



MONTCLAIR STATE
UNIVERSITY

Montclair State University
**Montclair State University Digital
Commons**

Theses, Dissertations and Culminating Projects

5-2023

The Full Degree Spanning Tree Problem

Sarah Acquaviva

Follow this and additional works at: <https://digitalcommons.montclair.edu/etd>



Part of the [Mathematics Commons](#)

Abstract

Given a graph G , we study the problem of finding a spanning tree T that maximizes the number of vertices of full degree; that is, the number of vertices whose degree in T equals its degree in G . We prove a few general bounds and then analyze this parameter on various classes of graphs including grid graphs, hypercubes, and random regular graphs. We also explore a related problem that focuses on maximizing the number of leaves in a spanning tree of a graph.

MONTCLAIR STATE UNIVERSITY

The Full Degree Spanning Tree Problem

by

Sarah Acquaviva

A Master's Thesis Submitted to the Faculty of

Montclair State University

In Partial Fulfillment of the Requirements

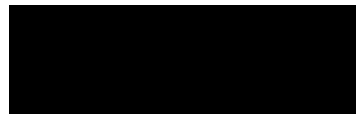
For the Degree of

Master of Science

May 2023

Thesis Committee:

College of Science and Mathematics



Dr. Deepak Bala, Thesis Sponsor

Department of Mathematics



Dr. Jonathan Cutler, Committee Member



Dr. Ashwin Vaidya, Committee Member

The Full Degree Spanning Tree Problem

A THESIS

Submitted in partial fulfillment of the requirements
for the degree of Master of Science

by

Sarah Acquaviva

Montclair State University

Montclair, NJ

2023

Copyright © 2023 by *Sarah Acquaviva*. All rights reserved.

Acknowledgements

I want to start by thanking my advisor, Dr. Deepak Bal. This project would not have been possible without him, not only because he was instrumental in advising it, but also because he sparked my interest in this subset of mathematics during his undergraduate combinatorics and graph theory course. I was fortunate to work with him during my time as both an undergraduate and graduate student at Montclair State, so there's a lot that I could say here, but I'll keep it short: Dr. Bal introduced me to graph theory, mathematics research, and the game of Set, all of which I am very grateful for.

Additionally, I want to thank Dr. Jonathan Cutler and Dr. Ashwin Vaidya for serving on my thesis committee. I appreciate all of the advice that they have shared with me throughout my graduate studies at MSU. Dr. Cutler's graduate graph theory course helped me connect the dots (pun intended) while writing the foundational proofs in this thesis, and I had the opportunity to travel with Dr. Vaidya to a conference where I presented some of the research contained in the following pages.

I am grateful to all members of the Department of Mathematics for their support in the form of travel funds, teaching assistantships, fun field trips, and quality classroom experiences.

Last but not least, thank you to my dog, my family, and my friends for their continuous support.

Contents

Acknowledgements	ii
List of Figures	iv
List of Tables	v
1 Introduction	1
1.1 Preliminaries	1
1.2 Background and Previous Results	6
2 Basic Graphs and General Bounds	8
2.1 Basic Graphs	8
2.2 General Bounds	9
3 Grid Graphs	12
4 Hypercubes	19
5 Random Regular Graphs	24
5.1 Setting up the Analysis of Algorithm 1	28
5.2 Expected One Step Changes	30
5.2.1 Phase 1	31
5.2.2 Phase 2	32
5.3 Trajectories from Differential Equations	33
5.4 Discussion of $r = 3, 4$	36
5.5 Further Directions	37
6 The Many Leaves Spanning Tree Problem	39
6.1 Setting up the Analysis of Algorithm 2	42
6.2 Expected One Step Changes	43
6.3 Trajectories from Differential Equations	45
6.4 Discussion of $r = 3, 4$	47
6.5 Further Directions	48
Bibliography	48

List of Figures

1.1	Example of a spanning tree on a graph with 12 vertices	2
1.2	A graph G with spanning trees T_1 and T_2 . T_1 has $\varphi(G)$ full degree vertices.	6
2.1	Spanning trees with φ full degree vertices on T_7 , C_4 , and K_4	9
3.1	The 3×4 grid graph $G(3, 4)$	12
3.2	Spanning tree with φ full degree vertices on $G(2, 6)$	14
3.3	Spanning tree with φ full degree vertices on $G(3, 7)$	17
3.4	Spanning tree portion with $\varphi(G)$ full degree vertices on $G(m, n)$, $m, n \rightarrow \infty$	18
4.1	Spanning tree that produces $\varphi(Q_3) = 2$ full degree vertices	23
4.2	Spanning tree that produces $\varphi(Q_4) = 4$ full degree vertices	23
5.1	Selecting $v \in L$ and exposing its neighbors	30
5.2	Trajectories for $r = 3$	36
5.3	Trajectories for $r = 4$	36
6.1	Trajectories for $r = 3$	47
6.2	Trajectories for $r = 4$	47

List of Tables

4.1	Number of full degree vertices in Q_k obtained via the construction described below. Values shown in bold are optimal; all others are lower bounds.	20
5.1	Values of f_r, u_r such that w.h.p. $f_r n \leq \varphi(G(n, r)) \leq u_r n$. The bounds f_r come from Algorithm 1 and $u_r = 1/(r - 1)$, deterministically from Theorem 2.2.3.	28
6.1	Values of l_r, u_r such that w.h.p. $l_r n \leq \lambda(G(n, r)) \leq u_r n$. The bounds l_r come from Algorithm 1 and u_r deterministically from Proposition 6.0.1. The bounds λ_{dm} are from [10].	42

Chapter 1

Introduction

This chapter will introduce a few basic graph theory concepts related to the full degree spanning tree problem, as well as provide some background information on the problem.

1.1 Preliminaries

We begin by briefly defining some graph theory terminology that is essential to the results in this thesis. A *graph* G is an ordered pair $G = (V, E)$ where V is a non-empty, finite set and $E \subseteq \binom{V}{2}$, where $\binom{V}{2} = \{X \subseteq V : |X| = 2\}$. $V = V(G)$ is called the *vertex set*, the elements of which are called *vertices*. $E = E(G)$ is called the *edge set*, the elements of which are called *edges*. A *subgraph* of a graph G is a graph H such that $V(H) \subseteq V(G)$ and $E(H) \subseteq E(G)$, and a *spanning subgraph* of G is a subgraph with vertex set $V(G)$.

For $u, v \in V$, the edge between u and v is denoted uv . We say that u and v are *adjacent* if $uv \in E(G)$. If $e = uv$, then the edge e is *incident* to the vertices u and v . We will be focusing on *simple graphs*, in which no vertex is adjacent to itself (i.e.,

the graph has no *loops*) and there is only one edge between adjacent vertices (i.e., the graph has no *multiple edges*). The *degree* of a vertex v in a graph G , denoted $d(v)$ or $d_G(v)$, is the number of edges incident to v . Two adjacent vertices are called *neighbors*. The *neighborhood* of v is $N(v) = \{u \in V(G) : uv \in E(G)\}$. Note that $|N(v)| = d(v)$.

Given a graph G and a vertex $v \in V(G)$, $G - v$ denotes the graph with vertex set $V(G) \setminus \{v\}$ and edge set $E(G) \setminus \{uv : u \in N_G(v)\}$. Similarly, given an edge $e \in E(G)$, $G - e$ denotes the graph with vertex set $V(G)$ and edge set $E(G) \setminus \{e\}$. In general, if we remove a set of vertices or a set of edges S from G , we are left with $G - S$.

A *path* is a list $v_0, e_1, v_1, e_2, v_2, \dots, e_k, v_k$ of vertices and edges with no repeated vertices, where $e_i = v_{i-1}v_i$ for $i = 1, 2, \dots, k$. In this definition, v_0 and v_k are called *endpoints*. The *length* of a path is the number of edges it contains. A graph G is *connected* if for each pair of vertices $u, v \in V(G)$ there is a uv -path (i.e., a path with endpoints u and v) in G . The *components* of a graph are its maximal connected subgraphs. A *cycle* is a closed path (i.e., a path where the endpoints are the same vertex) and a graph is *acyclic* if it contains no cycle.

Of extreme pertinence to this thesis, a *forest* is an acyclic graph, a *tree* is a connected acyclic graph, and a *spanning tree* of G is a spanning connected acyclic subgraph of G . A *leaf* is a vertex of degree one. Figure 1.1 shows an example of a spanning tree on a graph, where tree edges are represented by solid lines and non-tree edges are represented by dashed lines.

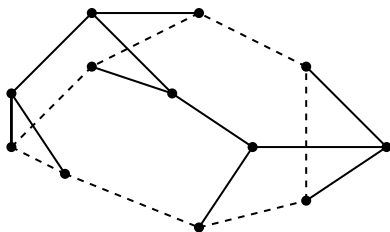


Figure 1.1: Example of a spanning tree on a graph with 12 vertices

Lemmas 1.1.1, 1.1.4, and 1.1.5, which follow, characterize some important properties of trees and serve as a basic foundation for the work in this thesis.

Lemma 1.1.1. *Every tree with $n \geq 2$ vertices has at least 2 leaves. Deleting a leaf from a tree on n vertices yields a tree on $n - 1$ vertices.*

Proof. A connected graph on at least 2 vertices contains an edge. In an acyclic graph, the endpoints of any maximal path (i.e., a path that cannot be made longer by adding another edge and vertex) must be leaves. Let v be a leaf in a tree T and let $T' = T - v$. Then T' is acyclic since it is a subgraph of T . A vertex of degree 1 cannot lie on a path between other vertices, so for $u, w \in V(T')$, every uw -path in T is still in T' , which means that T' is connected. Thus, T' is a tree on $n - 1$ vertices. \square

The proofs of Lemmas 1.1.4 and 1.1.5 make use of Lemma 1.1.3, which requires some extra definitions and a proposition. A *cut-edge* is an edge whose deletion increases the number of components in a graph. A *walk* is a list $v_0, e_1, v_1, e_2, v_2, \dots, e_k, v_k$ of vertices and edges where $e_i = v_{i-1}v_i$ for $i = 1, 2, \dots, k$. Note that a path is a restricted walk, as paths cannot repeat vertices but walks can.

Proposition 1.1.2. *Every uv -walk contains a uv -path.*

Proof. Let W be a uv -walk and consider the shortest uv -walk W' contained in W . If W' has a repeated vertex, then we can delete all edges between occurrences of the repeated vertex. This yields a shorter uv -walk, which contradicts the minimality of W' . Thus, W' is a uv -path. \square

Lemma 1.1.3. *An edge is a cut-edge if and only if it belongs to no cycle.*

Proof. Let $e = xy$ be an edge in G and let H be the component of G containing e . The removal of e affects no component other than H , so we'll prove that $H - e$ is connected iff e belongs to a cycle in H .

Suppose $H - e$ is connected. Then $H - e$ contains an xy -path. This path, along with e , forms a cycle in H . Thus, e belongs to a cycle in H .

Now suppose e belongs to a cycle C in H . Let u, v be any two vertices in $H - e$. We know H is connected by definition, so there must be a uv -path P in H . If P does not contain e , then we are done. If P contains e , then there is a ux -path in P , a xy -path in C , and a yv -path in P . Thus, stitching these together, we get a uv -walk in $H - e$. By Proposition 1.1.2, this uv -walk in $H - e$ contains a uv -path in $H - e$. Thus, $H - e$ is connected. \square

We now have enough information to state and prove Lemmas 1.1.4 and 1.1.5.

Lemma 1.1.4. *For a graph G on n vertices, the following are equivalent:*

- (i) G is connected and acyclic,
- (ii) G is connected and has $n - 1$ edges,
- (iii) G has $n - 1$ edges and is acyclic,
- (iv) For any $u, v \in V(G)$, there exists exactly 1 uv -path.

Proof. First we will show (i) \Rightarrow (ii) and (iii), so suppose G is connected and acyclic. If $n = 1$, $|E(G)| = 0 = n - 1$. Let G have $n > 1$ vertices and be connected and acyclic. By Lemma 1.1.1, G has a leaf x whose removal creates $G - x$, a tree on $n - 1$ vertices. So $G - x$ has $n - 2$ edges. Thus, G has $n - 2 + 1 = n - 1$ edges.

Now we will show (ii) \Rightarrow (i) and (iii), so suppose G is connected and has $n - 1$ edges. If G has a cycle, then we can remove an edge from the cycle and the remaining graph is connected by Lemma 1.1.3. Delete edges that lie on cycles from G until we are left with an acyclic graph G' . Since we only removed edges on cycles, G' is still

connected. Thus, we know from the first part of this proof that G' has $n - 1$ edges, so we did not actually remove any edges from G to obtain G' . Therefore, G is acyclic.

Now we will show $(iii) \Rightarrow (i)$ and (ii) , so suppose G has $n - 1$ edges and is acyclic. Let G_1, \dots, G_k be the components of G . Since every vertex of G appears in some G_i , we know

$$\sum_{i=1}^k |V(G_i)| = n.$$

Each G_i is connected by definition and, since G is acyclic, each G_i is acyclic. Thus, we know from the first part of this proof that each G_i has $|V(G_i)| - 1$ edges. Therefore,

$$n - 1 = |E(G)| = \sum_{i=1}^k |E(G_i)| = \sum_{i=1}^k (|V(G_i)| - 1) = \sum_{i=1}^k |V(G_i)| - \sum_{i=1}^k 1 = n - k.$$

So we have $k = 1$, i.e., G has one component. Thus, G is connected.

Now we will show $(i) \Rightarrow (iv)$, so suppose G is connected and acyclic. Then there is at least one path between every pair of vertices. If some pair is connected by at least two paths, let P, Q be the pair of such paths whose total length is shortest. If P and Q share an internal vertex, this contradicts the minimality of P, Q . But if there are no shared internal vertices, P and Q form a cycle, which contradicts the fact that G is acyclic. Thus, there is exactly one path between every pair of vertices in G .

Lastly, we will show $(iv) \Rightarrow (i)$, so suppose for any $u, v \in V(G)$, there exists exactly 1 uv -path. Then G is connected by definition. If G contained a cycle, then a pair of vertices would have at least two paths between them. So G is acyclic. \square

Lemma 1.1.5. *Every connected graph contains a spanning tree.*

Proof. Let G be a connected graph. If G is acyclic, then we are done. If G contains cycles, remove edges from the cycles until we have an acyclic graph. The remaining graph is connected by Lemma 1.1.3. Thus, this is a spanning tree of G . \square

We have now discussed the preliminaries that are the most essential to the remainder of this thesis. Any other necessary terminology will be defined as the need arises. In the next section, we will turn our attention to the full degree spanning tree problem and examine its background and a few previous results.

1.2 Background and Previous Results

A vertex v of the graph G is of *full degree* in a spanning tree T if $d_T(v) = d_G(v)$. Let $\varphi(G)$ be the maximum number of full degree vertices in any spanning tree of G . For example, in Figure 1.2 we see two spanning trees of the same graph G , where tree edges are represented by solid lines, non-tree edges are represented by dashed lines, and full degree vertices are solid. In this figure, T_1 has the maximum number of full degree vertices, so $\varphi(G) = 3$, while T_2 is a spanning tree that does not contain the maximum number of full degree vertices.

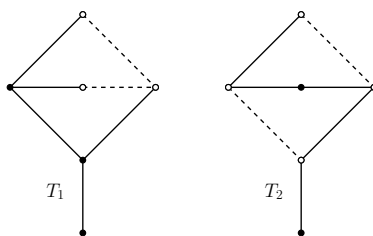


Figure 1.2: A graph G with spanning trees T_1 and T_2 . T_1 has $\varphi(G)$ full degree vertices.

The main goal of this thesis is to determine $\varphi(G)$ for different types of graphs. This problem is variously referred to as the *full degree spanning tree* (FDST) problem [4] or the *degree preserving spanning tree* (DPST) problem [5]. It was first studied by Lewinter [17], who found that if there exist spanning trees with k and l full degree vertices and $k < l$, then there exists a spanning tree with exactly m full degree vertices for every $k < m < l$. Pothof and Schut [19] suggested that the problem could be approximately solved by finding a minimum weight spanning tree of a graph

G , where the weight of each edge $uv \in E(G)$ is the sum of the degrees of u and v . However, they note that this heuristic does not work well in all graphs. Broersma et al. [5] and Bhatia et al. [4] independently obtained results on linear and almost linear time approximation algorithms for the FDST problem on certain types of graphs. Other previous results will be mentioned in the forthcoming chapters, but for the most part, the FDST problem has been studied from a complexity theoretic and algorithmic viewpoint (see [12, 15, 18] for some fairly recent work) and is, in general, NP-hard.

The FDST problem has the following nice application to water distribution networks (for more details see [4, 5, 19]). A *co-tree* in a graph G is a set of edges whose removal leaves a spanning tree in G . To measure the flow across each edge in a water network, one may place flow meters on the edges of a co-tree and then infer the flow on the edges of the tree. Alternatively, one may place pressure meters (which are apparently much less expensive) on the endpoints of an edge and compute the flow across that edge. In this set up, we would only need to place pressure meters on vertices that are incident to co-tree edges. In other words, we would *not* need to place a pressure meter on any vertex of full degree in the tree, and so maximizing the number of such vertices minimizes the cost of pressure meters. Further, as mentioned in [5], the FDST problem can be thought of as finding a spanning tree of a damaged network that contains as many undamaged vertices as possible, which is a problem that could certainly have a wide variety of other applications.

In the following chapters, we will explore the FDST problem on various classes of graphs. Chapter 2 provides relatively simple bounds on $\varphi(G)$ for general graphs. Chapter 3 discusses this parameter in grids and Chapter 4 does so for hypercubes. In Chapter 5, we provide and analyze an algorithm that determines lower bounds for $\varphi(G)$ on random regular graphs. We explore a related problem about maximizing the number of leaves in a spanning tree of a random regular graph in Chapter 6.

Chapter 2

Basic Graphs and General Bounds

2.1 Basic Graphs

We can easily determine $\varphi(G)$ for a few basic classes of graphs. Note that a path is connected and acyclic and is thus a tree. The graph C_n is a cycle on n vertices, and the complete graph K_n on n vertices is a graph with all $\binom{n}{2}$ possible edges.

Proposition 2.1.1. *For any tree T , cycle C_n on $n \geq 3$ vertices, and complete graph K_n on $n \geq 3$ vertices,*

$$\varphi(T) = |V(T)|,$$

$$\varphi(C_n) = n - 2,$$

$$\varphi(K_n) = 1.$$

Proof. Let T be a tree and T_1 be a spanning tree of T . It is obvious that $T = T_1$ and thus all vertices of T are full degree in T_1 . Let $n \geq 3$. For $uv \in E(C_n)$, note that $H = C_n - \{u, v\}$ is a path. So $\varphi(H) = |V(H)| = n - 2$. If we add u and v back into H to create a spanning tree of C_n , we cannot include edge uv . Thus,

$\varphi(C_n) = \varphi(H) = n - 2$. Lastly, note that a subgraph of K_n that has at least 2 full degree vertices contains a cycle, so $\varphi(K_n) = 1$. \square

Figure 2.1 illustrates examples of spanning trees with φ full degree vertices on a tree, a cycle, and a complete graph. Tree edges are represented by solid lines, non-tree edges are represented by dashed lines, and full degree vertices are solid.

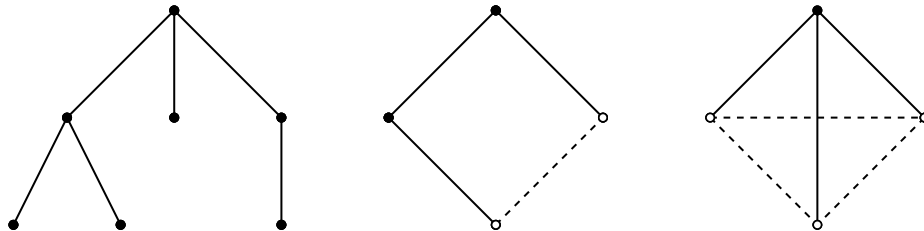


Figure 2.1: Spanning trees with φ full degree vertices on T_7 , C_4 , and K_4

2.2 General Bounds

The *maximum degree* of a graph G is $\Delta(G) = \max_{v \in V(G)} d(v)$. The *minimum degree* of a graph G is $\delta(G) = \min_{v \in V(G)} d(v)$. There are simple absolute bounds on $\varphi(G)$ in terms of the maximum degree $\Delta = \Delta(G)$ and the minimum degree $\delta = \delta(G)$, which are given in Theorem 2.2.3. In order to prove this theorem, we will need to use Lemma 2.2.1, which gives a well-known bound for the independence number of a graph. An *independent set* is a set of vertices such that none of the vertices in the set are adjacent. The *independence number* of a graph G , denoted $\alpha(G)$, is the maximum size of an independent set in G .

Lemma 2.2.1. *For any graph G on n vertices,*

$$\alpha(G) \geq \frac{n}{\Delta + 1}.$$

Proof. Let I be an independent set of maximum size in G so that $|I| = \alpha(G)$. Any

vertex in $V(G) - I$ must be adjacent to a vertex in I , so $|I| + |N(I)| = n$. We obtain an upper bound for $|N(I)|$ by summing over the degrees of I . Each vertex of I has degree at most Δ . Thus, we have

$$\begin{aligned} n &= |N(I)| + |I| \\ &\leq \sum_{v \in I} d(v) + |I| \\ &\leq \alpha(G)\Delta + \alpha(G) \\ &= \alpha(G)(\Delta + 1). \end{aligned}$$

Solving for $\alpha(G)$, we obtain the desired result. \square

We will also need to use Lemma 2.2.2 to prove Theorem 2.2.3. Lemma 2.2.2 is a fundamental result in graph theory that is variously referred to as the Degree Sum Formula, the First Theorem of Graph Theory, and the Handshake Lemma.

Lemma 2.2.2. *If $G = (V, E)$ is any graph, then*

$$\sum_{v \in V(G)} d(v) = 2|E(G)|.$$

Proof. Let $xy \in E(G)$. The edge xy is counted twice in the sum $\sum_{v \in V(G)} d(v)$: once in the $d(x)$ term and once in the $d(y)$ term. \square

We can now prove the aforementioned simple absolute bounds on $\varphi(G)$ in terms of Δ and δ .

Theorem 2.2.3. *For any connected graph G on n vertices, we have*

$$\frac{n}{\Delta(\Delta - 1) + 1} \leq \varphi(G) \leq \frac{n - 2}{\delta - 1}.$$

Proof. We first prove the lower bound. Let G^2 denote the square of G , i.e., the graph formed by joining all vertices at distance at most 2 in G . If I is an independent set in G^2 , then the vertices of I have disjoint neighborhoods in G . Thus all the vertices of I can be taken to be of full degree in a spanning tree of G . So

$$\varphi(G) \geq \alpha(G^2) \geq \frac{n}{\Delta(G^2) + 1} \geq \frac{n}{\Delta(\Delta - 1) + 1}$$

where we have used Lemma 2.2.1.

To prove the upper bound, we let T be a spanning tree of G and let $d_T(v)$ denote the degree of vertex v in T . Let X be the set of all full degree vertices in T . Since all full degree vertices have degree at least δ and all non-full degree vertices in T have degree at least 1,

$$\begin{aligned} 2(n - 1) &= \sum_v d_T(v) = \sum_{v \in X} d_T(v) + \sum_{v \notin X} d_T(v) \\ &\geq |X| \cdot \delta + (n - |X|) \cdot 1 \\ &= (\delta - 1)|X| + n \end{aligned}$$

where we have used Lemmas 1.1.4 and 2.2.2 in the first line. Thus,

$$|X| \leq \frac{n - 2}{\delta - 1}.$$

□

Chapter 3

Grid Graphs

The *Cartesian product* of two sets A and B is given by $A \times B = \{(a, b) : a \in A, b \in B\}$. So for graphs G and H , $V(G) \times V(H) = \{z = (x, y) : x \in V(G), y \in V(H)\}$. The *graph Cartesian product* of graphs G and H , denoted $G \square H$, is the graph with vertex set $V(G) \times V(H)$ such that two vertices $u = (u_1, u_2)$ and $v = (v_1, v_2)$ are adjacent only when either (i) $u_1 = v_1$ and u_2 is adjacent to v_2 in H or (ii) $u_2 = v_2$ and u_1 is adjacent to v_1 in G .

The $m \times n$ *grid graph* is the graph formed by $P_m \square P_n$, the graph Cartesian product of paths on m and n vertices. In other words, let $G(m, n)$ denote the grid graph with m rows of vertices and n columns of vertices. Each vertex is adjacent to the vertex immediately above and below it in its column and to the right and left of it in its row, as illustrated in Figure 3.1.

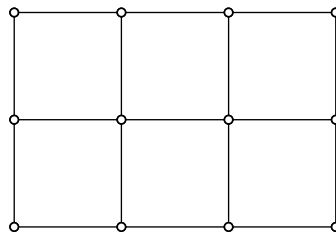


Figure 3.1: The 3×4 grid graph $G(3, 4)$

In this chapter, we determine $\varphi(G(m, n))$ in terms of n for a few different values of m . We begin with $G(2, n)$, the graph with 2 rows and n columns of vertices.

Theorem 3.0.1. *For $G = G(2, n)$, we have*

$$\varphi(G) = n.$$

Proof. For $j \in [2]$ and $i \in [n]$, let $x_{j,i}$ be the vertex in the j -th row and i -th column of $G = G(2, n)$. There exists a spanning tree of G with n vertices of full degree (e.g., a tree where all $x_{1,i}$ are of full degree, as shown in Figure 3.2). So, $\varphi(G) \geq n$.

We will now show that every spanning tree of G has at most n full degree vertices. Let T be a spanning tree of G and let $S_i = \{x_{1,i}, x_{2,i}\}$ and $B_k = \{x_{1,k}, x_{2,k}, x_{1,k+1}, x_{2,k+1}\}$ for $k \in [n-1]$. Let $F(H)$ denote the number of full degree vertices in any graph H . So, the sum of the number of full degree vertices across all B_k is given by

$$\sum_{k=1}^{n-1} F(B_k).$$

Each $F(B_k) \leq 2$, otherwise there will be a 4-cycle. If $F(S_1) = 2$, then $F(B_2) \leq 1$. Likewise, if $F(S_n) = 2$, then $F(B_{n-2}) \leq 1$. Let

$$\mathbb{1}_{F(S)=2} = \begin{cases} 1 & \text{if } F(S) = 2 \\ 0 & \text{otherwise.} \end{cases}$$

Thus, we have

$$\sum_{k=1}^{n-1} F(B_k) \leq 2(n-1) - \mathbb{1}_{F(S_1)=2} - \mathbb{1}_{F(S_n)=2}.$$

The sum of the number of full degree vertices across all B_k can also be counted by the sum of the number of full degree vertices across all S_i , where S_1 and S_n are counted

once and all others are counted twice. That is, using the fact that $F(T) = \sum_{i=1}^n F(S_i)$,

$$\begin{aligned} \sum_{k=1}^{n-1} F(B_k) &= F(S_1) + F(S_n) + 2 \sum_{i=2}^{n-1} F(S_i) \\ &= F(T) + \sum_{i=2}^{n-1} F(S_i) \\ &= 2F(T) - F(S_1) - F(S_n). \end{aligned}$$

So we have,

$$2F(T) - F(S_1) - F(S_n) \leq 2(n-1) - \mathbb{1}_{F(S_1)=2} - \mathbb{1}_{F(S_n)=2}.$$

Solving for $F(T)$, we obtain

$$F(T) \leq \frac{1}{2}(2n - 2 + [F(S_1) - \mathbb{1}_{F(S_1)=2}] + [F(S_n) - \mathbb{1}_{F(S_n)=2}]).$$

For $k \in \{1, n\}$, note that $F(S_k) \leq 2$. If $F(S_k) = 2$, then $F(S_k) - \mathbb{1}_{F(S_k)=2} = 1$. If $F(S_k) \leq 1$, then $F(S_k) - \mathbb{1}_{F(S_k)=2} \leq 1$. So,

$$F(T) \leq \frac{1}{2}(2n - 2 + [1] + [1]) = n.$$

Thus, $\varphi(G) \leq n$ and therefore $\varphi(G) = n$. □

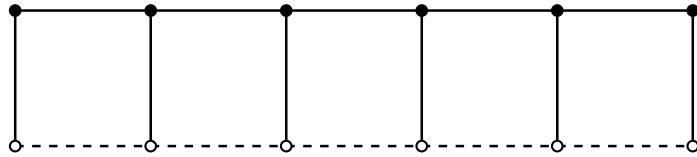


Figure 3.2: Spanning tree with φ full degree vertices on $G(2,6)$

An example of a spanning tree that produces φ full degree vertices on $G(2, n)$ is shown in Figure 3.2. Tree edges are represented by solid lines, non-tree edges are represented by dashed lines, and full degree vertices are solid.

The next theorem determines our parameter on $G(3, n)$, the graph with 3 rows of vertices and n columns of vertices. The proof of this theorem is, although slightly more complicated, similar to the proof of Theorem 3.0.1.

Theorem 3.0.2. *For $G = G(3, n)$, we have*

$$\varphi(G) = \left\lfloor \frac{4n}{3} + \frac{2}{3} \right\rfloor.$$

Proof. For $j \in [3]$ and $i \in [n]$, let $x_{j,i}$ be the vertex in the j -th row and i -th column of $G = G(3, n)$. If we take a tree of G where all $x_{1,i}$ are full degree and every third $x_{3,i}$ is full degree, then we have a spanning tree of G with $\left\lfloor \frac{4n}{3} + \frac{2}{3} \right\rfloor$ vertices of full degree, as shown in Figure 3.3. So, $\varphi(G) \geq \left\lfloor \frac{4n}{3} + \frac{2}{3} \right\rfloor$.

We will now show that every spanning tree of G has at most $\left\lfloor \frac{4n}{3} + \frac{2}{3} \right\rfloor$ full degree vertices. Let T be a spanning tree of $G(3, n)$ and let $S_i = \{x_{1,i}, x_{2,i}, x_{3,i}\}$. For $k \in [n - 2]$, let $B_k = \{x_{j,r} : j \in [3], r \in \{k, k + 1, k + 2\}\}$. Let $F(H)$ denote the number of full degree vertices in any graph H . So, the sum of the number of full degree vertices across all B_k is given by

$$\sum_{k=1}^{n-2} F(B_k).$$

Each $F(B_k) \leq 4$, otherwise there will be a cycle. If $F(S_1) = 3$, then $F(B_2) \leq 3$. Likewise, if $F(S_n) = 3$, then $F(B_{n-3}) \leq 3$. Let

$$\mathbb{1}_{F(S)=3} = \begin{cases} 1 & \text{if } F(S) = 3 \\ 0 & \text{otherwise.} \end{cases}$$

Thus, we have

$$\sum_{k=1}^{n-2} F(B_k) \leq 4(n - 2) - \mathbb{1}_{F(S_1)=3} - \mathbb{1}_{F(S_n)=3}.$$

The sum of the number of full degree vertices across all B_k can also be counted by the sum of the number of full degree vertices across all S_i , where S_1 and S_n are counted once, S_2 and S_{n-1} are counted twice, and all others are counted thrice. That is, using the fact that $F(T) = \sum_{i=1}^n F(S_i)$,

$$\begin{aligned} \sum_{k=1}^{n-1} F(B_k) &= F(S_1) + F(S_n) + 2[F(S_2) + F(S_{n-1})] + 3 \sum_{i=3}^{n-2} F(S_i) \\ &= 3F(T) - 2[F(S_1) + F(S_n)] - [F(S_2) + F(S_{n-1})]. \end{aligned}$$

So we have

$$3F(T) - 2[F(S_1) + F(S_n)] - [F(S_2) + F(S_{n-1})] \leq 4(n-2) - \mathbf{1}_{F(S_1)=3} - \mathbf{1}_{F(S_n)=3}.$$

Solving for $F(T)$, we obtain

$$F(T) \leq \frac{4n}{3} - \frac{8}{3} + a + b + c$$

where we let $a = \frac{-\mathbf{1}_{F(S_1)=3} - \mathbf{1}_{F(S_n)=3}}{3}$, $b = \frac{2[F(S_1) + F(S_n)]}{3}$ and $c = \frac{F(S_2) + F(S_{n-1})}{3}$ for ease of notation. Note that one of the following must occur in T :

1. $F(S_1) = 3 = F(S_n)$. If this occurs, we have $a = -\frac{2}{3}$, $b = 4$, and $c = 0$. So $a + b + c = \frac{10}{3}$.
2. $F(S_1) \neq 3 \neq F(S_n)$. If this occurs, we have $a + b + c \leq \frac{10}{3}$. We obtain the maximum $a + b + c = \frac{10}{3}$ when $F(S_1) = 2 = F(S_n)$, which implies $F(S_2) = 1 = F(S_{n-1})$.
3. $F(S_1) = 3$ or $F(S_n) = 3$ but not both. If this occurs, we have $a + b + c \leq \frac{10}{3}$. Without loss of generality, assume $F(S_1) = 3$, which implies $F(S_2) = 0$. We obtain the maximum $a + b + c = \frac{10}{3}$ when $F(S_n) = 2$ and $F(S_{n-1}) = 1$.

Thus, in all three cases, we have

$$F(T) \leq \frac{4n}{3} - \frac{8}{3} + \frac{10}{3} = \frac{4n}{3} + \frac{2}{3}.$$

Therefore, since we can only count vertices using whole numbers, $\varphi(G) \leq \lfloor \frac{4n}{3} + \frac{2}{3} \rfloor$ and thus $\varphi(G) = \lfloor \frac{4n}{3} + \frac{2}{3} \rfloor$. \square

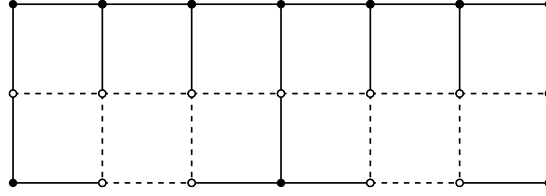


Figure 3.3: Spanning tree with φ full degree vertices on $G(3,7)$

An example of a spanning tree that produces φ full degree vertices on $G(3,n)$ is shown in Figure 3.3. Tree edges are represented by solid lines, non-tree edges are represented by dashed lines, and full degree vertices are solid.

Our last theorem of this chapter focuses on $G(m,n)$, where m and n are each a large number tending towards infinity. For two sequences A_n and B_n , we say $A_n \sim B_n$ if $\lim_{n \rightarrow \infty} \frac{A_n}{B_n} = 1$.

Theorem 3.0.3. *For $G = G(m,n)$ with m, n tending towards infinity, we have*

$$\varphi(G) \sim \frac{mn}{3}.$$

Proof. There exists a spanning tree of the grid $G = G(m,n)$, where m and n are tending towards infinity, with at least $\frac{mn}{3}$ vertices of full degree (e.g., a tree where all vertices in every third column of G are of full degree as shown in Figure 3.4). So, $\varphi(G) \geq \frac{mn}{3}$.

Let $V' = \{v \in V(G) : d_G(v) = 4\}$ and let $X \subset V'$ be the set of full degree vertices from V' in a spanning tree T of G . Since all vertices in X have degree 4 in T and all

non-full degree vertices from V' have degree at least 1 in T ,

$$\begin{aligned}
 2(mn - 1) &\geq \sum_{v \in V'} d_T(v) = \sum_{v \in X} d_T(v) + \sum_{v \notin X} d_T(v) \\
 &\geq 4 \cdot |X| + 1 \cdot |V' \setminus X| \\
 &= 4 \cdot |X| + (mn - 2m - 2n + 4 - |X|).
 \end{aligned}$$

Solving for $|X|$, we obtain

$$|X| \leq \frac{mn + 2m + 2n - 6}{3} = \frac{mn}{3} + O(m + n).$$

Therefore, $\varphi(G) \sim \frac{mn}{3}$. □

A portion of a spanning tree that produces $\varphi(G)$ full degree vertices on $G = G(m, n)$, with m, n tending towards infinity, is shown in Figure 3.4. Tree edges are represented by solid lines, non-tree edges are represented by dashed lines, and full degree vertices are solid.

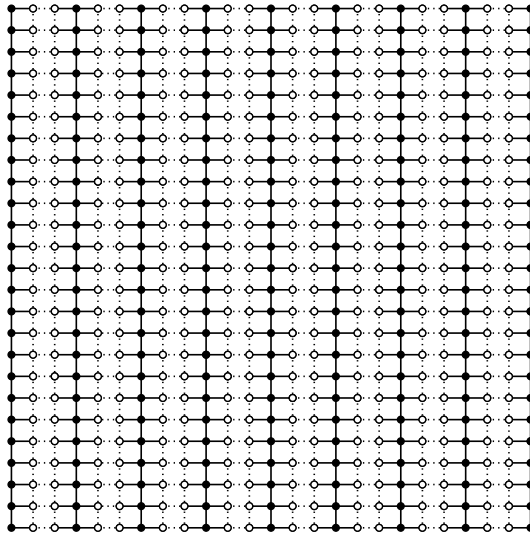


Figure 3.4: Spanning tree portion with $\varphi(G)$ full degree vertices on $G(m, n)$, $m, n \rightarrow \infty$

Chapter 4

Hypercubes

A graph is k -regular if every vertex in the graph has degree k , and so $\delta = k = \Delta$ in a k -regular graph. A *hypercube* Q_k is a k -regular graph whose vertices are subsets of $[k] := \{1, 2, \dots, k\}$. Two vertices are adjacent if their subsets differ in exactly one element. Note that Q_k has 2^k vertices, and we say that Q_k is a hypercube in k dimensions.

Choi and Guan [7] found that $\varphi(Q_k) = 2^k/k$ if $k = 2^m$ for a positive integer m and $\varphi(Q_k) < 2^k/k$ if $k \neq 2^m$. We can combine their results with our results from Section 2.2 as follows in Corollary 4.0.1.

Corollary 4.0.1. *For a hypercube Q_k , if $k = 2^m$ for $m \in \mathbb{Z}^+$, then*

$$\varphi(Q_k) = \frac{2^k}{k}.$$

Otherwise,

$$\frac{2^k}{k^2 - k + 1} < \varphi(Q_k) < \frac{2^k}{k}.$$

Proof. This result follows directly from Theorem 2.2.3 in this thesis and Theorems 2.3 and 2.4 in [7]. We have strict inequality when k is not a power of 2 since $\varphi \in \mathbb{Z}$. \square

For $1 \leq k \leq 24$, Table 4.1 gives values for the number of full degree vertices F_k in Q_k obtained via a construction, which is described below. As our table shows, this construction yields the exact values of $\varphi(Q_k)$ given in Corollary 4.0.1 for $k = 2^m$ with $m \in \mathbb{Z}^+$. Theorem 4.0.6 relates the values given in Table 4.1 to $\varphi(Q_k)$, which provide some improvements over the lower bounds given in Corollary 4.0.1.

k	F_k	k	F_k
1	2	13	512
2	2	14	1024
3	2	15	2048
4	4	16	4096
5	4	17	5120
6	8	18	10240
7	16	19	19456
8	32	20	38912
9	40	21	73728
10	76	22	147456
11	144	23	294912
12	288	24	589824

Table 4.1: Number of full degree vertices in Q_k obtained via the construction described below. Values shown in bold are optimal; all others are lower bounds.

Before we prove Theorem 4.0.6, we will need to prove a few lemmas.

Lemma 4.0.2 (The Pigeonhole Principle). *Suppose n and k are positive integers with $n > k$. If n pigeons are distributed amongst k pigeonholes, then some hole has at least 2 pigeons.*

Proof. Let n and k be positive integers with $n > k$. Suppose n pigeons are distributed amongst k pigeonholes and every hole has at most 1 pigeon. Then, if we sum the number of pigeons in each hole, we have at most k pigeons, a contradiction. Thus, some hole has at least 2 pigeons. \square

Lemma 4.0.3. *If A and B are each m -element subsets of $[n]$, where $n < 2m$, then $A \cap B \neq \emptyset$.*

Proof. Let A and B be m -element subsets of $[n]$, where $n < 2m$. Let S be the multiset (which allows elements to be repeated) containing the elements of A and B . So we have $|S| = 2m$. By Lemma 4.0.2, where the elements of $[n]$ are the “pigeonholes” and the elements of S are the “pigeons,” some element of $[n]$ must appear twice in S . Call this element x . Since elements of sets must be unique, x can be repeated in neither A nor B . Thus, x must be an element of both A and B , and so $A \cap B \neq \emptyset$. \square

Lemma 4.0.4. *If A, B, C are each 3-element subsets of $[5]$, two of A, B, C must contain 2 common elements.*

Proof. Let A, B, C each be 3-element subsets of $[5]$. Without loss of generality, we know $|B \cap C| \geq 1$ by Lemma 4.0.3. If $|B \cap C| = 2$, then we are done. If $|B \cap C| = 1$, then $[k] = B \cup C$ and thus each element of A must be an element of B or C , so the desired result follows from Lemma 4.0.2 where the elements of A are the “pigeons” and the sets B, C are the “pigeonholes.” \square

Lemma 4.0.5. *Given a hypercube Q_k , each $v \in V(Q_k)$ is adjacent to at most 1 full degree vertex in a spanning tree of Q_k .*

Proof. See the proof of Lemma 2.1 in [7]. \square

We now have enough information to prove Theorem 4.0.6, which is stated below.

Theorem 4.0.6. *For $1 \leq k \leq 5$, $\varphi(Q_k) = F_k$. For $6 \leq k \leq 24$, $\varphi(Q_k) = F_k$ if $k = 2^m$ for $m \in \mathbb{Z}^+$ and $\varphi(Q_k) \geq F_k$ otherwise, where values of F_k are shown in Table 4.1.*

Proof. Note that Q_1 is a single edge, i.e., a path on 2 vertices, and so $\varphi(Q_1) = 2$ by Proposition 2.1.1. For all $2 \leq k \leq 24$, $\varphi(Q_k) \geq F_k$ since these values are from a construction. Let $m \in \mathbb{Z}^+$. The results for $k = 2^m$ follow from Corollary 4.0.1. Also from Corollary 4.0.1, $\varphi(Q_3) < 2.\bar{6}$, and thus $\varphi(Q_3) = 2$.

We will now show that $\varphi(Q_5) \leq 4$. Let L_i be the set of vertices whose labels are subsets of size i and let f_i be the number of full degree vertices in L_i . Then the number of full degree vertices in Q_5 is given by $\sum_{i=0}^5 f_i$. Pick an initial vertex to be full degree. Due to the symmetry of the hypercube, this can be any vertex, so pick vertex \emptyset . Then $f_0 = 1$ and, by Lemma 4.0.5, $f_2 = 0$. Clearly, $f_5 \leq 1$. By Lemma 4.0.5, we have $f_1 \leq 1$ and $f_4 \leq 1$. We have $f_3 \leq 2$, since three full degree vertices from L_3 would create a cycle by Lemma 4.0.4. We will now proceed by considering three cases:

Case 1: Suppose $f_5 = 1$. Then $f_3 = 0$ by Lemma 4.0.5. Thus, $\sum_{i=0}^5 f_i \leq 4$.

Case 2: Suppose $f_5 = 0 = f_4$. Then $\sum_{i=0}^5 f_i \leq 4$.

Case 3: Suppose $f_5 = 0$ and $f_4 = 1$. If $f_1 = 0$, then $\sum_{i=0}^5 f_i \leq 4$, so suppose $f_1 = 1$.

Without loss of generality, say that vertex $\{5\}$ is full degree. By Lemma 4.0.5, no vertex from L_3 containing the element 5 may be full degree. That is, any full degree vertex of L_3 must be a subset of $[4]$. Any two such vertices S_1 and S_2 will share two common elements by Lemma 4.0.3. It follows that if both S_1 and S_2 are full degree, then a cycle is formed by $S_1, S_2, S_1 \cap S_2$, and any vertex from L_4 that contains $S_1 \cap S_2$. Thus, $f_3 < 2$ and $\sum_{i=0}^5 f_i \leq 4$.

Therefore, $\varphi(Q_5) \leq 4$ and thus $\varphi(Q_5) = 4$. □

We will now describe the construction from [7] used to obtain the values given in Table 4.1. A *binary string of length k* is a k -digit sequence of 0's and 1's. Note that we may alternatively define the hypercube Q_k as a graph whose vertices are binary strings of length k . Two vertices in a hypercube are adjacent if their strings differ in exactly one position. The *star* centered at vertex v in graph G , denoted $S_G(v)$, is the graph with vertex set $\{v\} \cup N_G(v)$ and edge set $\{vx : x \in N(v)\}$. Stars S_1 and S_2 are *vertex disjoint* if $V(S_1) \cap V(S_2) = \emptyset$. In a spanning tree of a graph, we refer to a

star as full degree if its center is a full degree vertex. A *code* is a set of vertex disjoint full degree stars. Note that to form Q_k , we may start with $Q = Q_{k-1}$ and $Q' = Q_{k-1}$, and then add edges between the vertices of Q and Q' that share the same label. In order to find a lower bound for $\varphi(Q_k)$, we can construct a spanning tree of Q_k by first finding a code in Q and then finding the same code in Q' . We connect the center of each star in Q with the center of the star in Q' that has the same label. Thus, each center is a full degree vertex, and so a code of size z in Q_{k-1} gives us $F_k = 2z$.

In [3], Best et al. present a table for $A(n, d)$, which is the maximum number of length n binary strings such that any two of the strings differ in at least d positions. In the context of our construction on Q_k , we are finding length $k - 1$ binary strings that differ in at least 3 positions, i.e., our vertex disjoint full degree stars. Then $A(k - 1, 3)$ represents the size of a code in Q_{k-1} . By Theorem 1 in [3], $A(k - 1, 3) = A(k, 4)$. Thus, $F_k = A(k, 3) = 2A(k - 1, 3) = 2A(k, 4)$. For $5 \leq k \leq 16$, the values of F_k given in Table 4.1 were calculated using values of $A(k, 4)$ given in Table 1 of [3]. Note that $A(6, 4) = 4 = A(5, 3) = F_5$ by Theorem 1 in [3].

Examples of spanning trees that produce φ full degree vertices on Q_3 and Q_4 are shown in Figures 4.1 and 4.2, respectively. Tree edges are represented by solid lines, non-tree edges are represented by dashed lines, and full degree vertices are solid.

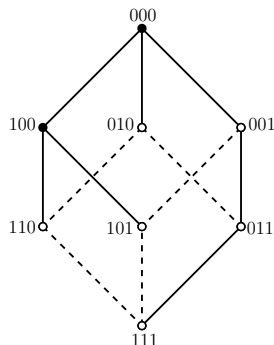


Figure 4.1: Spanning tree that produces $\varphi(Q_3) = 2$ full degree vertices

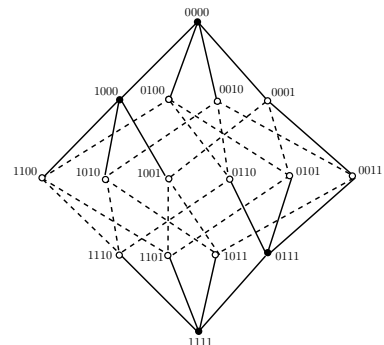


Figure 4.2: Spanning tree that produces $\varphi(Q_4) = 4$ full degree vertices

Chapter 5

Random Regular Graphs

Recall that a graph is r -regular if every vertex in the graph has degree r . We have the following bounds that arise from our results in Section 2.2.

Corollary 5.0.1. *For a connected r -regular graph G on n vertices,*

$$\frac{n}{r^2 - r + 1} \leq \varphi(G) \leq \frac{n}{r - 1}.$$

Proof. These bounds are a direct result of Theorem 2.2.3. □

The upper bound in Corollary 5.0.1 is essentially tight as can be seen by taking $G = K_{r-1} \square C_{n/(r-1)}$. In this construction, one can take one vertex from each of $n/(r-1) - 2$ copies of K_{r-1} to be of full degree and so $\varphi(G) \geq n/(r-1) - 2$. See Section 5.5 for a question about the tightness of the lower bound.

In the case of cubic graphs (i.e., 3-regular graphs), this problem has been studied before under a different guise. Note that, for $S \subset V(G)$ for any graph G , a subgraph H of G is *induced by S* if $V(H) = S$ and $E(H) = \{uv \in E(G) : u, v \in S\}$. We denote the subgraph of G induced by S as $G[S]$.

Let G be a connected graph with n vertices. A *dominating set* of vertices is a subset $S \subseteq V(G)$ such that every vertex in G is either in S or adjacent to a vertex in S . If $G[S]$ is connected, then S is said to be a *connected dominating set*. Let $\gamma_C(G)$ be the minimum number of vertices in a connected dominating set of G and let $\lambda(G)$ be the maximum number of leaves in a spanning tree of G . Aptly named, the problem of determining $\lambda(G)$ is known as the many leaves problem and the problem of determining $\gamma_C(G)$ is known as the connected dominating set problem. The following proposition, which is observed in [6] and [10], shows that the many leaves problem is equivalent to the connected dominating set problem.

Proposition 5.0.2. *For a connected graph G on n vertices,*

$$\lambda(G) = n - \gamma_C(G).$$

Proof. Let G be a connected graph on n vertices. A spanning tree of G exists by Lemma 1.1.5, so let T be a spanning tree of G with leaf set L . We have that $V(T) \setminus L$ forms a tree by Lemma 1.1.1. Also, note that every leaf in L is adjacent to a vertex in $V(T) \setminus L$. Thus, $V(T) \setminus L$ is a connected dominating set and so $\lambda(G) \leq n - \gamma_C(G)$.

Now consider a connected dominating set D of G . Then, by definition, $G[D]$ is connected. Note that the complement of D is given by $\overline{D} = \{x \in V(G) : x \notin D\}$. Let T' be a spanning tree of $G[D]$. We can attach all of the vertices in \overline{D} to T' as leaves to obtain T , a spanning tree of G . Thus, there is a spanning tree of G in which $V(G) \setminus D$ is the leaf set and so $\lambda(G) \geq n - \gamma_C(G)$. Therefore, $\lambda(G) = n - \gamma_C(G)$. \square

These parameters have been very well studied (see e.g., [6], [13], [20], and others). Some previous results for these problems are given in Chapter 6. We have the following observation which shows that the problems of finding λ , φ and γ_C are all equivalent in connected cubic graphs.

Proposition 5.0.3. *For any connected cubic graph G , we have $\lambda(G) = \varphi(G) + 2$. Furthermore, for any $r \geq 4$ and any connected r -regular graph G , we have $\lambda(G) \geq (r - 2)\varphi(G) + 2$.*

Proof. Let $r \geq 3$ and let G be a connected r -regular graph. Let T be a spanning tree of G and let x_i be the number of vertices in T of degree i for $i \in [r]$. Then

$$\begin{aligned} 2(n - 1) &= \sum_{v \in T} d_T(v) = \sum_{i=1}^r i x_i \\ &= 2 \left(\sum_{i=1}^r x_i \right) - x_1 + \sum_{i=3}^r (i - 2) x_i \\ &= 2n - x_1 + \sum_{i=3}^r (i - 2) x_i \\ &\geq 2n - x_1 + (r - 2) x_r \end{aligned}$$

where the last inequality is equality in the case $r = 3$. Thus $x_1 \geq (r - 2)x_r + 2$ and the result follows. \square

We seek to understand the average behavior of φ for r -regular graphs by considering random regular graphs. A *random r -regular graph*, denoted $G(n, r)$, is a graph chosen uniformly at random from all r -regular graphs on vertex set $[n] := \{1, 2, \dots, n\}$. For background on random regular graphs, see Wormald's survey [22]. Duckworth [9] and Duckworth and Mans [10] developed an algorithm for the connected dominating set problem and analyzed it on random d -regular graphs for d fixed. An event occurs *with high probability (w.h.p.)* if the probability that the event occurs tends towards 1 as n tends towards ∞ . Duckworth's algorithm [9] produces a connected dominating set on the random cubic graph $G \sim G(n, 3)$ that w.h.p. has size less than $0.5854n + o(n)$. Thus, w.h.p. his algorithm gives $\gamma_C(G) \leq 0.5854n + o(n)$ and, by Propositions 5.0.2 and 5.0.3, $\varphi(G) = \lambda(G) \geq 0.4146n + o(n)$.

We approach this problem by providing an algorithm that attempts to maximize the number of full degree vertices in spanning trees of random regular graphs. Somewhat surprisingly, our simple algorithm gives a decent improvement over the bounds given in [9, 10] for random cubic graphs. A *breadth-first search* is an algorithm that explores all of the vertices on one level of a tree before exploring vertices on the next level. Algorithm 1 is based on a breadth-first search and thus only explores from certain vertices. The algorithm iteratively builds a forest T . At each step, a random vertex v (either a leaf of the current forest or an as-yet unseen vertex) is chosen and the neighbors of v are exposed. If at most one of v 's neighbors lies in the current forest T , then v may be safely added to T as a full degree vertex. Recall that $S_G(v)$ represents the star centered at v in G . The main results of this chapter, the bounds determined by Algorithm 1, are given in Theorem 5.0.4.

Input: Connected r -regular graph $G = (V, E)$.

Output: Tree T with full degree vertices F .

Select arbitrary $v \in V$;

$T = S_G(v)$;

$L = N_G(v)$, $Z_r = V \setminus V_T$;

while $L \cup Z_r \neq \emptyset$ **do**

if $L \neq \emptyset$ **then**

 | Select $v \in L$ u.a.r.

else

 | Select $v \in Z_r$ u.a.r.

if $|V_T \cap N_G(v)| \leq 1$ **then**

 | $T = T \cup S_G(v)$;

 | $F = F \cup \{v\}$;

 | $X = Z_r \cap N_G(v)$;

 | Move X from Z_r to L

else

 | $Z_r = Z_r \setminus N_G(v)$;

 | $L = L \setminus N_G(v)$;

end

Complete the forest T to a spanning tree arbitrarily;

Algorithm 1: Full Degree Tree

Theorem 5.0.4. *When run on a random cubic graph $G \sim G(n,3)$, Algorithm 1 w.h.p. produces a tree T with at least $0.4591n$ vertices of full degree. Thus $\varphi(G) \geq 0.4591n$, $\lambda(G) \geq 0.4591n$, and $\gamma_C(G) \leq 0.5409n$. When run on a random r -regular graph $G(n,r)$ with $4 \leq r \leq 10$, Algorithm 1 w.h.p. produces a tree with at least $f_r n$ vertices of full degree, where values of f_r are shown in Table 5.1.*

r	f_r	u_r
3	.4591	.5000
4	.2699	.3333
5	.1811	.2500
6	.1315	.2000
7	.1006	.1667
8	.0799	.1429
9	.0652	.1250
10	.0545	.1111

Table 5.1: Values of f_r, u_r such that w.h.p. $f_r n \leq \varphi(G(n,r)) \leq u_r n$. The bounds f_r come from Algorithm 1 and $u_r = 1/(r-1)$, deterministically from Theorem 2.2.3.

In the following sections, we use the differential equations method to prove Theorem 5.0.4. We describe the expected one-step changes of several parameters throughout the algorithm in Section 5.2. Then, in Section 5.3, we apply a general theorem of Wormald [21, 23] to show concentration of this system of random variables around their expected trajectories in order to complete the proof of Theorem 5.0.4.

5.1 Setting up the Analysis of Algorithm 1

Note that a *matching* is a set of non-loop edges with no shared endpoints. The vertices incident to edges of a matching M are said to be *saturated* by M . A *perfect matching* saturates all vertices in a graph. Also, note that a *multigraph* is a graph that is allowed to have multiple edges, which are distinct edges that have the same endpoints.

In our set up, we make use of the *configuration model* to analyze our algorithm on $G(n, r)$ (see, e.g., [22] for more details on the description that follows). Suppose rn is even and consider a set of rn *configuration points* partitioned into n labeled *buckets* v_1, \dots, v_n each of size r . A *pairing* of these points is a perfect matching of the configuration points. Given a pairing P , we may obtain a multigraph $G(P)$ by contracting each of the buckets to one vertex. It is well known that the restriction of this probability space to simple graphs is $G(n, r)$ and that for fixed r , the probability that the pairing generates a simple graph is bounded away from 0 (independently of n). Thus, any event that holds w.h.p. over the space of random pairings also holds w.h.p. for $G(n, r)$.

We analyze our algorithm by tracking certain parameters throughout the execution of the algorithm. We only reveal partial information about $G(n, r)$ (or more precisely, the pairing) as the algorithm progresses. We let T represent the current forest being built. Each iteration of the while loop will be called a *step*. At each step, we *process* a vertex v . When v is processed, we reveal its neighbors in the configuration (some of v 's neighbors may already have been revealed). Let $L = L(t)$ represent the vertices that are in T and have $r - 1$ unrevealed configuration points. All vertices of L are leaves of T , but not all leaves of T are in L . For $i \in [r]$, let $Z_i = Z_i(t)$ denote the set of vertices that are not in T and have i unrevealed neighbors at time step t . Thus, we either process a vertex that is a leaf of T (when $v \in L$) or a vertex that has never been seen (when $v \in Z_r$). A step is a *success* if at most one of v 's neighbors already lies in T (when $v \in L$, the previously revealed neighbor certainly lies in T). If the step is a success, we may add v and its neighbors to T . We let $F = F(t)$ represent the set of vertices of degree r (i.e., the set of full degree vertices) in T at time step t .

We consider our algorithm to have two phases. In Phase 1, we only process vertices from L . We say Phase 2 begins when the first vertex from Z_r is processed. In Phase 2, we process vertices from both L and Z_r . In Figure 5.1, we illustrate the possibilities

when processing a vertex v from L in the case of a random cubic graph. In that case, there are three possibilities, shown in Figure 5.1: (1) both of v 's newly revealed neighbors are not currently in T , (2) only one of these neighbors is currently in T , or (3) both of these neighbors are currently in T . If both of these neighbors are not currently in T (Case 1 in Figure 5.1), then we can make v full degree by adding all of its neighbors and incident edges to T , i.e., the step is a success. In Cases 2 and 3, adding v and its neighbors to T could potentially create a cycle and so we do not attempt to do so. We note that since T is a forest, having 2 neighbors in T does not guarantee the creation of a cycle, but it is simpler for our analysis to err on the side of caution.

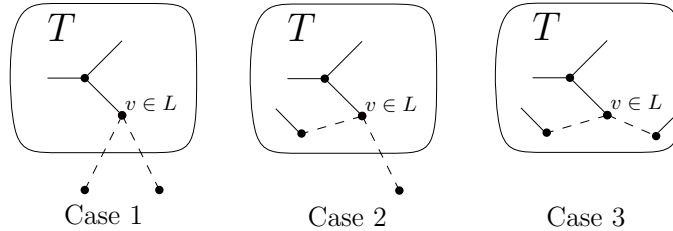


Figure 5.1: Selecting $v \in L$ and exposing its neighbors

5.2 Expected One Step Changes

We will now use the set up we described above to define some equations for our parameters that arise from Algorithm 1. In a common abuse of notation, we refer to L , Z_i , and F as the sets they represent as well as the sizes of those sets. Let

$$Z := \sum_{i=1}^r iZ_i$$

so that Z represents the number of unrevealed configuration points corresponding to the Z_i 's. Let $M = M(t)$ represent the number of unrevealed configuration points at time step t . There are two types of operations that we perform. Following notation

from [23], let Op_1 denote the operation of processing a vertex from L and let Op_2 denote the operation of processing a vertex from Z_r . Let $\text{op}_t \in \{\text{Op}_1, \text{Op}_2\}$ represent the operation performed at time step t of the algorithm.

5.2.1 Phase 1

Let \mathcal{F}_t represent the revealed part of the configuration model at time t . As mentioned before, in Phase 1, we only process vertices from L . For random variable $X = X(t)$, let $\Delta X(t) = X(t+1) - X(t)$. For $i \in [r-1]$,

$$\begin{aligned} \mathbb{E}[\Delta Z_i(t) \mid \mathcal{F}_t, \text{op}_t = \text{Op}_1] &= \\ (r-1) \cdot \left(-\frac{iZ_i}{M} + \frac{(i+1)Z_{i+1}}{M} \cdot \left(1 - \left(\frac{Z}{M} \right)^{r-2} \right) \right) + O\left(\frac{1}{n}\right) \end{aligned} \quad (5.1)$$

$$\mathbb{E}[\Delta Z_r(t) \mid \mathcal{F}_t, \text{op}_t = \text{Op}_1] = (r-1) \cdot \left(-\frac{rZ_r}{M} \right) + O\left(\frac{1}{n}\right) \quad (5.2)$$

$$\begin{aligned} \mathbb{E}[\Delta L(t) \mid \mathcal{F}_t, \text{op}_t = \text{Op}_1] &= \\ -1 + (r-1) \cdot \left(-\frac{(r-1)L}{M} + \frac{rZ_r}{M} \cdot \left(\frac{Z}{M} \right)^{r-2} \right) + O\left(\frac{1}{n}\right) \end{aligned} \quad (5.3)$$

$$\mathbb{E}[\Delta F(t) \mid \mathcal{F}_t, \text{op}_t = \text{Op}_1] = \left(\frac{Z}{M} \right)^{r-1} + O\left(\frac{1}{n}\right) \quad (5.4)$$

$$\mathbb{E}[\Delta M(t) \mid \mathcal{F}_t, \text{op}_t = \text{Op}_1] = -2(r-1). \quad (5.5)$$

To see (5.1) and (5.2), note that the vertex v that we are processing has $r-1$ unrevealed configuration points. If any of these points pair with a point in a Z_i vertex, then that vertex is no longer a Z_i vertex. This happens with probability $\frac{iZ_i + O(1)}{M + O(1)} = \frac{iZ_i}{M} + O(1/n)$, where the error term is $O(1/n)$ since we will assume that $M = \Omega(n)$. The error terms in the rest of the explanation are similar and will be ignored. We gain a Z_i vertex if the step is not a success but one of the revealed points pairs with a Z_{i+1} point. The factor $(i+1)Z_{i+1}/M$ represents the probability that

a point pairs with a Z_{i+1} point and $1 - \left(\frac{Z}{M}\right)^{r-2}$ represents the probability that the other revealed points do not all land in Z (which would mean a success).

In (5.3), the -1 accounts for the loss of the L vertex that we are processing. Again, we will lose L vertices when the revealed points pair to L vertices. We gain L vertices if the step is a success and one of the points pairs with a Z_r vertex. The expected change in F , given by (5.4), is just the probability that the step is a success, which is $\left(\frac{Z}{M}\right)^{r-1}$. Finally, note that at each step we reveal $r - 1$ pairs from the configuration, which explains (5.5).

5.2.2 Phase 2

The expected one step changes in Phase 2 when processing a vertex from L are given by equations (5.1)–(5.5).

The expected one step changes when processing a vertex from Z_r are as follows. For ease of notation let $P := \left(\frac{Z}{M}\right)^{r-1} + (r - 1) \cdot \left(\frac{Z}{M}\right)^{r-2} \cdot \left(1 - \frac{Z}{M}\right)$. Note that this represents the probability of a success step when performing Op_2 conditional on one revealed point pairing with a Z vertex. For $i \in [r - 1]$,

$$\mathbb{E}[\Delta Z_i(t) \mid \mathcal{F}_t, \text{op}_t = \text{Op}_2] = r \cdot \left(-\frac{iZ_i}{M} + \frac{(i+1)Z_{i+1}}{M} \cdot (1 - P) \right) + O\left(\frac{1}{n}\right) \quad (5.6)$$

$$\mathbb{E}[\Delta Z_r(t) \mid \mathcal{F}_t, \text{op}_t = \text{Op}_2] = -1 + r \cdot \left(-\frac{rZ_r}{M} \right) + O\left(\frac{1}{n}\right) \quad (5.7)$$

$$\mathbb{E}[\Delta L(t) \mid \mathcal{F}_t, \text{op}_t = \text{Op}_2] = r \cdot \left(-\frac{(r-1)L}{M} + \frac{rZ_r}{M} \cdot P \right) + O\left(\frac{1}{n}\right) \quad (5.8)$$

$$\mathbb{E}[\Delta F(t) \mid \mathcal{F}_t, \text{op}_t = \text{Op}_2] = \left(\frac{Z}{M}\right)^r + r \cdot \left(\frac{Z}{M}\right)^{r-1} \cdot \left(1 - \frac{Z}{M}\right) + O\left(\frac{1}{n}\right) \quad (5.9)$$

$$\mathbb{E}[\Delta M(t) \mid \mathcal{F}_t, \text{op}_t = \text{Op}_2] = -2r \quad (5.10)$$

The explanations for these are similar to those for Op_1 . The main differences are

that (i) we now reveal r pairs from the configuration model rather than $r - 1$, and (ii) a step is still a success if one of the revealed points pairs with a T (i.e., “non- Z ”) vertex.

5.3 Trajectories from Differential Equations

For ease of notation, we let $a = r + 3$ and set $\mathbf{Z} = (Z_1, \dots, Z_a) = (Z_1, \dots, Z_r, L, F, M)$. For $i \in [a]$ and $j \in [2]$, we define $f_{i,j}(t/n, Z_1(t)/n, \dots, Z_a(t)/n)$ to be the expression given on the right hand side shown in (5.1)-(5.10) for $\mathbb{E}[\Delta Z_i(t) | \mathcal{F}_t, \text{op}_t = \text{Op}_j]$ ignoring the $O(1/n)$ terms so as to remove the dependence on n . So for example, letting $x = t/n$, $\mathbf{z} = (z_1, \dots, z_a)$ and $z = \sum_{i=1}^r iz_i$ where $z_i(x) = Z_i(t)/n$, we have $f_{i,1}(x, \mathbf{z}) = (r - 1) \left(-\frac{iz_i}{z_{r+3}} + \frac{(i+1)z_{i+1}}{z_{r+3}} \cdot \left(1 - \left(\frac{z}{z_{r+3}} \right)^{r-2} \right) \right)$ for $i \in [r - 1]$.

To prove concentration of our random variables, we will use the differential equations method. Wormald proved the following theorem, which is often used as a black box in such situations. The theorem is stated as it appears in [8].

Theorem 5.3.1 (Wormald [21]). *Given random variables Y_i , for $1 \leq y \leq a$ (where $a > 0$ is a constant), representing components of a time discrete Markov process $\{G_t\}_{t \geq 0}$, assume that $D \subseteq \mathbb{R}^{a+1}$ is closed and bounded and contains the set*

$$\{(0, y_1, \dots, y_a) : \Pr[Y_i(0) = y_i n, 1 \leq i \leq a] \neq 0 \text{ for some } n\}.$$

If for all i and all t , $|Y_i(t + 1) - Y_i(t)| \leq \beta$ for some constant β and $|\mathbb{E}[Y_i(t + 1) - Y_i(t) | G_t] - f_i(t/n, Y_1(t)/n, \dots, Y_a(t)/n)| \leq \lambda$ for some $\lambda = o(1)$ and some functions f_i which are Lipschitz continuous on an open set containing D , then, for $(0, \hat{z}_1, \dots, \hat{z}_a) \in D$, the system of differential equations

$$\frac{dz_i}{dx} = f_i(x, z_1, \dots, z_a)$$

has a unique solution for $z_i : \mathbb{R} \rightarrow \mathbb{R}$ in D satisfying the initial condition $z_i(0) = \hat{z}_i$.
 Moreover, asymptotically almost surely

$$Y_i(t) = nz_i(t/n) + o(n),$$

uniformly for all t and all i , where $z_i(0) = \hat{z}_i = \frac{1}{n}Y_i(0)$.

The details of the following application of the differential equations method are omitted, but they are by now standard (see for example [10, 11, 22, 23]). In Phase 1, we have an ordinary application of Theorem 5.3.1. In Phase 2, we have a prioritized algorithm that performs a mixture of 2 types of steps with preference given to a particular type of step. The results of Wormald [23] essentially say that we may instead analyze a deprioritized algorithm that selects vertices according to a pre-determined probability function (which in effect blends the two types of steps appropriately). We note that our functions $f_{i,j}$ are well behaved (e.g., they have continuous and bounded derivatives) as long as $z_{r+3} = M/n$ stays bounded away from 0. As our numerical solutions show¹, this is the case for all values of r considered in Table 5.1.

We now describe the blending of the steps for Phase 2 as described in [23]. Suppose in Phase 2, an Op_2 creates, in expectation, α many vertices of L and suppose that performing an Op_1 decreases the number of L vertices, in expectation, by τ . Then we would expect an Op_2 to be followed by α/τ many Op_1 steps. Then in Phase 2, we would expect the proportion of Op_2 steps to be $1/(1 + \alpha/\tau) = \tau/(\tau + \alpha)$ and the proportion of Op_1 steps to be $\alpha/(\tau + \alpha)$. Let $x = t/n$ and let $\mathbf{z} = (z_1, \dots, z_a)$. Then (recalling that L is now represented by Z_{r+1}) the asymptotic values of α and τ are given by

$$\alpha = f_{r+1,2}(x, \mathbf{z}), \quad \tau = -f_{r+1,1}(x, \mathbf{z}).$$

¹A Maple worksheet that can be used to verify the claimed results can be found at <https://msuweb.montclair.edu/~bald/research.html>

We let

$$p := \frac{\alpha}{\tau + \alpha}$$

and

$$F(x, \mathbf{z}, i, k) = \begin{cases} f_{i,1}(x, \mathbf{z}) & \text{if } k = 1 \\ p \cdot f_{i,1}(x, \mathbf{z}) + (1 - p) \cdot f_{i,2}(x, \mathbf{z}) & \text{if } k = 2. \end{cases}$$

Then for Phase $k \in [2]$, and $i \in [a]$, we let

$$\frac{d\tilde{z}_i}{dx} = F(x, \tilde{\mathbf{z}}, i, k) \tag{5.11}$$

with initial conditions for Phase 1 given by $\tilde{z}_i(0) = 0$ for $i \in \{1, \dots, r-1, r+1, r+2\}$, $\tilde{z}_r(0) = 1$, $\tilde{z}_{r+3}(0) = r$. Numerical solutions of the Phase 1 system show that in this phase, \tilde{z}_{r+1} increases and then decreases until it hits zero at which time Phase 2 begins. Let $\rho_1^r > 0$ be the first time when $\tilde{z}_{r+1}(\rho_1^r) = 0$. The numerical solutions show that all other tracked variables are bounded away from 0 at time ρ_1^r . The initial conditions for Phase 2 are given by the final values of Phase 1, i.e., $\tilde{z}_i(\rho_1^r)$ for all $i \in [a]$. Numerical solutions of the Phase 2 system then imply that Phase 2 ends at a time $\rho_2 = \rho_2^r$, when $\tilde{z}_r(\rho_2) = 0$. A more detailed explanation of the solutions in the cases $r = 3$ and 4 can be found in Section 5.4. The conclusion of the differential equations method is that, by Theorem 5.3.1,

$$Z_i(t) = n\tilde{z}_i(t/n) + o(n)$$

for all i and for all $0 \leq t \leq \rho_2^r n$. The number of full degree vertices can then be represented by

$$f_r n = \tilde{z}_{r+2}(\rho_2^r) \cdot n.$$

We have now completed the proof of Theorem 5.0.4.

5.4 Discussion of $r = 3, 4$

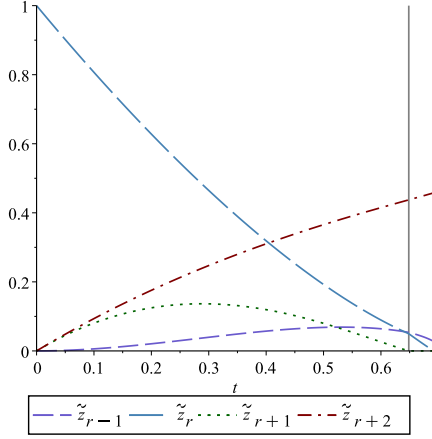


Figure 5.2: Trajectories for $r = 3$

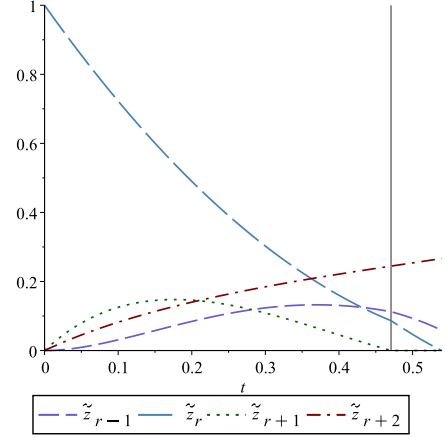


Figure 5.3: Trajectories for $r = 4$

In Figure 5.2 we see some of the solutions to the system given in (5.11) for $G \sim G(n, 3)$. Phase 1 ends at the vertical line at time $\rho_1 \approx 0.6485$. The initial conditions for Phase 2 are then given by $\tilde{z}_1(\rho_1) \approx 0.0193$, $\tilde{z}_2(\rho_1) \approx 0.0536$, $\tilde{z}_3(\rho_1) \approx 0.0498$, $\tilde{z}_4(\rho_1) = 0$, $\tilde{z}_5(\rho_1) \approx 0.4375$, $\tilde{z}_6(\rho_1) \approx 0.4060$. Thus at the end of Phase 1, the algorithm has found a forest with $\approx 0.4375n$ vertices of full degree. We note that this already implies that w.h.p. $\lambda(G) \geq 0.4375n$ and $\gamma_G(G) \leq 0.5625n$, an improvement over the best known bounds given in [9] for these problems. Phase 2 ends when $\tilde{z}_3 = 0$ at time $\rho_2 \approx 0.6922$. As one can see, \tilde{z}_2 (and hence \tilde{z}_{r+3}) is bounded away from 0 at time ρ_2 .

In Figure 5.3 we see some of the solutions to the system given in (5.11) for $G \sim G(n, 4)$. Phase 1 ends at the vertical line at time $\rho_1 \approx 0.4707$. The initial conditions for Phase 2 are then given by $\tilde{z}_1(\rho_1) \approx 0.0119$, $\tilde{z}_2(\rho_1) \approx 0.0548$, $\tilde{z}_3(\rho_1) \approx 0.1124$, $\tilde{z}_4(\rho_1) \approx 0.0864$, $\tilde{z}_5(\rho_1) = 0$, $\tilde{z}_6(\rho_1) \approx 0.2445$, $\tilde{z}_7(\rho_1) \approx 1.1757$. Phase 2 ends when $\tilde{z}_4 = 0$ at time $\rho_2 \approx 0.5397$.

5.5 Further Directions

We point out that some of the functions in the Phase 1 system given in Section 5.2.1 can be solved for analytically. In particular, we have, using the initial conditions $\tilde{z}_{r+3}(0) = r$ and $\tilde{z}_r(0) = 1$, that in Phase 1,

$$\tilde{z}_{r+3}(x) = r - 2(r-1)x$$

and hence

$$\tilde{z}_r(x) = \left(1 - \frac{2(r-1)}{r}x\right)^{r/2}.$$

Unfortunately, the equations for the other variables depend on $z = \sum_i iz_i$, which we don't currently see how to deal with.

It is an intriguing open problem to determine whether random regular graphs contain full degree spanning trees with an optimal (or asymptotically optimal) number of full degree vertices.

Problem 5.5.1. *Does $G \sim G(n, r)$ satisfy $\varphi(G) = \frac{n}{r-1}(1 + o(1))$?*

Perhaps applying the second moment method to a particular tree with only vertices of degree r and 1 could be used to show this. We note that as a function of r , our f_r seem to decay at a rate faster than $1/r$, so it seems unlikely that a simple modification of Algorithm 1 will succeed in proving this.

The lower bound in Theorem 2.2.3 can be thought of as arising from a greedy algorithm that removes a vertex, and its first and second neighborhoods, at each step. By analyzing a slightly more sophisticated algorithm (see Lemma 5.2 in [2]) that allows the neighborhoods of the chosen vertices to overlap in one vertex, one can prove the lower bound $\varphi(G) \geq \frac{2}{\Delta^2 + \Delta + 2} \cdot n = (1 + o_\Delta(1)) \frac{2}{\Delta^2} \cdot n$. This lower bound is almost tight (up to a constant factor) as can be seen by the following construction

which essentially appears in [4]. Let $G = (K_{\Delta/2} \square K_{\Delta/2}) \square C_{4n/\Delta^2}$. Here, one can take at most one vertex to be of full degree from each copy of $K_{\Delta/2} \square K_{\Delta/2}$ and so $\varphi(G) \leq 4n/\Delta^2$. It would be interesting to improve this factor of 2.

Problem 5.5.2. *Determine the best possible deterministic lower bound for $\varphi(G)$ for all connected graphs with $\Delta(G) = \Delta$.*

Chapter 6

The Many Leaves Spanning Tree

Problem

Recall that the many leaves problem focuses on determining $\lambda(G)$, which is the maximum number of leaves in any spanning tree of a graph G . The many leaves problem is related to the FDST problem by Proposition 5.0.3 and is, in general, NP-hard. This problem has been very well studied. For every connected cubic graph G , Storer [20] found that $\lambda(G) \geq \lceil (n/4) + 2 \rceil$. As stated in [13], Linial generalized Storer's result by conjecturing that, for a graph G with minimum degree δ , $\lambda(G) \geq \frac{\delta-2}{\delta+1}n + c_\delta$, where c_δ depends on δ . Kleitman and West [16] proved Linial's conjecture for $\delta = 3$ and $c_\delta = 2$. An *isomorphism* between a graph G and a graph H is a bijection $f : V(G) \rightarrow V(H)$ such that $uv \in E(G)$ iff $f(u)f(v) \in E(H)$. If there is an isomorphism from G to H , then G is *isomorphic* to H . Griggs et al. [13] found that $\lambda(G) \geq \lceil (n/3) + (4/3) \rceil$ for every connected cubic graph G that does not have a subgraph isomorphic to K_4 with one edge removed. Griggs and Wu [14] determined that $\lambda(G) \geq (2n/5 + 8/5)$ for any connected graph G with minimum degree $\delta = 4$. They also determined that $\lambda(G) \geq n/2 + 2$ for any connected graph G with $\delta = 5$.

We have the following general upper bound for any connected regular graph, which gives rise to the deterministic upper bounds u_r given in Table 6.1.

Proposition 6.0.1. *For a connected r -regular graph G on n vertices,*

$$\lambda(G) \leq \left(1 - \frac{1}{r-1}\right) n + \frac{2}{r-1}.$$

Proof. Let G be a connected r -regular graph on n vertices, T be a spanning tree of G , and L be the leaf set of T . Since all non-leaf vertices have degree at most r in T ,

$$\begin{aligned} 2(n-1) &= \sum_v d_T(v) = \sum_{v \in L} d_T(v) + \sum_{v \notin L} d_T(v) \\ &\leq 1 \cdot |L| + r \cdot (n - |L|) \\ &= (1-r) \cdot |L| + rn. \end{aligned}$$

Solving for $|L|$, we obtain

$$|L| \leq \left(1 - \frac{1}{r-1}\right) n + \frac{2}{r-1}.$$

Therefore, $\lambda(G) \leq \left(1 - \frac{1}{r-1}\right) n + \frac{2}{r-1}$. □

Recall that the many leaves problem is equivalent to the connected dominating set problem by Proposition 5.0.2. The connected dominating set problem has been well-studied; of the most relevance to our results, Duckworth and Mans [10] developed an algorithm for the connected dominating set problem and analyzed it on random d -regular graphs for d fixed, as discussed in Chapter 5. Bounds on γ_C for various $d \geq 4$ are given in Table 1 of [10], and bounds for λ can be obtained from this table by applying Proposition 5.0.2. These bounds are given by λ_{dm} in Table 6.1 of this chapter.

In Chapter 5, we studied φ by considering random regular graphs. Here, we will study the average behavior of λ for r -regular graphs in the same manner. We provide an algorithm that attempts to maximize the number of leaves in spanning trees of random regular graphs. This algorithm yields a decent improvement over the bounds in [9, 10]. Algorithm 2 is a modified breadth first search and iteratively builds a forest T . At each step, a random vertex v of a certain type is chosen and v 's neighbors are exposed (some neighbors may have been previously exposed). If none of v 's newly exposed neighbors lie in the current forest T , then these neighbors may be safely added to T , creating new leaves. Theorem 6.0.2 is the main result of this chapter and gives the bounds determined by Algorithm 2.

Input: Connected r -regular graph $G = (V, E)$.
Output: Tree T with leaves L .
Select arbitrary $v \in V$;
 $T = S_G(v)$;
 $Y_{r-1} = N_G(v)$, $L = N_G(v)$, $Z_r = V \setminus V_T$;
 $Y_k = \emptyset$ for $k \in \{1, 2, \dots, r-2\}$, $Z_k = \emptyset$ for $k \in \{1, 2, \dots, r-1\}$;
while $\bigcup_{j=1}^{r-1} Y_j \neq \emptyset$ **do**
 Select $v \in Y_j$ u.a.r. for the largest j s.t. $Y_j \neq \emptyset$;
 $Y_j = Y_j \setminus \{v\}$;
 if $|V_T \cap N_G(v)| \leq r - j$ **then**
 $T = T \cup S_G(v)$;
 $U = \{u \in N_G(v) : u \notin T\}$;
 $L = \{L \cup U\} \setminus \{v\}$;
 for $x \in N_G(v)$ **do**
 if $x \in Z_i$, move x from Z_i to Y_{i-1} ;
 if $x \in Y_i$, move x from Y_i to Y_{i-1} ;
 end
 else
 for $x \in N_G(v)$ **do**
 if $x \in Z_i$, move x from Z_i to Z_{i-1} ;
 if $x \in Y_i$, move x from Y_i to Y_{i-1} ;
 end
 end
end
Complete the forest T to a spanning tree arbitrarily;

Algorithm 2: Many Leaves

Theorem 6.0.2. *When run on a random r -regular graph $G(n, r)$ with $3 \leq r \leq 6$, Algorithm 2 w.h.p. produces a tree with at least $l_r n$ leaves, where values of l_r are shown in Table 6.1.*

r	λ_{dm}	l_r	u_r
3	.41458	.4559	.5000
4	.53439	.5872	.6666
5	.61393	.6524	.7500
6	.66065	.6933	.8000

Table 6.1: Values of l_r, u_r such that w.h.p. $l_r n \leq \lambda(G(n, r)) \leq u_r n$. The bounds l_r come from Algorithm 1 and u_r deterministically from Proposition 6.0.1. The bounds λ_{dm} are from [10].

As we did for Theorem 5.0.4, we use the differential equations method to prove Theorem 6.0.2. We describe the expected one-step changes of several parameters throughout the algorithm in Section 6.2. Then, in Section 6.3, we apply Theorem 5.3.1 to show concentration of this system of random variables around their expected trajectories in order to complete the proof of Theorem 6.0.2.

6.1 Setting up the Analysis of Algorithm 2

As described in Section 5.1, we make use of the configuration model to analyze our algorithm on $G(n, r)$. We again track certain parameters throughout the execution of the algorithm and only reveal partial information about $G(n, r)$ as the algorithm progresses. Each iteration of the while loop is again called a *step*. At each step, we *process* a vertex v . When v is processed, we reveal its neighbors in the configuration (some of v 's neighbors may already have been revealed). Let T represent the current forest being built. For $i \in [r]$, let $Y_i = Y_i(t)$ denote the set of vertices that are in T and have i unrevealed configuration point neighbors at time step t . Define $Y_r := 0$, as a vertex cannot simultaneously be in T and have r unexposed neighbors. Let $Z_i = Z_i(t)$ denote the set of vertices that are not in T and have i unrevealed neighbors at time

step t . A step is a *success* if at most $r - i$ of v 's neighbors already lie in T . In this case, we may add v 's neighbors to T . We let $L = L(t)$ represent the set of all leaves of T at time step t .

We consider our algorithm to have $r - 1$ phases. In Phase 1, we only process vertices from Y_{r-1} . We say Phase k begins when the first vertex from Y_{r-k} is processed. In Phase k , we process vertices from $Y_{r-k}, \dots, Y_{r-2}, Y_{r-1}$.

6.2 Expected One Step Changes

In a common abuse of notation, we refer to L , Y_i , and Z_i as the sets they represent as well as the sizes of those sets. Let

$$Z := \sum_{i=1}^r iZ_i$$

so that Z represents the number of unrevealed configuration points corresponding to the Z_i 's. Let

$$Y := \sum_{i=1}^r iY_i$$

so that Y represents the number of unrevealed configuration points corresponding to the Y_i 's. Let $M = M(t)$ represent the number of unrevealed configuration points at time step t . There are $r - 1$ types of operations that we perform. Following the same notation as used in Section 5.2, for $j \in [r - 1]$, let Op_j denote the operation of processing a vertex from Y_j . Let $\text{op}_t \in \{\text{Op}_1, \text{Op}_2, \dots, \text{Op}_{r-1}\}$ represent the operation performed at time step t of the algorithm.

Let \mathcal{F}_t represent the revealed part of the configuration model at time t . In Phase k , we only process vertices from $Y_{r-k}, \dots, Y_{r-2}, Y_{r-1}$ and so we only perform operations from $\{\text{Op}_{r-k}, \text{Op}_{r-k+1}, \dots, \text{Op}_{r-1}\}$. For random variable $X = X(t)$, let $\Delta X(t) =$

$X(t+1) - X(t)$. Recall that $Y_r := 0$ and let

$$\mathbb{1}_{i,j} = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{otherwise.} \end{cases}$$

The following equations, which define the expected one step changes for each of our parameters, hold in all $r - 1$ phases. For $i \in [r - 1]$,

$$\begin{aligned} \mathbb{E}[\Delta Z_i(t) \mid \mathcal{F}_t, \text{op}_t = \text{Op}_j] = \\ j \cdot \left(-\frac{iZ_i}{M} + \frac{(i+1)Z_{i+1}}{M} \cdot \left(1 - \left(\frac{Z}{M} \right)^{j-1} \right) \right) + O\left(\frac{1}{n}\right) \end{aligned} \quad (6.1)$$

$$\mathbb{E}[\Delta Z_r(t) \mid \mathcal{F}_t, \text{op}_t = \text{Op}_j] = j \cdot \left(-\frac{rZ_r}{M} \right) + O\left(\frac{1}{n}\right) \quad (6.2)$$

$$\begin{aligned} \mathbb{E}[\Delta Y_i(t) \mid \mathcal{F}_t, \text{op}_t = \text{Op}_j] = \\ -\mathbb{1}_{i,j} + j \cdot \left(\frac{Z^{j-1}(-iY_i + (i+1)Y_{i+1} + (i+1)Z_{i+1})}{M^j} \right) + O\left(\frac{1}{n}\right) \end{aligned} \quad (6.3)$$

$$\mathbb{E}[\Delta L(t) \mid \mathcal{F}_t, \text{op}_t = \text{Op}_j] = (j-1) \cdot \left(\frac{Z}{M} \right)^j + O\left(\frac{1}{n}\right) \quad (6.4)$$

$$\mathbb{E}[\Delta M(t) \mid \mathcal{F}_t, \text{op}_t = \text{Op}_j] = -2j. \quad (6.5)$$

In (6.1) and (6.2), note that the vertex $v \in Y_j$ that we are processing has j unrevealed configuration points. If any of these points pair with a point in a Z_i vertex, then that vertex is no longer a Z_i vertex. This happens with probability $\frac{iZ_i + O(1)}{M + O(1)} = \frac{iZ_i}{M} + O(1/n)$ where the error term is $O(1/n)$ since we will assume that $M = \Omega(n)$. The error terms in the rest of the explanation are similar and will be ignored. We gain a Z_i vertex if the step is not a success but one of the revealed points pairs with a Z_{i+1} point. The factor $(i+1)Z_{i+1}/M$ represents the probability that a point pairs with a Z_{i+1} point and $1 - \left(\frac{Z}{M}\right)^{j-1}$ represents the probability that the step is not a success (i.e., the other revealed points do not all land in Z).

In (6.3), the $-\mathbb{1}_{i,j}$ accounts for the loss of the vertex that we are processing if this vertex is from Y_i . Again, we will lose Y_i vertices when the revealed points pair to Y_i vertices. We gain Y_i vertices if the step is a success and one of the points pairs with a Z_{i+1} vertex. We also gain Y_i vertices if one of the points pairs with a Y_{i+1} vertex, which could happen if the step is a failure. To see (6.4), note that we gain $j - 1$ leaves when the step is a success, the probability of which is $(\frac{Z}{M})^j$. Finally, note that at each step we reveal j pairs from the configuration, which explains (6.5).

6.3 Trajectories from Differential Equations

For ease of notation, we set $\mathbf{Z} = (Z_1, \dots, Z_a) = (Z_1, \dots, Z_r, Y_1, \dots, Y_r, L, M)$ where $a = 2r + 2$. For $i \in [a]$ and $j \in [r - 1]$, we define $f_{i,j}(t/n, Z_1(t)/n, \dots, Z_a(t)/n)$ to be the expression given on the right hand side shown in (6.1)-(6.5) for $\mathbb{E}[\Delta Z_i(t) | \mathcal{F}_t, \text{op}_t = \text{Op}_j]$ ignoring the $O(1/n)$ terms so as to remove the dependence on n . So for example, letting $x = t/n$, $\mathbf{z} = (z_1, \dots, z_a)$ and $z = \sum_{i=1}^r iz_i$ where $z_i(x) = Z_i(t)/n$, we have $f_{i,r-1}(x, \mathbf{z}) = (r - 1) \left(-\frac{iz_i}{z_{2r+2}} + \frac{(i+1)z_{i+1}}{z_{2r+2}} \cdot \left(1 - \left(\frac{z}{z_{2r+2}} \right)^{r-2} \right) \right)$ for $i \in [r - 1]$.

Again, the details of the following application of the differential equations method have been omitted. Algorithm 2 is a prioritized algorithm that performs a mixture of $r - 1$ types of steps, but recall that we may instead analyze a deprioritized algorithm that effectively blends the steps appropriately. We note that our functions $f_{i,j}$ are well behaved (e.g., they have continuous and bounded derivatives) as long as $z_{2r+2} = M/n$ stays bounded away from 0. As our numerical solutions show, this is the case for all values of r considered in Table 6.1 until the end of the final phase. Thus, we solve our equations numerically in the same manner as in [1] and do not obtain strict inequalities involving the derivatives at the termination of each phase. For all phases except the last one, z_{2r+2} stays bounded away from 0, so for the last phase, we only

consider our equations on the domain where $z_{2r+2} \geq \epsilon$ for some small fixed ϵ and all other variables are non-negative.

Below, we describe the blending of the $r - 1$ steps. Again, this process is described in [23]. Since we generally have more than 2 types of steps in each phase, the blending done here is necessarily more complicated than the blending done in Section 5.3. For a fixed k , let $\tau_{j,k}$ be the proportion of steps that process a vertex from Y_j during Phase k . Then we have the following system of equations for $\tau_{j,k}(x, z_1, z_2, \dots, z_a) = \tau_{j,k}(x, \mathbf{z})$, $r - k \leq j \leq r - 1$:

$$1 = \sum_{j=r-k}^{r-1} \tau_{j,k}(x, \mathbf{z}), \quad (6.6)$$

$$0 = \sum_{j=r-k}^{r-1} \tau_{j,k}(x, \mathbf{z}) \cdot f_{i,j}(x, \mathbf{z}), \text{ for } r - k + 1 \leq i \leq r - 1. \quad (6.7)$$

Letting $x = t/n$ and $z_i(x) = Z_i(t)/n$ as described earlier, we have that the solution $\tau_{j,k}$ to the system given in (6.6) and (6.7) is the proportion of steps that process a vertex from Y_j during Phase k as desired. Recall that in Phase k , we only process vertices from $Y_{r-k}, \dots, Y_{r-2}, Y_{r-1}$ and so $\sum_{j=r-k}^{r-1} \tau_{j,k}(x, \mathbf{z}) = 1$, which explains (6.6). Also, recall that during Phase k vertices from $Y_{r-k+1}, \dots, Y_{r-1}$ are essentially processed as soon as they are created. That is, we will never amass these types of vertices and so we can expect $\sum_{j=r-k}^{r-1} \tau_{j,k}(x, \mathbf{z}) \cdot f_{i,j}(x, \mathbf{z}) = 0$ for $r - k + 1 \leq i \leq r - 1$, which explains (6.7). Thus, by the differential equations method, we have that each z_i should approximately be \tilde{z}_i , where the \tilde{z}_i are deterministic functions that satisfy the following system of differential equations:

$$\frac{dz_i}{dx} = F(x, \tilde{\mathbf{z}}, i, k) := \sum_{j=r-k}^{r-1} \tau_{j,k}(x, \tilde{\mathbf{z}}) \cdot f_{i,j}(x, \tilde{\mathbf{z}}) \quad (6.8)$$

with initial conditions $\tilde{z}_i(0) = 0$ for $i \in \{1, \dots, r - 1, r + 1, \dots, 2r + 1\}$, $\tilde{z}_r(0) = 1$,

and $\tilde{z}_{2r+2}(0) = r$. Numerical solutions show that in Phase 1, \tilde{z}_{2r-1} increases and then decreases until it hits zero, at which time Phase 2 begins. Let $\rho_1^r > 0$ be the first time that $\tilde{z}_{2r-1}(\rho_1^r) = 0$. The initial conditions for Phase 2 are given by the final values of Phase 1, i.e. $\tilde{z}_i(\rho_1^r)$ for all $i \in [a]$. Likewise, the numerical solutions imply that Phase k ends at time $\rho_k = \rho_k^r$ when $\tilde{z}_{2r-k}(\rho_k) = 0$. The initial conditions for Phase $k+1$ are given by the final values of Phase k , i.e. $\tilde{z}_i(\rho_k)$ for all $i \in [a]$. The conclusion of the differential equations method is that, by Theorem 5.3.1,

$$Z_i(t) = n\tilde{z}_i(t/n) + o(n)$$

for all i and for all $0 \leq t \leq \rho_{r-1}^r n$. The number of leaves can then be represented by

$$l_r n = \tilde{z}_{2r+1}(\rho_{r-1}^r) \cdot n.$$

We have now completed the proof of Theorem 6.0.2.

6.4 Discussion of $r = 3, 4$

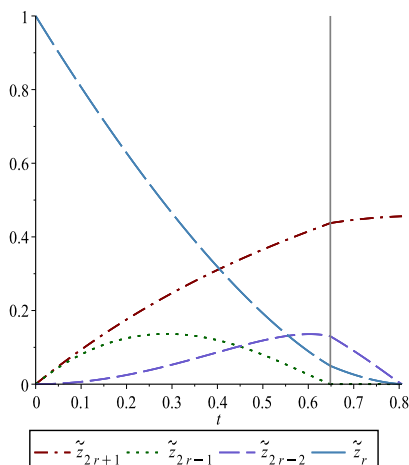


Figure 6.1: Trajectories for $r = 3$

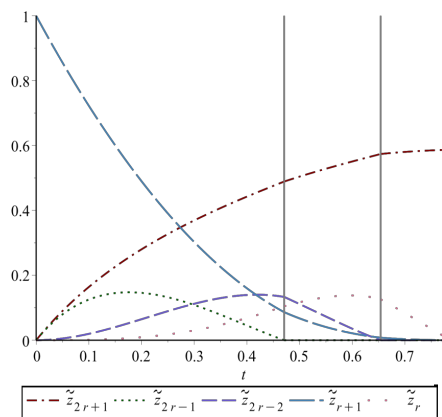


Figure 6.2: Trajectories for $r = 4$

Figure 6.1 shows some of the solutions to the system given in (6.8) for $G \sim G(n, 3)$. Phase 1 ends at the vertical line at time $\rho_1 \approx 0.6485$. The initial conditions for Phase 2 are then given by $\tilde{z}_1(\rho_1) \approx 0.0193$, $\tilde{z}_2(\rho_1) \approx 0.0536$, $\tilde{z}_3(\rho_1) \approx 0.0498$, $\tilde{z}_4(\rho_1) \approx 0.1300$, $\tilde{z}_5(\rho_1) = 0$, $\tilde{z}_6(\rho_1) = 0$, $\tilde{z}_7(\rho_1) \approx 0.4375$, and $\tilde{z}_8(\rho_1) \approx 0.4060$. Phase 2 ends when $\tilde{z}_4 = 0$ at time $\rho_2 \approx 0.8062$.

Figure 6.2 shows some of the solutions to the system given in (6.8) for $G \sim G(n, 4)$. Phase 1 ends at the vertical line at time $\rho_1 \approx 0.4707$. The initial conditions for Phase 2 are then given by $\tilde{z}_1(\rho_1) \approx 0.0119$, $\tilde{z}_2(\rho_1) \approx 0.0548$, $\tilde{z}_3(\rho_1) \approx 0.1124$, $\tilde{z}_4(\rho_1) \approx 0.0864$, $\tilde{z}_5(\rho_1) \approx 0.1046$, $\tilde{z}_6(\rho_1) \approx 0.1334$, $\tilde{z}_7(\rho_1) = 0$, $\tilde{z}_8(\rho_1) = 0$, $\tilde{z}_9(\rho_1) \approx 0.4890$, and $\tilde{z}_{10}(\rho_1) \approx 1.1757$. Phase 2 ends when $\tilde{z}_6 = 0$ at time $\rho_2 \approx 0.6537$, represented by the second vertical line. The initial conditions for Phase 3 are then given by $\tilde{z}_1(\rho_1) \approx 0.0257$, $\tilde{z}_2(\rho_1) \approx 0.0411$, $\tilde{z}_3(\rho_1) \approx 0.0292$, $\tilde{z}_4(\rho_1) \approx 0.0078$, $\tilde{z}_5(\rho_1) \approx 0.1261$, $\tilde{z}_6(\rho_1) = 0$, $\tilde{z}_7(\rho_1) = 0$, $\tilde{z}_8(\rho_1) = 0$, $\tilde{z}_9(\rho_1) \approx 0.5739$, and $\tilde{z}_{10}(\rho_1) \approx 0.3526$. Phase 3 ends when $\tilde{z}_5 = 0$ at time $\rho_2 \approx 0.7884$.

6.5 Further Directions

We note that Table 6.1 does not present bounds for extensive values of r . At present, our Maple worksheet cannot compute solutions to (6.8) for $r \geq 7$. We will be exploring other means of generating numerical solutions to our system for higher r values.

The bounds for λ presented in Table 6.1 are an improvement over the bounds for λ that can be obtained from the φ bounds in Table 5.1 for $r > 3$. However, for $r = 3$, Algorithm 1 produces a better bound than Algorithm 2, which leads us to wonder whether there is a third algorithm that gives improved bounds for all r .

Problem 6.5.1. *Develop an algorithm that produces improved lower bounds on $\lambda(G)$, where $G \sim G(n, r)$, for all values of r .*

Bibliography

- [1] N. Alon, P. Pralat, and N. Wormald. Cleaning regular graphs with brushes. *SIAM J. Discrete Math.*, 23(1):233–250, 2008/09.
- [2] D. Bal and E. Schudrich. On the size Ramsey number of all cycles versus a path. *Discrete Math.*, 344(5):Paper No. 112322, 10, 2021.
- [3] M. R. Best, A. E. Brouwer, F. J. MacWilliams, A. M. Odlyzko, and N. J. A. Sloane. Bounds for binary codes of length less than 25. *IEEE Trans. Inform. Theory*, IT-24(1):81–93, 1978.
- [4] R. Bhatia, S. Khuller, R. Pless, and Y. J. Sussmann. The full-degree spanning tree problem. *Networks*, 36(4):203–209, 2000.
- [5] H. Broersma, O. Koppius, H. Tuinstra, A. Huck, T. Kloks, D. Kratsch, and H. Müller. Degree-preserving trees. *Networks*, 35(1):26–39, 2000.
- [6] Y. Caro, D. B. West, and R. Yuster. Connected domination and spanning trees with many leaves. *SIAM J. Discrete Math.*, 13(2):202–211, 2000.
- [7] S. Choi and P. Guan. A spanning tree of the 2^m -dimensional hypercube with maximum number of degree-preserving vertices. *Discrete Math.*, 117(1-3):275–277, 1993.

- [8] J. Díaz and D. Mitsche. The cook-book approach to the differential equation method. *Computer Science Review*, 4(3):129–151, 2010.
- [9] W. Duckworth. Minimum connected dominating sets of random cubic graphs. *Electron. J. Combin.*, 9(1):Research Paper 7, 13, 2002.
- [10] W. Duckworth and B. Mans. Connected domination of regular graphs. *Discrete Math.*, 309(8):2305–2322, 2009.
- [11] W. Duckworth and M. Zito. Large independent sets in random regular graphs. *Theoret. Comput. Sci.*, 410(50):5236–5243, 2009.
- [12] S. Gaspers, S. Saurabh, and A. A. Stepanov. A moderately exponential time algorithm for full degree spanning tree. In *Theory and applications of models of computation*, volume 4978 of *Lecture Notes in Comput. Sci.*, pages 479–489. Springer, Berlin, 2008.
- [13] J. R. Griggs, D. J. Kleitman, and A. Shastri. Spanning trees with many leaves in cubic graphs. *J. Graph Theory*, 13(6):669–695, 1989.
- [14] J. R. Griggs and M. Wu. Spanning trees in graphs of minimum degree 4 or 5. *Discrete Math.*, 104(2):167–183, 1992.
- [15] J. Guo, R. Niedermeier, and S. Wernicke. Fixed-parameter tractability results for full-degree spanning tree and its dual. *Networks*, 56(2):116–130, 2010.
- [16] D. J. Kleitman and D. B. West. Spanning trees with many leaves. *lecture at SIAM Discrete Math. Conf. San Francisco*, 1988.
- [17] M. Lewinter. Interpolation theorem for the number of degree-preserving vertices of spanning trees. *IEEE Trans. Circuits and Systems*, 34(2):205, 1987.
- [18] D. Lokshtanov, V. Raman, S. Saurabh, and S. Sikdar. On the directed full degree spanning tree problem. *Discrete Optim.*, 8(1):97–109, 2011.

- [19] I. W. Pothof and J. Schut. Graph-theoretic approach to identifiability in a water distribution network. *Univesity of Twente Memorandum*, (1283), 1995.
- [20] J. A. Storer. Constructing full spanning trees for cubic graphs. *Inform. Process. Lett.*, 13(1):8–11, 1981.
- [21] N. C. Wormald. Differential equations for random processes and random graphs. *Ann. Appl. Probab.*, 5(4):1217–1235, 1995.
- [22] N. C. Wormald. Models of random regular graphs. 267:239–298, 1999.
- [23] N. C. Wormald. Analysis of greedy algorithms on graphs with bounded degrees. *Discrete Math.*, 273(1-3):235–260, 2003. EuroComb'01 (Barcelona).