Utrecht University

Thesis Project

Utrecht University

Department of Mathematics

Uniform Sampling of Bi-colored Random Graphs

Supervisor:
Dr. Ivan Kryven

Author:
Andi Lin

Second Reader:
Prof. Rob Bisseling

Daily supervisor:
Dr. Rik Versendaal

February 11, 2022

Abstract

We propose a near-linear complexity algorithm for uniform sampling of simple random graphs with bi-colored edges imposed by two given colored degree sequences. This problem is also known as three-colour discrete tomography and is closely related to the sequence packing problem, both of which are known to be NP hard in the general setting. That being said, our algorithm provides a fast means of asymptotically uniform sampling for large graphs. The algorithm is applicable when the maximum degree, d_{max} is of order $O(m^{\frac{1}{4}-\tau})$ with m being the total number of red and blue edges, and any small enough $\tau > 0$.

Keywords: Random Graphs, Simple Graphs, Colored edges, Randomised Approximation Algorithms

Symbols and Notations used

$[n]$	All numbers 1 to n
$\delta,\ \epsilon$	Arbitrary small real numbers
$i \sim\ j$	The edge connecting vertex i to j
$i \sim_B\ j$	The blue edge connecting vertex i to j
$i \sim_R\ j$	The red edge connecting vertex i to j
$\bar{d^B},\ \bar{d^R}$	Blue degree sequence and Red degree sequence
$\hat{d^B},\ \hat{d^R}$	Residual blue and red degree sequence
$m_B,\ m_R$	Total number of blue edges, red edges respectively
$k_B,\ k_R$	Number of blue edges, red edges formed at stage k
$O(x),\ \Omega(x),\ o(x)$	Different order of growth of functions, defined on page 4
$\mathbb{E}(X)$	Expectation of a (random) variable
$\mathbb{P}(X)$	Probability of an event X
q_k	Probability of removing an edge (or an edge not present at stage k)
p_k	Probability of keeping an edge (or an edge present at stage k)
$M(G)$	Set of matchings that leads to G
$N \in S(M)$	Set of all orderings, N, of a fixed matching M
G_{P_K}	Expected state of the algorithm at stage k
$\mathbb{E}_{p_k}[X]$	Expectation of X within the G_{p_k} model
H_m	m-th Harmonic number
$E_k^B,\ E_k^R$	Set of all suitable matchings at k-th stage of an ordering
$\bigsqcup$	Disjoint union of sets

Contents

Chapter I

Introduction and Preliminaries

1 Introduction

The theory of random graphs lies at the intersection between graph theory and probability theory. Ever since its first definition by Paul Erdős and Alfréd Rényi [15], it has brought together different fields of research, such as discrete mathematics, probability theory, theoretical computer science, and statistical physics. Since then, it have been extensively studied since their introduction, and became one of the central themes of contemporary mathematics, partly because they are closely related to various random discrete structures such as random surfaces, random maps, random matrices, random satisfiability problems[12]; and these mathematical concepts have found very natural applications in different real life problems. An example would be analysis of human interactions and communication where one of the oldest and best-known examples is the 'six degree of separation' phenomenon describe any two people can be connected by a chain of acquaintances on average six persons long.

A graph is simple if it has no self loops and multi-edges and a degree sequence is graphical if there is a graph that satisfies it. Erdős and Gallai provided a necessary and sufficient conditions for a finite sequence of natural numbers to be the degree sequence of a simple graph [10]. The study of generating graphs satisfying a degree is an interesting, especially in the areas of hypothesis testing. Hence formally, one could ask the following related problem:

1. Given the sample space of all graphs that satisfies a given degree sequence, can we ensure we sample only the simple graphs?

2. Can we perform the sampling uniformly ?

One possible way to solve question one is to use rejection sampling with respect to the configuration model. In essence, this naive method constructs a graph that satisfies the degree sequence, however we reject the output whenever there is an multi-edge or self loops. According to a result by [4], this method is exponential in (the square of) the average degree, which is not ideal. So what we learnt is that enforcing simplicity in sampling is not a trivial problem and there have been numerous research undertaken to resolve this. Two such methods are the Markov Chain Monte Carlo (MCMC) algorithms that approximate the desired sample by taking the last element of an ergodic Markov chain [16][6][11][17][1] and Fast sequential construction algorithms that build a graph by placing m edges one-by-one, starting with an empty graph[13][3][2]. Both methods greatly improve in terms of complexity as both have polynomial (in fact linear) time complexity with respect to the degree sequence. However in this thesis we are mainly interested in the latter. Steger and Wormald's algorithm [18] samples from regular graphs and the algorithm by Bayati, Kim and Vu[13] generalizes it further to arbitrary degree sequences while still maintaining linear complexity with respect to the number of edges. However these algorithms only solve the first problem posted. The one downside is that the proposed algorithm only achieve uniform sampling asymtoptically, meaning the bias and the error of the sampling as n (or number of edges) tend to positive infinity.
In this thesis we will focus on generalizing the algorithm by Bayati, Kim and Vu[13] by generalizing the algorithm to receive two input sequences-a blue and red sequence. Specifically we will sample

a simple graph with colored edges satisfying the input sequences. At the end we will show that the run-time of the algorithm is still linear in n (or the number of edges) while still achieving uniformity asymptotically.
Perhaps as a starting motivation for this thesis, we can revisit the context of human interactions and connectivity. We know that any two people can be connected by a chain of acquaintances on average six persons long. But one can also decide to impose conditions like social status, financial standings, gender, races etc and investigate how it affects connectivity between individuals.
The thesis will be structured into six core chapters, beginning with the preliminaries and notation. From there we will formally introduce the notion of configuration model and the modification proposed by Bayati, Kim and Vu [13] to generate simple graph. In chapter two, we will propose a more generalize version of [13] and explore the relevant heuristics. Chapter three, four and five will be a three part analysis with the focus on investigating the asymptotic behaviour of our algorithm. We then end off with a conclusion to summarize our findings and possible future works. Last we wish to acknowledge the admirable work of [7] who not only worked on the directed version of the algorithm by Bayati, Kim and Vu [13], but also serves as the basis of inspiration for this thesis.

2 Notations

We will begin by introducing some important notations and preliminaries that will be used throughout the thesis. As we proceed deeper into the chapters, new notations will be introduced as new concepts and definition arise.

We start by defining a graph. A graph G is an ordered pair (V, E), where V is a set of n vertices and $E \subset V \times V$ is the set of edges connecting the vertices. We use $v_i \sim v_j$ to denote the ordered paired (v_i, v_j) which represents vertex i being connected to vertex j.
A graph G is *undirected* if for every $v_i, v_j \in V, v_i \sim v_j \iff v_j \sim v_i$. Since there are n vertices in a graph, for simplicity, we will denote each $v_i \in V$ by simply $i \in [n]$, meaning the $i-th$ vertex of G. In the future chapters, we will frequently be performing summation indexed by edges. Example as follow:

$$\sum_{i\sim j} 1 = m = \frac{1}{2}\sum_{i\in[n]} d_i$$

This example illustrate a simple counting of total number of edges in a graph G, which we will denote by m, from here on out. Finally we say a graph is *simple* if there are no self loops and for every $i \neq j$, there is at most one edge connecting i to j.

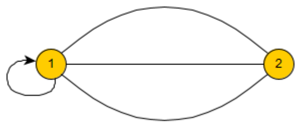


Figure I.1: Example of self-loops and multiedge in an undirected graph

In this thesis, unless otherwise stated, we will assume all graphs that we fixed to be *undirected* and **simple**. Every $i \in [n]$ has an associated degree d_i, which is the number on edges connecting to it. Hence a degree sequence of length n is of the form $(d_1, \ldots, d_j, \ldots d_n)$, where each d_i is the degree of i. Throughout the thesis, we will denote this by $\bar{d}$. Naturally when given $\bar{d}$, one might ask if there is a graph G such that each vertex i has precisely d_i many adjacent vertices.If there is, we then say G *satisfies* the degree sequence $\bar{d}$ and that the degree sequence is *graphical*.

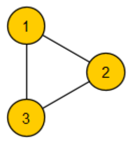


Figure I.2: Example of a graph satisfying the degree sequence (2, 2, 2)

In our thesis, we are interested in two degree sequences differentiated by the colors blue and red . Specifically we have the degree sequences $\bar{d^B} = (d_1^B, \dots, d_n^B)$ and $\bar{d^R} = (d_1^R, \dots, d_n^R)$ where each $i \in [n]$ has d_i^B many adjacent vertices connected via a blue edge and d_i^R many red edges connected through a red edge. This meant that we have the following:

$$d_i = d_i^B + d_i^R$$
$$\bar{d} = \bar{d^B} + \bar{d^R}$$

With the introduction of colored edges, we would then have the need to indicate if i and j is connected via a blue edge or red a red edge. We will use $i \sim_B j$ and $i \sim_R j$ to denote vertex i being connected to vertex j via a blue or red edge respectively. Like before, in the upcoming chapters, we will be frequently performing summation indexed by blue and red edges.

$$\sum_{i \sim_B j} 1 = m_B = \frac{1}{2} \sum_{i \in [n]} d_i^B$$
$$\sum_{i \sim_R j} 1 = m_R = \frac{1}{2} \sum_{i \in [n]} d_i^R$$

The above example illustrate a simple counting of blue and red edges using the notation we introduced. Since the number of blue edges is m_B and the number of red edges is m_R, we get that

$$m = m_B + m_R$$

We say that a two colored degree sequence is satisfiable if there is a there is a graph G such that each vertex i has precisely d_i^B adjacent vertices via blue edges and d_i^R adjacent vertices via red edges. We will sometimes refer to such a G as a bi-colored edged graph. Finally we will assume throughout the thesis the following

- m_B and m_R are of the same order as m. Hence, $m_B, m_R = O(m)$
- The maximal blue and red degree is of the same order as $d_{max} = Max\ (d_i)_{i \in [n]}$

In future chapters, we will be performing detailed analysis on some on the algebraic expressions. In particular, we will be comparing the growth rate between functions f and g. Thus we will be introducing the following definition.

Definition. *We say $f(x) = O(g(x))$ if and only if there exist a constants and $C > 0$ and a real number x_0 such that*

$$|f(x)| \leq C|g(x)|$$

for all $x \geq x_0$.

Furthermore, We say $f(x) = o(g(x))$ if and only if for every positive $\epsilon > 0$ there exists a real number x_0 such that

$$|f(x)| \leq \epsilon|g(x)|$$

for all $x \geq x_0$.

Intuitively one may view the former definition as saying f 'does not grow faster than' g and the latter definition as f 'grows **strictly** slower than' g.

Remark 1. *One can also say f 'grows at least as fast as' g with the definition $f = \Omega(g(x))$.*

3 Configuration Models

The first and perhaps most important definition that we will introduce is the notion of *configuration model.* Intuitively, a configuration model is nothing more than a (random) graph that obtained through a *random matching of half edges.* We will go into details on the specifics in a while, but the main caveat is that the process accepts input of a degree sequence $\bar{d}$ which is fixed beforehand, and its main goal is to construct a G that satisfies such a $\bar{d}$. Hence, it is not hard to see why configuration model lays the foundation for all future algorithm and analysis in this thesis.

We will now begin explaining what exactly this random matching process entails. First let us begin with just one degree sequence $\bar{d}$, and without loss of generality assume that each $d_i > 0$. The reason being if $d_i = 0$, then vertex i is isolated and hence not relevant in the construction. Next to each vertex i, we associate d_i many *half-edges*, denoted by w_i to it. A half edge of i only becomes an (*full*) edge when paired with another half-edge of j, connecting i to j.

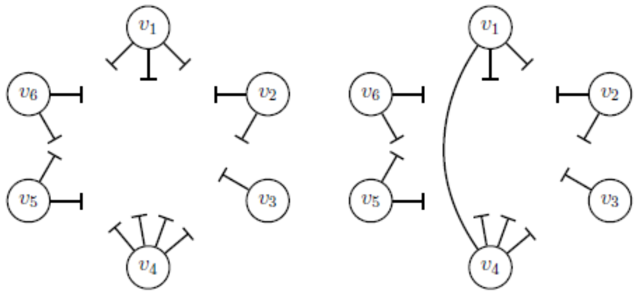


Figure I.3: Example of half edges and matchings

The above example concerning the degree sequence $(3, 2, 1, 4, 2, 2)$ shows the assigned half-edges to the vertices and we see on the right picture that we paired one half edge of vertex 1 to another half edge of vertex 4, forming an edge connecting these two vertices. Now let l_n be the total number of half-edges assigned, the randomness of the process comes into play when we consider pairing the half-edges in a uniform way. Meaning that by fixing one half edge w_i of vertex i, with probability $\frac{1}{l_n - 2k}$ we paired it with another arbitrary half-edge of our choice at the k-th iteration of the process.

We repeat this random matching process for $\frac{l_n}{2}$ iterations, where at each stage after pairing, the half-edges are removed from the list of half-edges that need to be paired. The resulting graph G is called the **configuration model with degree sequence** $\bar{d}$, abbreviated as $CM_n(\bar{d})$.

(a) Bayati's algorithm

The matching process described by the configuration model this does not exactly provide the answer to our problem if we insist on the output being a simple graph as the process we describe will likely produce a *multigraph* - a graph with self loops and multi-edges. In fact, a result by [14] shows that the probability of the process in configuration model producing a simple graph is in essence

$$\mathbb{P}(CM_n(\bar{d}) = simple) \propto e^{-\lambda_n^S - \lambda_n^M}$$

Here λ_n^S, λ_n^M are both proportional to the degrees, d_i's, of the degree sequence. Thus what this means is that the bigger the values of d_i's the more unlikely we are able to produce a simple graph from the process.

However, the solution to this conundrum turns out to be quite elegant, as we need only to insist that at each iteration, we only focus on pairing vertices $i \neq j$ that have no edge connecting them. The only caveat is that we can can no longer do the pairing uniformly. This idea was explored and proposed in [13]'s thesis, and the following is the proposed algorithm.

Algorithm: Graph sampling

```
Input  : $n$-length degree sequence $\bar{d}$
Output : $G$ satisfying the input degree sequence or output failure
  1. Let $E$ be the set of edges formed. Let $\hat{d} = (\hat{d}_1, \dots, \hat{d}_n)$ be a vector of $n$ integers.
     Initialize $E$ to empty set and $\hat{d} = \bar{d}$.
  2. Choose two vertices $v_i, v_j \in V$ with probability proportional to $\hat{d}_i\hat{d}_j(1 - \frac{d_i d_j}{4m})$
     among all pairs $v_i,\ v_j$ with $i \neq j$ and $\{v_i, v_j\} \notin E$.
  3. Add $\{v_i, v_j\}$ to $E$ and decrease $\hat{d}_i$ and $\hat{d}_j$ by 1.
  4. Repeat step 2 until no more edges can be added
  5. If $|E| < m$, output **FAILURE**. Otherwise output $G = (V, E)$.
```

Perhaps it is worthwhile to explore some of the heuristics in this algorithm, in particular step 2 of the algorithm. The first thing to note is that the algorithm it no longer allows for arbitrary matching between half edges, which is reflected by the checking for $\{v_i, v_j\} \notin E$. However if we choose to maintain uniform matching of half-edges and proceed to step 3 at this point, this will result in a bias during the sampling. Consider the following:

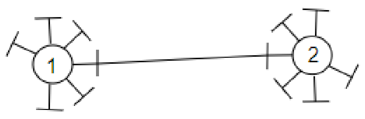


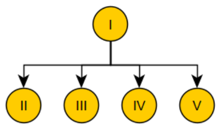
Figure I.4: Vertex 1, 2 with high degree

The above diagram illustrates vertex $1, 2$ with very high degree (compared with other vertices). Now if we were to match half edges uniformly, then very likely we will end up matching a half edge w_1 of vertex 1 with another half edge w_2 of vertex 2 as shown in the diagram. This will result in a bias towards graphs containing the edge (v_1, v_2). It is precisely because of this reason, that we have to offset this bias in the latter step of step 2, which is reflected by the term $\hat{d}_i\hat{d}_j(1 - \frac{d_i d_j}{4m})$. We will explain in detail how such a bias offset is being derived in chapter III.

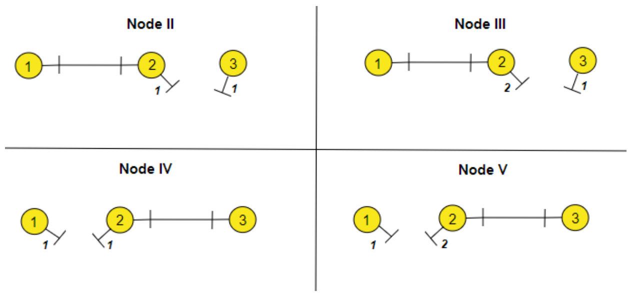
It is shown in the paper by [13] that when given a graphical degree sequence $\bar{d}$, the probability of failure due to mismatching of half-edges tends to 0 asymptotically as the number of edges, m tends to ∞. This means that if a given $\bar{d}$ (with sufficiently large degree sum) produces a **FAILURE** output, then the algorithm by [13] provides statistical evidence that such $\bar{d}$ is in fact not graphical. Now we shall move on to the next important notion in our thesis - *execution tree.*

(b) Execution Tree

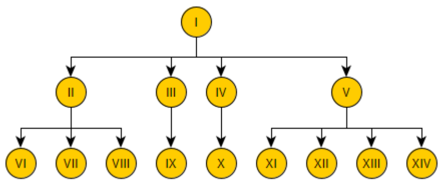
An *Execution tree* is an important tool to help us analyze and understand the (asymptotic) analysis of algorithms in the upcoming section, and it is a tool to study the algorithm proposed in [13]. In essence, a *execution tree*, T,is a data-structure that keeps track of all possible states of the partial graph by stage k. As an example, let us consider the execution tree when we run the algorithm with input sequence $(1, 2, 1)$



Now node I would be the initial state with no pairings of half edges, and nodes II, III, IV, V would then store the following possible states of the partial graph after one matching of half-edges has been performed; do note that the half-edges are labelled.



So by viewing all states of the algorithm in terms of an execution tree, it makes it much easier to analyze the algorithm at each stage k as every path from the root to any node in stage k represents one way to form a subgraph. In fact if we were to go back to the topic of biases caused by uniform matching of half-edges, this fact can be reflected in the execution tree, T. Consider the following arbitrary execution tree T.



Each path from the root to the leave will represent one graph G, and the edge connecting the parent to its child is weighted by p_k defined to be probability of transitioning from the parent state to the children state.With this, the probability of obtaining such G is simply by multiplying (and possibly summing) the relevant p_k's from the root to the leave(s) corresponding to G. Suppose we perform uniform matching of half edges, then one can verify there is bias towards certain graph by simply computing the probability of reaching certain leaves. Conversely, to obtain uniform sampling, the problem becomes ensuring that every path to the leave(s) corresponding to G has equal total weight. This idea will actually be used in the upcoming section when we are deriving the bias and the offsets for the generalized algorithm that we will propose.

Remark. *To be more explicit, technically there is one more nodes in the execution tree in the above example if we consider pairing the half edge of vertex* 1 *to the half edge of vertex* 3*. But we omit this in the illustration as we only intend to demonstrate the key idea of what an execution tree does - the storing of states/partial graphs up to stage k.*

Before we end the chapter, it is worth pointing out that the algorithm by [13] only samples graph that satisfies **one** degree sequence uniformly at an asymptotic level. We however wish to generalize it one step further for it to sample graphs to satisfy two degree sequences.

Chapter II

Algorithm and Heuristics

In this chapter we will formally introduce our proposed generalization of the algorithm by [13]. As we proceed deeper into the chapter, we will discuss the related heuristics and a more generalized execution tree induced by our algorithm. Finally we end the chapter by formerly stating the main theorem about our algorithm - uniform sampling of bi-colored edged graphs that satisfies two colored degree sequences.

1 Algorithm

The algorithm of [13] only solves half of our problem as the input is a single degree sequence $\bar{d}$. We require an algorithm that accepts two (colored) degree sequences. If we do not want to reinvent the wheel, it would be natural for us to try to generalize the algorithm in [13]. Hence we propose the following modification to the algorithm.

Algorithm: Bi-colored graph sampling

```
Input  : $n-$length Red degree sequence $\bar{d}^R$ and Blue degree sequence $\bar{d}^B$
Output : $G$ satisfying the input degree sequences or output failure

Let $E_B, E_R$ be the set of Blue edges and Red edges respectively. Let $\hat{d}^B = (\hat{d}^B_1, ...., \hat{d}^B_n)$ and
$\hat{d}^R = (\hat{d}^R_1, ...., \hat{d}^R_n)$ be two vectors of $n$ integers.
1. Initialize $E_B, E_R$ to empty set and $\hat{d}^B = \bar{d}^B,\ \hat{d}^R = \bar{d}^R$.

2. if there are $i, j$, s.t. $\hat{d}^B_i > 0, \hat{d}^R_j > 0$ then
       $P \sim \mathrm{Uniform}[0,1]$
   else if for all $i$, $\hat{d}^B_i = 0$ and there is $j$ s.t. $\hat{d}^R_j > 0$ then
       P=1
   else if for all $j$, $\hat{d}^R_j = 0$ and there is $i$ s.t. $\hat{d}^B_i > 0$ then
       P=0;
   else
       Go to step 4;
   end

3. if $P \leq \frac{m_B}{m}$ then
       a Choose two vertices $v_i, v_j \in V$ with probability proportional to $\hat{d}^B_i\hat{d}^B_j(1 - \frac{d^B_i d^B_j}{4m})$ among all pairs $v_i,\ v_j$ with
         $i \neq j$ and $\{v_i, v_j\} \notin E_B, E_R$.
       b Add $(v_i, v_j)$ to $E_B$ and decrease $\hat{d}^B_i$ and $\hat{d}^B_j$ by 1.
       c Go to step 2
   else
       a Choose two vertices $v_i, v_j \in V$ with probability proportional to $\hat{d}^R_i\hat{d}^R_j(1 - \frac{d^R_i d^R_j}{4m})$ among all pairs $v_i,\ v_j$ with
         $i \neq j$ and $\{v_i, v_j\} \notin E_B, E_R$.
       b Add $(v_i, v_j)$ to $E_R$ and decrease $\hat{d}^R_i$ and $\hat{d}^R_j$ by 1.
       c Go to step 2
   end
```

```
4. if $|E_B| < m_B$ or $|E_R| < m_R$ then
   |  Output FAILURE
else
   |  Output $G = (V,\ E_B,\ E_R)$
end
```

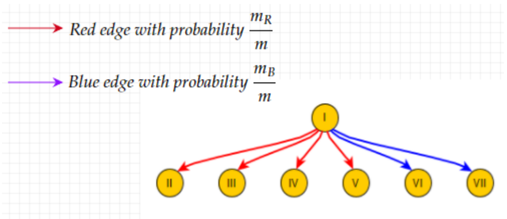
The proposed algorithm follows a very similar structure to the algorithm of [13], with the key difference being step 2 and step 3. Now let us compare it with step 2 of the algorithm by [13]. In the algorithm by [13], since there is only one degree sequence $\bar{d}$, so any graph G that satisfies $\bar{d}$ will have single colored edges. This means during the construction process, we know at by the end of k-th iteration, there will be exactly k many single colored edge(s) formed.
However in our case of two colored degree sequences, at each k-th iteration, we have to decide whether we are going to form a blue edge or red edge. We have decided to propose that we make this decision in a probabilistic manner instead of deterministic. The reason for this will be elaborated in subsection (b).

The next thing to note is the difference in interpretation of the out **FAILURE** between the algorithm by [13] and our proposed algorithm. We have discussed in subsection (a) that an output of **FAILURE** is statistical evidence that the given (single) degree sequence $\bar{d}$ is not graphical. In our case, our algorithm is less conclusive in that regard as a **FAILURE** in our case only provide evidence that there is no graph G that can satisfies the two colored sequence simultaneously, however we cannot conclude whether or not the individual degree sequence is graphical.
The next and perhaps the most noteworthy difference is that of the execution tree. Since the new execution serves as visual foundation for all future analysis, we will discuss it formally and in detail.

(a) Generalized Execution tree

Before we delve into the heuristics and analysis of our algorithm, it is vital that we spend some time to analyze the new execution tree since it serves as the main tool to model our algorithm up to each stage $k \leq m$ and is heavily used in chapter III.

To help understand the new execution tree, let us run the our algorithm with the input colored sequences $\bar{d}^B = (0, 1, 1), \bar{d}^R = (2, 1, 1)$. We will get the following partial execution tree T at depth 1.


Red edge with probability $\frac{m_R}{m}$
Blue edge with probability $\frac{m_B}{m}$
I
II
III
IV
V
VI
VII

In the diagram above, the red arrow represents forming a red edge with probability $\frac{m_R}{m}$, likewise for the blue arrow. Nodes II to VII store the following states of the algorithm.

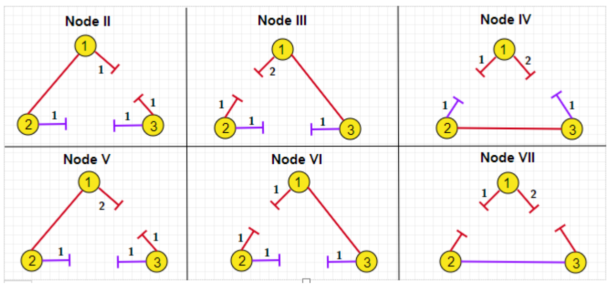


Figure II.1: Execution tree nodes

We now draw our attention to node IV. Observe that there is now no way to produce a graph G that satisfies the residue red degree sequence $\hat{d^R} = (2,0,0)$ and residue blue degree sequence $\hat{d^B} = (0,1,1)$ and hence our algorithm will output **FAILURE**. Now compare this to the execution tree of the algorithm by [13]. Without colors, node IV is **still able** to output a valid graph satisfying the uni-colored degree sequence $(2,2,2)$.

So the first difference is that the new algorithm has fewer nodes that leads to successful output, however, what is more important is perhaps the reason why this is the case. The crux of this issue lies with the order in which we decided to form our blue or red edges. Using II.1 again, we see that if we form a blue edge first, we will always have a successful output but this is not the case if we start with forming a red edge. This provide a good intuition that the ordering of the formation of the edges directly influence the success rate of the algorithm. In fact, the ordering also affects uniformity of the sampling process, this will be thoroughly explored in the upcoming chapters.

Now the next thing to take away from the execution tree is that it is possible for multiple leaves to store the same graph G that satisfies the input. In fact using our example $\bar{d^B} = (0,1,1), \bar{d^R} = (2,1,1)$, there is exactly one graph that satisfies it, so this means the leaves of the execution tree must all store such a graph. So now suppose the input blue and red sequences are graphical, hence there is a graph G that satisfies the sequences. One could ask then how many leaves stores such a G?

Lemma 1. *For any G that satisfies $\bar{d^B}$ and $\bar{d^R}$, the size of leaves in the execution tree that stores G is $m! \prod\limits_{i\in[n]} (d_i^B!)(d_i^R!)$.*

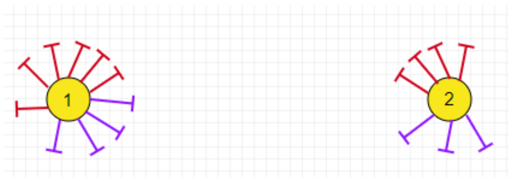
Proof. We know there are $m!$ ways for the edges to be form. Once an order is fixed, for each i, there are $(d_i^B!)(d_i^R!)$ ways for the half-edges of i to be paired, and each way represents a path from the root leading to the leaves storing G. Hence we have that the total number of leaves storing G is $m! \prod\limits_{i\in[n]} (d_i^B!)(d_i^R!)$. □

Remark. *From the above lemma, one sees that the number of leaves is actually independent of G and only depends on the degree sequences.*

(b) Heuristics

This section will be dedicated to the heuristics behind our proposed algorithm. Mainly, we will discuss in detail the motivations behind some of the steps in our proposed algorithm. We will start by addressing the elephant in the room - why do we use probabilistic means to decide which colored edge to form at stage k
II.1 shows us that the order in which the edges are formed directly affects how successful the algorithm is in generating a graph that satisfies the input. This would motivate the question on how do we come up with a suitable ordering of red and blue edges so that it not only maximizes the success rate but also generates all possible graphs uniformly at random.

However coming up with such a deterministic sequence of blue and red is not a trivial task, as the order has a huge impact on the uniformity. To illustrate this, let us simplify the matter by once again using uniform matching of half-edges. Consider the following picture:



The above picture focuses on vertices 1 and 2 among the n vertices, where they have a much higher blue and red degree compared to other vertices. Now if we decide (deterministically) to form a blue edge here, then there will be a bias towards graph containing a blue edge between vertices 1 and 2. The same goes if we decide to form a red edge. This conundrum motivates the probabilistic choice between blue and red edges, as we are essentially flipping an unfair coin with probability P_B of obtaining blue and P_R of obtaining red . This then reduces the problem from choosing a suitable ordering of red and blue to choosing a suitable probability weight for the coin flip.

Remark. *The example given above also provide an intuition on why it is not wise to perform the algorithm by [13] on the blue degree sequence first, follow by the red degree sequence. Because this essentially is equivalent to the following ordering:*

$$(blue, \dots, blue, red, \dots, red)$$

where the blue edges appear m_B times first, follow by the red edges appearing m_R times after.

It turns out there is a natural choice for P_B and P_R which we will derive in the upcoming section. However before we proceed, it might be worthwhile to consider why it is not wise to set $P_B = P_R = \frac{1}{2}$ straight away at the start. Let us first consider the scenario where $m_B \gg m_R$. If we set $P_B = P_R = \frac{1}{2}$ in this case where there are much more blue edges than red edges, then we are forming *red* edges much more often than required. The same can be said if after some k iterations the remaining red edges becomes much more than the remaining blue edges. This motivates the idea that a good choice of P_B and P_R would be one that respects the remaining colored edges, and this idea will be explored in the upcoming section.

Chapter III

Analysis Part One: Coin-flip and offsets

The main goal of this section is to determine what would be a suitable value for P_B and P_R using the intuition discussed in subsection (b). After which we will give a proper treatment to the derivation of the bias when performing the matching of half edges by following similar technique used in [13]. It should be noted that the derivation and analysis taken will have an asymptotic focus, meaning when m is sufficiently large.

1 Derivation of P_B and P_R

The main goal of this section is to determine what would be a suitable value for P_B and P_R, defined to be the probability forming a blue edge or red edge in each iteration of the algorithm. At the end of subsection (b), we analyze why it is not ideal to propose a fair coin flip right off the start. We also discussed that a good choice of P_B, P_R would be one respects the remaining colored edges. Now let k_B be defined as the number of blue edges formed by the end of stage k, with k_R defined symmetrically. Then the following would be a reasonable, albeit naive definition.

$$P_B^k := \frac{m_B - k_B}{m - k}$$

Here $P_R^k = 1 - P_B^k$ is defined symmetrically. We now note that k_B and k_R is a random variable for $k > 0$. This means that for $k > 0$, P_B^k and P_R^k is also random. As mentioned, we are interested in asymptotic results; which is why at this point it is worthwhile to investigate around which value does P_B^k and P_R^k concentrates. The first step to do this is to make use of the notion of a *martingale.*

Definition. *[5]*Martingale *is a discrete-time stochastic process (i.e., a sequence of random variables)* $X_1, X_2, X_3, \ldots$ *that satisfies for any time* n*,*

- $\mathbb{E}[|X_n|] < \infty$
- $\mathbb{E}[X_{n+1}|X_1, \ldots, X_n] = X_n$

That is, the conditional expected value of the next observation, given all the past observations, is equal to the most recent observation.

With this, it is now reasonable to claim that P_B^k and P_R^k is in fact a martingale since we are choosing to form a blue or red edge in each stage k base on their respective ratio of unpaired half-edges to the total amount of unpaired half edges, which is why in *expectation* we should see P_B^k and P_R^k follow the properties of a Martingale. Hence define $X_k := P_B^k$

Claim. *For all* $0 \leq k \leq m-1$, X_k *is a Martingale.*

Proof. Clearly $\mathbb{E}[X_k] < \infty$. Now $\mathbb{E}[X_{k+1}|X_k] = X_k(\frac{m_B - k_B - 1}{m-k-1}) + (1 - X_k)(\frac{m_B - k_B}{m-k-1})$ This is because by definition X_k is the probability of entering the blue case and hence $1 - X_k$ is the probability of entering the red case. Next by substituting the definition of X_k, we arrive at the expression $(\frac{m_B - k_B}{m-k})(\frac{m_B - k_B - 1}{m-k-1}) + (1 - \frac{m_B - k_B}{m-k})(\frac{m_B - k_B}{m-k-1})$. All that remains is to simplify this expression and we arrive at the equation $\mathbb{E}[X_{k+1}|X_k] = X_k$. The proof that P_R^k is a martingale is completely symmetric. ☐

This simple yet extremely useful result tells us one very important fact, that is the ratio of un-formed blue edges to the total remaining un-formed edges is **expected** to remain constant. Furthermore, by using the law of total expectation, we arrive at the following.

$$\mathbb{E}[\mathbb{E}[X_{k+1}|X_k]] = \mathbb{E}[X_{k+1}] = \mathbb{E}[X_k] = \ldots = \mathbb{E}[X_0] = \frac{m_B}{m} = P_B^0 := P_B$$

So what we have arrived at is that the the expectation of P_B^k is $\frac{m_B}{m}$. The next step is to show that in fact, P_B^k concentrates around its expectation. If this is true, then we arrive at a very natural choice for P_B (and P_R). In order to show this concentration, we will make use of the following theorem.

Theorem 1. *[9][Azuma] Suppose* $\{X_k : k = 0, 1, 2, 3, \ldots\}$ *is a martingale and for all* k, $|X_{k+1} - X_k| \leq c_k$. *Then for all positive integers* N *and all positive reals* ϵ, $\mathbb{P}(|X_N - X_0| \geq \epsilon) \leq 2\exp\left(\frac{-\epsilon^2}{2\sum_{k=1}^{N} c_k^2}\right)$.

This is theorem by [9] is extremely useful in showing that P_B^k concentrates around its expectation, which is precisely X_0 in this case. And all we have to do now is to bound the difference between P_B^{k+1} and P_B^k, which we will show in the following lemma.

Lemma 2. *Let* $X_k := P_B^k$, *then for all* k, $|X_{k+1} - X_k| \leq c_k$ *for some* c_k.

Proof. $|X_k - X_{k+1}| \leq |\frac{m_B - k_B}{m-k} - \frac{m_B - k_B - 1}{m-k-1}| = |\frac{(m-k)-(m_B-k_B)}{(m-k)^2-(m-k)}|$. Since $m_B - k_B$ is non negative, we have that $|\frac{(m-k)-(m_B-k_B)}{(m-k)^2-(m-k)}| \leq \frac{(m-k)}{(m-k)^2-(m-k)} = \frac{1}{m-k-1} = c_k$. ☐

With this and theorem by [9]. We see that with high probability (w.h.p) that P_B^k concentrates around its expectation and thus this is a good motivation for us to set

$$P_B^k = \ldots = P_B^0 = P_B = \frac{m_B}{m}$$

The case for P_R^k can be argued symmetrically.

Remark. *Perhaps it is worthwhile to remark that the reason that it took some effort to justify our choice of* P_B *and* P_R *is due to the fact that* k_B, k_R *is a random variable, that means a lot of times we have to deal with its expectation and investigate how much its actual value deviates from it. We will see a similar conundrum soon in section 2*

2 Proposition of bias and offset

Now that we have proposed a choice for P_B and P_R, what remains is to propose a suitable set of bias for choosing which v_i, v_j to form an edge with, for the reason that is discussed in subsection (a). In order to motivate our choice, we will use similar technique as in [13]. The idea is that we start off by assuming we form an arbitrary edge (red or blue) uniformly, then lets assume our algorithm terminates successfully and output a G that satisfies the input red and blue degree sequence. So fix such a G, we will then compute what is $\mathbb{P}(G)$, the probability that our algorithm produces such a G in general. From there we derive the bias towards G and an offset for it.

Remark. *Before we continue, we will remark that we will be working a lot with the expectation of a random variable and use it as a motivation behind most of our proposed definitions or values. However the formal justification for them will be discussed in the later chapters on the analysis.*

The first question that is perhaps worth addressing is what is the *residue blue / red degree* or unpaired blue / red half-edges of vertex i, which we will denote as $\hat{d_i^B}$ and $\hat{d_i^R}$ from here on. This is because in the setting in which we match half-edges uniformly, the probability of there being an edge (whichever color) connecting to vertex i is directly proportional to the respective residue degree. However the residue degree is a random variable and hence we will investigate its value in expectation.

Lemma 3. *Let $i \in [n]$, $0 \leq k \leq m-1$. If the matching of the blue half edges is done uniformly at random, then*

- *The expected number of unpaired blue half-edges at the end of stage k for v_i is $d_i^B(1-\frac{k}{m})$.*
- *The expected number of unpaired red half-edges at the end of stage k for v_i is $d_i^R(1-\frac{k}{m})$.*

Proof. Fix and i and k, the expected number of (total) blue edges formed is kP_B. Since the matching of half edges is done uniformly at random, the expected number of matched/paired half edges for v_i is $kP_B \cdot \frac{d_i^B}{m_B}$. Hence the expected number of unpaired half edges for v_i is $d_i^B - kP_B \cdot \frac{d_i^B}{m_B}$. Since $P_B = \frac{m_B}{m}$, the expression simplifies to $d_i^B(1-\frac{k}{m})$. The proof for the red case is completely symmetric. $\square$

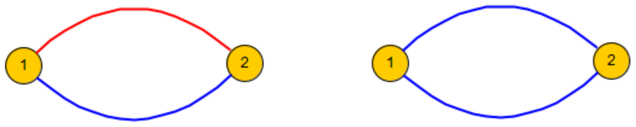
Remark 2. *From the proof above, let $q_k = 1 - \frac{k}{m}$. Observe that now q_k can be viewed as the probability that an arbitrary edge is not present* ***by*** *stage k. Further more observe that by our choice of P_B and P_R, this probability becomes independent of the colors.*

Now since we are interesting in the probability that our algorithm output a graph G which is simple, it is then worthwhile to investigate how many ways are there to there for valid matchings of half edges at stage k so that we have a successful output. As an example, the number valid matchings of blue half edges is

$$\binom{2m_B - 2k_B}{2} - \Delta_k^B \tag{III.1}$$

Where $\binom{2m_B-2k_B}{2}$ is the number of arbitrary ways that to match blue half edges at stage k and Δ_k^B is the number of matchings of blue half edges at stage k that leads to blue self loops and blue multi-edge.

Remark. *blue multi-edge in our context, we are counting both the following cases:*



The case for red is also the same symmetrically.

However in our example, the number of arbitrary blue and red pairings at stage k is a random variable because k_B and k_R is a random variable. So once again, we will investigate its expectation. In other words, we wish to know what is $\mathbb{E}[\binom{2m_B-2k_B}{2}]$. A wishful thinking is that we could simply push the expectation function into the choose function as follows

$$\mathbb{E}[\binom{2m_B - 2k_B}{2}] = \binom{2m_B - 2\mathbb{E}[k_B]}{2}$$

However, the "choose-2" function is not linear, which means that if k_B and $\hat{d_i^B}$ deviates from its expectation even by a little, we would end up with more choices by a factor! Hence before we proceed, we need to investigate how much the R.H.S of the equation deviates from the L.H.S. The result can be phrase in terms of the following lemma.

Lemma 4. *For each $k \leq m - m^{1-\delta}$, where δ is small and positive, then w.h.p the following holds*

- $\mathbb{E}[\binom{2m_B-2k_B}{2}] = \binom{2m_B-2\mathbb{E}[k_B]}{2} + o(m^2)$

Proof. We need to make use of Azuma's inequality 1 once again. We will use the c_k in the proof of Lemma 2. Now let $\epsilon = (m-k)^{-1/2-\tau/2}$ for some small $\tau > 0$. Then by 1, this means that $\mathbb{P}(|X_N - X_0| \geq \epsilon) \leq 2\exp\left(\frac{-\epsilon^2}{2\sum_{i=1}^k c_i^2}\right) \leq 2\exp\left(\frac{-1/(m-k)^{-1-\tau}(m-k)^2}{2k/(m-k-1)^2}\right)$. Since asymptotically $m-k-1 \approx m-k$, we further simplify to $2\exp\left(\frac{-(m-k)^{1+\tau}}{2k}\right) \leq 2\exp\left(\frac{-(m-k)^{1+\tau}}{2m}\right)$. Since $k \leq m^{1-\frac{1}{m^\delta}}$, this means $2\exp\left(\frac{-(m-k)^{1+\tau}}{2m}\right)$ is at most $2\exp\left(\frac{-(m^{1-\delta})^{1+\tau}}{2m}\right)$, which we can choose a suitable τ (that depends on δ) so that the exponent goes to $-\infty$.

So we know that with with low probability, $|X_N - X_0|$ differs by more than $(m-k)^{-1/2-\tau/2}$. This would imply with high probability (w.h.p), $|X_N - X_0|$ is bounded by $(m-k)^{-1/2-\tau/2}$. Explicitly, w.h.p, we have the following

$$\begin{aligned} &|\frac{m_B - k_B}{m-k} - \frac{m_B}{m}| \leq (m-k)^{-1/2-\tau/2} \\ \implies &|(m_B - k_B) - m_B q_k| \leq (m-k)^{\frac{1}{2}-\tau/2} \\ &\implies (m_B - k_B) = m_B q_k + o(m) \end{aligned}$$

Now to get the conclusion we want, we substitute $(m_B - k_B) = m_B q_k + o(m)$ into $\binom{2m_B - 2k_B}{2} = \frac{(2m_B - 2k_B - 1)(2m_B - 2k_B)}{2}$ $\square$

Now we are ready to approximate the expected number of arbitrary matchings at stage k.

Lemma 5. *For any $k \leq m(1 - \frac{1}{m^\delta})$, if m_B (and m_R) is of $O(m)$, then the expected number of arbitrary matchings is asymptotically*

- *$2(m_B)^2(q_k)^2$ as $m \to \infty$ for the blue case*
- *$2(m_R)^2(q_k)^2$ as $m \to \infty$ for the red case*

Proof. Once again we make use of Lemma 4 and we have that $\mathbb{E}[\binom{2m_B - 2k_B}{2}] = \binom{2m_B - 2P_B k}{2} + o(m^2)$. Next we will expand this expression as follows,

$$\begin{aligned} \binom{2m_B - 2P_B k}{2} + o(m^2) &= \frac{(2m_B - 2P_B k)(2m_B - 2P_B k - 1)}{2} + o(m^2) \\ &= 2m_B^2(1 - \frac{P_B k}{m_B} - \frac{1}{2m_B})(1 - \frac{P_B k}{m_B}) + o(m^2) \end{aligned}$$

Since $1 - \frac{P_B k}{m_B} = q_k$, we get that $2m_B^2(1 - \frac{P_B k}{m_B} - \frac{1}{2m_B})(1 - \frac{P_B k}{m_B}) + o(m^2) = 2m_B^2 q_k^2(1 - \frac{2m}{m_B(m-k)} + \frac{o(m^4)}{m_B^2(m-k)^2})$. Finally since m_B is of $O(m)$, this means the last 2 terms $\to 0$ as $m \to \infty$, and with that, we have the conclusion we want. The proof for the red case is completely symmetric. $\square$

Perhaps it is worthwhile now for a quick summary on what we achieved. We started by investigating what is the number of valid matchings of half edges at a given stage k and we arrive at equation III.1. From there we decide to estimate $\binom{2m_B - 2k_B}{2}$ by its expectation, and by Lemma 4, we conclude that we can push in the expectation function into the choose 2 function at a cost of a deviation of $o(m^2)$. After which we can finally come up with a approximation for the expectation of $\binom{2m_B - 2k_B}{2}$ by Lemma 5. However we still need to come up with a good approximation for Δ_k^B in Equation III.1. Once again, we will estimate its value by its expectation, but for the moment we will postulate that the following remark holds.

Remark 3. *For each $0 \leq k \leq m-1$, the following holds*

1. *The expected number of blue self loops at stage k is $\approx 2m(q_k)^2\lambda(\bar{d}^B)$ where $\lambda(\bar{d}^B) = \frac{\sum_{i \in [n]} \binom{d_i^B}{2}}{2m}$. The red case is defined symmetrically.*
2. *The expected number of blue multi-edge at stage k is $\approx \sum_{i \sim_B j} d_i^B d_j^B (q_k)^2(1 - q_k)$. The red case is defined symmetrically.*

With the above lemmas, we are ready to approximate $\mathbb{P}(G)$, where G is the output of our sampling algorithm. Now let $L(G)$ be the set of all leaves in the execution tree storing G. Next, Let ξ_R^k, ξ_B^k are the number of suitable blue matchings and red matchings in stage k respectively. Then we arrive at the following,

$$\begin{aligned} \mathbb{P}(G) &= \sum_{L(G)} \prod_{k=0}^{m-1} \frac{1}{\xi_R^k + \xi_B^k} \\ &= |L(G)| \cdot \prod_{k=0}^{m-1} \frac{1}{\xi_R^k + \xi_B^k} \end{aligned}$$

By Lemma 1, we know the expression for $|L(G)|$ explicitly and that it is actually independent of G. So all that remains is to approximate ξ_B^k and ξ_R^k. Once again, we will approximate them by their expectation by using *Lemma* 5, *Remark* 3. With this, we arrive at

$$\xi_B^k \approx 2m_B^2 q_k^2 - 2mq_k^2\lambda(\bar{d}^B) - 4m_B q_k^2(1-q_k)\gamma_G^B$$

where $\gamma_G^B = \sum_{i\sim_B j} d_i^B d_j^B (q_k)^2(1-q_k)/4m_B$ and ξ_R^k is evaluated symmetrically. Thus we have the following:

$$\xi_B^k + \xi_R^k = 2m^2 q_k^2 \cdot [P_R^2 + P_B^2 - \frac{1}{m}(\lambda(\bar{d}^B)) + \lambda(\bar{d}^R))) - \frac{2(1-q_k)}{m}(P_B\gamma_G^B + P_R\gamma_G^R)]$$

Since $P_B^2 + P_R^2 = (P_B + P_R)^2 - 2P_BP_R = 1 - 2P_BP_R$, Hence

$$\prod_{k=0}^{m-1} \frac{1}{\xi_B^k + \xi_R^k} = \prod_{k=0}^{m-1}[1 - \frac{1}{m}(\lambda(\bar{d}^B)) + \lambda(\bar{d}^R)) - \frac{2(1-q_k)}{m}(P_B\gamma_G^B + P_R\gamma_G^R) - 2P_BP_R]^{-1}$$
$$= \exp[-1 \cdot \sum_{k=0}^{m-1} \ln(1 - \frac{1}{m}(\lambda(\bar{d}^B)) + \lambda(\bar{d}^R)) - \frac{2(1-q_k)}{m}(P_B\gamma_G^B + P_R\gamma_G^R) - 2P_BP_R))]$$

Now we will use the approximation $\ln(1-x) = x + O(x^2)$. In our case, the terms in $O(x^2) \to 0$ as $m \to \infty$, with the exception of the term $(P_B^2 P_R^2)$. Thus we can rewrite the above as

$$\prod_{k=0}^{m-1} \frac{1}{\xi_B^k + \xi_R^k} \approx \exp[\sum_{k=0}^{m-1} \frac{1}{m}(\lambda(\bar{d}^B)) + \lambda(\bar{d}^R)) + \sum_{k=0}^{m-1} \frac{2(1-q_k)}{m}(P_B\gamma_G^B + P_R\gamma_G^R) + f(P_B, P_R))]$$

In which $f(P_B, P_R) = \sum_{k=0}^{m-1} 2P_BP_R + 4P_B^2P_R^2$. At this point, it would be helpful to observe that the only term in the evaluation of $\mathbb{P}(G)$ that depends on G is the factor $\exp[\sum_{k=0}^{m-1} \frac{2(1-q_k)}{m}(P_B\gamma_G^B + P_R\gamma_G^R)]$, the rest of the factors only depends on the input red and blue sequences. This means the probability of outputting G, and hence the bias, is only dependent on that factor. Evaluating further, we arrive at the following:

$$\exp[\sum_{k=0}^{m-1} \frac{2(1-q_k)}{m}(P_B\gamma_G^B + P_R\gamma_G^R)] \approx \exp\left[\frac{\sum_{i\sim_B j} d_i^B d_j^B}{4m}\right] \cdot \exp\left[\frac{\sum_{i\sim_R j} d_i^R d_j^R}{4m}\right]$$

So what this means is that adding one blue edge connecting i, j in G produce a bias of $\exp[\frac{d_i^B d_j^B}{4m}]$. Hence at every stage we will offset the probability connecting i, j by $\exp[\frac{d_i^B d_j^B}{4m}]^{-1} \approx 1 - \frac{d_i^B d_j^B}{4m}$. The same reasoning applies to the red case.

Before we conclude this chapter, it is useful what we have achieved. The goal of this chapter is to give motivation and intuition behind our choice of the weights for the coin-flip P_B, P_R and the bias-offsets when performing the matchings of colored half-edges. However, the main goal of this thesis is to show that given our definitions, we can show the following theorem

Theorem 2. *For an arbitrary $\tau > 0$ and for any degree sequences $\bar{d}^B, \bar{d}^R$ with maximum degree of $O(m^{\frac{1}{4}-\tau})$, our algorithm generates any graph G that satisfy the aforementioned degree sequences with probability within $1 \pm o(1)$ factor of uniformity.*

Chapter IV

Analysis Part Two: Asymptotic results

The work so far is mostly heuristic in nature, where we provide motivation on the proposed generalization of [13]'s algorithm. However we still need to prove that our generalized algorithm does in fact generate all graphs that satisfies the input colored degree sequences uniformly at random as $m \to \infty$. The goal in mind is formally stated in Theorem 2. However the proof of Theorem 2 is an endeavor, and we will summarize, in essence, the key milestones involved as followings

Milestones 1.

1. *Analyze our algorithm with the proposed value, and identify expressions that will affect uniformity of the sampling*
2. *Propose an approximation for the expression in step 1*
3. *Show that our approximated value proves Theorem 2*
4. *Prove that the actual expression in step 1 concentrates around our proposed approximation*

At this point, the steps described will seem convoluted, and it will be our main focus to shed light on them. To this end, we will omit some cumbersome computation that does not provide any extra useful insights on the main ideals of the different notions that we will introduce.

1 Asymptotic Behaviour

We will start by tackling the first step in 1. Formally we will be showing the following proposition.

Proposition 1. *For all for G that satisfies the input degree sequences, the probability of our algorithm generating it can be written as*

$$\mathbb{P}(G) = \prod_{i\in[n]} (d_i^B!)(d_i^R!)[\prod_{i\sim_B j}(1-\frac{d_i^B d_j^B}{4m})\prod_{i\sim_R j}(1-\frac{d_i^R d_j^R}{4m})]$$
$$\times \sum_{N\in S(M)}\prod_{0\le k\le m-1}\frac{1}{\binom{2m_B-2k_B}{2}-\Psi_k^B(N)+\binom{2m_R-2k_R}{2}-\Psi_k^R(N)}$$

for some values $\Psi_k^B(N)$ and $\Psi_k^R(N)$.

Proof. To begin proving the proposition, we use a similar technique as used in section 2. We first wish to find an expression for $\mathbb{P}(G)$, where G satisfies the colored degree sequences, with our proposed P_B and P_R and the relevant bias-offset now. So let $M(G)$ be the set of all valid matching of half-edges that lead to G. Observe that for any $M, M' \in M(G)$, M, M' differ only by a permutation of the half edges belonging to each vertex i. So this means that

$$\mathbb{P}(G) = \sum_{M \in M(G)} \mathbb{P}(M) = \prod_{i \in [n]} (d_i^B!)(d_i^R!)\mathbb{P}(M)$$

for a fixed $M \in M(G)$. The reasoning is similar to that used in Lemma 1. So we reduce the problem to finding an expression for $\mathbb{P}(M)$. The next observation is that once we fix a matching M, we can denote $S(M)$ to be the order in which the red/blue edges occurs. This further reduces our problem in the sense that we can rewrite $\mathbb{P}(M) = \sum_{N \in S(M)} \mathbb{P}(N)$.

This is where our real work begins as we proceed to find an expression for $\mathbb{P}(N)$. Now for each $N = (e_1, \dots, e_m)$, $\mathbb{P}(N)$ is nothing more than

$$\mathbb{P}(N) = \prod_{0 \le k \le m-1} \mathbb{P}(e_{k+1} \mid e_1, \dots, e_k) \tag{IV.1}$$

Now this can then be rewritten in terms of our algorithm. Let us first have the following definition.

Definition 1. *E_k^B, E_k^R is the set of all possible blue and red pairings, respectively, after choosing $(e_1, \dots, e_k)$.*

Then we can rewrite (IV.1)

$$\begin{aligned}
\mathbb{P}(N) &= \prod_{0 \le k \le m-1} \mathbb{P}(e_{k+1} \mid e_1, \dots, e_k) && \text{(IV.2)}\\
&= \prod_{0 \le k \le m-1} \frac{\text{weight of } e_k}{\text{sum of weights of all other pairings in } E_k^B \text{and } E_k^R} && \text{(IV.3)}\\
&= \prod_{i \sim_B j} (1 - \frac{d_i^B d_j^B}{4m}) \prod_{i \sim_R j} (1 - \frac{d_i^R d_j^R}{4m}) \prod_{0 \le k \le m-1} \frac{1}{C_B + C_R} && \text{(IV.4)}
\end{aligned}$$

Here $C_B = \sum_{(u,v) \in E_k^B} \hat{d_u^B} \hat{d_v^B} (1 - d_u^B d_v^B / 4m)$ and $C_R = \sum_{(u,v) \in E_k^R} \hat{d_u^R} \hat{d_v^R} (1 - d_u^R d_v^R / 4m)$.

However C_B and C_R are too cumbersome to work with and does not provide insights into further analysis, hence we need to rewrite it further. Denote the number of **unsuitable** blue matchings after choosing the edges after choosing $(e_1, \dots, e_k)$ by Δ_k^B, similarly for red . Now if we focus on the factor

$$\sum_{(u,v) \in E_k^B} \hat{d_u^B} \hat{d_v^B}$$

By definition of E_k^B, this means we are summing over all possible valid matchings to form blue edges at stage k where we account for the different choices of blue half edges for each matching.

Then what this means is that $\sum_{(u,v) \in E_k^B} \hat{d_u^B} \hat{d_v^B}$ can be written as $\binom{2m_b - 2k_b}{2} - \Delta_k^B$. Using this expression, and performing substitution into C_B, one finds that C_B can be written as $\binom{2m_b - 2k_b}{2} - \Psi_k^B(N)$ where $\Psi_k^B(N) = \Delta_k^B + \sum_{(u,v) \in E_k^B} \hat{d_u^B} \hat{d_v^B} (d_u^B d_v^B / 4m)$. The red case, Ψ_k^R can be defined symmetrically.

Now combining what we know so far we arrive at the following:

$$\mathbb{P}(G) = \prod_{i \in [n]} (d_i^B!)(d_i^R!)[\prod_{i \sim_B j} (1 - \frac{d_i^B d_j^B}{4m}) \prod_{i \sim_R j} (1 - \frac{d_i^R d_j^R}{4m})]$$

$$\times \sum_{N \in S(M)} \prod_{0 \le k \le m-1} \frac{1}{\binom{2m_b - 2k_b}{2} - \Psi_k^B(N) + \binom{2m_r - 2k_r}{2} - \Psi_k^R(N)}$$

Which completes the proof of our proposition. □

What comes next is that we now make use of the assumption we made in Theorem 2 that $\bar{d}^B$ and $\bar{d}^R$ has maximum degree of $O(m^{\frac{1}{4}-\tau})$, combined with the approximation $1-x = e^{-x+O(x^2)}$. We can conclude the following

$$\mathbb{P}(G) = \prod_{i\in[n]} (d_i^B!)(d_i^R!)\ e^{-\gamma_G^B - \gamma_G^R + o(1)} \sum_{N\in S(M)} \prod_{0\le k\le m-1} \frac{1}{\binom{2m_b-2k_b}{2} - \Psi_k^B(N) + \binom{2m_r-2k_r}{2} - \Psi_k^R(N)}$$

At this stage we are done with step one of 1, as the terms above that depend on G are $\gamma_G^B, \gamma_G^R, \Psi_k^B(N)$ and $\Psi_k^R(N)$. As such, these are the terms that affect uniformity in sampling.
Now we are ready to tackle steps two and three in Milestones 1, where the first order of work cut out for us is to come up with an approximation, ψ_k^B, ψ_k^R, for $\Psi_k^B(N)$ and $\Psi_k^R(N)$. After which we will show the following lemma regarding ψ_k^B and ψ_k^R.

Lemma 6. *If $d_{max} = O(m^{\frac{1}{4}-\tau})$, then with high probability, the following relation holds*

$$\prod_{0\le k\le m-1} \frac{1}{\binom{2m_b-2k_b}{2} - \psi_k^B + \binom{2m_r-2k_r}{2} - \psi_k^R} \propto e^{\gamma_G^B + \gamma_G^R + \lambda(\bar{d}^B) + \lambda^2(\bar{d}^B) + \lambda(\bar{d}^R) + \lambda^2(\bar{d}^R) + o(1)}$$

Suppose we manage to prove Lemma 6, then this means we can substitute Ψ_k's with ψ_k's and combining it with the expression we got for $\mathbb{P}(G)$ in Proposition 1, we see that asymptotically $\mathbb{P}(G)$ is independent of G that satisfies the input degree sequences. This would then imply Theorem 2, which is the goal of this thesis.

(a) G_{p_k} model and its heuristics

The goal of at hand is to propose a good approximations ψ_k^B and ψ_k^R for $\Psi_k^B(N)$ and $\Psi_k^R(N)$. Before we begin, we note that we will be making a few changes to the notation throughout this (sub)section. The first is that since we are working with a fixed ordering $N \in S(M)$, we will write $\Psi_k^B(N)$ as simply Ψ_k^B (likewise for Ψ_k^R). The next thing to note is that discussion is completely symmetric between the blue and red case , and hence our discussion will only revolve around the blue case.

Before we proceed to the computations, it is indeed worthwhile to have another discussion on the the heuristic behind how we are going to define ψ_k^B. First let us recall that $\Psi_k^B = \Delta_k^B + \sum_{(u,v)\in E_k^B} \hat{d_u^B}\hat{d_v^B}(d_u^B d_v^B/4m)$, where Δ_k^B counts the number of unsuitable blue edges that can be formed at stage k, and E_k^B is the set of suitable blue edges that can be formed after choosing the edges $(e_1, \ldots, e_k)$ of N. Then Ψ_k^B is a random variable and naively we might just decide to take it's expectation as an approximation. However this is extremely difficult to compute, if we consider the following

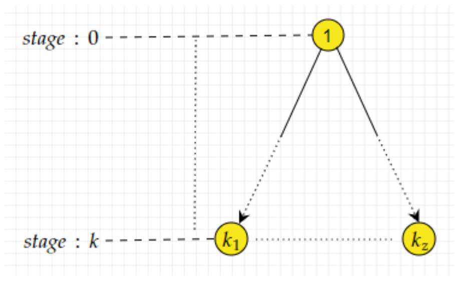
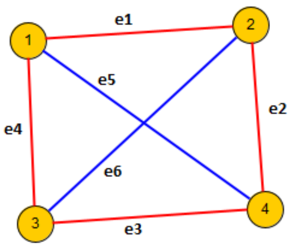


Figure IV.1: Stage k execution tree

Here the diagram depicts an execution tree up to stage k, and the nodes k_1 to k_z are the partial graphs of G at stage k. Then for each partial graph stored in k_1 to k_z, it can have it's own variation of (the expectation of) Ψ_k^B ! Hence we need a more *effective* way to approximate Ψ_k^B. As it turns out, we have another way of looking at this problem. Consider the following graph that satisfies the sequence $\bar{d}^B = (1,1,1,1), \bar{d}^R = (2,2,2,2)$.



So let's assume that our algorithm output this graph after $m = 6$ iterations. Now one could ask what would be it's *expected* partial graph at, for example, stage 4 ?

Remark 4. *In order for the question to make sense, one need to assume some kind of distribution on $S(M)$, which if we recall, is the set of all orderings of the edges of G. So throughout the rest of the section, we will impose uniform distribution on $S(M)$.*

Turns out we can define the *expected* partial graph of G, as the outcome of the following steps:

1. Iterate through all edges, $e_i,\ i \in \{1, \dots, m\}$, of G
2. For each edge, e_i:
   - For some probability q_k^B we remove e_i if it is blue .
   - For some probability q_k^R we remove e_i if it is red .
3. The end result would be the expected graph of G at stage k.

Perhaps what is really interesting to note about this process is that choice of q_k^B and q_k^R. If we recall Remark 2, from our choice of P_B and P_R we can set $q_k^B = q_k^R = q_k = 1 - \frac{k}{m}$. Which means the the process we just described becomes independent of the colors of the edges.

1. Iterate through all edges, $e_i,\ i \in \{1, \dots, m\}$, of G
2. For each edge, e_i, remove e_i with probability q_k.
3. The end result would be the expected graph of G at stage k.

We will henceforth call this *expected* partial graph the G_{p_k} model, where $p_k = 1 - q_k$.

Remark 5. *One might notice at this point is that by definition G_{p_k} might not have exactly k edges as G_{p_k} is the result of m coin flips. In fact one can observe that if we look at all the possible G_{p_k} with **exactly** k edges, this is in $1-1$ correspondence with the nodes of the execution tree at depth k. And the probability of G_{p_k} having exactly k edges is bounded from below by the following lemma due to [13]*

Proposition 2. *For all k, $\mathbb{P}(|E(G_{p_k})| = k) \geq \frac{1}{n}$*

Now let us summarize what we have done. By IV.1 we realize it is impractical to compute the different possible variations of Ψ_k^B for each partial graph stored in node k_1 to k_z. So now we instead propose to compute the expected value Ψ_k^B in the G_{p_k} model, which is what we will define ψ_k^B to be.

(b) Computing ψ_k^B (and ψ_k^R)

With the discussion in subsection (a), we will now begin computing Ψ_k^B in the G_{p_k} model. We first need to perform some bookings as we rewrite the current definition of Ψ_k^B. From its definition, we observe that Ψ_k^B can be written as $\Delta_k^B + \Lambda_k^B$ where

- $\Delta_k^B = \binom{2m_b - 2k_b}{2} - \sum\limits_{(i,j)\in E_k^B} \hat{d_i^B}\hat{d_j^B}$
- $\Lambda_k^B = \sum\limits_{i\neq j} \hat{d_i^B}\hat{d_j^B}\frac{d_i^B d_j^B}{4m} - \sum\limits_{\substack{(i,j)\notin E_k^B \\ i\neq j}} \hat{d_i^B}\hat{d_j^B}\frac{d_i^B d_j^B}{4m}$

Now by definition Δ_k^B counts the number of unsuitable pairings which is simply the sum of number of self-loops with multi-edges. We will denote them by $\Delta_k^{(B,1)}$, $\Delta_k^{(B,2)}$ where

$$\Delta_k^{(B,1)} = \sum_{i\in[n]} \binom{\hat{d}_i^B}{2} \tag{IV.5}$$

$$\Delta_k^{(B,2)} = \Delta_k^B - \Delta_k^{(B,1)} \tag{IV.6}$$

Next we will also rewrite Λ_k^B as follows

$$\begin{aligned} 4m\Lambda_k^B &= \sum_{i\neq j} \hat{d}_i^B \hat{d}_j^B d_i^B d_j^B - \sum_{\substack{(i,j)\notin E_k^B \\ i\neq j}} \hat{d}_i^B \hat{d}_j^B d_i^B d_j^B \\ &= \frac{(\sum_{i\in[n]} \hat{d}_i^B d_i^B)^2 - \sum_{i\in[n]} (\hat{d}_i^B)^2 (d_i^B)^2}{2} - \sum_{\substack{(i,j)\notin E_k^B \\ i\neq j}} \hat{d}_i^B \hat{d}_j^B d_i^B d_j^B \end{aligned}$$

We will rewrite the three-terms as $\Lambda_k^{(B,1)}, \Lambda_k^{(B,2)}$ and $\Lambda_k^{(B,3)}$ respectively. Thus we have

$$\Lambda_k^B = \frac{(\Lambda_k^{(B,1)})^2 - \Lambda_k^{(B,2)}}{8m} - \frac{\Lambda_k^{(B,3)}}{4m} \tag{IV.7}$$

Since our goal is to approximate Ψ_k^B, it would be extremely useful to prove some bounds.

Lemma 7. *For all k, the following holds:*

1. $\Delta_k^B \leq \frac{(2m-2k)d_{max}}{2}$
2. $\Lambda_k^{(B,1)} \leq d_{max}(2m-2k)$
3. $\Lambda_k^B \leq \frac{(2m-2k)^2 d_{max}^2}{8m}$

Proof. For 1. we first try to bound the number of possible blue self-loops. At stage k, there are at most $2m-2k$ half-edges left regardless of color. Now fix an half edge that belongs to vertex i. There are at most $d_{max}-1$ other possible half-edges to pair with to form self loops. Thus total blue self loops is bounded by $\frac{2m-2k(d_{max}-1)}{2}$. Next for the blue multi-edge, following the same argument, we fixed one half edge of vertex i and another half edge of vertex $j \neq i$. Then there are at most $(d_{max}-1)^2$ many ways to form double edges when i is already connected to j. So we upper-bound blue multiedges by $\frac{(2m-2k)(d_{max}-1)^2}{2}$. Using the fact $\Delta_k^B = \Delta_k^{(B,2)} + \Delta_k^{(B,1)}$, we get the bound that we want.

For 2., we have the following chain of inequalities: $\Lambda_k^{(B,1)} \leq d_{max} \sum_{i\in[n]} \hat{d}_i^B \leq d_{max} \sum_{i\in[n]} \hat{d}_i = d_{max}(2m-2k)$

Finally for 3. By definition, $\Lambda_k^B = \sum\limits_{(i,j)\in E_k^B} \hat{d}_i^B \hat{d}_j^B \frac{d_i^B d_j^B}{4m} \leq \frac{d_{max}^2}{4m} \sum\limits_{(i,j)\in E_k^B} \hat{d}_i^B \hat{d}_j^B \leq \frac{d_{max}^2}{4m} \binom{2m-2k}{2}$ □

Now we are ready to begin to compute ψ_k^B. Before we begin, we will use the subscript p_k to denote properties and computation performed involving our approximation model G_{p_k}. Our first tasks is to compute the terms in Λ_k^B and Δ_k^B as defined in Equation IV.7 and Equation IV.5. The computation can be summarized in the following lemma:

Lemma 8. *For every k, the following holds:*

i. $\mathbb{E}_{p_k}[\Delta_k^{(B,1)}] = \frac{(2m-2k)^2}{2}(\frac{\lambda(\bar{d}^B)}{m})$, *where* $\lambda(\bar{d}^B) = \frac{\sum_{i\in[n]} \binom{d_i^B}{2}}{2m}$

ii. $\mathbb{E}_{p_k}[\Delta_k^{(B,2)}] = \frac{(2m-2k)^2}{2}(k \sum\limits_{i\sim j}(d_i^B - 1)(d_j^B - 1)/2m^3)$

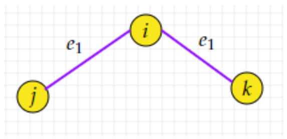
iii. $\mathbb{E}_{p_k}[\Lambda_k^{(B,1)}] = (2m-2k)\frac{\sum_{i\in[n]}(d_i^b)^2}{2m}$

iv. $\mathbb{E}_{p_k}[\Lambda_k^{(B,2)}] = (2m-2k)^2\frac{\sum_{i\in[n]}(d_i^B)^4}{4m^2} + 2k(2m-2k)\frac{\sum_{i\in[n]}(d_i^B)^3}{4m^2}$

v. $\mathbb{E}_{p_k}[\Lambda_k^{(B,3)}] = \frac{(2m-2k)^2}{2}(\frac{k\sum_{i\sim j} d_i^B d_j^B(d_i^B-1)(d_j^B-1)}{2m^3})$

Proof.

i. For i. We first note that in the G_{p_k} model, the probability of an edge to be removed is $q_k = 1-\frac{k}{m}$. Now fix an i and consider any two arbitrary $j,\ k(j\neq k)$ and consider the following:

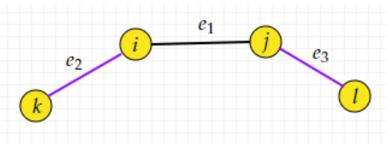


Only when edges e_1 and e_2 is removed, can we formed a self loop. Furthermore since G is simple and and satisfies the input blue sequences, that means all the d_i^B many half-edges of i are paired in a valid way. So there are $\binom{d_i^B}{2}$ many ways for i to be connected to arbitrary $j,\ k$. And for each ways, it is with probability $(q_k)^2$ that the edges are removed in the G_{p_k} model. Thus the expected number of self loops for vertex i in G_{p_k} is $\binom{d_i^B}{2}(q_k)^2$. So now if we sum over all $i\in[n]$, we get the following

$$\begin{aligned}\mathbb{E}_{p_k}[\Delta_k^{(B,1)}] &= \sum_{i\in[n]}\binom{d_i^B}{2}(q_k)^2\\ &= 2m(q_k)^2\sum_{i\in[n]}\frac{\binom{d_i^B}{2}}{2m}\\ &= \frac{(2m-2k)^2}{2}(\frac{\lambda(\bar{d^B})}{m})\end{aligned}$$

where $\lambda(\bar{d^B}) = \frac{\sum_{i\in[n]}\binom{d_i^B}{2}}{2m}$

ii. For ii., the proof is similar. Fix $i,\ j$ and consider any two arbitrary $k,\ l$ and consider the following:



Then only when e_1 remains and e_2, e_3 removed. Can we form a multiedge between $i,\ j$. By the same reasoning as in i., there are $(d_i^B-1)(d_j^B-1)$ ways for arbitrary $k,\ l$ to be connected to i and j via some e_1 and e_2. And with probability $(q_k)^2$ that they are removed from G_{p_k}. Furthermore it is with probability $p_k = \frac{k}{m}$ that the edge $e_1 = (i,j)$ remains. So summing over all $i\sim j$ we get the following:

$$\begin{aligned}\mathbb{E}_{p_k}[\Delta_k^{(B,2)}] &= \sum_{i\sim j}\frac{k}{m}(1-\frac{k}{m})^2(d_i^B-1)(d_j^B-1)\\ &= \frac{(2m-2k)^2}{2}\left(k\sum_{i\sim j}(d_i^B-1)(d_j^B-1)/(2m^3)\right)\end{aligned}$$

Where the last equality comes from rewriting the first equality.

iii. For iii.), by definition $\mathbb{E}_{p_k}[\Lambda_k^{(B,1)}] = \sum_{i\in[n]} d_i^B \mathbb{E}[\hat{d_i^B}]$. By Lemma 3 this is equal to $\sum_{i\in[n]} d_i^B(d_i^B q_k)$, which is exactly what we need.

iv. To prove iv. We need only to evaluate what is $\mathbb{E}_{p_k}[\hat{d_i^B}^2]$ in the model of G_{p_k}. However, this can be rewritten as $\mathbb{E}_{p_k}[\hat{d_i^B}(\hat{d_i^B}-1)+\hat{d_i^B}] = \mathbb{E}_{p_k}[\hat{d_i^B}(\hat{d_i^B}-1)]+\mathbb{E}_{p_k}[\hat{d_i^B}] = 2\mathbb{E}_{p_k}[\binom{\hat{d_i^B}}{2}]+q_k d_i^B$. However $\mathbb{E}_{p_k}[\binom{\hat{d_i^B}}{2}]$ is the expected number of blue self loops for vertex i in G_{p_k}, which we already shown in i. Thus the expression further simplify to $(q_k)^2(d_i^B)(d_i^B-1)+q_k d_i^B = \mathbb{E}_{p_k}[\hat{d_i^B}^2]$. Performing subsition into $\mathbb{E}_{p_k}[\Lambda_k^{(B,2)}]$ will conclude the required result.

v. Finally for v. by definition one only needs to evaluate $\mathbb{E}_{p_k}(\hat{d_i^B}\hat{d_j^B})$ where $i \neq j$ and $(i,j) \notin E_k^B$, where E_k^B is defined in Definition 1. Now since $i \neq j$, $\hat{d_i^B}\hat{d_j^B}$ counts the number of ways to match the residue blue half edges(degrees) of i to the residue blue half edges(degrees) of j to form a blue edge. Since $(i,j) \notin E_k^B$, then there must **already** be an edge (regardless of color) present between i and j. Hence we are counting the number of ways to form multiedges between i and j and the value is simply ii. of Lemma 8. Performing the relevant substitution and we will arrive at the result needed.

□

Remark. *The first two results of Lemma 8 justifies the choice we made in the approximation of self loops and multi-edge in Remark 3. We will further remark that the first two results shows the elegance of the G_{p_k} model in which the dependence on the colors of the edges is weaken significantly. Consider the proof of ii. If we are still heavily dependent on the colors, we would then need to split into two cases of blue and blue each with a different probability of keeping/removing an edge.*

At this point, technically, we are ready to define ψ_k^B. However once again, we wish to rewrite it even further in terms of algebraic bounds for the analysis that lies ahead. To achieve this, we will first prove a simple technical lemma.

Lemma 9. *For all $i \in [n]$ and positive integer r, the following equation holds:*

$$\sum_{i\in[n]} (d_i^B)^r = \sum_{i\sim_B j} ((d_i^B)^{r-1} + (d_j^B)^{r-1}) = O(md_{max}^{r-1})$$

Proof. For the terms in $\sum_{i\sim_B j} ((d_i^B)^{r-1} + (d_j^B)^{r-1})$, we are summing $(d_i^B)^{r-1}$, d_i^B many times. Symmetrically for $(d_j^B)^{r-1}$. Hence

$$\sum_{i\sim_B j} ((d_i^B)^{r-1} + (d_j^B)^{r-1}) = \sum_{i,j\in[n]} (d_i^B)^r + (d_j^B)^r = \sum_{i\in[n]} (d_i^B)^r$$

On the other hand, $\sum_{i\sim_B j} ((d_i^B)^{r-1} + (d_j^B)^{r-1}) \leq \sum_{i\sim_B j} 2(d_{max})^{r-1} = O(md_{max}^{r-1})$ □

We can state formally what we wish to rewrite with the following lemma:

Lemma 10. *For all k, the following holds:*

- $\mathbb{E}_{p_k}[\Lambda_k^{(B,2)}/8m] = \frac{(2m-2k)^2}{2}(O(\frac{d_{max}^3}{m^2}) + O(\frac{d_{max}^2}{m^2}\frac{2k}{2m-2k}))$
- $\mathbb{E}_{p_k}[\Lambda_k^{(B,3)}/4m] = \frac{k(2m-2k)^2}{2}O(\frac{d_{max}^4}{m^3})$

Proof. The proof is simply by using Lemma 9 and Lemma 8, □

Now we will formally define ψ_k^B (and hence ψ_k^R symmetrically) with the following lemma.

Lemma 11. *For every k, the following holds:*

$$\psi_k^B = \mathbb{E}_{p_k}[\Delta_k^{(B,1)}] + \mathbb{E}_{p_k}[\Delta_k^{(B,2)}] + \mathbb{E}_{p_k}[\frac{(\Lambda_k^{(B,1)})^2 - \Lambda_k^{(B,2)}}{8m} - \frac{\Lambda_k^{(B,3)}}{4m}]$$
$$= \frac{(2m-2k)^2}{2}(\frac{\lambda(\bar{d^B})}{m} + \frac{k\sum_{i\sim_B j}(d_i^B-1)(d_j^B-1)}{2m^3} + \frac{(\sum_{i\in[n]}(d_i^B)^2)^2}{16m^3} + t_k)$$

Where $t_k = O(\frac{kd_{max}^4}{m^3} + \frac{kd_{max}^2}{(m-k)m^2})$.

Proof. The proof is simply putting together terms from Lemma 8 and Lemma 10. □

Now that we have finish defining ψ_k^B (and ψ_k^R) we have completed step two of Milestones 1. The next task to complete to to show that our definitions satisfies the relation stated in *Lemma* 6. To do that, we once again need a technical lemma for book keeping in the upcoming computations.

Lemma 12. *For all k and $d_{max} = O(m^{\frac{1}{4}-\tau})$, ψ_k^B is bounded above by $O(\frac{d_{max}^2(2m-2k)^2}{2m})$*

Proof. We will use Lemma 9 on Lemma 11 to get that $\psi_k^B = \frac{(2m-2k)^2}{2}(O(\frac{d_{max}}{2m} + \frac{kd_{max}^2}{2m^2} + \frac{d_{max}^2}{16m} + \frac{kd_{max}^4}{m^3} + \frac{kd_{max}^2}{(m-k)m^2}))$. Now using the fact that $k \le m$ and that $d_{max} = O(m^{\frac{1}{4}-\tau})$, then $O(\frac{d_{max}^2(2m-2k)^2}{2m})$ dominates all the linear terms written above. □

Now we have defined ψ_k^B, with ψ_k^R defined symmetrically. We are now ready to prove Lemma 6 which we shall recall as followings

Lemma. *If $d_{max} = O(m^{\frac{1}{4}-\tau})$, then with high probability, the following relation holds*

$$\prod_{0\le k\le m-1} \frac{1}{\binom{2m_b-2k_b}{2} - \psi_k^B + \binom{2m_r-2k_r}{2} - \psi_k^R} \propto e^{\gamma_G^B+\gamma_G^R+\lambda(\bar{d^B})+\lambda^2(\bar{d^B})+\lambda(\bar{d^R})+\lambda^2(\bar{d^R})+o(1)}$$

Proof. For simplicity we will use χ_G^B to denote $\sum_{i\sim_B j}(d_i^B-1)(d_j^B-1)$. The red case is denoted symmetrically. We begin by first multiplying $\prod_{0\le k\le m-1}\binom{2m_B-2k_B}{2} + \binom{2m_R-2k_R}{2}$ to Lemma 6. Then we can rewrite the new equation as follows:

$$\exp\left[\sum_{k=0}^{m-1}\log\left(1 + \frac{\frac{\lambda(\bar{d^B})+\lambda(\bar{d^R})}{m} + \frac{k(\chi_G^B+\chi_G^R)}{2m^3} + \frac{(\sum_{i\in[n]}(d_i^B)^2)^2+(\sum_{i\in[n]}(d_i^R)^2)^2}{16m^3} + t_k}{(\frac{m_B-k_B}{m-k})^2 + (\frac{m_R-k_R}{m-k})^2 - \frac{1}{2m-2k} - O(\frac{d_{max}^2}{m})}\right)\right]$$

Now the key here is that we now use the fact that $\frac{m_B-k_B}{m-k}$ and $\frac{m_R-k_R}{m-k}$ is a martingale and hence by azuma's inequality, we have that w.h.p they are equal to P_B and P_R respectively. Now using the fact that $P_B^2 + P_R^2 = 1 - 2P_BP_R$ Thus we are able to rewrite the equation as follows:

$$\exp\left[\sum_{k=0}^{m-1}\log\left(1 + \frac{\frac{\lambda(\bar{d^B})+\lambda(\bar{d^R})}{m} + \frac{k(\chi_G^B+\chi_G^R)}{2m^3} + \frac{(\sum_{i\in[n]}(d_i^B)^2)^2+(\sum_{i\in[n]}(d_i^R)^2)^2}{16m^3} + t_k}{1 - \frac{1}{2m-2k} - O(\frac{d_{max}^2}{m}) - 2P_BP_R}\right)\right]$$
$$= \exp\left[\sum_{k=0}^{m-1}\log\left(1 + \left(\frac{1}{1-2P_BP_R}\right)\frac{\frac{\lambda(\bar{d^B})+\lambda(\bar{d^R})}{m} + \frac{k(\chi_G^B+\chi_G^R)}{2m^3} + \frac{(\sum_{i\in[n]}(d_i^B)^2)^2+(\sum_{i\in[n]}(d_i^R)^2)^2}{16m^3} + t_k}{1 - \left(\frac{1}{1-2P_BP_R}\right)\left(\frac{1}{2m-2k}\right) - O(\frac{d_{max}^2}{m})\left(\frac{1}{1-2P_BP_R}\right)}\right)\right]$$

At this point, we observe that $P_BP_R = O(1)$ because m_B and m_R are of same order as m, and hence $1 - O(1) = O(1)$. This help us to further simplify to the following

$$\exp\left[\sum_{k=0}^{m-1}\log\left(1 + O(1)\cdot\frac{\frac{\lambda(\bar{d^B})+\lambda(\bar{d^R})}{m} + \frac{k(\chi_G^B+\chi_G^R)}{2m^3} + \frac{(\sum_{i\in[n]}(d_i^B)^2)^2+(\sum_{i\in[n]}(d_i^R)^2)^2}{16m^3} + t_k}{1 - O(1)\cdot\left(\frac{1}{2m-2k}\right) - O(\frac{d_{max}^2}{m})}\right)\right] \quad \text{(IV.8)}$$

we will make a technical claim to help simplify Equation IV.8.

Claim 1. $\frac{\psi_k^B+\psi_k^R}{(2m-2k)^2} \cdot \left(\frac{O(1)}{2m-2k} + O(\frac{d_{max}^2}{m}) + O\left(\left(\frac{1}{2m-2k} + O(\frac{d_{max}^2}{m})\right)^2\right)\right) = O(\frac{d_{max}^4}{m^2})$

Proof. We first make use the of algebraic bound derived in Lemma 12 to rewrite the equation of the L.H.S of our claim to the following

$$O\left(\frac{d_{max}^2}{m}\right)\left(\frac{O(1)}{2m-2k} + O(\frac{d_{max}^2}{m}) + O\left(\left(\frac{O(1)}{2m-2k} + O(\frac{d_{max}^2}{m})\right)^2\right)\right)$$

Next we expand the squared terms to arrive at

$$O\left(\frac{d_{max}^2}{m}\right)\Bigg(\frac{O(1)}{2m-2k} + O\left(\frac{d_{max}^2}{m}\right) + O\bigg(\frac{O(1)}{(2m-2k)^2} + O\left(\frac{d_{max}^4}{m^2}\right) + O\left(\frac{d_{max}^2}{m(m-k)}\right)\bigg)\Bigg)$$

Now multiplying all the terms above, we arrive finally arrive at

$$O\left(\frac{d_{max}^2}{(2m-2k)m}\right) + O\left(\frac{d_{max}^4}{m^2}\right) + O\left(\frac{d_{max}^2}{m(2m-2k)^2}\right) + O\left(\frac{d_{max}^6}{m^3}\right) + O\left(\frac{d_{max}^4}{m^2(m-k)}\right)$$

Now using the assumption that $d_{max} = O(m^{\frac{1}{4}-\tau})$, we can see that the term $O(\frac{d_{max}^4}{m^2})$ dominates all other terms. Hence we arrive at the conclusion we need. □

Now let us continue from Equation IV.8. Rewrite the numerator as $\frac{\psi_k^B+\psi_k^R}{(2m-2k)^2}$ and by using the algrabraic estimate $\frac{1}{1-x} = 1 + x + O(x^2)$, we see that we can write the terms in the big bracket of Equation IV.8 as

$$\begin{aligned}
&1 + O(1)\cdot\frac{\psi_k^B+\psi_k^R}{(2m-2k)^2}\bigg(1 + \frac{O(1)}{2m-2k} + O\left(\frac{d_{max}^2}{m}\right) \\
&\quad + O\bigg(\left(\frac{O(1)}{2m-2k} + O\Big(\frac{d_{max}^2}{m}\Big)\right)^2\bigg)\bigg) \\
&= 1 + O(1)\frac{\psi_k^B+\psi_k^R}{(2m-2k)^2} + Claim\ 1 \\
&= 1 + O(1)\frac{\psi_k^B+\psi_k^R}{(2m-2k)^2} + O(\frac{d_{max}^4}{m^2})
\end{aligned}$$

We next wish to refine the algebraic bound a little by writing out

$$1 + O(1)\frac{\psi_k^B+\psi_k^R}{(2m-2k)^2} + O\left(\frac{d_{max}^4}{m^2}\right) \tag{IV.9}$$

$$= 1 + \frac{\lambda(\bar{d^B}) + \lambda(\bar{d^R})}{m} + \frac{k(\chi_G^B + \chi_G^R)}{2m^3} + \frac{(\sum_{i\in[n]}(d_i^B)^2)^2 + (\sum_{i\in[n]}(d_i^R)^2)^2}{16m^3} + t_k + O\left(\frac{d_{max}^4}{m^2}\right) \tag{IV.10}$$

$$= 1 + \frac{\lambda(\bar{d^B}) + \lambda(\bar{d^R})}{m} + \frac{k(\chi_G^B + \chi_G^R)}{2m^3} + \frac{(\sum_{i\in[n]}(d_i^B)^2)^2 + (\sum_{i\in[n]}(d_i^R)^2)^2}{16m^3} + O\left(\frac{d_{max}^4}{m^2} + \frac{kd_{max}^2}{(m-k)m^2}\right) \tag{IV.11}$$

Where by Lemma 12, $\frac{\psi_k^B+\psi_k^R}{(2m-2k)^2} = O(\frac{d_{max}^2}{m})$ to conclude that the constant $O(1)$ has no influence in the growth of $\frac{\psi_k^B+\psi_k^R}{(2m-2k)^2}$, so we can ignore it for the rest of the computation. Now we make use of the fact that $\log(1+x) = x - O(x^2)$, then by taking logarithm of Equation IV.9, we arrive at the following:

$$\frac{\lambda(\bar{d^B})+\lambda(\bar{d^R})}{m}+\frac{k(\chi_G^B+\chi_G^R)}{2m^3}+\frac{(\sum\limits_{i\in[n]}(d_i^B)^2)^2+(\sum\limits_{i\in[n]}(d_i^R)^2)^2}{16m^3}+O(\frac{d_{max}^4}{m^2}+\frac{kd_{max}^2}{(m-k)m^2})$$
$$-O(\frac{\lambda(\bar{d^B})+\lambda(\bar{d^R})}{m}+\frac{k(\chi_G^B+\chi_G^R)}{2m^3}+\frac{(\sum\limits_{i\in[n]}(d_i^B)^2)^2+(\sum\limits_{i\in[n]}(d_i^R)^2)^2}{16m^3}$$
$$+O(\frac{d_{max}^4}{m^2}+\frac{kd_{max}^2}{(m-k)m^2}))^2$$
$$=\frac{\lambda(\bar{d^B})+\lambda(\bar{d^R})}{m}+\frac{k(\chi_G^B+\chi_G^R)}{2m^3}+\frac{(\sum\limits_{i\in[n]}(d_i^B)^2)^2+(\sum\limits_{i\in[n]}(d_i^R)^2)^2}{16m^3}+O(\frac{d_{max}^4}{m^2}+\frac{kd_{max}^2}{(m-k)m^2})$$

Where the last equality is due to the highlighted terms is of smaller order than $O(\frac{d_{max}^4}{m^2}+\frac{kd_{max}^2}{(m-k)m^2})$. Now substituting the equation back to Equation IV.8, we arrive at

$$\exp\left[\sum_{k=0}^{m-1}\frac{\lambda(\bar{d^B})+\lambda(\bar{d^R})}{m}+\frac{k(\chi_G^B+\chi_G^R)}{2m^3}+\frac{(\sum\limits_{i\in[n]}(d_i^B)^2)^2+(\sum\limits_{i\in[n]}(d_i^R)^2)^2}{16m^3}+O(\frac{d_{max}^4}{m^2}+\frac{kd_{max}^2}{(m-k)m^2})\right] \tag{IV.12}$$

We will now make another technical claim to aid in the simplification of the equation.

Claim 2. $\sum\limits_{k=0}^{m-1}\frac{k}{m-k}=O(m\log(m))$

Proof. We can rewrite $\sum\limits_{k=0}^{m-1}\frac{k}{m-k}$ as $\sum\limits_{k=1}^{m}\frac{m-k}{k}$. This is then equal to

$$\begin{aligned}\sum_{k=1}^{m}\frac{m-k}{k}&=\sum_{k=1}^{m}(m\frac{1}{k}-1)\\&=mH_m\ -\ m\\&=(m)(H_m-1)\end{aligned}$$

Where H_m is the m-th Harmonic number [8]. Now by a result in [8], $H_m=O(\log(m))\implies m(H_m-1)=O(m\log(m))$. □

Now we will continue from Equation IV.12. By summing over the index k. We arrive at

$$\exp\left[\lambda(\bar{d^B})+\lambda(\bar{d^R})+\frac{m(m-1)(\chi_G^B+\chi_G^R)}{4m^2}+\frac{(\sum\limits_{i\in[n]}(d_i^B)^2)^2+(\sum\limits_{i\in[n]}(d_i^R)^2)^2}{16m^2}\right.$$
$$\left.+O(\frac{d_{max}^4}{m}+\frac{d_{max}^2}{m^2}\cdot(\sum_{k=0}^{m-1}\frac{k}{m-k}))\right]$$

Now we invoke Claim 2 to simplify it the following:

$$\exp\left[\lambda(\bar{d^B})+\lambda(\bar{d^R})+\frac{m(m-1)(\chi_G^B+\chi_G^R)}{4m^3}+\frac{(\sum\limits_{i\in[n]}(d_i^B)^2)^2+(\sum\limits_{i\in[n]}(d_i^R)^2)^2}{16m^2}\right.$$
$$\left.+O(\frac{d_{max}^4}{m}+\frac{d_{max}^2}{m}\log(m))\right]$$
$$=\exp\left[\lambda(\bar{d^B})+\lambda(\bar{d^R})+\frac{(\chi_G^B+\chi_G^R)}{4m}+\frac{(\sum\limits_{i\in[n]}(d_i^B)^2)^2+(\sum\limits_{i\in[n]}(d_i^R)^2)^2}{16m^2}\right.$$
$$\left.-\frac{\chi_G^B+\chi_G^R}{4m^2}+O(\frac{d_{max}^4}{m}+\frac{d_{max}^2}{m}\log(m))\right]$$

Now by definition of χ_G^B (and hence χ_G^R), it is upper bounded by $O(d_{max}^2)$ this means we can arrive at the following

$$\exp\left[\lambda(\bar{d^B}) + \lambda(\bar{d^R}) + \frac{(\chi_G^B + \chi_G^R)}{4m} + \frac{(\sum\limits_{i\in[n]}(d_i^B)^2)^2 + (\sum\limits_{i\in[n]}(d_i^R)^2)^2}{16m^2} \right.$$
$$\left. + O(\frac{d_{max}^4}{m} + \frac{d_{max}^2}{m}\log(m))\right]$$

Finally we will use the fact that $d_{max} = O(m^{\frac{1}{4}-\tau})$ to strength the algebraic bound $O(\frac{d_{max}^4}{m} + \frac{d_{max}^2}{m}\log(m))$ to $o(1)$, thus arriving at

$$\exp\left[\lambda(\bar{d^B}) + \lambda(\bar{d^R}) + \frac{(\chi_G^B + \chi_G^R)}{4m} + \frac{(\sum\limits_{i\in[n]}(d_i^B)^2)^2 + (\sum\limits_{i\in[n]}(d_i^R)^2)^2}{16m^3} + o(1)\right] \tag{IV.13}$$

Finally we will make a final technical claim to arrive at the conclusion we want in Lemma 6.

Claim 3. $\lambda^2(\bar{d^B}) = \frac{1}{4} + \frac{(\sum_{i\in[n]}(d_i^B)^2)^2}{16m^2} - \frac{\sum_{i\sim_B j} d_i + d_j}{4m}$. *The red case is defined symmetrically.*

Proof. By expanding $\lambda(\bar{d^B})$, we see that $\lambda^2(\bar{d^B})$ can be written as

$$\left(\frac{\sum\limits_{i\in[n]}(d_i^B)^2 - d_i^B}{4m}\right)^2 = \left(\frac{\sum\limits_{i\in[n]}(d_i^B)^2}{4m} - \frac{1}{2}\right)^2 = \frac{(\sum\limits_{i\in[n]}(d_i^B)^2)^2}{16m^2} + \frac{1}{4} - \frac{(\sum\limits_{i\in[n]}(d_i^B)^2)}{4m}$$

. By Lemma 9, $\frac{(\sum_{i\in[n]}(d_i^B)^2)}{4m} = \frac{\sum_{i\sim_B j} d_i + d_j}{4m}$ □

Continuing from Equation IV.13, we expand the definition of χ_G^R and χ_G^B to arrive at the following

$$\exp\left[\lambda(\bar{d^B}) + \lambda(\bar{d^R}) - \frac{\sum\limits_{i\sim_B j}(d_i^B + d_j^B)}{4m} + \frac{(\sum\limits_{i\in[n]}(d_i^B)^2)^2}{16m^2} + \frac{1}{4} \right.$$
$$- \frac{\sum\limits_{i\sim_R j}(d_i^R + d_j^R)}{4m} + \frac{(\sum\limits_{i\in[n]}(d_i^R)^2)^2}{16m^2} + \frac{1}{4}$$
$$\left. + \frac{\sum\limits_{i\sim_B j} d_i^B d_j^B}{4m} + \frac{\sum\limits_{i\sim_R j} d_i^R d_j^R}{4m} + o(1)\right]$$

Now by using Claim 3, we have finally arrived at

$$(1 + o(1))\exp\left[\lambda(\bar{d^B}) + \lambda(\bar{d^R}) + \lambda^2(\bar{d^B}) + \lambda^2(\bar{d^R}) + \gamma_G^B + \gamma_G^R\right]$$

Which completes the proof of Lemma 6 □

Chapter V

Analysis Part Three: Concentration results

With the first two part of the analysis, we have achieved step one, two and three of Milestones 1. All that remains is step four, and we will elaborate in details on what the task ahead entails. The goal is to show that for an ordering $N \in S(M)$, $\Psi_k^B(N)$ concentrates around $\psi_k^B(G)$ as defined in Lemma 11, where the same goes for the red case. To put into more formal terms, we have the following theorem:

Theorem 3. *For every $0 \le k \le m-1$, let $\psi_k^B(G)$ be as defined in Lemma 11. Then the following relation is true*

$$\sum_{N \in S(M)} \prod_{k=0}^{m-1} \frac{1}{\binom{2m_B-2k_B}{2} - \Psi_k^B(N) + \binom{2m_R-2k_R}{2} - \Psi_k^R(N)}$$
$$= (1+o(1))m! \prod_{k=0}^{m-1} \frac{1}{\binom{2m_B-2k_B}{2} - \psi_k^B(G) + \binom{2m_R-2k_R}{2} - \psi_k^R(G)}$$

Let us pry a little deeper into Theorem 3 and multiply both side of the equation by $\prod_{k=0}^{m-1} \binom{2m_B-2k_B}{2} - \psi_k^B(G) + \binom{2m_R-2k_R}{2} - \psi_k^R(G)$. Then we divide by $m!$. We would then arrive at the following equation

$$\sum_{N \in S(M)} \frac{1}{m!} f(N) = 1 + o(1) \tag{V.1}$$

Where $f(N) = \prod_{k=0}^{m-1} \frac{\binom{2m_B-2k_B}{2} - \psi_k^B(G) + \binom{2m_R-2k_R}{2} - \psi_k^R(G)}{\binom{2m_B-2k_B}{2} - \Psi_k^B(N) + \binom{2m_R-2k_R}{2} - \Psi_k^R(N)}$.

Now if we recall from Remark 4, we imposed uniform distribution on $S(M)$ and there are precisely $m!$ many orderings in the set $S(M)$. This means to each $f(N)$ we are giving it a weight of $\frac{1}{m!}$. Then Equation V.1 becomes $\mathbb{E}[f(N)] = 1 + o(1)$. Hence we will restate Theorem 3 in terms of the following equivalent lemma

Lemma 13. *For every $0 \le k \le m-1$, let $\psi_k^B(G)$ be as defined in Lemma 11. Then the following relation is true*

$$\mathbb{E}[f(N)] = 1 + o(1)$$

Where $f(N) = \prod_{k=0}^{m-1} \frac{\binom{2m_B-2k_B}{2} - \psi_k^B(G) + \binom{2m_R-2k_R}{2} - \psi_k^R(G)}{\binom{2m_B-2k_B}{2} - \Psi_k^B(N) + \binom{2m_R-2k_R}{2} - \Psi_k^R(N)}$.

Remark 6.

The Proof of Lemma 13 can be achieve by the following steps:

1. *Partition the set $S(M)$ into smaller subsets.*
2. *For each partition, we study the deviation of f.*
3. *Show that which f concentrates in one particular partition.*

As before we will be fixing an arbitrary (satisfactory) G and an arbitrary $N \in S(M)$. Thus, we will be writing $\psi_k^B(G)$ as ψ_k^B and $\Psi_k^B(N)$ as Ψ_k^B. Perhaps as a foreword, the proof of Lemma 13 is rather involved and has several non-intuitive definitions and constructions. Like in the previous parts of the chapters on analysis, we will, as much as possible, focus on the intuition behind rather than the actual full computation.

1 Partitioning $S(M)$

At this moment, it is perhaps worthwhile to see how we can interpret Ψ_k^B (and Ψ_k^N). By the discussion at the start of subsection (b), we can write $\Psi_k^B + \Psi_k^R$ as

$$\Psi_k^B + \Psi_k^R = \Delta_k^B + \Delta_k^R + \Lambda_k^B + \Lambda_k^R$$

Where Δ_k's counts the number of respective unsuitable pairings at stage k and the Λ_k's can be written sub of linear terms involving unsuitable pairings also. That means intuitively, we can view $\Psi_k^B + \Psi_k^N)$ as the sum of unsuitable blue pairings plus the sum of unsuitable red pairings at stage k adjusted accordingly to some weight. This is useful in providing intuition on how to define the partitions using the following steps.

1. In this first step, we consider orderings from $N \in S(M)$ where the the number of unsuitable red and blue pairings (weight-adjusted) does not exceed a **constant** (strictly less than 1) fraction of the number of all available blue plus red pairings. We can achieve this by first defining $S'(M)$ as follows:

$$S'(M) := \left\{ N \in S(M \mid \Psi_k^B(N) + \Psi_k^B(N) \leq (1-\epsilon)\left[\binom{2m_B - 2k_B}{2} + \binom{2m_R - 2k_R}{2}\right] \right.$$
$$\left. : \forall\ 0 < k < m \right\}$$

   Where $0 < \epsilon \leq \frac{1}{3}$. Now our first partition will be $S(M) - S'(M)$. $N \in S(M) - S'(M)$ are orderings that are extremely 'badly behaved' where the number of unsuitable matchings are - for all intends and purposes - unbounded.

2. Now we wish to remove ordering in which $\Psi_k^B(N) - \psi_k^B + \Psi_k^R(N) - \psi_k^R$ deviates by more than a certain function independent of G and $N \in S(M)$. We define $\mathcal{A} \subseteq S'(M)$ as:

$$\mathcal{A} := \left\{ N \in S'(M) \mid \Psi_k^B(N) - \psi_k^B + \Psi_k^R(N) - \psi_k^R > 2T_k\left((\log n)^{1+\delta}\right) \right\}$$

   Where the δ is small (e.g. 0.01) and the function T_k will be defined later on in the chapter. However we will remark at this point that the function T_k is simply a mathematical construct for the computations ahead. In essence, $\mathcal{A}$ captures the notion of $\Psi_k^B + \Psi_k^R$ deviating 'too much' from $\psi_k^B + \psi_k^R$.

3. On this third step, we will define $\mathcal{B} \subseteq S'(M) - \mathcal{A}$. We wish $\mathcal{B}$ to contain orderings in which towards the end of the algorithm, there are still unsuitable pairings. This is reflected in the folllowing definition:

$$\mathcal{B} := \left\{ N \in S'(M) - \mathcal{A} \mid \text{for some } k \text{ with } 2m - 2k \leq (\log n)^{1+2\delta}, \right.$$
$$\left. \Psi_k^B(N) + \Psi_k^R(N) \geq 1 \right\}$$

4. Finally for the last partition, we simply have $\mathcal{C} = S'(M) - (\mathcal{A} \cup \mathcal{B})$

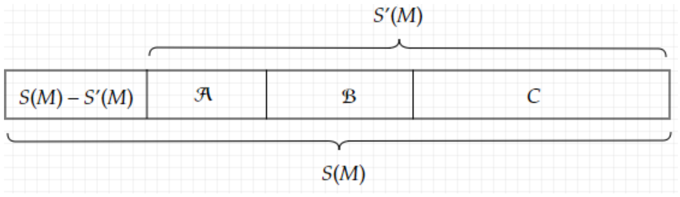


Figure V.1: Partitions of $S(M)$

Now let $1_{\mathcal{A}}$ be the indicator function that returns 1 if $N \in \mathcal{A}$ and returns 0 otherwise. (The indicator functions for the other partitions is defined symmetrically). Our goal for the rest of the chapter is to show the following algebraic bound

$$\mathbb{E}[f(N)1_{\mathcal{A}}] = o(1) \tag{V.2}$$

$$\mathbb{E}[f(N)1_{\mathcal{B}}] = o(1) \tag{V.3}$$

$$\mathbb{E}[f(N)1_{\mathcal{C}}] \leq 1 + o(1) \tag{V.4}$$

$$\mathbb{E}[f(N)1_{\mathcal{C}}] \geq 1 - o(1) \tag{V.5}$$

$$\mathbb{E}[f(N)1_{S(M)-S'(M)}] = o(1) \tag{V.6}$$

Suppose we manage to show all results from Equation V.2 to Equation V.6. Then instead of considering the entire $S(M)$, we need only consider orderings $N \in \mathcal{C} \subseteq S(M)$ instead. Then Lemma 13 follows immediately from Equation V.4 and Equation V.5.

(a) Technical definitions and theorems

The goal of this subsection is to consolidate our ”toolbox” as we introduce notations ,definitions and technical theorems for the upcoming computations and proofs. Perhaps as a foreword, the constructions of this subsection is extremely un-intuitive and readers who are not purist may just take these constructions and theorems for granted and continue on to the next section.

Now the remainder of the thesis δ will be a positive small real number smaller than 0.01. Next, we let $\omega := (\log n)^{\delta}$ and we have the following recursive definition.

Definition 2. *For every $i \in \mathbb{N}$,*

$$\lambda_0 = \omega \log(n)$$
$$\lambda_i = 2\lambda_{i-1} = 2^i \lambda_0, \ i \geq 1$$

Furthermore, let $L \in \mathbb{N}$ such that for some large c, $\lambda_{L-1} < cd_{max}^2 \log n < \lambda_L$.

Hence from now on, We will now only consider $i \in \{0, 1, 2, \ldots, L\}$. Next we will now begin to define T_k when we are defining $\mathcal{A} \subseteq S'(M)$. We first have the following

Definition 3. *Let $q_k = 1 - \frac{k}{m}$ and $p_k = 1 - q_k$. For all $0 \leq k \leq m-1$, let c be some large constant. Then*

$$\beta_k(x) = c\sqrt{x(md_{max}^2 q_k^2 + x^2)(d_{max}^2 q_k + x)} \tag{V.7}$$

$$\eta_k(x) = c\sqrt{x(md_{max}^2 q_k^3 + x^3)(d_{max}^2 q_k^2 + x^2)} \tag{V.8}$$

$$\nu_k = 8md_{max}^2 q_k^3 \tag{V.9}$$

Furthermore, for all $0 \leq k < m$, define $T_k : \mathbb{R} \to \mathbb{R}$ as

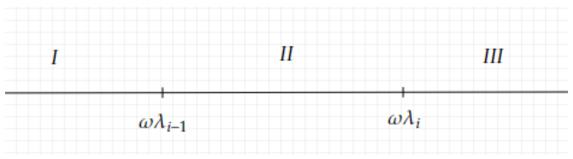
$$T_k(x) \mapsto \begin{cases} 3\beta_k(x) + 2\min(\eta_k(x), \nu_k) & \text{if } 2m - 2k \geq \omega x \\ \frac{x^2}{w} & \text{otherwise} \end{cases}$$

Remark. *One need not be daunted by the un-intuitive constructions. As mentioned at the start of section, these are purely mathematical constructs. Though as we proceed on with the chapter, we will get some insights behind the definitions once we invoked the result by [19].*

Now by the definition of Equation IV.7, we propose a simple but useful inequality that can aid us further on.

Lemma 14. *For all* $0 \leq k < m$ *and* $i \in \{1, \dots, L\}$, $T_k(\lambda_i) \leq 8T_k(\lambda_{i-1})$

Proof. Fix an $i \in \{1, \dots, L\}$, we will show that the claim is true for every positive $2m - 2k < 2m$ on the intervals below.



- **Case I**: We have $2m - 2k < \omega\lambda_{i-1}$ and $2m - 2k < \omega\lambda_i$. By performing by applying T_k to λ_i and λ_{i-1} we get the following chain

$$T_k(\lambda_i) = \frac{\lambda_i^2}{\omega} = \frac{4 \cdot (2^{i-1})^2 \lambda_0^2}{\omega} \leq \frac{8 \cdot (2^{i-1})^2 \lambda_0^2}{\omega} = 8T_k(\lambda_{i-1})$$

- **Case II**: So we have We have $2m - 2k \geq \omega\lambda_{i-1}$ and $2m - 2k < \omega\lambda_i$. Then by direct computation, we see that

$$T_k(\lambda_i) = \frac{\lambda_i^2}{\omega} = \frac{4\lambda_{i-1}^2}{\omega} \leq 8 \cdot 3\beta_k(\lambda_{i-1}) + 8 \cdot 2 \min(\eta_k(\lambda_{i-1}), \nu_k)$$

  Where the inequality holds because η_k and v_k is non negative and $8\beta_k(\lambda_{i-1}) \geq \frac{4\lambda_{i-1}^2}{\omega}$

- **Case III**: So we have We have $2m - 2k \geq \omega\lambda_{i-1}$ and $2m - 2k \geq \omega\lambda_i$. So we are in the first case of Definition 3. We first apply β on the λ_i's, using the fact that $\lambda_i = 2\lambda_{i-1}$.

$$\beta_k(\lambda_i) = c\sqrt{2\lambda_{i-1}(md_{max}^2 q_k^2 + 4\lambda_{i-1}^2)(d_{max}^2 q_k + 2\lambda_{i-1})}$$
$$8\beta_k(\lambda_{i-1}) = c\sqrt{64\lambda_{i-1}(md_{max}^2 q_k^2 + \lambda_{i-1}^2)(d_{max}^2 q_k + \lambda_{i-1})}$$

  Now we make use of the fact that the λ_i's is monotone increasing and the β_k function is non-negative to enable us to compare them without the square root function. Then one can see that

$$2c\lambda_{i-1}(md_{max}^2 q_k^2 + 4\lambda_{i-1}^2)(d_{max}^2 q_k + 2\lambda_{i-1}) \leq 64c\lambda_{i-1}(md_{max}^2 q_k^2 + \lambda_{i-1}^2)(d_{max}^2 q_k + \lambda_{i-1})$$

  So $\beta_k(\lambda_i) \leq 8\beta_k(\lambda_{i-1})$. In fact using the same argument, we can also show that $\eta_k(\lambda_i) \leq 8\eta_k(\lambda_{i-1})$. Then we can conclude the following:

$$\min(\eta_k(\lambda_i), \nu_k) \leq \min(8\eta_k(\lambda_{i-1}), \nu_k) \leq \min(8\eta_k(\lambda_{i-1}), 8\nu_k) = 8\min(\eta_k(\lambda_{i-1}), \nu_k)$$

  Hence $T_k(\lambda_i) \leq 8T_k(\lambda_{i-1})$

□

2 Alternate Partitioning of $S'(M)$

Now we have the required tools to begin proving Equation V.2, Equation V.3, Equation V.4 and Equation V.5. Heuristically speaking, to prove Equation V.2, we sub-partition $\mathcal{A}$ and show that the probability of N landing in each sub-partition bounded, and if N did land in these partitions then $f(N)$ is also bounded by some "nice" function. Hence the first matter at hand is to define the sub-partitions, first consider the following chain of inclusion defined on $S'(M)$

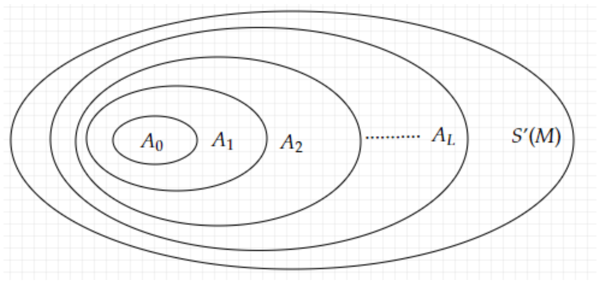


Figure V.2: Chain of inclusion of $A_{i\leq L} \subseteq S'(M)$

Where $A_0 \subseteq A_1 \subseteq A_2 \subseteq \dots A_L \subseteq S'(M)$ is defined as follows:

$$A_i = \left\{N \in S'(M) \mid \Psi_k^B(N) - \psi_k^B(N) + \Psi_k^R(N) - \psi_k^R(N) < T_k(\lambda_i),\ \forall\ 0 \leq k < m\right\} \qquad \text{(V.10)}$$

Furthermore, this will induce a partition defined by $A_i - A_{i-1}$, as seen by Figure V.2 and to partition the entire $S'(M)$ we simply define

$$A_\infty := \left\{N \in S'(M) \mid \exists\ 0 \leq k < m,\ \Psi_k^B(N) - \psi_k^B(N) + \Psi_k^R(N) - \psi_k^R(N) \geq T_k(\lambda_L)\right\} = S'(M) - A_L \qquad \text{(V.11)}$$

It is perhaps worthwhile to discuss on what we are trying to achieve with the partitioning $S'(M)$ again when are have already done it in Figure V.1. Heuristically, one can view the partitions in Figure V.1 as a consequence of Equation V.10 and Equation V.11. In fact we will be writing the partitions $\mathcal{A}, \mathcal{B}$ and $\mathcal{C}$ in terms of the A_i. The idea is that we will be dealing with the A_i's from here on as they are much easier to work with.

(a) Rewriting $\mathcal{A}, \mathcal{B}$, $\mathcal{C}$

By Definition 2, we can write

$$\mathcal{A} = \left\{N \in S'(M) \mid \Psi_k^B(N) - \psi_k^B + \Psi_k^R(N) - \psi_k^R > 2T_k(\lambda_0)\right\}$$
$$= S'(M) - A_0 = A_\infty + \bigsqcup_{i=1}^{L}(A_i - A_{i-1})$$

Since $\mathcal{A}, \mathcal{B}$ and $\mathcal{C}$ are partitions, this means $\mathcal{B} + \mathcal{C} \subseteq A_0$. Hence we will carve up A_0 into smaller subsets $B_i \subset A_0$ so that we can write $\mathcal{B}$ and $\mathcal{C}$ in terms of them. To achieve that, lets consider an integer K such that $2^{K-1} < \omega\lambda_0 \leq 2^K$. Now consider the chain of subsets $B_0 \subseteq B_1 \dots \subseteq B_K = A_0$ defined by

$$B_j := \left\{N \in A_0 \mid \Psi_k^B(N) + \Psi_k^R(N) < 2^j, \forall k \geq (2m - \omega\lambda_0)/2\right\}$$

Claim 4. $\mathcal{C} = B_0$

Proof. Recall by definition $\mathcal{C} = S'(M) - (\mathcal{A} \sqcup \mathcal{B})$. Pick any $N \in B_0$, then $N \notin A$ because $A = A_\infty + \bigcup_{i=1}^{L} A_i - A_{i-1}$. At the same time $N \notin \mathcal{B}$ as well because

$$B_0 := \left\{N \in A_0 \mid \Psi_k^B(N) + \Psi_k^R(N) < 1, \forall k \geq (2m - \omega\lambda_0)/2\right\}$$

Hence N has the negation to the conditions defined on $\mathcal{B}$, thus $B_0 \subseteq \mathcal{C}$. In fact, by definition B_0 is precisely $\mathcal{B}^c$. So if we pick any $N \in \mathcal{C}$, then $N \notin \mathcal{B}$ because $\mathcal{B}$ is a partition and thus $N \in \mathcal{B}^c = B_0 \implies C \subseteq B_0$. □

Corollary 1. $\mathcal{B} = \bigsqcup_{j=1}^{K} B_j - B_{j-1}$

Proof. If we refer to Figure V.1, we see that $\mathcal{C} + \mathcal{B} = S'(M) - \mathcal{A} = A_0$. This means $\mathcal{B} = A_0 - \mathcal{C} = A_0 - B_0 = \bigsqcup_{j=1}^{K} B_j - B_{j-1}$. □

Once again for the purist, we will address the some issues involving the construction of the A_i's and the B_i's. Let us focus on A_i first, after we fix k then inclusion chain is well defined because of the monotone increasing nature of T_k. The next issue to address is - why is it enough to just consider the inclusion chain of length K. Note that a priori we might need an inclusion chain of length greater than K to cover the entirety of A_0. To answer this we first fix any $N \in A_0$, then by definition for every $0 \leq k < m$, we have

$$\Psi_k^B(N) - \psi_k^B + \Psi_k^R(N) - \psi_k^R < T_k(\lambda_0)$$

Now if we consider $2m - 2k < \omega\lambda_0 \iff k > m - \omega\lambda_0$, then by Definition 3 we have

$$\Psi_k^B(N) - \psi_k^B + \Psi_k^R(N) - \psi_k^R < (\log n)^2 \iff \Psi_k^B(N) + \Psi_k^R(N) < (\log n)^2 + \psi_k^B + \psi_k^R$$

Claim 5. *For all $k > m - \omega\lambda_0$, $\psi_k^B + \psi_k^R = o(1)$*

Proof. By Lemma 12, we have $\psi_k^B + \psi_k^R = O(\frac{d_{max}^2(2m-2k)^2}{2m})$. Since $2m - 2k < \omega\lambda_0 \iff k > m - \omega\lambda_0$, we have $\psi_k^B + \psi_k^R = \omega^2\lambda_0^2 \cdot O(\frac{d_{max}^2}{2m}) = O(\frac{d_{max}^2}{2m})$ since $\omega^2\lambda_0^2$ is independent of m. Now we make use of the fact the $d_{max} = O(m^{\frac{1}{4}-\tau})$ to conclude that $\psi_k^B + \psi_k^R = o(1)$. $\square$

So by using the above claim we have the following inequalities

$$\begin{aligned} &\Psi_k^B(N) + \Psi_k^R(N) < (\log n)^2 + \psi_k^B + \psi_k^R \\ \iff &\Psi_k^B(N) + \Psi_k^R(N) < (\log n)^2 + o(1) \implies \Psi_k^B(N) + \Psi_k^R(N) < (\log n)^2 + 1 \end{aligned}$$

Now we use the fact that for every real number x there exist integer K such $K - 1 \leq x < K$. So for $\log_2((\log n)^2 + 1)$ there exist integer K such that

$$K - 1 < \log_2((\log n)^2 + 1) \leq K \implies 2^{K-1} \leq (\log n)^2 + 1 < 2^K$$

Hence we need only consider the K-length inclusion chain as defined above.

(b) Bounding $\mathbb{E}[f(N)]$ on the partitions

Since we have rewritten $\mathcal{A}, \mathcal{B}$ and $\mathcal{C}$, it is perhaps worthwhile to update the diagram of Figure V.1 to the following.

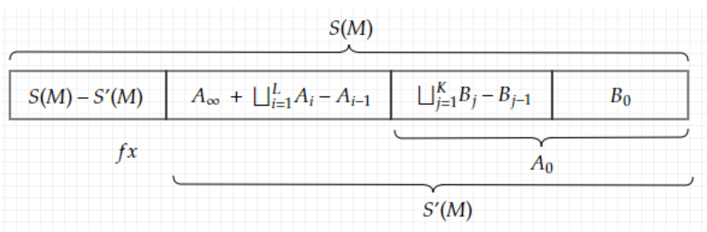


Figure V.3: New partitions of $S(M)$

Now with respect to each of the partitions, we have the following lemma.

Lemma 15. *For all $1 \leq i \leq L$,*

(a) $\mathbb{P}(N \in A_i - A_{i-1}) \leq e^{-\Omega(\lambda_i)}$

(b) For every $N \in A_i - A_{i-1}$, $f(N) \leq e^{o(\lambda_i)}$

Lemma 16. *For a large enough constant c, the following holds*

(a) $\mathbb{P}(N \in A_\infty) \leq e^{-cd_{max}^2 \log n}$

(b) For every $N \in A_\infty$, $f(N) \leq e^{4d_{max}^2 \log n}$

Lemma 17. *For all $1 \leq j \leq K$, the following holds* *(a)* $\mathbb{P}(N \in B_j - B_{j-1}) \leq e^{-\Omega(2^{j/2}\log n)}$

(b) For all $N \in B_j - B_{j-1},\ f(N) \leq e^{O(2^{3j/4})}$

Lemma 18. *For all $N \in \mathcal{C}$, we have $f(N) \leq 1 \pm o(1)$*

Now the above four lemmas allow us to show Equation V.2,Equation V.3, Equation V.4 and Equation V.5, which we will state the the following corollaries with proof.

Corollary 2. *Assuming Lemma 15 and Lemma 16, then Equation V.2 holds.*

Proof. For a fixed N, We rewrite Equation V.2 as

$$\begin{aligned} \textit{Equation V.2} &= \sum_{i=1}^{L} \mathbb{E}[f(N)1_{A_i - A_{i-1}}] + \mathbb{E}[f(N)1_{A_\infty}] \\ &= \sum_{i=1}^{L} \mathbb{P}(N \in A_i - A_{i-1}) \cdot f(N) + \mathbb{P}(N \in A_\infty) \cdot f(N) \end{aligned}$$

Now we invoke Lemma 15 and Lemma 16 to see that

$$\begin{aligned} &\sum_{i=1}^{L} \mathbb{P}(N \in A_i - A_{i-1}) \cdot f(N) + \mathbb{P}(N \in A_\infty) \cdot f(N) \\ &\qquad \leq \sum_{i=1}^{L} e^{-\Omega(\lambda_i)+o(\lambda_i)} + e^{-cd_{max}^2 \log n + 4d_{max}^2 \log n} \end{aligned}$$

Now it suffices to check the the individual summand is $o(1)$. First we have

$$e^{-\Omega(\lambda_i)+o(\lambda_i)} = \frac{e^{o(\lambda_i)}}{e^{\Omega(\lambda_i)}}$$

Where the numerator grows **strictly** slower than λ_i and the denominator grows at least as fast as $\lambda_i \implies$ the fraction is of order $o(1)$. Now

$$e^{-cd_{max}^2 \log n + 4d_{max}^2 \log n} = \frac{1}{e^{(c-4)(d_{max}^2 \log n)}}$$

Now choose any large enough $c \gg 4$ and we see that the above is also of order $o(1)$. □

Corollary 3. *Assuming Lemma 17, Equation V.3 is true.*

Proof. Once again for a fixed N, By rewriting Equation V.3, we now have

$$\sum_{j=1}^{K} \mathbb{P}(N \in B_j - B_{j-1}) \cdot f(N) \leq \sum_{j=1}^{K} \frac{e^{O(2^{3j/4})}}{e^{\Omega(2^{j/2}\log n)}}$$

Since $j \leq K$ and by definition of K, we can deduce that $2^{K/4} \ll \log n$. Then we can conclude that $2^{3j/4} \ll 2^{j/2}\log n)$. Hence the equation above is of $o(1)$. □

Corollary 4. *Assuming Lemma 18, then Equation V.4 and Equation V.5 holds.*

Proof. This is because if the actual value is bounded, then the expectation follows the same bound as well. □

3 Research limitation

Regrettably at this moment due to time constraint, we are unable to prove the above lemmas and Equation V.6. In essence we feel that the proof should be somewhat symmetric to that of the single color case in [13]. However as our results so far is a generalization of the work in [13], hence it is likely that some aspect of the proof discussed in [13] will also need to be changed. Hence, we feel that it is worthwhile as a future endeavor to delve into the technicality and provide a thorough treatments to proving the above lemmas.

Chapter VI

Conclusion

In this thesis, we investigated how one might generalize the algorithm by [13] to two colors. The main contribution of this thesis is that, we proposed to use a biased coin flip to generate a m-length sequence of blue and red edges to decide which edge is to be formed at the current k-th iteration of the algorithm. Using the notion of a martingale, we managed to arrive at the choice of the bias, P_k^B and P_k^R for the coin flip. Our first main result, and perhaps also the most crucial one in this thesis is that P_k^B and P_k^R can be defined independent of k.

The second key result that we arrived at with this thesis is with the G_{p_k} model - a model that we used to approximate the state of our algorithm at stage k. Due to our choice of P_B^k and P_R^k, we are able view the edges as independent identically distributed random variable regardless of their colors. This is extremely interesting and vital in future analysis because firstly we see that the dependence between the blue and red edges is gone in the G_{p_k} model. Furthermore we see that the i.i.d nature of the edges allow us to disregard colors and orderings, reducing many parts of our analysis to a single color case. Ultimately, it is this that led us to the final main result of our thesis in which we show that in the G_{p_k} model, our algorithm does in fact generate a satisfactory graph G uniformly at random.

Remark. *We would also like to discuss a little on the time complexity of the algorithm. One can see that if we are not concerned with whether the output is successful, then the complexity can be no worse than perform the algorithm by Bayati[13] twice. And it has been shown that Bayati's algorithm has a runtime complexity of* $O(md_{max})$

Future work

We believe there are many interesting prospective future work directions besides from what we discussed in the research limitations.

I. The further generalization from 2-color to arbitrary n-color. We feel that this is perhaps the closest next step to our work in this thesis as a lot of the heuristics remains largely the same.

II. In this thesis, the key modification we proposed in our algorithm is the use of a coin-flip to generate a random sequence of blue and red edges. However, we feel that there might be a deterministic sequence that still preserves the uniformity of the sampling, so long as it respects the ratio of the blue (or red) edges.

The application of our algorithm is quite diverse as (random) graphs is a very natural tool for modelling and understanding problems at both application and abstract level. Perhaps one of the most notable example would be in the area of hypothesis testing and economy where human interactions need to be modeled; the colors can be used to represent social status, race, age etc. However other prospective application can be found in the areas of discrete tomography.
As a final word, we are confident and excited over the prospect that our algorithm can find its use by professionals in different fields and we feel elated that our effort, however small, plays a part in their work.

Bibliography

[1] Rao A Ramachandra, Jana Rabindranath, and Bandyopadhyay Suraj. “A markov chain monte carlo method for generating random (0, 1)-matrices with given marginals”. In: *Sankhya: The Indian Journal of Statistics, Series A, pages 225–242* (1996).

[2] Arman Andrii, Pu Gao, and Wormald Nicholas. “Fast uniform generation of random graphs with given degree sequences”. In: *2019 IEEE 60th Annual Symposium on Foundations of Computer Science (FOCS), pages 1371–1379. IEEE* (2019).

[3] Angelika, Steger, and Nicholas. C. Wormald. “Generating random regular graphs quickly”. In: *Combinatorics, Probability and Computing, 8(4):377–396* (1999).

[4] Bollobás Béla. “A probabilistic proof of an asymptotic formula for the number of labelled regular graphs”. In: *European Journal of Combinatorics* (1980).

[5] Williams. David. “Probability with Martingales”. In: *Cambridge University Press* (1991).

[6] Tinhofer Gottfried. “Generating Graphs Uniformly at Random”. In: *Computing Supplementum, vol 7. Springer, Vienna* (1990).

[7] Femke van Ieperen. “Percolation in random directed graphs with an arbitrary degree distribution”. In: *Utrecht University* (2020).

[8] Havil. J. “Gamma: Exploring Euler’s Constant”. In: *Princeton, NJ: Princeton University Press* (2003).

[9] Azuma. K. “Weighted Sums of Certain Dependent Random Variables”. In: *Tôhoku Mathematical Journal* (1967).

[10] Aigner Martin and Eberhard Triesch. “Realizability and uniqueness in graphs”. In: *Freie Universität Berlin* (1993).

[11] Müller-Hannemann Matthias and Berger Annabell. “Uniform sampling of digraphs with a fixed degree sequence”. In: *In International Workshop on Graph-Theoretic Concepts in Computer Science, pages 220–231. Springer* (2010).

[12] Kang Mihyun. “Random Graphs: Theory and Applications from Nature to Society to the Brain”. In: *Technische Universitat Graz, Internat. Math. Nachrichten Nr. 227 (2014), 1–24* (2014).

[13] Bayati Mohsen, Kim Jeong Han, and Saberi Amin. “A sequential algorithm for generating random graphs”. In: *Springer Science+Business Media, LLC* (2009).

[14] Angel Omer, Hofstad Remco van der, and Holmgren Cecilia. “Limit laws for self-loops and multiple edges in the con guration model”. In: *arXiv:1603.07172v2 [math.PR] 2 Feb 2017* (2017).

[15] Erdős Paul and Rényi Alfréd. “On Random Graphs I”. In: *Publ. Math. Debrecen 6, p. 290–297* (1959).

[16] L. Erdős Péter, Greenhill Catherine, and Mezei Tamás Róbert. “The mixing time of the switch markov chains: a unified approach”. In: *Cornell University, arXiv:1903.06600* (2019).

[17] Gao Pu and Greenhill Catherine. “Mixing time of the switch markov chain and stable degree sequences”. In: *Discrete Applied Mathematics, 291:143–162* (2021).

[18] A. Steger and N. C. Wormald. “Generating random regular graphs quickly”. In: *University of Melbourne, Technische Universitat Munchen* ().

[19] Vu. V.H. “Concentration of non-lipschitz functions and applications”. In: *Random Structures and Algorithms* (2002).