Parameterized algorithms for k - path counting

Omkar Bhalerao

Parameterized algorithms for k - path counting

Dissertation submitted in partial fulfillment of the requirements for the degree of

Master of Technology
in
Computer Science

by

Omkar Bhalerao
[Roll No: CS2040]

under the guidance of

Dr. Arijit Ghosh
Advanced Computing and Microelectronics Unit

and

Prof. Saket Saurabh
Institute of Mathematical Sciences, Chennai

Indian Statistical Institute
Kolkata-700108, India

July 2022

To my family

CERTIFICATE

This is to certify that the dissertation titled **"Parameterized algorithms for approximate k - path counting"** submitted by **Omkar Bhalerao** to Indian Statistical Institute, Kolkata, in partial fulfillment for the award of the degree of **Master of Technology in Computer Science** is a bonafide record of work carried out by him under our supervision and guidance. The dissertation has fulfilled all the requirements as per the regulations of this institute and, in our opinion, has reached the standard needed for submission.

Arijit Ghosh

Dr. Arijit Ghosh
Assistant Professor
Advanced Computing and Microelectronics Unit
Indian Statistical Institute, Kolkata

[illegible]

Prof. Saket Saurabh
Professor
Theoretical Computer Science Unit
Institute of Mathematical Sciences,
Chennai

Acknowledgement

I am extremely grateful to my advisor, *Prof. Saket Saurabh* for not only allowing me to work on this problem, but for also giving me the freedom to explore various techniques along the way. His valuable guidance and advice at different point during this thesis has definitely played a pivotal role in shaping this work.

I would also like to express my gratitude to *Prof. Arijit Ghosh*, who has been an amazing mentor to me during my journey at the ISI. His ideas, encouragement, enthusiasm and guidance has helped me in shaping my thoughts and motivated me strive harder.

I am grateful to all the faculty members at ISI for their constant support. I am especially thankful to *Prof. Arijit Bishnu and Prof. Arijit Ghosh*, since it was their course on Randomized Algorithms which sparked my interest in the subject. Last but not the least, I would like to thank my family,

since this thesis could not have come into existence without their support. I am especially grateful to my parents, who had to take up on some of my responsibilities during the completion of this thesis, and bear with me during this period.

Omkar Bhalerao
Indian Statistical Institute
Kolkata - 700108, India.

Abstract

The problem of counting the number of subgraphs of a specific kind within an input graph $G = (V, E)$ has been extensively studied in literature. However, when it comes to designing good algorithms for such problems, even the most simple cases, such as counting k - paths pose some challenges. One of the many possible ways to cope with the hardness of such problems is by studying them from the lens of Parameterized algorithms. In this thesis, we explore FPT approximation schemes (FPT-AS) for counting k - paths in G.

In the first part, we look at additional parameterizations for the problem of k - path counting. In particular, for directed graphs, we design a randomized FPT-AS, whose running time depends upon the size of the hitting set for all k - paths in the graph, along with the parameter k. In case of undirected graphs, we design a randomized FPT-AS, with running time dependent on k and the size of the max-cut in G, conditioned on the existence of efficient sensitivity oracles of certain kind. For both these algorithms, our primary tool is the KNAPSACK problem.

In the second part of the thesis, we look at the problem of counting the number of isomorphic copies of a connected subgraph on k vertices within graphs of bounded degree. We do so by introducing the notion of a subgraph-separating family, which is a natural extension to the Random-Separation technique. Subsequently, by establishing its equivalence with parsimonious families in regular graphs, we design a deterministic FPT-AS for $\#k$-PATH.

Contents

List of Figures

Chapter 1

Introduction and Literature Survey

The problem of counting structures within a graph G has puzzled computer scientists for a long time. Given a graph H, at times, it may be possible to determine the existence of H within G in polynomial time, but the corresponding counting problem, which we typically denote by $\#H$, may not necessarily be solvable in polynomial time. This phenomenon was first demonstrated by Valiant [22] in the context of matchings in a graph. In particular, he showed that a polynomial time algorithm for counting the number of matchings in G is unlikely to exist. In order to cope with the hardness of such problems, we divert our attention to the realm of parameterized complexity.

Given any instance I of some problem P and an associated parameter k, any algorithm with running time $O(f(k)|I|^{O(1)})$ that solves P is said to be Fixed Parameter Tractable (FPT), where f is some computable function of k. In the context of counting subgraphs in G, the most simple structure that one could possibly think of is a path of length k. The corresponding decision version of this problem has been extensively studied in the literature of parameterized algorithms [5, 8, 13, 14, 16, 20, 24]. Hence, one might initially hope to find an algorithm to exactly count the number of copies of a k - path in G. However, it was not long before we figured that even the problem of counting k - paths is not as easy as it seems. In particular, Flum and Grohe [12] proved that it is highly unlikely to find an FPT algorithm for counting k - paths in a graph, essentially bringing us back to square-one. So, instead of completely abandoning the problem, we started investigating FPT - approximation schemes for approximately counting the number of k - paths in a graph. In particular, our objective was to design algorithms with running time $O(f(k, \frac{1}{\epsilon})n^{O(1)})$, which could produce close approximations to the actual count of the k - paths in the input graph. It is only recently that this problem has received widespread attention, with a host of new techniques introduced to tackle this problem.

The $\#k$ - PATH problem has played a pivotal role in the development of parameterized algorithms for approximate counting. The first algorithm to address this question was proposed nearly 15 years ago [6], in which the authors combined the technique of color coding with the classical Karp-Luby technique for approximate counting, resulting in a randomized, exponential space, $O(k^{O(k)}n^{O(1)})$ - time algorithm. Later, the running time was improved by Alon et al. in [2], in which they proposed an algorithm which reduced the dependence on k to single exponential. This algorithm, along with the previous one was primarily based on the color-coding technique, introduced in [5]. The motivation for this problem was based on trying to understand the Protien-Protein Interaction (PPI) networks within uni-celullar organisms. Later, in 2007, Alon and Gutner proposed the first deterministic algorithm for the $\#k$ - PATH problem [3], which had a running time of $(2e)^{k+O(\log^3 k)}m \log n$ for $\epsilon = k^{-O(1)}$, with a continued exponential dependence on the space requirement.

After nearly a decade, In 2018, Brand et al. [9] proposed a randomized FPT-AS with running time of $O(4^k m\epsilon^{-2})$ and exponential dependence on the space. Subsequently Bj¨orklund et al. [17] also gave a deterministic algorithm with similar running time. In 2021, Lokshtanov et al. [18] proposed an algorithm, which bridged the gap between the existing $4^k\mathsf{poly}(n)$ algorithms and the conjectured [15] $2^k\mathsf{poly}(n)$ algorithm, by proposing an FPT-AS with running time $O(2.619^k\mathsf{poly}(n)\frac{1}{\epsilon^2})$.

Chapter 2

Techniques for counting k - paths

2.1 Introduction

In this chapter, we briefly discuss the techniques which have been used to design FPT-AS for $\#k$-PATH problem. In particular, we briefly describe the techniques of Color-coding and Extensor-coding.

2.2 Color-Coding

The principle idea behind the color-coding technique is that by coloring the input graph $G = (V, E)$ with a suitable number of colors, it might be possible to efficiently look for the substructure that we are interested in.

In the context of k - paths, if we independently assign to every vertex of G a color, chosen uniformly at random from a set of $[k]$ colors, then with probability at least $\frac{1}{e^k}$, given a k - path $\mathcal{P}$, every vertex on $\mathcal{P}$ will receive a different color. This reduces the problem of finding k - paths to the relatively easier problem of finding colorful k - paths, which can be solved deterministically in $O(2^k n^{O(1)})$ time using dynamic programming.

However, the technique by itself cannot be employed directly for approximate counting of k - paths in G. In order to extend its functionality in the context of $\#k$-PATH problem, Alon and Gutner introduced the notion of a balanced family of hash functions [3]. More concretely,

Definition 2.2.1. Let $1 \leq k \leq n$ and $\delta > 1$. A family of functions from $[n]$ to $[k]$ is a δ - *balanced (n, k) - family of hash functions* if there exists a constant $T > 0$ such that for every $S \in \binom{[n]}{k}$, the number of functions in the family which are injective on S lies within $\frac{T}{\delta}$ and δT.

In [3], the authors proposed a construction of a relatively small δ - balanced (n, k) - family of hash functions. It is now easy to note that if we color the vertices of G using the functions defined in the above family, then every k - path $\mathcal{P}$ will be colored colorfully by at least $\frac{T}{\delta}$ and at most $T\delta$ colorings within the family. Hence the problem of counting k - paths reduces to counting colorful k - paths, which can be solved in $2^k n^{O(1)}$ - time using dynamic-programming. If we denote the δ - balanced family of (n, k) - functions by $\mathcal{F}$, and for a given $f \in \mathcal{F}$, the number of colorful k-paths in G with respect to f by N_f, then $\sum_{f \in \mathcal{F}} \frac{N_f}{T}$ represents an approximation to the count of the k - paths in G.

Thus, for the above technique to work, it is absolutely essential that $|\mathcal{F}|$ is relatively small. This is guaranteed by the following theorem:

Theorem 2.2.1. *[3] For a fixed $1 \leq \delta \leq 2$, a δ - balanced (n, k) - family of perfect hash functions of size $2^{O(k \log \log k)}$ can be constructed in $2^{O(k \log \log k)} n \log n$ time.*

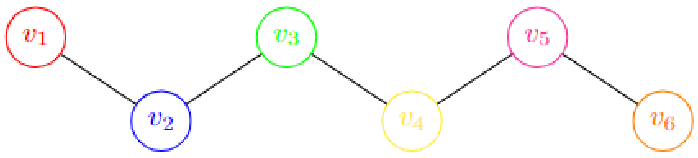

Figure 2.1: A colorful path

2.3 Extensor-Coding

2.3.1 A brief detour into Exterior Algebra

Here we briefly discuss the details of exterior algebra. A relatively detailed exposition to the subject can be found in [7]. Consider a field $\mathbb{F}$ and a positive integer k. Let $V = \mathbb{F}^k$ be a k-dimensional vector space over $\mathbb{F}$ with basis elements $e_1, e_2, \cdots, e_k$. We wish to extend V to a 2^k-dimensional vector space W. In order to do so, it is sufficient to put together a set of 2^k linearly independent vectors and let W be their span i.e. we let W to be the set of all possible linear combinations of the elements in the basis set $\mathcal{B}$. We can uniquely identify the elements in $\mathcal{B}$ by indexing them with subsets of $[k]$. In particular, $\mathcal{B} = \{e_I \mid I \subseteq [k]\}$.

Consider any $x \in W$. Then $x = \sum_{I \in \mathcal{B}} a_I e_I$, where the constants a_I are drawn from the field $\mathbb{F}$. In order to compute linear combinations of the elements in W, we need to be able to add vectors in W and take scalar product of a vector in W with a scalar $a \in \mathbb{F}$. Both these operations are very similar to what we would usually do when dealing with polynomials. For instance, if $x = 3e_{\{1,2\}} + 4e_{\{3\}}$ and $y = e_{\{2\}} - 2e_{\{1,2\}}$, then $x + 2y = 2e_{\{2\}} - e_{\{1,2\}} + 4e_{\{3\}}$.

The next obvious question is how to compute the basis elements in $\mathcal{B}$. In order to address this question, we transform W into an algebra $\Lambda(\mathbb{F}^k)$ by introducing the notion of a wedge product. In particular, for any $e_I \in \mathcal{B}$, $e_I = \bigwedge_{i \in I} e_i$. The wedge product satisfies the properties of bilinearity and associativity, which allows us to restrict our attention to the behaviour of the wedge product for the elements in $\mathcal{B}$ only. This is an easy consequence of bilinearity of Λ, which facilitates in extending the definition of wedge product to elements in $\Lambda(\mathbb{F}^k)$.

Given two sets $I, J \subseteq [k]$, if $I \cap J \neq \phi$, then $e_I \bigwedge e_J = 0$. Otherwise, $e_I \bigwedge e_J = \pm\, e_{I \cup J}$. Equipped with the above definition, it is immediately evident that $\Lambda(\mathbb{F}^k)$ is closed under $\bigwedge$. In order to determine the sign associated with the wedge product $e_I \bigwedge e_J$, we first sort the elements in I and J, and then combine them in that order into a sequence σ. Then $e_I \bigwedge e_J = (-1)^{\text{sign}(\sigma)} e_{I \cup J}$. Hence, we can conclude that the wedge product between the elements in all possible subsets of the canonical basis vectors for $\mathbb{F}^k$ constitute the basis set for $\Lambda(\mathbb{F}^k)$. This leads to the following crucial observations for the wedge product between elements in $\Lambda(\mathbb{F}^k)$:

1. For any $i, j \in [k]$, $e_i \bigwedge e_j = e_{\{i,j\}} = -e_j \bigwedge e_i$, i.e. the wedge product is anti-commutative on the elements of $\mathbb{F}^k$.
2. An immediate consequence of this anti-commutativity is that for any $x \in \mathbb{F}^k, x \bigwedge x = 0$.
3. Further, for any $v_1, v_2, \cdots, v_k \in \mathbb{F}^k$, $\bigwedge_{i\in[k]} v_i = \det(v_1|v_2|\cdots|v_k) \bigwedge_{i\in[k]} e_i = \det(v_1|v_2|\cdots|v_k)e_{[k]}$.

Equipped with these properties, we now take a look at how this technique can be employed for counting k - paths. The elements of $\Lambda(\mathbb{F}^k)$ are called *extensors.*

2.3.2 k - path counting using extensors

Suppose $G = (V, E)$ is a simple, directed graph on n vertices and m edges. Let us assign a unique k - dimensional Vandermode extensor to every vertex in G. In particular, associate with every vertex $v_i inV$, a unique vector $\chi(v_i)$ in $\mathbb{F}^k$, which is given by $(1, j, j^2, \cdots, j^{k-1})$ where $j = f(i)$ for some injective function f. Further, assign a unique constant y_{uv} to every edge $(u, v) \in E(G)$. Let $W_s(G)$ denote the collection of all possible walks of length $s - 1$ in G.

Given any walk $w = u = v_1 - v_2 - \cdots - v_k = v$ in $W_k(G)$, define the walk extensor $\chi(w)$ associated with w as $\chi(w) = \chi(u) \bigwedge y_{u,v_1} \bigwedge \chi(v_1) \bigwedge y_{v_1,v_2} \bigwedge \chi(v_2) \bigwedge \cdots \bigwedge \chi(v_k)$. Let $Z = \sum_{w\in W_{k-1}(G)} \chi(w)$. We now make the following observations regarding the properties of these extensors:

1. Note that a path is distinguished from a walk by a repeated vertex. More precisely, every walk has at least one repeated vertex whereas no vertex is repeatedly visited in a path. This implies that for any $w \in W_k(G)$, if w is a walk, then $\chi(w) = 0$. So, the only k - walks which contribute to Z are the ones which are k - paths.
2. Every k - path will make a contribution towards Z. More precisely, the coefficient associated with every k - path will be non-zero. This is because, for any k - path $w \in W_k$, we can pull out the constants from $\chi(w)$, effectively producing a wedge product between k unique Vandermonde extensors. This translates to computing the determinant of the matrix, whose columns are these k - dimensional Vandermonde extensors. Since they are unique, the determinant of this matrix will always be non-zero.

In the light of the above mentioned properties, it follows that every k - path gets accounted within Z, and has a unique monomial in the variables $\{y_{u,v} \mid (u, v) \in E(G)\}$ as its coefficient. Now, if we set each $y_{u,v}$ to 1 and sample the vectors $\chi(v_i)$ uniformly at random from $\{-1, 1\}^k$ for every $v_i \in V$, then the expected value of the coefficient of $e_{[k]}$ in Z is proportional to the the number of k - paths in G. [9]. Thus, through exterior algebra, it is possible to design an FPT-AS for $\#k$ - PATH problem.

2.4 Summary

In this chapter, we took a close look at some of the techniques that are widely used in designing FPT-AS for the $\#k$ - PATH problem. In particular, we briefly discussed the color-coding and extensor-coding techniques for counting k - paths in an input graph G.

Chapter 3

Counting Paths using KNAPSACK

3.1 Introduction

In this chapter, we carry out multi-variate analysis f algorithms for approximately counting the number of k - paths in both, directed and undirected graphs. We present separate randomized FPT algorithms for the above problem. The first algorithm is parameterized by k and the size of the hitting set for k - paths in G. The second algorithm, under certain assumptions, gives an FPT algorithm, parameterized by k, the size of the hitting set for specific paths and the size of the max-cut in G. Further, under certain additional assumptions, the running time of this algorithm can be modified to be dependent only on k and the size of the max-cut in G.

The primary object in each of the above algorithms is the KNAPSACK problem.

3.2 KNAPSACK Problem

The classical 0/1 knapsack problem is as follows:

Let $I = \{i_1, i_2, \cdots, i_n\}$ be a set of n items. For every $1 \leq j \leq n$, the item i_j has an associated non-negative weight w_j and a non-negative profit p_j. Given a knapsack of capacity W, the problem asks to fill the knapsack with a subset of items from I, so that the total profit incurred from the items placed in the knapsack is maximized, subject to the constraint that their total weight does not exceed W.

The KNAPSACK problem is known to be NP-Complete. However, given an instance Π of the KNAPSACK problem, if OPT(Π) denotes the total profit incurred from an optimal solution to Π, then there exists an FPTAS (Fully Polynomial Time Approximation Scheme), which for any $0 < \epsilon < 1$, returns a solution that ϵ - approximates OPT(Π). In particular, if $P^{\mathcal{A}}$ denotes the profit earned from the solution to Π obtained by running the FPTAS - $\mathcal{A}$ on it, then $P^{\mathcal{A}} \in [(1-\epsilon)\mathrm{OPT}(\Pi), \mathrm{OPT}(\Pi)]$ for any $0 < \epsilon < 1$. Further, the running time of the algorithm is $O(n^2 \lfloor \frac{n}{\epsilon} \rfloor)$. In conclusion:

Lemma 3.2.1. *[23] There exists a fully polynomial approximation scheme for the KNAPSACK problem, with running time* $O(n^2 \lfloor \frac{n}{\epsilon} \rfloor)$

3.3 Sensitivity Oracle for k - Path Counting

Sensitivity oracles are typically studied in the context of dynamic graphs. Given a positive integer l and an directed graph $G = (V, E)$, an ideal l - sensitive oracle for k - path counting takes G as its initial input and pre-processes it in order to extract necessary information. Subsequently, when queried with a sequence of at most l updates to the edges of G, in the form of edge insertions or deletions, the oracle returns the count of the k - paths in the updated graph. For an l - sensitive oracle to be efficient, it is imperative that the time it needs to pre-process G is $O(f(k)n^{O(1)})$ and the time to respond to the queries is $O(\mathsf{poly}(l)f(k)n^{O(1)})$.

However, note that it is difficult to construct an ideal l - sensitive oracle for counting k - paths in dynamic setting. Instead, we resort to an ϵ - approximate l - sensitive oracle, which approximates the number of k - paths in the updated graph. In particular, the estimate returned by the oracle lies within a $((1-\epsilon), (1+\epsilon))$ factor of the actual count of the k - paths in the modified graph. The following lemma guarantees the existence of such oracles.

Lemma 3.3.1. *[1] For an initial graph G on n vertices and m edges, there exists a random sensitivity oracle that, given any $\epsilon > 0$ and k, produces and estimate to the number of k paths , that with probability > 99 %, is within ϵ relative error, with pre-processing time $\frac{1}{\epsilon^2}4^k\mathsf{poly}(k)\mathsf{poly}(n)$ and update time $\frac{1}{\epsilon^2}l^2 2^{\omega k}$.*

3.4 KNAPSACK Problem and k - path counting

Let $G = (V, E)$ be a simple directed graph on n vertices and m edges, and $\mathcal{F}^k$ be the family of k - paths in G. Let $\mathcal{H}^k \subseteq E$ be a hitting set of smallest size for the paths in $\mathcal{F}^k$ i.e. every k - path in G necessarily passes through at least one edge in $\mathcal{H}^k$. Further, assume that $|\mathcal{H}^k| = t$ and $S \subseteq E$ contains $\mathcal{H}^k$.

Suppose $S = \{e_1, e_2, \cdots, e_s\}$, where $s = |S|$. Consider the following instance of the KNAPSACK problem, which we shall denote by Π^K:

1. $I = S$
2. Every edge in S is assigned unit weight.
3. $W = |S|$, and
4. For every edge $e_i \in S$, define the profit p_i associated with e_i as the number of k - paths in $G \setminus \{e_j \mid i+1 \leq j \leq s\}$, that pass through e_i and haven't already been accounted in the profit p_j for any $j < i$.

Intuitively, it is evident that

1. every k - path in $\mathcal{F}^k$ will contribute towards the profit of some edge in S and
2. every k - path will be accommodated in the profit of exactly one edge in S.

The reason for the latter claim is as follows: Suppose we want to determine p_i for some $1 < i < s$. Let $P := e_{i_1} - e_{i_2} - \cdots - e_{i_{k-1}}$ be a k - path in $G^i := G \setminus \{e_j \mid i+1 \leq j \leq s\}$, that passes through e_i. If P were to pass through e_j for some $j > i$, then it would not have existed in G^i in the first place.

Consequently, at this moment, its contribution to p_j for every $j > i$ is 0. Further, based on the way in which we have defined the profits, it follows that unless P's contribution to the profits has already been taken care of in some $p_j, j < i$, it will be counted in p_i. Once it has been accounted in p_i, no matter what $j > i$ we choose, P will not contribute to p_j. These arguments are formalised in Lemma-3.

In order to compute the profits for the edges in S, we employ a 1-sensitive oracle $\mathcal{O}$. Note here that we assume $\mathcal{O}$ to be an ideal sensitivity oracle for k - path counting. Given access to $\mathcal{O}$, algorithm-1 outlines a procedure to compute the profits of the edges in S. We assume that $\mathcal{O}$ is equipped with a subroutine COUNT-$\mathcal{O}(e)$, which returns the number of k - paths in $G^{'} \cup \{e\}$, where $G^{'}$ represents the current state of the original graph.

Algorithm 1 PROFIT-S($S, G, \mathcal{O}$)

```
$C^0 \leftarrow 0$
$G^S = G \setminus S$          ▷ $G \setminus S$ is the graph obtained by deleting the edges in $S$ from $G$
$\mathcal{O}(G^S)$             ▷ Feed $G \setminus S$ as input to $\mathcal{O}$ for pre-processing
for $i \leftarrow 1$ to $s$ do
    $C^i \leftarrow$ COUNT-$\mathcal{O}(e_i)$
    $p_i \leftarrow C^i - C^{i-1}$
end for
return $\{p_1, \cdots, p_s\}$
```

Lemma 3.4.1. *Every k - path in G will contribute to the profit of exactly one edge in S, where the profits are computed using PROFIT-S($S, G, \mathcal{O}$)*

Proof. Note that since S contains a hitting set for $\mathcal{F}^k$, it is clear that every k - path necessarily passes through at least one edge in S. Consequently, every k - path will make its way into the profit of at least one edge in S. To show that each k - path contributes to the profit of exactly one edge, we proceed by proving the Loop-invariant described below. In order to do so, we induct on the number of iterations of the "for" - loop in the subroutine PROFIT-S($S, G, \mathcal{O}$).

The Loop-invariant: "At the end of the i^{th} - iteration of the for - loop, p_i will be equal to the number of k - paths in $G^i := G \setminus \{e_j \mid i+1 \leq j \leq s\}$, which go through the edge e_i. Furthermore, every other k - path in G^i would be accommodated in the profit of exactly one of the edges in $\{e_1, e_2, \cdots, e_{i-1}\}$" For the sake of convenience, we shall denote this statement by $P(i)$.

Base-case: Suppose $i = 1$. At the end of the 1st - iteration, p_1 should be equal to the number of k - paths in G^1, that pass through e_1. This is precisely the number represented by C_1. Since the subroutine PROFIT-S($S, G, \mathcal{O}$) assigns C_1 to p_1 in the 1st iteration of the for loop, we indeed have $p_1 = C_1$. Further, since S contains a hitting set for all k - paths in G, it is not possible for $G \setminus S$ to contain any k - path. Hence, the base case follows immediately.

Induction-Step: Assume $P(i)$ to be true for every $i \leq j-1$ and consider the j^{th} iteration of the for loop.

Let $\Delta_j = C_j - C_{j-1}$. Δ_j is the number of k - paths introduced in G^{j-1} due to the insertion of the edge e_j. Note that every such newly created k - path will definitely pass through e_j in $G^{j-1} \cup \{e_j\} = G^j$, and hence would contribute to p_j. Observe that none of these k- paths contribute to p_j for every $j < i$, because if any such k-path did make a contribution to some p_j, where$j < i$, then it should have existed prior to the addition of e_j, which is a contradiction.

Now suppose $P = e_{i_1} - e_{i_2} - \cdots - e_{i_k}$ is a k - path in G^j, that does not pass through e_j. Let S_p denote the subset of edges on P which belong to S and let e_l be the most recently inserted edge among the edges in S_p. Since $e_l \neq e_j$ (and hence no other edge in S_p), it follows that P was created prior to the insertion of e_j. More concretely, it was introduced the moment e_l was inserted in G^{l-1}. Note that it is not possible for P to have come into existence unless e_l was added. Now, by the induction hypothesis, P would contribute to p_l, for some $l < j$. Further, its contribution to p_j will be 0, since it gets counted once in C_j and then its contribution to p_j gets subtracted in C_{j-1}. Hence the induction hypothesis holds.

Termination: A similar argument will establish the correctness of $P(i)$ when $i = s$.

Note that the above claim implies that every k - path will contribute to the profit associated with exactly one edge. This proves the lemma. □

Theorem 3.4.1. *Let Π^K be an instance of the KNAPSACK problem defined as above. Then the profit incurred from any optimal solution to Π^K will be equal to $|\mathcal{F}^k|$*

Proof. It is obvious to note that the optimal solution to Π^k will incorporate all of S. Hence, the profit earned by including all the edges from S into the knapsack will be $\sum_{i=1}^{s} p_i$. By lemma-3.4.1, every k - path gets counted exactly once in $\sum_{i=1}^{s} p_i$, therefore, $\sum_{i=1}^{s} p_i = |\mathcal{F}^k|$. □

Consequently, to obtain an exact count of the number of k - paths in G, it is sufficient to solve Π^k exactly. However, note here that we assumed $\mathcal{O}$ to be an ideal 1-sensitive oracle. Turns out, under certain assumptions, we may not be able to construct an ideal 1 - sensitive oracle. Therefore, we rely on an ϵ_1 - approximate 1-sensitive oracle, which we shall denote by $\mathcal{O}^{\epsilon_1}$. The existence of such an oracle is guaranteed by lemma-3.3.1. The following lemma essentially states that by using $\mathcal{O}^{\epsilon_1}$, we can compute estimates of the profits associated with the edges in S, upto a relative error of ϵ_1, where $0 < \epsilon_1 < 1$.

Lemma 3.4.2. *The profits $\hat{p_i}$, obtained by replacing $\mathcal{O}$ with $\mathcal{O}^{\epsilon_1}$ within the subroutine PROFIT-S($S, G, \mathcal{O}$), satisfy $(1-\epsilon)p_i \leq \hat{p_i} \leq (1+\epsilon)p_i$ for every $1 \leq i \leq s$ with high probability.*

Proof. Let $\hat{C_i}$ denote the estimate of the count of k - paths in G, obtained by invoking $\mathcal{O}^{\epsilon_1}$ in the i^{th} - iteration of the for loop within the subroutine PROFIT-S($S, G, \mathcal{O}$). Then, from the definition of $\mathcal{O}^{\epsilon_1}$ and lemma-3.4.2, with high probability, $(1-\epsilon_1)C_i \leq \hat{C_i} \leq (1+\epsilon_1)C_i$ for each $1 \leq i \leq s$. Since $\hat{p_i} = \hat{C_i} - \widehat{C_{i-1}}$, we can conclude that with high probability, every $\hat{p_i}$ satisfies the conditions stated in the lemma. □

We are now in a position to use the FPTAS $\mathcal{A}$ on Π^k, with ϵ_2 being an additional error parameter. This leads us to the following theorem:

Theorem 3.4.2. *Let G be a directed graph on n vertices. Assume that $S \subseteq E$ contains a hitting set for all k - paths in G. Given S, let Π^k be the instance of the KNAPSACK problem, obtained using the procedure whose details are outlined above. Then, given any $0 < \epsilon_1, \epsilon_2 < 1$, there exists a randomized FPT algorithm, which with high probability, returns an estimate of the number of k - paths in G within a relative error of $((1-\epsilon_1)(1-\epsilon_2)|\mathcal{F}^k|, (1+\epsilon_1)|\mathcal{F}^k|)$, and has running time $O(\frac{1}{\epsilon_1\epsilon_2}2^{(2k+\frac{\omega k}{2})}\mathsf{poly}(n))$.*

Proof. For the sake of convenience, given a value x, we shall denote its estimate by $\hat{x}$. From lemma-3.4.2, it is clear that with high probability, $(1-\epsilon_1)p_i \leq \hat{p_i} \leq (1+\epsilon_1)p_i$ where p_i is the profit associated with the edge e_i in S. With these estimates as profits, the total profit incurred from any optimal solution to Π^k satisfies $(1-\epsilon_1)|\mathcal{F}^k| \leq |\hat{\mathcal{F}^k}| \leq (1+\epsilon_1)|\mathcal{F}^k|$. This is a consequence of theorem-3.4.1. Finally, by executing $\mathcal{A}$ on Π^k, we get the desired approximation factor for $|\mathcal{F}^k|$.

As for the running time, the majority of the time the algorithm spends is in pre-processing $G \setminus S$, followed by responding to the $|S|$ queries concerning edge insertions. This takes $O(\frac{1}{\epsilon_1} 2^{(2k+\frac{\omega k}{2})}\mathsf{poly}(n))$ - time. Combining this with the running time of $O(\frac{1}{\epsilon_2}\mathsf{poly}(n))$, introduced due to the execution of $\mathcal{A}$ on Π^k, gives us the running time stated in Theorem-2. □

It is clear that the correctness of our algorithm is crucially dependent on the existence of a set S, that contains $\mathcal{H}^k$. In order to find S, we resort to Chromatic-Coding.

3.5 Finding S through Chromatic-Coding

The technique of Chromatic-Coding was introduced by Alon et. al [4] for giving a subexponential algorithm for Feedback Arc Set in Tournaments. This technique is used primarily for edge modification problems. Suppose the number of edge modifications required in a given problem in upper bounded by k. Further, assume that every vertex of the input graph is colored independently with a color, chosen uniformly at random from a palette of $o(k)$ colors. The hope is that, the edges which belong to the solution set of the given problem, will end up being properly colored i.e. both the end points of every edge in the solution set will receive distinct colors. As a result, it would suffice to consider only the properly colored edges of the graph, while looking for a solution to the problem in question. The following theorem formalises this intuition.

Theorem 3.5.1. *[11] If the vertices of a simple graph G on k edges are colored independently and uniformly at random with $\sqrt{8k}$ colors, then the probability that $E(G)$ is properly colored is at least $2^{-\sqrt{\frac{k}{2}}}$.*

Let $G=(V,E)$ be a simple directed graph on n vertices and m edges. Assume the size of the minimal hitting set for all k - paths in G to be exactly t. Let us denote this set by $\mathcal{H}$. If we color each vertex of G independently with a color, chosen uniformly at random from a palette of $\sqrt{8t}$ colors, then from theorem-3.5.1, we can conclude that with probability at least $2^{-\sqrt{\frac{t}{2}}}$, $G^{'} = (V, \mathcal{H})$ will be properly colored. Consequently, if we let S to be the subset of all those edges in G, both of whose end-points receive different colors, then with certain probability, S will contain $\mathcal{H}$. Equipped with such an S, we can obtain the corresponding KNAPSACK instance and resort to theorem-3.4.2 in order to approximately count the number of k - paths in G.

In order to boost the probability of finding an S which will contain $\mathcal{H}$, we carry out $2^{\sqrt{\frac{t}{2}}}$ independent executions of the algorithm for estimating $|\mathcal{F}^k|$, and subsequently return the maximum of all the estimates obtained so far. This implicitly involves the coloring step, described in the preceding paragraph. The probability that none of coloring-steps produce a set S with the given property is at most

$$\left(1-\frac{1}{2^{\sqrt{\frac{t}{2}}}}\right)^{2^{\sqrt{\frac{t}{2}}}} \leq \frac{1}{e}$$

Therefore, with probability at least $1-\frac{1}{e}$, one of the $2^{\sqrt{\frac{t}{2}}}$ estimates will be obtained by operating on a set S, which will contain $\mathcal{H}$. This leads to algorithm-2. Hence we have the following result:

Theorem 3.5.2. *Let G be a directed graph on n vertices and Let t be the size of the smallest hitting set for the paths in $\mathcal{F}^k$. Then with constant probability, algorithm-2 returns an estimate of the number* *of k - paths in G, which lies within a factor of $((1-\epsilon_1)(1-\epsilon_2),(1+\epsilon_1))$ - times the actual number of k - paths in G, and has running time $O(\frac{1}{\epsilon_1\epsilon_2}2^{(2k+\frac{\omega k}{2}+\sqrt{\frac{t}{2}})}\mathsf{poly}(n))$*

Proof. Let $S_1, S_2, \cdots, S_T$ denote the sets obtained by coloring the vertices of G independently for $T = 2^{\sqrt{\frac{t}{2}}}$ iterations. Note that none of the estimates of $|\mathcal{F}^k|$ can exceed $(1+\epsilon_1)|\mathcal{F}^k|$. So, for the variable 'max' to be at least $(1-\epsilon_2)(1-\epsilon_1)|\mathcal{F}^k|$, it must be the case that at least one of the S_j's contains $\mathcal{H}^k$. Additionally, conditioned on the existence of such S_j, the estimates for the profits of the edges in S_j should satisfy the conditions in lemma-3.4.2. The former event occurs with probability at least $1-\frac{1}{e}$ and the latter with probability at least 0.99. The running time estimate follows immediately from the algorithm itself. This completes the proof. □

Algorithm 2 PROFIT-S$(G, \mathcal{O}^{\epsilon_1}, \epsilon_2)$

```
max ← 0
for $i \leftarrow 1$ to $2^{\sqrt{\frac{t}{2}}}$ do
    $S \leftarrow \phi$
    for $v \in V$ do
        $c(v) \leftarrow$ a color chosen independently and uniformly at random from $[\sqrt{8k}]$ - colors
        Color $v$ with $c(v)$
        for $w \in N^+(v)$ do                    ▷ $N^+(v)$ consists of the "out" neighbours of $v$
            if $w$ is colored and $c(v) \neq c(w)$ then
                $S \leftarrow S \cup \{(v,w)\}$                ▷ Adding properly colored edges to $S$
            end if
        end for
    end for
    $G^S = G \setminus S$                ▷ Equipped with $S$, we proceed to compute the profits
    $\mathcal{O}^{\epsilon_1}(G^S)$
    $C^0 \leftarrow 0$
    for $i \leftarrow 1$ to $|S|$ do
        $C^i \leftarrow$ COUNT-$\mathcal{O}^{\epsilon_1}(e_i)$
        $p_i \leftarrow C^i - C^{i-1}$
    end for
    $\Pi^k \leftarrow (S, \{1,1,\cdots,1\}, |S|, \{p_1,\cdots,p_{|S|}\})$ ▷ The set of 1's are the unit weights for the edges in $S$
    $P \leftarrow \mathcal{A}(\Pi^k, \epsilon_2)$            ▷ $P$ is the profit obtained by running the FPTAS $\mathcal{A}$ on $\Pi^k$.
    if max > $P$ then
        max ← $P$                        ▷ "max" contains the maximum profit.
    end if
end for
return max
```

3.6 Hitting sets in cuts

3.6.1 Preliminaries and notations

Let $G=(V,E)$ be a simple undirected graph on n vertices and m edges. Assume that the minimum degree of G, denoted by $\delta(G)$, is at least 3. Given $A \subset V$, define ∂A, which we shall refer to as the boundary of the cut $(A, V \setminus A)$, as the set of edges in G which cross the cut $(A, V \setminus A)$. For every

$u \in V$ and integers $1 \leq x < y \leq n$, let $\mathcal{F}^u_{[x,y]}$ denote the family of l - length paths in G, which originate from u, where $x \leq l \leq y$. We shall represent the hitting set corresponding to the paths in $\mathcal{F}^u_{[x,y]}$ by $\mathcal{H}^u_{[x,y]}$. In general, $\mathcal{F}_{[x,y]}$ will represents the family of paths with length between x and y, inclusive of x and y both. Note that $\mathcal{H}^u_{[2,k-1]}$ will contain a hitting set for the k - paths in G that start at u. We shall denote the family of such k - paths by $\mathcal{F}^u$ and its hitting set by $\mathcal{H}^u$. Further, we shall continue denoting the collection of k - paths in G by $\mathcal{F}^k$ and its corresponding hitting set by $\mathcal{H}^k$. We assume that G contains a hitting set $\mathcal{H}_{[2,k-1]}$ of size at most t. Finally, let $\mathcal{O}^{\epsilon_1} = \{\mathcal{O}^{\epsilon_1}_{u,v} \mid u, v \in V,\, u \neq v\}$, where $\mathcal{O}^{\epsilon_1}_{u,v}$ is an ϵ_1 - approximate 1 - sensitive oracle for the number of k - paths from u to v.

We now present the following definition: Let $\mathcal{F}$ be a family of connected subgraphs of G, such that every subgraph in $\mathcal{F}$ has non-empty intersection with a set T of vertices. Suppose $S \subseteq E$ is a hitting set for $\mathcal{F}$. The set of vertices that are not reachable from T in $G \backslash S$ constitute a *shadow* of S.

Equipped with the definition of the shadow of a hitting set, we now state the following result:

Theorem 3.6.1. *[21] Let $\mathcal{F}$ be a family of connected subgraphs of G, such that every subgraph in $\mathcal{F}$ has non-empty intersection with a set T of vertices. If G has a hitting set of size at-most t for the subgraphs in $\mathcal{F}$, then with probability $2^{-O(t)}$, G contains an optimal hitting set S for $\mathcal{F}$ which has the following properties:*

1. *shadow of S is covered by Z,*
2. *no edge of S is contained in Z,*

where Z is a subset of V, which can be computed in $2^{O(t)}n^{O(1)}$ time.

3.6.2 Overview of the algorithm

We begin with an overview of the algorithm. Suppose we fix a vertex $u \in V$ and consider the set $\mathcal{F}^u_{[2,k-1]}$. Observe that

1. every graph in $\mathcal{F}^u_{[2,k-1]}$ is connected and
2. every path in $\mathcal{F}^u_{[2,k-1]}$ has a non-empty intersection with $T = \{u\}$.

Since G has a hitting set $\mathcal{H}_{[2,k-1]}$ of size at most t, it follows that G also consists of a hitting set for the paths in $\mathcal{F}^u_{[2,k-1]}$, whose size is upper bounded by t. Therefore, from theorem-3.6.1 we can conclude that, that with probability at least $2^{-O(t)}$, there exists a minimum hitting set $\mathcal{H}'_{[2,k-1]}$ in G for the paths in $\mathcal{F}^u_{[2,k-1]}$, which has the following properties:

1. Z covers the shadow of $\mathcal{H}'_{[2,k-1]}$ and
2. every edge of $\mathcal{H}'_{[2,k-1]}$ lies outside Z,

where $Z \subseteq V$ can be determined in $2^{O(t)}n^{O(1)}$ time.

The following discussion is based on the assumption that G indeed contains such a hitting set. Since none of the edges in $\mathcal{H}'_{[2,k-1]}$ belong to Z, every edge in $\mathcal{H}'_{[2,k-1]}$ either crosses the cut $(Z, V \setminus Z)$, or has both of its endpoints within Z. Theorem-3.6.2 essentially states that it is not possible for any edge in $\mathcal{H}'_{[2,k-1]}$ to be completely within $G \setminus Z$. Therefore, we have a cut $(Z, V \setminus Z)$, which has the property that every edge in $\mathcal{H}'_{[2,k-1]}$ lies in $\partial(Z)$. By letting $S = \partial(Z)$, we have essentially found a subset of edges in G, which contains a hitting set for all the k - paths in $\mathcal{F}^u_{[2,k-1]}$, and hence $\mathcal{F}^u$. This allows us to adopt the techniques we saw in section-3 to find an approximation to $|\mathcal{F}^u|$.

More precisely, conditioned on the existence of $\mathcal{O}^{\epsilon_1}$, we can effectively translate the above problem of estimating $|\mathcal{F}^u|$ into an instance of the KNAPSACK problem. In particular, we let $I = S$, assign unit weight to the edges in S and let $W = |S|$. To compute the profit $\hat{p^i_u}$ for every $e_i \in S$, we first collect the estimates $\hat{p^i_{u,v}}$ for e_i, obtained from the oracles $\mathcal{O}^{\epsilon_1}_{u,v}$ for every $v \in V \setminus \{u\}$, and then set $\hat{p^i_u} = \sum_{v \in V \setminus \{u\}} \hat{p^i_{u,v}}$. Observe that $\sum_{e_i \in S} \hat{p^i_u}$ gives us an estimate of $|\mathcal{F}^u|$. This is because every k - path in $\mathcal{F}^u$ ends in some $w \in V \setminus \{u\}$. Consequently, by adopting the proof of lemma-3.4.1 for the oracles in $\mathcal{O}^{\epsilon_1}$, we can conclude that every such k - path is bound to contribute to the estimated profit of exactly one edge in S. Once we have the estimates $\hat{p^i_u}$ for the profits of the edges e_i in S, we can run our FPTAS $\mathcal{A}$ to obtain the desired approximation to $|\mathcal{F}^u|$. But note that this is contingent on the fact that S contains a hitting set for $\mathcal{F}^u$, which in turn is determined by the set Z. To boost the probability of succeeding in finding a set which will contain the desired hitting set, we obtain $2^{O(t)}$ independent instances of the KNAPSACK problem, each producing some estimate of $|\mathcal{F}^u|$, and subsequently select the maximum of these estimates. Let us denote by S_u, the set that achieves this maximum.

We now repeat the procedure described in the preceding paragraph for every vertex in G, and then compute $\sum_{v \in V} |\hat{\mathcal{F}^v}|$ to obtain an estimate of the number of k - paths . However, note that every k path will be counted twice, once for each of its endpoints. For instance, consider a k - path from u to v in G. This path will be counted once in the profit of some edge in S_u and once in the profit of some edge in S_v. Consequently, we must return $\frac{\sum_{v \in V} |\hat{\mathcal{F}^v}|}{2}$ as the estimate of the number of k - paths in G.

Let $\beta(G)$ denote the size of the max-cut in G. Then it is quite clear that $|S| \leq \beta(G)$. One can easily note that the running time of this algorithm is $O(2^{O(t)} f(k) g(k) \beta(G) \mathsf{poly}(n))$, where $f(k)$ and $g(k)$ denote the maximum pre-processing and response time taken by any oracle in $\mathcal{O}^{\epsilon_1}$.

Now suppose that t is the size of the optimal hitting set for paths in $\mathcal{F}_{[2,k-1]}$. Then, since $\partial(Z)$ contains an optimal hitting set for the paths in $\mathcal{F}_{[2,k-1]}$, it has to be the case that $t \leq \beta(G)$. This gives us a running time of $O(2^{O(\beta(G))} f(k) g(k) \beta(G) \mathsf{poly}(n))$.

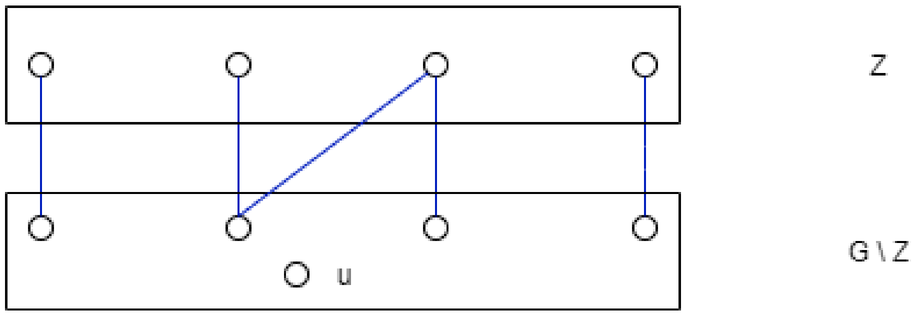

Figure 3.1: The cut $(Z, V \setminus Z)$

3.6.3 The gory details

We now prove the following theorem:

Theorem 3.6.2. *Every edge which belongs to $\mathcal{H}'_{[2,k-1]}$ crosses the cut $(Z, G \setminus Z)$.*

Proof. We will prove this statement by contradiction. Let $(x, y) \in \mathcal{H}'_{[2,k-1]}$ so that both x and y belong to $G \setminus Z$ i.e. the edge (x, y) lies completely inside $G \setminus Z$. Since neither x nor y belong to Z, it follows that both x and y lie outside the shadow of $\mathcal{H}'_{[2,k-1]}$. Hence there exist paths P_x and P_y from u to x and y respectively in $G \setminus \mathcal{H}'_{[2,k-1]}$. Note that $u \in G \setminus Z$, since u is reachable from itself. Since the edges in $\mathcal{H}'_{[2,k-1]}$ hit every path of length between 2 and $k-1$, it must be the case that the lengths of P_x and P_y either exceed $k-1$ or are at most 1. If any one of the two paths satisfies the former condition, then we immediately have a contradiction, since we would have found a path of length $k-1$, which is disjoint from $\mathcal{H}'_{[2,k-1]}$. Since neither of the two paths can be of length 0, let us assume that the lengths of P_x and P_y are 1.

Since $\delta(G) \geq 3$, y must have at least one neighbour t which is different from x and u. Similarly, x must have at least one neighbour w, which is different from y and u. Let $N(x)$ and $N(y)$ denote the neighbourhoods of x and y in G. If for any $t \in N(y)$, the edge $(y, t) \notin \mathcal{H}'_{[2,k-1]}$, then we immediately have a contradiction, since we would end up with a $u - y - t$ - path in $G \setminus \mathcal{H}'_{[2,k-1]}$. Consequently, every edge incident on y should be in $\mathcal{H}'_{[2,k-1]}$. By a similar argument, every edge incident on x should be contained within $\mathcal{H}'_{[2,k-1]}$.

Now, consider any path that passes through (x, y). Any such path enters x via one of its neighbours and leaves y through one its neighbours. Therefore, every such path will be hit by at least one of the edges incident on x and y. Since all the edges incident on x and y belong to $\mathcal{H}'_{[2,k-1]}$, we can conclude that every path through x, y is hit by $\mathcal{H}'_{[2,k-1]}$. Consequently, if we remove (x, y) from $\mathcal{H}'_{[2,k-1]}$, we will obtain a smaller set, which continues to intersect every path through (x, y). This contradicts the minimality of $\mathcal{H}'_{[2,k-1]}$. Hence, we can conclude that no edge in $\mathcal{H}'_{[2,k-1]}$ can lie completely within $G \setminus Z$. □

Equivalently, theorem-3.6.2 can be stated in the following terms: with probability at least $2^{-O(t)}$, there exists a minimal hitting set for $\mathcal{F}^u_{[2,k-1]}$, whose edges lie within $\partial(Z)$, where Z can be computed in $2^{O(t)}\mathsf{poly}(n)$ time.

Equipped with theorem-3.6.2 and the discussion in the preceding section, we have the following theorem:

Theorem 3.6.3. *Let $G = (V, E)$ be a simple undirected graph on n vertices and m edges. Assume $\delta(G) \geq 3$. Let $\mathcal{H}_{[2,k-1]}$ be a hitting set for the paths in $\mathcal{F}_{[2,k-1]}$ of size at most t. Assume on the existence of an efficient $\mathcal{O}^{\epsilon_1}$, and let $f(k)$ and $g(k)$ denote the functions of k, representing the maximum pre-processing and response times among the oracles in $\mathcal{O}^{\epsilon_1}$. Then for any $0 < \epsilon_1, \epsilon_2 < 1$,*

1. *there exists a randomized FPT algorithm, which approximates the number of k - paths within a multiplicative error of $((1-\epsilon_1)(1-\epsilon_2), (1+\epsilon_1))$ with running time $O(\frac{1}{\epsilon_1\epsilon_2}2^{O(t)}f(k)g(k)\beta(G)\mathsf{poly}(n))$*

2. *If $\mathcal{H}_{[2,k-1]}$ is an optimal hitting set, then the same algorithm has running time $O(\frac{1}{\epsilon_1\epsilon_2}2^{O(\beta(G))}f(k)g(k)\beta(G)\mathsf{poly}(n))$*

3.7 Summary

To summarize, we explored additional parameterizations for the problem of k - path counting in this chapter. Our primary tool for the algorithms discussed in this chapter was the KNAPSACK problem. Further, we formulated randomized FPT algorithms for the $\#k$ - PATH problem, parameterized by the size of hitting set of the paths in $\mathcal{F}^u$ and the max-cut $\beta(G)$.

Chapter 4

FPT Algorithm for counting connected graphs on k vertices in graphs of bounded degree

4.1 Introduction

Let $G = (V, E)$ be a simple, undirected and d - regular graph on n vertices and H be a simple, undirected and connected graph on k vertices. We present a randomized FPT algorithm, which can be used for approximately counting the number of copies of H in G. Throughout, we will assume that G is connected. However, our algorithm can be easily extended to disconnected graphs. Further, our algorithm can be directly employed to obtain an approximation to the number of copies of H in graphs of bounded degree.

Given any $0 < \epsilon < 1$, we introduce the notion of an ϵ - H - *separating family*, which allows us to extend the technique of *Random Separation* [10, 11] to approximately count the copies of H in G. We begin our discussion with the specific case of H being a path on k vertices (or equivalently a k - path), and later examine how these ideas can be generalised to count any simple, connected subgraph on k vertices.

4.2 Notations

As stated before, let $G = (V, E)$ be a simple, undirected and d - regular graph on n vertices and H be a simple, undirected and connected graph on k vertices. Given a set X and a positive integer j, we shall denote the collection of all j - element subsets of X by $\binom{X}{j}$. Note that if $j > |X|$, then $\binom{X}{j} = \phi$. Suppose $E = [N]$, i.e. the edges of G are labelled with integers from $[N] = \{1, 2, \cdots, N\}$, where $N = \frac{dn}{2}$, provided $|V| = n$. Let $\mathcal{C}_k^P$ denote the family of all k - paths and C_k^H denote the family of all copies of H in G. Furthermore, we define a pair of distinct edges to be adjacent to each other if they are incident on the same vertex. Let $\Delta(G)$ denote the maximum degree of G.

4.3 ϵ - k - path - separating family

Given a 2-coloring $\chi : E \to \{R, B\}$ of the edges of G and a k - path $\mathcal{P} \in \mathcal{C}_k^P$, we say that χ *successfully separates* $\mathcal{P}$ if and only if :

1. every edge on $\mathcal{P}$ is colored red(R) and
2. every edge which is not on $\mathcal{P}$, but is adjacent to some edge on $\mathcal{P}$ is colored blue(B).

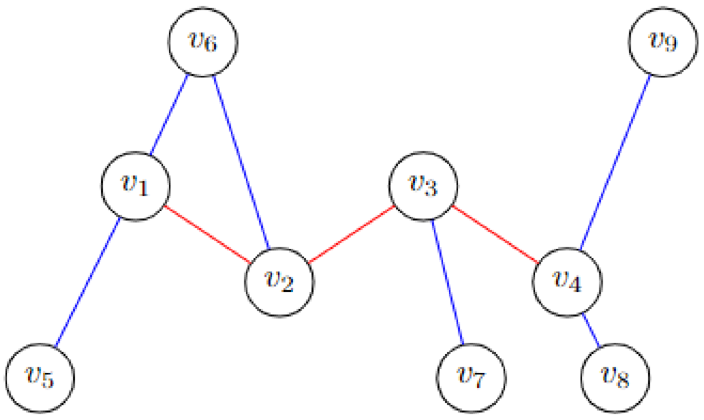

Figure 4.1: Random Separation

With the above definition at hand, we now proceed to define an ϵ - k - path - separating family.

Definition 4.3.1. Given a simple, undirected graph $G = (V, E)$, a family $\mathcal{F}$ of 2 - colorings of the edges of G is said to be an ϵ - k - *path - separating family* if there exists a constant $T > 0$ such that for every $\mathcal{P} \in \mathcal{C}_k^P$, if $N_{\mathcal{P}}$ denotes the number of colorings in $\mathcal{F}$ which successfully separate $\mathcal{P}$, then $(1 - \epsilon)T \leq N_{\mathcal{P}} \leq (1 + \epsilon)T$ for any $0 < \epsilon < 1$.

The following result provides a probabilistic construction of an ϵ - k - path - separating family for d - regular graphs with N edges. Note that construction presented below can also be used to construct an ϵ - k - path - separating family for graphs of bounded degree.

Theorem 4.3.1. *There exists a family $\mathcal{F}$ of 2-colorings of the edges of G of size $O(\frac{1}{\epsilon^2}k2^{dk}\log_e(N))$, such that with probability at least $1 - \frac{2}{k^k}$, for every k - path $\mathcal{P} \in \mathcal{C}_k^P$, the number of colorings in $\mathcal{F}$ that succeed in separating $\mathcal{P}$ lies within the interval $((1 - \epsilon)T, (1 + \epsilon)T)$ for any $0 < \epsilon < 1$, where $T = T(N, \epsilon, k, d)$ is a positive constant.*

Proof. Let $\mathcal{F} = \{\chi_1, \chi_2, \cdots, \chi_s\}$ be a family of 2 - colorings of the edges of G. For every $1 \leq i \leq s$, construct $\chi_i \in \mathcal{F}$ by coloring every edge of G independently with either Red or Blue color, chosen uniformly at random. Note that the colorings in $\mathcal{F}$ are independent of each other.

Let us fix a k - path $\mathcal{P} \in \mathcal{C}_k^P$ and let $\mathcal{E}_{i,\mathcal{P}}$ denote the event that χ_i succeeds in separating $\mathcal{P}$. For the moment, let us assume that $\mathbb{P}(\mathcal{E}_{i,\mathcal{P}}) = \rho$. Note that ρ will be independent of i.

For every $\chi_i \in \mathcal{F}$, define an indicator random variable $X_{i,\mathcal{P}}$ as follows:

$$X_{i,\mathcal{P}} = \begin{cases} 1 & \text{if } \chi_i \text{ successfully separates } \mathcal{P} \\ 0 & \text{otherwise} \end{cases} \tag{4.1}$$

Therefore, if the random variable $Y_{\mathcal{P}}$ denotes the total number of colorings in $\mathcal{F}$ that succeed in separating $\mathcal{P}$, then $Y_{\mathcal{P}} = \sum_{i=1}^{s} X_{i,\mathcal{P}}$. Hence $\mathbb{E}[Y_{\mathcal{P}}] = \mathbb{E}[\sum_{i=1}^{s} X_{i,\mathcal{P}}] = \sum_{i=1}^{s} \mathbb{E}[X_{i,\mathcal{P}}] = \sum_{i=1}^{s} \mathbb{P}(\mathcal{E}_{i,\mathcal{P}}) = s\rho$.

Since the $X_{i,\mathcal{P}}$'s are independent, using Chernoff bound, for any $0 < \epsilon < 1$, we have that $\mathbb{P}(|Y_{\mathcal{P}} - s\rho| \geq \epsilon s\rho) \leq 2e^{-\frac{\epsilon^2 s\rho}{3}}$.

Let $T = s\rho$. Using the union bound, the probability of there being a k - path $\mathcal{P}$ in $\mathcal{C}_k^P$, for which the number colorings in $\mathcal{F}$ which successfully separate $\mathcal{P}$ does not lie within the interval $((1-\epsilon)T, (1+\epsilon)T)$, is upper bounded by $C = 2|\mathcal{C}_k^P|e^{-\frac{\epsilon^2 s\rho}{3}} \leq 2\binom{N}{k}e^{-\frac{\epsilon^2 s\rho}{3}} \leq 2(\frac{Ne}{k})^k e^{-\frac{\epsilon^2 s\rho}{3}}$.

Now observe that $\rho \geq \frac{1}{2^{dk}}$. Consequently, for $s = \frac{3}{\epsilon^2}k2^{dk}\log_e(eN)$, it follows that $2(\frac{Ne}{k})^k e^{-\frac{\epsilon^2 s\rho}{3}} \leq 2(\frac{Ne}{k})^k \frac{1}{(Ne)^k} = \frac{2}{k^k}$. Therefore, with probability at least $1 - \frac{2}{k^k}$, for every $\mathcal{P} \in \mathcal{C}_k^P$, $|Y_{\mathcal{P}} - T| \leq \epsilon T$ for any choice of ϵ between 0 and 1. □

4.4 An algorithm for approximately counting k - paths

Suppose we are given an ϵ - k - path - separating family $\mathcal{F}$, a simple, undirected and connected d - regular graph $G = (V, E)$ and some $0 < \epsilon < 1$. For each $\chi_i \in \mathcal{F}$, let $G_{\chi_i,R}$ denote the subgraph of G, obtained by collecting those edges which have been assigned Red color under χ_i and let $\{G_{\chi_i,R}\}$ be the family of such subgraphs, obtained from all the colorings in $\mathcal{F}$. Then note that *every path $\mathcal{P} \in \mathcal{C}_k^P$ occurs as a connected component in $N_{\mathcal{P}}$ - many subgraphs of the family $\{G_{\chi_i,R}\}$, where* $(1-\epsilon)T \leq N_p \leq (1+\epsilon)T$. This observation leads to the following straight forward algorithm to approximately count the number of k - paths in G. (refer Algorithm 1).

Algorithm 3 Approximately counting k paths in d - regular graphs : COUNT($\mathcal{F}, T, \epsilon, G$)

```
count ← 0
for each $\chi \in \mathcal{F}$ do
    Color the edges of $G$ with $\chi$.
    $G^{\chi} \leftarrow G_{\chi,R}$
    for every connected component $\mathcal{C}$ of $G^{\chi}$ on $k$ vertices do
        if $\mathcal{C}$ is a $k$ - path then                ▷ Can be checked in $k^{O(d \log d)}$ time
            count ← count + 1
        end if
    end for
end for
return $\frac{\text{count}}{T}$
```

Theorem 4.4.1. *The count of the k - paths returned by COUNT($\mathcal{F}, T, \epsilon, G$) lies within the interval $((1-\epsilon)|\mathcal{C}_k^P|, (1+\epsilon)|\mathcal{C}_k^P|)$ with high probability.*

Proof. Let us fix a k - path $\mathcal{P} \in \mathcal{C}_k^P$. Let $\mathcal{F}_{\mathcal{P}}$ denote the subfamily of $\{G_{\chi_i,R}\}$, consisting of graphs in which $\mathcal{P}$ occurs as a connected component. These graphs correspond precisely to those colorings in $\mathcal{F}$, which succeed in separating $\mathcal{P}$. Further, let $\mathcal{E}_{\mathcal{F},\mathcal{P}}$ denote the event that at least $(1-\epsilon)T$ and at most $(1+\epsilon)T$ of the colorings in $\mathcal{F}$ successfully separate $\mathcal{P}$. Then clearly, $\mathbb{P}(\mathcal{E}_{\mathcal{F},\mathcal{P}}) \geq \mathbb{P}(\cap_{\mathcal{P} \in \mathcal{C}_k^P} \mathcal{E}_{\mathcal{F},\mathcal{P}}) \geq 1 - \frac{2}{k^k}$.

So, with high probability, $(1-\epsilon)T \leq |\mathcal{F}_{\mathcal{P}}| \leq (1+\epsilon)T$. Infact, based on the previous argument, for every $\mathcal{P} \in \mathcal{C}_k^P$, it follows that with high probability, $(1-\epsilon)|T \leq |\mathcal{F}_{\mathcal{P}}| \leq (1+\epsilon)T$. Therefore, $(1-\epsilon)T|\mathcal{C}_k^P| \leq \sum_{\mathcal{P}\in\mathcal{C}_k^P} |\mathcal{F}_{\mathcal{P}}| \leq (1+\epsilon)T|\mathcal{C}_k^P|$. Observe that the value assumed by the variable *count*, which is defined within the subroutine *COUNT($\mathcal{F}, T, \epsilon, G$)*, is exactly $\sum_{\mathcal{P}\in\mathcal{C}_k^P} |\mathcal{F}_{\mathcal{P}}|$ at the end of the outer for loop. Therefore, with high probability, $(1-\epsilon)|\mathcal{C}_k^P| \leq \frac{count}{T} \leq (1+\epsilon)|\mathcal{C}_k^P|$ □

Theorem 4.4.2. *The running time of COUNT($\mathcal{F}, T, \epsilon, G$) is* $O(\frac{1}{\epsilon^2}2^{dk}k^{O(d\log d)}\mathsf{poly}(n))$

Proof. Given any $\chi \in \mathcal{F}$, it takes $O(N)$ time to identify the red colored edges with respect to χ in G. This is precisely the time needed to construct the graph $G_{\chi,R}$. Since the number of connected components in $G_{\chi,R}$ cannot exceed $n = |V(G)|$, it follows that the time taken to determine all the k - paths among the components of $G_{\chi,R}$ is $O(k^{O(d\log d)}\mathsf{poly}(n))$ [19]. Since $\mathcal{F}$ houses $O(\frac{1}{\epsilon^2}k2^{dk}\log_e(N))$ many functions, it follows that the running time of the algorithm is $O(\frac{1}{\epsilon^2}2^{dk}k^{O(d\log d)}\mathsf{poly}(n))$. □

4.5 Generalizing to any connected graph H on k vertices.

We now shift our attention to the case when H is a simple, undirected and connected graph on k vertices, while G continues to remain a d - regular graph on n vertices and N edges.

Given a 2-coloring $\chi : E \to \{R, B\}$ of the edges of G and a subgraph $\mathcal{H} \in \mathcal{C}_k^H$, we say that χ *successfully separates* $\mathcal{H}$ if and only if :

1. every edge in $\mathcal{H}$ is colored red(R) and
2. every edge which is not in $\mathcal{H}$, but adjacent to some edge in $\mathcal{H}$ is colored blue(B).

As before, we now proceed to define the notion of an ϵ - H - separating family.

Definition 4.5.1. Given a simple, undirected graph $G = (V, E)$, a family of 2 - colorings of the edges of G is said to be an ϵ - H - *separating family* if there exists a constant $T > 0$ such that for every $\mathcal{H} \in \mathcal{C}_k^H$, if $N_{\mathcal{H}}$ denotes the number of colorings in $\mathcal{F}$ which successfully separate $\mathcal{H}$, then $(1-\epsilon)T \leq N_{\mathcal{H}} \leq (1+\epsilon)T$ for any $0 < \epsilon < 1$.

The existence of small ϵ - H - separating families for d - regular graphs can be shown using exactly the same construction, that was used in the proof of Theorem-1. For the sake of completeness, we present the proof again in the following theorem. As noted before, the procedure outlined in the following theorem can be directly employed for the construction of an ϵ - H - separating family for graphs of bounded degree.

Theorem 4.5.1. *There exists an ϵ - H - separating family $\mathcal{F}$ of size $O(\frac{1}{\epsilon^2}k2^{dk}\log_e(n))$ such that with probability at least $1 - \frac{2}{k^k}$, for every $\mathcal{H} \in \mathcal{C}_k^H$, the number of colorings in $\mathcal{F}$ that succeed in separating $\mathcal{H}$ lies within the interval $((1-\epsilon)T, (1+\epsilon)T)$ for any $0 < \epsilon < 1$, where $T = T(n, \epsilon, k, d)$ is a positive constant.*

Proof. Let $\mathcal{F} = \{\chi_1, \chi_2, \cdots, \chi_s\}$ be a family of 2 - colorings of the edges of G. For each $1 \leq i \leq s$, in order to construct $\chi_i \in \mathcal{F}$, we color every edge of G independently with either Red or Blue color, chosen uniformly at random. Note that the colorings in $\mathcal{F}$ are independent of each other.

Let us fix some $\mathcal{H} \in \mathcal{C}_k^H$ and let $\mathcal{E}_{i,\mathcal{H}}$ denote the event that χ_i succeeds in separating $\mathcal{H}$. Let $\mathbb{P}(\mathcal{E}_{i,\mathcal{H}}) = \rho$.

Note that ρ will be independent of i.

For any $S \subseteq E(G)$ and $\sigma \in \{R, B\}$, let $\mathcal{E}_{S,\sigma}$ denote the event that the edges in S are colored with σ. Further, assume that $\Gamma(\mathcal{H})$ consists of those edges in G, which are not in $\mathcal{H}$, but adjacent to some edge in $E(\mathcal{H})$. Then $\rho = \mathbb{P}(\mathcal{E}_{E(\mathcal{H}),R} \cap \mathcal{E}_{\Gamma(\mathcal{H}),B})$. Since the edges in G are colored independently and every edge is equally likely to receive either R or B color, it follows that $\rho = \mathbb{P}(\mathcal{E}_{E(\mathcal{H}),R})\mathbb{P}(\mathcal{E}_{\Gamma(\mathcal{H}),B}) = \frac{1}{2^{|E(\mathcal{H})|+|\Gamma(\mathcal{H})|}}$. Since G is d regular, $|E(\mathcal{H})| + |\Gamma(\mathcal{H})| \leq dk$. Therefore, $\rho \geq \frac{1}{2^{dk}}$.

Now, for every $\chi_i \in \mathcal{F}$, define an indicator random variable $X_{i,\mathcal{H}}$ as follows:

$$X_{i,\mathcal{H}} = \begin{cases} 1 & \text{if } \chi_i \text{ successfully separates } \mathcal{H} \\ 0 & \text{otherwise} \end{cases} \tag{4.2}$$

Therefore, if the random variable $Y_{\mathcal{H}}$ denotes the total number of colorings in $\mathcal{F}$ that succeed in separating $\mathcal{H}$, then $Y_{\mathcal{H}} = \sum_{i=1}^{s} X_{i,\mathcal{H}}$. Hence $\mathbb{E}[Y_{\mathcal{H}}] = \mathbb{E}[\sum_{i=1}^{s} X_{i,\mathcal{H}}] = \sum_{i=1}^{s} \mathbb{E}[X_{i,\mathcal{H}}] = \sum_{i=1}^{s} \mathbb{P}(\mathcal{E}_{i,\mathcal{H}}) = s\rho$.

Since the random variables $X_{i,\mathcal{H}}$ are independent, using Chernoff bound, for any $0 < \epsilon < 1$, we have $\mathbb{P}(|Y_{\mathcal{H}} - s\rho| \geq \epsilon s\rho) \leq 2e^{-\frac{\epsilon^2 s\rho}{3}}$.

Let $T = s\rho$. Using union bound, the probability that for some $\mathcal{H} \in \mathcal{C}_k^H$, the number of successful separating colorings in $\mathcal{F}$ does not lie within the interval $((1-\epsilon)T, (1+\epsilon)T)$ is upper bounded by $C = 2|\mathcal{C}_k^H|e^{-\frac{\epsilon^2 s\rho}{3}} \leq 2\binom{n}{k}e^{-\frac{\epsilon^2 s\rho}{3}} \leq 2(\frac{ne}{k})^k e^{-\frac{\epsilon^2 s\rho}{3}} \leq 2(\frac{ne}{k})^k e^{-\frac{\epsilon^2 s}{3\cdot 2^{dk}}}$.

Since $s = \frac{3}{\epsilon^2}k2^{dk}\log_e(en)$, it follows that $2(\frac{ne}{k})^k e^{-\frac{\epsilon^2 s\rho}{3}} \leq 2(\frac{ne}{k})^k \frac{1}{(ne)^k} = \frac{2}{k^k}$. Therefore, with probability at least $1 - \frac{2}{k^k}$, for every $\mathcal{H} \in \mathcal{C}_k^H$, $|Y_{\mathcal{H}} - T| \leq \epsilon T$ for any choice of ϵ between 0 and 1. □

4.6 Approximately Counting the number of copies of a connected subgraph H on k vertices

The idea behind the algorithm to approximately count the number of copies of H in G is exactly same as that of the algorithm presented in section - 4.

Suppose we are given an ϵ - H - separating family $\mathcal{F}$, a simple, undirected and connected d - regular graph $G = (V, E)$ and some $0 < \epsilon < 1$. For each $\chi_i \in \mathcal{F}$, let $G_{\chi_i,R}$ denote the subgraph of G formed by its red colored edges, after coloring G with χ_i. Let $\{G_{\chi_i,R}\}$ be the family of such subgraphs, obtained from the colorings in $\mathcal{F}$. Then note that *every $\mathcal{H} \in \mathcal{C}_k^H$ occurs as a connected component in $N_{\mathcal{H}}$ - many subgraphs of the family $\{G_{\chi_i,R}\}$, where $(1-\epsilon)T \leq N_{\mathcal{H}} \leq (1+\epsilon)T$.* This observation leads to the following straight forward algorithm to approximately count the copies of H in G (refer Algorithm 2). As before, the correctness of the algorithm is an immediate consequence of the definition of $\mathcal{F}$ itself. The following theorem proves the correctness of the subroutine COUNT-H($\mathcal{F}, T, \epsilon, G, H$).

Theorem 4.6.1. *The count of the number of copies of H returned by COUNT-H($\mathcal{F}, T, \epsilon, G$) lies within the interval $((1-\epsilon)|\mathcal{C}_k^H|, (1+\epsilon)|\mathcal{C}_k^H|)$ with high probability.*

Proof. Let us fix some $\mathcal{H} \in \mathcal{C}_k^H$. Let $\mathcal{F}_{\mathcal{H}}$ denote the subfamily of $\{G_{\chi_i,R}\}$, that contains the graphs in which $\mathcal{H}$ occurs as a connected component. These correspond precisely to those colorings in $\mathcal{F}$, which succeed in separating $\mathcal{H}$. Further, let $\mathcal{E}_{\mathcal{F},\mathcal{H}}$ denote the event that at least $(1-\epsilon)T$ and at most $(1+\epsilon)T$ of the colorings in $\mathcal{F}$ successfully separate $\mathcal{H}$. Then clearly, $\mathbb{P}(\mathcal{E}_{\mathcal{F},\mathcal{H}}) \geq \mathbb{P}(\cap_{\mathcal{H}\in\mathcal{C}_k^H}\mathcal{E}_{\mathcal{F},\mathcal{H}}) \geq 1 - \frac{2}{k^k}$.

Algorithm 4 Approximately counting number of copies of H in d - regular graphs : COUNT-H($\mathcal{F}, T, \epsilon, G, H$)

```
count $\leftarrow 0$
for each $\chi \in \mathcal{F}$ do
    Color the edges of $G$ with $\chi$.
    $G^{\chi} \leftarrow G_{\chi,R}$
    for every connected component $\mathcal{C}$ of $G^{\chi}$ on $k$ vertices do
        if $\mathcal{C}$ is isomorphic to $H$ then          ▷ Can be checked in $k^{O(d \log d)}$ time
            count $\leftarrow$ count + 1
        end if
    end for
end for
return $\frac{\text{count}}{T}$
```

So, with high probability, $(1-\epsilon)|T \leq |\mathcal{F}_{\mathcal{H}}| \leq (1+\epsilon)T$. Infact, based on the previous argument, it is evident that for every $\mathcal{H} \in \mathcal{C}_k^H$, with high probability, $(1-\epsilon)T \leq |\mathcal{F}_{\mathcal{H}}| \leq (1+\epsilon)T$. Therefore, $(1-\epsilon)T|\mathcal{C}_k^H| \leq \sum_{\mathcal{H} \in \mathcal{C}_k^H} |\mathcal{F}_{\mathcal{H}}| \leq (1+\epsilon)T|\mathcal{C}_k^H|$. Observe that the value assumed by the variable *count*, which is defined within the subroutine *COUNT-H($\mathcal{F}, T, \epsilon, G$)*, is exactly $\sum_{\mathcal{H} \in \mathcal{C}_k^H} |\mathcal{F}_{\mathcal{H}}|$ at the end of the outer for loop. Therefore, with high probability, $(1-\epsilon)|\mathcal{C}_k^H| \leq \frac{count}{T} \leq (1+\epsilon)|\mathcal{C}_k^H|$. □

Note that the running time of the algorithm is $O(|\mathcal{F}|k^{O(d \log d)}\mathsf{poly}(n)) = O(\frac{1}{\epsilon^2}2^{dk}k^{O(d \log d)}\mathsf{poly}(n))$.

Theorem 4.6.2. *The running time of COUNT-H($\mathcal{F}, T, \epsilon, G$) is $O(\frac{1}{\epsilon^2}2^{dk}k^{O(d \log d)}\mathsf{poly}(n))$*

Proof. Given any $\chi \in \mathcal{F}$, it takes $O(N)$ time to identify the red colored edges under χ in G. This is precisely the time needed to construct the graph $G_{\chi,R}$. Since the number of connected components in $G_{\chi,R}$ cannot exceed $n = |V(G)|$, it follows that the time taken to determine all the components of $G_{\chi,R}$, which are isomorphic to H is $O(k^{O(d \log d)}\mathsf{poly}(n))$ [19]. Since $\mathcal{F}$ contains $O(\frac{1}{\epsilon^2}k2^{dk}\log_e(n))$ many functions, it follows that the running time of the algorithm is $O(\frac{1}{\epsilon^2}2^{dk}k^{O(d \log d)}\mathsf{poly}(n))$. □

It is not very difficult to note that *all the ideas presented in the preceding sections are indeed extensible to the case of graphs with bounded degree.*

4.7 A connection with Parsimonious Families in regular graphs

As before, assume $G = (V, E)$ to be a connected, undirected and d - regular graph on n vertices and N edges, where $N = \frac{nd}{2}$. Similarly, assume H to be an undirected and connected graph on k vertices and let $\mathcal{C}_k^H$ be the family of subgraphs of G which are isomorphic to H. Given any $\mathcal{H} \in \mathcal{C}_k^H$, let $p = |E(\mathcal{H})|$ and $q = |\Gamma(\mathcal{H})|$, where $\Gamma(\mathcal{H})$ consists of all the edges of G, which are not in $\mathcal{H}$, but adjacent to some edge in $E(\mathcal{H})$. Note that while $\Gamma(\mathcal{H})$ may differ across the subgraphs in $\mathcal{C}_k^H$, $|\Gamma(\mathcal{H})|$ will be the same for every $\mathcal{H} \in \mathcal{C}_k^H$, since G is a regular graph.

Suppose we could deterministically construct a family $\mathcal{F}$ of 2-colorings of the edges of G, which has the property that for any $P \in \binom{E}{p}$ and $Q \in \binom{E}{q}$ such that $P \cap Q = \phi$, $(1-\epsilon)T \leq |\mathcal{F}[P : Q]| \leq (1+\epsilon)T$ for some positive constant T, where $|\mathcal{F}[P : Q]|$ denotes the number of colorings in $\mathcal{F}$ which map the edges in P to Red color and edges in Q to Blue color, then we would be done. In particular, if we set $p = |E(H)|$ and $q = |\Gamma(H)|$, then since $|E(\mathcal{H})| \cap |\Gamma(H)| = \phi$, each $\mathcal{H} \in \mathcal{C}_k^H$ will be separated by $N_{\mathcal{H}}$ -

many colorings within $\mathcal{F}$, where $(1-\epsilon)T \leq N_{\mathcal{H}} \leq (1+\epsilon)T$ for some choice of $T > 0$. We now define the notion of parsimonious families, introduced in [17]

Definition 4.7.1. [17] Let $n, p, q \in \mathbb{N}$ and $0 < \delta < 1$. Denote $K = p+q$. A family $\mathcal{F}$ of sets over a universe $\mathcal{U}$ of size n is a *δ-parsimonious (n, p, q) - universal family* if there exists $T = T(n, p, q, \delta) > 0$ such that for each pair of disjoint sets $A \in \binom{\mathcal{U}}{p}$ and $B \in \binom{\mathcal{U}}{q}$, it holds that $(1-\delta)T \leq |\mathcal{F}[A : B]| \leq (1+\delta)T$, where $|\mathcal{F}[A : B]|$ denotes the number of sets in $\mathcal{F}$ which contain P and are disjoint from Q.

Let $p = |E(H)|, q = |\Gamma(H)|$, $K = \frac{dk}{2}$ and $\mathcal{U} = E(G)$. Further, every subset A of $\mathcal{U}$ in $\mathcal{F}$ can be viewed as a function $f_A : \mathcal{U} \to \{0, 1\}$ where for every $u \in \mathcal{U}$

$$f_A(u) = \begin{cases} 1 & \text{if } u \in A \\ 0 & \text{otherwise} \end{cases} \tag{4.3}$$

Equivalently, we can interpret the function f_A as assigning either Red or Blue colors to the edges of G, where 0 corresponds to Red color and 1 corresponds to Blue color. Then from the definition of an ϵ - parsimonious (N, p, q) - family, it follows that the number of colorings in $\mathcal{F}$ which successfully separate every $\mathcal{H} \in \mathcal{C}_k^H$ lies between $(1-\epsilon)T$ and $(1+\epsilon)T$ for some $T > 0$. This is precisely the notion of an $\epsilon - H$ - separating family. Therefore, we can conclude that *for regular graphs, ϵ - H - separating families are equivalent to ϵ - parsimonious families.* Since we know that deterministic constructions for ϵ - parsimonious families do exist [11], we can conclude that for regular graphs, there exists a deterministic algorithm which can approximately count the number of copies of H in G.

Algorithm 5 Approximately counting number of copies of H in d - regular graphs **deterministically** : DET-COUNT-H(ϵ, G, H)

```
count ← 0
$\mathcal{F} \leftarrow \epsilon -$ parsimonious $(N, |E(H)|, |\Gamma(H)|)$ - family      ▷ Can be constructed deterministically
for $A \in \mathcal{F}$ do
    $V(G^A) \leftarrow V(G)$
    $E(G^A) \leftarrow \phi$
    for $a \in A$ do
        $E(G^A) \leftarrow E(G^A) \cup \{a\}$      ▷ Note that the elements of the universe are edges of G
    end for
    for every connected component $\mathcal{C}$ of $G^A$ on $k$ vertices do
        if $\mathcal{C}$ is isomorphic to $H$ then      ▷ Can be checked in $k^{O(d \log d)}$ time
            count ← count + 1
        end if
    end for
end for

return $\frac{\text{count}}{T}$
```

4.8 Summary

In this chapter, we studied the problem of counting k - connected subgraphs in graphs of bounded degree. In particular, we showed that there exists an FPT-AS for counting an connected subgraph on k vertices in graphs of bounded degree. Further, we also showed an explicit connection between $\epsilon - H$ - separating families and ϵ - parsimonious families for regular graphs, which lead to a deterministic algorithm for the above problem. Note that all the algorithms proposed above are essentially extenions of the Random Separation technique, and utilise polynomial space.

Chapter 5

Conclusion

In this thesis we primarily focused on coming up with additional parameterizations for the $\#k$-PATH problem. In particular, we designed separate FPT algorithms for approximate counting of k - paths in the input graph $G = (V, E)$. The running time of the former algorithm is dependent upon k and the size of the hitting set for all k - paths in G. The latter algorithm runs in time, dependent on k and the size of the max-cut in G, conditioned on the existence of efficient, approximate sensitivity oracles for $\#k$-PATH.

We also designed an FPT algorithm, which can be employed for approximately counting the number of k - paths in graphs of bounded degree. This algorithm is essentially an extension of the Random Separation technique. We also described a deterministic FPT-AS for $\#k$-PATH problem in regular graphs, by establishing the equivalence between parsimonious families and separating families.

Bibliography

[1] Alman, J., and Hirsch, D. Parameterized sensitivity oracles and dynamic algorithms using exterior algebras. *CoRR abs/2204.10819* (2022).

[2] Alon, N., Dao, P., Hajirasouliha, I., Hormozdiari, F., and Sahinalp, S. C. Biomolecular network motif counting and discovery by color coding. In *Proceedings 16th International Conference on Intelligent Systems for Molecular Biology (ISMB), Toronto, Canada, July 19-23, 2008* (2008), pp. 241–249.

[3] Alon, N., and Gutner, S. Balanced families of perfect hash functions and their applications. *ACM Trans. Algorithms 6*, 3 (2010), 54:1–54:12.

[4] Alon, N., Lokshtanov, D., and Saurabh, S. Fast FAST. In *Automata, Languages and Programming, 36th International Colloquium, ICALP 2009, Rhodes, Greece, July 5-12, 2009, Proceedings, Part I* (2009), S. Albers, A. Marchetti-Spaccamela, Y. Matias, S. E. Nikoletseas, and W. Thomas, Eds., vol. 5555 of *Lecture Notes in Computer Science*, Springer, pp. 49–58.

[5] Alon, N., Yuster, R., and Zwick, U. Color coding. In *Encyclopedia of Algorithms - 2008 Edition*, M. Kao, Ed. Springer, 2008.

[6] Arvind, V., and Raman, V. Approximation algorithms for some parameterized counting problems. In *Algorithms and Computation, 13th International Symposium, ISAAC 2002 Vancouver, BC, Canada, November 21-23, 2002, Proceedings* (2002), P. Bose and P. Morin, Eds., vol. 2518 of *Lecture Notes in Computer Science*, Springer, pp. 453–464.

[7] Babai, L., Frankl, P., and of Chicago. Department of Computer Science, U. *Linear Algebra Methods in Combinatorics. Part 1.* University of Chicago, Department of Computer Science, 1988.

[8] Björklund, A., Kamat, V., Kowalik, L., and Zehavi, M. Spotting trees with few leaves. *SIAM J. Discret. Math. 31*, 2 (2017), 687–713.

[9] Brand, C., Dell, H., and Husfeldt, T. Extensor-coding. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2018, Los Angeles, CA, USA, June 25-29, 2018* (2018), I. Diakonikolas, D. Kempe, and M. Henzinger, Eds., ACM, pp. 151–164.

[10] Cai, L., Chan, S. M., and Chan, S. O. Random separation: A new method for solving fixed-cardinality optimization problems. In *Parameterized and Exact Computation, Second International Workshop, IWPEC 2006, Zürich, Switzerland, September 13-15, 2006, Proceedings* (2006), H. L. Bodlaender and M. A. Langston, Eds., vol. 4169 of *Lecture Notes in Computer Science*, Springer, pp. 239–250.

[11] Cygan, M., Fomin, F. V., Kowalik, L., Lokshtanov, D., Marx, D., Pilipczuk, M., Pilipczuk, M., and Saurabh, S. *Parameterized Algorithms.* Springer, 2015.

[12] Flum, J., and Grohe, M. The parameterized complexity of counting problems. In *43rd Symposium on Foundations of Computer Science (FOCS 2002), 16-19 November 2002, Vancouver, BC, Canada, Proceedings* (2002), IEEE Computer Society, p. 538.

[13] Fomin, F. V., Lokshtanov, D., Panolan, F., and Saurabh, S. Efficient computation of representative families with applications in parameterized and exact algorithms. *J. ACM 63*, 4 (2016), 29:1–29:60.

[14] Kneis, J., Mölle, D., Richter, S., and Rossmanith, P. Divide-and-color. In *Graph-Theoretic Concepts in Computer Science, 32nd International Workshop, WG 2006, Bergen, Norway, June 22-24, 2006, Revised Papers* (2006), F. V. Fomin, Ed., vol. 4271 of *Lecture Notes in Computer Science*, Springer, pp. 58–67.

[15] Koutis, I., and Williams, R. Algebraic fingerprints for faster algorithms. *Commun. ACM 59*, 1 (2016), 98–105.

[16] Koutis, I., and Williams, R. LIMITS and applications of group algebras for parameterized problems. *ACM Trans. Algorithms 12*, 3 (2016), 31:1–31:18.

[17] Lokshtanov, D., Björklund, A., Saurabh, S., and Zehavi, M. Approximate counting of k-paths: Simpler, deterministic, and in polynomial space. *ACM Trans. Algorithms 17*, 3 (2021), 26:1–26:44.

[18] Lokshtanov, D., Saurabh, S., and Zehavi, M. Efficient computation of representative weight functions with applications to parameterized counting (extended version). In *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms, SODA 2021, Virtual Conference, January 10 - 13, 2021* (2021), D. Marx, Ed., SIAM, pp. 179–198.

[19] Luks, E. M. Isomorphism of graphs of bounded valence can be tested in polynomial time. *J. Comput. Syst. Sci. 25*, 1 (1982), 42–65.

[20] Marx, D. A parameterized view on matroid optimization problems. In *Algorithms and Complexity in Durham 2006 - Proceedings of the Second ACiD Workshop, 18-20 September 2006, Durham, UK* (2006), H. Broersma, S. S. Dantchev, M. Johnson, and S. Szeider, Eds., vol. 7 of *Texts in Algorithmics*, King's College, London, p. 158.

[21] Marx, D. Randomized techniques for parameterized algorithms. In *IPEC 2012* (Ljubljana, Slovenia, 2012).

[22] Valiant, L. G. The complexity of computing the permanent. *Theor. Comput. Sci. 8* (1979), 189–201.

[23] Vazirani, V. V. *Approximation Algorithms*, 1 ed. Springer Berlin, Heidelberg.

[24] Williams, R. Finding paths of length k in $\mathrm{o}^{*}(2^{\mathrm{k}})$ time. *Inf. Process. Lett. 109*, 6 (2009), 315–318.