{O}(2R(r+1,\ell+1))}$ and it can be enumerated in time $2^{\mathcal{O}(\log^c(|S|))}n^{\mathcal{O}(2R(r+1,\ell+1))}$. We show that $\mathcal{S}$ is indeed a $(\mathcal{F}_1, \mathcal{F}_2)$-separating family for G. Consider any disjoint vertex subsets U_1 and U_2 of $V(G)$ such that $G[U_1] \in \mathcal{F}_1$ and $G[U_2] \in \mathcal{F}_2$. We need to show that there is a set $T \in \mathcal{S}$ such that $U_1 \subseteq T$ and $T \cap U_2 = \emptyset$. Since the two families $\mathcal{F}_1$ and $\mathcal{F}_2$ are hereditary, $G[U_1 \cap S] \in \mathcal{F}_1$ and $G[U_2 \cap S] \in \mathcal{F}_2$. Since $\mathcal{S}'$ is a $(\mathcal{F}_1, \mathcal{F}_2)$-separating family for $G[S]$ there is a set $A \in \mathcal{S}'$ such that $S \cap U_1 \subseteq A$ and $(S \cap U_2) \cap A = \emptyset$. Since $G[U_1], G[V_1] \in \mathcal{F}_1$ and $G[U_2], G[V_2] \in \mathcal{F}_2$, by Corollary 3.6, we know that $|U_1 \cap V_2| < R(r+1, \ell+1)$ and $|U_2 \cap V_1| < R(r+1, \ell+1)$. Now consider the set $T = A \cup (V_1 \setminus (U_2 \cap V_1)) \cup (U_1 \cap V_2)$. Since $|U_1 \cap V_2| < R(r+1, \ell+1)$ and $|U_2 \cap V_1| < R(r+1, \ell+1)$, by the definition of $\mathcal{S}$, $T \in \mathcal{S}$. Notice that $U_1 \subseteq T$ and $U_2 \cap T = \emptyset$. Hence, we are done. ◀

Lemma 4.3 gives us Theorem 1.5. Suppose there was an approximation algorithm $\mathcal{A}$ for $(\mathcal{F}_1, \mathcal{F}_2)$-Partition, where the approximation factor is defined by a computable function f depending only on the size of an optimal solution, and let the running time of $\mathcal{A}$ be $T(n)$ on an n-vertex input graph. Then, a $(\mathcal{F}_1, \mathcal{F}_2)$-seperating family, for G, of cardinality $2^{\mathcal{O}(\log^c f(\mathsf{opt}^G_{\mathcal{F}_1,\mathcal{F}_2}))}n^{\mathcal{O}(1)}$ can be enumerated in time $2^{\mathcal{O}(\log^c f(\mathsf{opt}))}n^{\mathcal{O}(1)}$, where c is the same constant as in Theorem 1.5, which is at most $r + \ell$.

5 Applications in Parameterized and Exact Algorithms

In this section we relate the results obtained in previous sections to exact and FPT algorithms. The main result of this section is to show that the $(\mathcal{F}_1, \mathcal{F}_2)$-p-Partition problem is FPT. In fact, we propose an algorithm strategy that might result in faster running times than that of the best known algorithms for certain pairs $(\mathcal{F}_1, \mathcal{F}_2)$. We also provide combinatorial bounds on the number of maximal induced subgraphs that have a vertex bipartition (A, B), where $G[A] \in \mathcal{F}_1$ and $G[B] \in \mathcal{F}_2$. We also give a strategy to design an enumeration algorithm for all such maximal induced subgraphs. Similarly, we can find the maximum(minimum) size of such an induced subgraph. These results and their corollaries can be found in the full version of the paper.

Parameterized Algorithms

The question of what is the maximum size of an induced subgraph, that has a vertex bipartition (A, B) with $G[A] \in \mathcal{F}_1$ and $G[B] \in \mathcal{F}_2$, brings us to the question of how 'far' a graph is from becoming a graph with the desired bipartition. The $(\mathcal{F}_1, \mathcal{F}_2)$-p-Partition problem addresses this question. In this part, we look at this problem and a technique to solve this problem, when the pair of families are a good pair of families. The technique we use is an adaptation of the popular iterative compression technique.

▶ **Observation 5.1.** *If an instance (G, k) is a* YES *instance of $(\mathcal{F}_1, \mathcal{F}_2)$-p-Partition, then for any induced subgraph $G' \leq_s G$, (G', k) is also a* YES *instance of $(\mathcal{F}_1, \mathcal{F}_2)$-p-Partition.*

Let $\mathcal{F}_1$ and $\mathcal{F}_2$ be a good pair of graph families, and for any positive integer k, the families $\mathcal{F}_1 + kv$ and $\mathcal{F}_2 + kv$ have FPT recognition algorithms, that is, there are algorithms which take as input a graph G and an integer k, decides whether $G \in \mathcal{F}_i + kv, i \in \{1, 2\}$ and runs in time $f(k)|V(G)|^{\mathcal{O}(1)}$. For ease of notation, if $\mathcal{F}_1$ and $\mathcal{F}_2$ be a good pair of graph families, and the families $\mathcal{F}_1 + kv$ and $\mathcal{F}_2 + kv$ have FPT recognition algorithms, then we call $(\mathcal{F}_1, \mathcal{F}_2)$ an FPT-*good pair of families*.

We obtain a fast FPT algorithm for $(\mathcal{F}_1, \mathcal{F}_2)$-p-Partition, as claimed in Theorem 1.6 by incorporating the iterative compression technique. For more details about the algorithmic technique of iterative compression we refer to the book (chapter 4 [3]).

Proof Sketch for Theorem 1.6. Let (G, k) be an input instance. The algorithm is based on the iterative compression technique. Due to Observation 5.1, the iterative compression technique is meaningful for this problem. The iteration step is exactly as described in [3](chapter 4). The description of the compression problem and an algorithm to solve the same is given below.

The input of the compression problem is a graph G' and a vertex set $S \subseteq V(G')$, of size at most $k + 1$. The set S satisfies the property that there is a partition $V_1 \uplus V_2$ of $V(G) \setminus S$ such that $G[V_1] \in \mathcal{F}_1$ and $G[V_2] \in \mathcal{F}_2$. The compression problem outputs YES if there is a vertex set S' of size at most k such that there is a partition $V_1' \uplus V_2'$ of $V(G) \setminus S'$ with $G[V_1'] \in \mathcal{F}_1$ and $G[V_2'] \in \mathcal{F}_2$. Otherwise, the output is NO. Lemma 4.3 can be used to solve the compression problem in time $2^{\mathcal{O}(\log^c k)} n^{\mathcal{O}(1)} \cdot 2k \max\{T_1(n, k), T_2(n, k)\}$.

By Lemma 4.3, we know that there is an enumeration algorithm which outputs a $(\mathcal{F}_1, \mathcal{F}_2)$-separating family $\mathcal{S}$ of cardinality $2^{\mathcal{O}(\log^c |S|)} n^{\mathcal{O}(1)}$ in time $2^{\mathcal{O}(\log^c |S|)} n^{\mathcal{O}(1)}$. Now for each $S \in \mathcal{S}$ and each pair of non-negative integers k_1, k_2 such that $k_1 + k_2 \leq k$, we run $\mathcal{A}_1$ on $(G[S], k_1)$ and $\mathcal{A}_2$ on $(G - S, k_2)$. We output YES if both $\mathcal{A}_1$ and $\mathcal{A}_2$ outputs YES. Otherwise our algorithm will output NO. The full proof is in the full version of the paper. ◀

There are several corollaries of Theorem 1.6, to be found in the full version.

6 Conclusion

In this paper, we studied the parameterized communication complexity of the function $\mathsf{GDISJ}_{G,\mathcal{F}_1,\mathcal{F}_2}$. We also obtained separating families for good pairs of families, and used them to give combinatorial bounds, exact algorithms and FPT algorithms. An important question here is to see if the lower bounds for CIS_G can be used to obtain non-trivial lower bounds for $\mathsf{GDISJ}_{G,\mathcal{F}_1,\mathcal{F}_2}$, when $(\mathcal{F}_1, \mathcal{F}_2)$ is a good pair of graph families. Also, it would be interesting to study the communication complexity of these functions in terms of other parameters of the input. We would like to thank Pranabendu Misra and an anonymous reviewer for insightful comments.

References

1. Kazuyuki Amano. Some improved bounds on communication complexity via new decomposition of cliques. *Discrete Applied Mathematics*, 166:249–254, 2014. doi:10.1016/j.dam.2013.09.015.
2. Nicolas Bousquet, Aurélie Lagoutte, and Stéphan Thomassé. Clique versus independent set. *Eur. J. Comb.*, 40:73–92, 2014. doi:10.1016/j.ejc.2014.02.003.
3. Marek Cygan, Fedor V. Fomin, Lukasz Kowalik, Daniel Lokshtanov, Dániel Marx, Marcin Pilipczuk, Michal Pilipczuk, and Saket Saurabh. *Parameterized Algorithms*. Springer, 2015. doi:10.1007/978-3-319-21275-3.
4. Marek Cygan and Marcin Pilipczuk. Split vertex deletion meets vertex cover: New fixed-parameter and exact exponential-time algorithms. *Inf. Process. Lett.*, 113(5-6):179–182, 2013. doi:10.1016/j.ipl.2013.01.001.
5. R. Diestel. *Graph Theory*. Springer, Berlin, second ed., electronic edition, February 2000.
6. Tomas Feder, Pavol Hell, Sulamita Klein, and Rajeev Motwani. Complexity of graph partition problems. In *Proceedings of the Thirty-first Annual ACM Symposium on Theory of Computing*, STOC'99, pages 464–472, New York, NY, USA, 1999. ACM. doi:10.1145/301250.301373.
7. Mika Göös. Lower bounds for clique vs. independent set. In *IEEE 56th Annual Symposium on Foundations of Computer Science, FOCS 2015, Berkeley, CA, USA, 17-20 October, 2015*, pages 1066–1076, 2015. doi:10.1109/FOCS.2015.69.
8. Mika Göös, T. S. Jayram, Toniann Pitassi, and Thomas Watson. Randomized communication vs. partition number. *Electronic Colloquium on Computational Complexity (ECCC)*, 22:169, 2015. URL: http://eccc.hpi-web.de/report/2015/169.
9. Mika Göös, Toniann Pitassi, and Thomas Watson. Deterministic communication vs. partition number. In *IEEE 56th Annual Symposium on Foundations of Computer Science, FOCS 2015, Berkeley, CA, USA, 17-20 October, 2015*, pages 1077–1088, 2015. doi:10.1109/FOCS.2015.70.
10. Vince Grolmusz and Gábor Tardos. A note on non-deterministic communication complexity with few witnesses. *Theory Comput. Syst.*, 36(4):387–391, 2003. doi:10.1007/s00224-003-1158-7.
11. Hao Huang and Benny Sudakov. A counterexample to the Alon-Saks-Seymour conjecture and related problems. *Combinatorica*, 32(2):205–219, 2012. doi:10.1007/s00493-012-2746-4.
12. Eyal Kushilevitz and Noam Nisan. *Communication Complexity*. Cambridge University Press, New York, NY, USA, 1997.
13. László Lovász. Communication complexity: a survey. *Paths, flows, and VLSI-layout*, pages 235–265, 1990.
14. László Lovász. Stable sets and polynomials. *Discrete Mathematics*, 124(1-3):137–153, 1994. doi:10.1016/0012-365X(92)00057-X.
15. Manami Shigeta and Kazuyuki Amano. Ordered biclique partitions and communication complexity problems. *Discrete Applied Mathematics*, 184:248–252, 2015. doi:10.1016/j.dam.2014.10.029.
16. Mihalis Yannakakis. Expressing combinatorial optimization problems by linear programs. *Journal of Computer and System Sciences*, 43(3):441–466, 1991. doi:10.1016/0022-0000(91)90024-Y.
17. Andrew Chi-Chih Yao. Some complexity questions related to distributive computing(preliminary report). In *Proceedings of the Eleventh Annual ACM Symposium on Theory of Computing*, STOC'79, pages 209–213, New York, NY, USA, 1979. ACM. doi:10.1145/800135.804414.