Structure in Stable Matching Problems

by

William Justin Toth

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Master of Mathematics
in
Combinatorics and Optimization

Waterloo, Ontario, Canada, 2016

Abstract

In this thesis we provide two contributions to the study of structure in stable matching problems. The first contribution is a short new proof for the integrality of Rothblum's linear description of the convex hull of incidence vectors of stable matchings in bipartite graphs. The key feature of our proof is to show that extreme points of the formulation must have a $0, 1$-component.

The second contribution is a computer search procedure for instances of cyclic stable matching problems with three genders as proposed by Knuth. We provide sufficient conditions for the existence of a stable matching in this context. We also investigate bijections of the problem instance vertex set to itself which preserve the set of stable matchings (up to permutation). Such bijections define "symmetric" problem instances. We study this notion of symmetry, and use it to cut down on the number of problem instances in our search. We implemented our proposed computational procedure in Java and end with a discussion of the results running computational experiments using our code on problem instances of size 5.

Acknowledgements

I would like to thank Jochen Koenemann and Kanstantsin Pashkovich. Their guidance has been invaluable.

This research has been supported by the Natural Sciences and Engineering Research Council (NSERC) of Canada.

Dedication

This is dedicated to my parents Bill and Cheryl, and my brother Michael, for their unwavering support in all my endeavours.

Table of Contents

List of Figures

Chapter 1

Introduction

This chapter gives motivation, history of related work, and how our results fit into the current body of knowledge.

1.1 Motivation and Related Work

These days online dating applications are all the rage for eligible singles, but before these apps made finding a date as easy as a swipe of the finger, one way people met potential suitors was through their social circle. Imagine you really like playing cupid and you happen to know n heterosexual male and heterosexual female friends who are all single and looking for dates. Since you're such a good friend you know each of your friends' preferences over who they would like to be paired with in the opposite gender, and even who they deem unacceptable. You would like to set up your single friends together in n pairs and you have a goal: you don't want your friends to mutually circumvent your dating suggestions. Say some are your friends are Adam, Bob, Amy, and Brianne. If you pair Adam and Amy together, and Bob and Brianne, but Adam and Brianne mutually prefer each other to Amy and Bob respectively then they might want to ditch their suggested dates for each other. You hope to avoid this.

In the previous dating game your single friends with their preferences form what is called by economists a two-sided matching market [43]. The hope that your friends don't mutually want to ditch their dates is the condition of stability which is a highly desirable property in matching markets. A pair of friends who mutually prefer each other to their matches

Adam: Brianne > Amy > Carly	Amy : Bob > Adam > Chad
Bob: Brianne > Carly > Amy	Brianne: Adam > Chad > Bob
Chad : Carly > Amy > Brianne	Carly: Chad > Bob > Adam

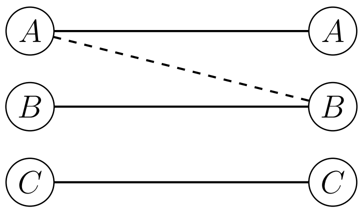

Figure 1.1: A simple example of stable marriage

Solid edges indicate a matching and dashed edge indicates a blocking pair who prefer each other to their math. The pair (Chad, Carly) will appear in any stable matching since they are each other's top choice.

are called a blocking pair, and the existence of a blocking pair prevents stability. Other (perhaps more practical) examples of two-sided matching markets are the matching of medical school residents with hospitals for internships [41], college applicants with colleges [18], and labour markets between employers and employees [43]. The critical features of two-sided matching markets are the presence of two distinct groups without crossover who have interests in forming some mutual relationship with a member of the opposite group, and not only that but each member has preferences who they become involved with in the opposite group.

The abstract mathematical model of these phenomena is called a stable matching problem. In particular the model with two sides and strict preferences, referred to as Stable Marriage was first introduced by Gale and Shapley to investigate college admissions [18]. Formally stable marriage asks for a matching in a bipartite graph such that no two vertices not matched together mutually prefer each other to their partners in the matching. See section 2.3 of chapter 2 for more detail.

In Gale and Shapley's work they prove that stable matchings always exist for stable marriage problems through giving their famous deferred acceptance algorithm to compute stable matchings. See subsection 2.3.2 for details. For this work Shapley earned the Nobel Prize in Economics [10].

Stable matchings in two-sided markets come with a lot of interesting structure. In Gale and Shapley's original paper [18] they observed that their algorithm favoured one side of the market over the other. Deferred acceptance has one side of the market play the role of proposers, and the other side playing the role of proposees. Proposers propose to proposees in order from their first to last choice and proposees tentatively accept or reject. They accept if they have no partner or they prefer the proposer to their current tentative partner. For a formal description see 2.3.2. As it turns out the algorithm gives the proposers the best possible partner they could hope for in any stable matching.

A major landmark in stable marriage was the work of Knuth [30]. Knuth demonstrated that when given two stable matchings, if all members of one side of the market like the first stable matching at least as well as the second then all members of the other side of the market like the second stable matching at least as well as the first. In plainer language the interests of both sides of the market are opposed.

A1: A2 > B2 > C2	A2 : C1 > A1 > B1
B1: B2 > A2 > C2	B2: A1 > B1 > C1
C1 : C1 > A2 > B2	C2: B1 > C1 > A1

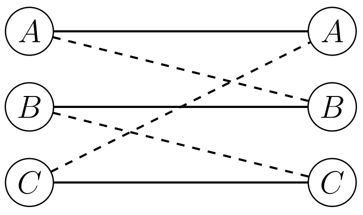

Figure 1.2: Comparing two stable matchings

Solid edges indicate one stable matching and dashed edges indicate another stable matching. The solid matching is preferred by everyone on side 1 and the dashed matching is preferred by everyone on side 2.

In his work Knuth also mentions that the set of stable matchings of a two-sided market forms an algebraic structure called a distributive lattice. He credits Conway with the result. For details of a proof of this result see subsection 2.3.3 of chapter 2 and in particular the proof of theorem 2.3.6.

One line of investigation related to our work in Chapter 3 is to consider involves problems wherein each possible pairing in a stable matching problem is weighted by some possible utility, and our goal is to find a stable matching of maximum total weight. More formally if we let E be the set of possible pairs in a stable matching instance we have a weight function $w : E \to \mathbb{Q}$ and our goal is to find a stable matching $M \subseteq E$ such that $\sum_{e \in M} w(e)$ is maximized over the set of all stable matchings. In a paper of Irving, Leather, and Gusfield [25] they give an algorithm which solves weighted stable marriage by applying network flow theory methods to the set of "rotations" in a problem instance. Another approach to solving weighted stable marriage would be via linear programming. Vande Vate was the first to formulate stable matchings in the stable marriage problem as the feasible region of a linear program and prove that the extreme points of his formulation were integer-valued vectors [49]. In doing so he implies a means of solving the weighted stable matching variant through the use of linear programming techniques. Vande Vate's formulation was only for the case of no unacceptable matches and his proof was complicated. Rothblum gave a formulation for the more general case where unacceptable matches were allowed, and Rothblum's proof via an extreme point argument was simpler than Vande Vate's original proof. [44].

Another line of investigation gaining more attention in recent years is the question of stable matching markets with more than two sides. In Knuth's work on stable matching he proposes a variant where we consider three-sided markets instead of two. In this context matchings are triples with one member from each side of the market, and matchings are stable if there does not exist a triple where each member mutually prefers the triple to the triple they are matched in. One important application for this type of model would be three-way kidney exchange in the context of organ transplant procedures in hospitals [45]. There are essentially two variants to this problem. The first is where each participant has preferences over the possible pairs they could be matched with. Ng and Hirschberg [39] consider this variant in their work and demonstrate that finding stable matchings for such problems in NP-hard.

The other possible variant, which we will focus our attention on in Chapter 4, is the cyclic three-sided matching problem. In this problem participants from the first side of the market has preferences over the members of the second side (with indifference towards the third), the second side has preferences over the third, and the third side has preferences over the first side. It is an open conjecture by Knuth that stable matchings always exist for this problem when there are no unacceptable triples. As late as 2006 Eriksson et al. [15] showed that for complete preference lists with at most 4 agents in each side there always exists a stable matching. In 2010 Biro et al. [7] show a counterexample with 6 agents in each side but with incomplete preferences allowed. Later Farczadi et al. [16] proved that a natural approach through extending Gale and Shapley's deferred acceptance may be intractable by proving that the problem of deciding if any matching of two sides of a three-sided market can be extended to a stable three-sided matching is NP-complete.

1.2 Surrounding Literature

Gale and Shapley's work on stable marriage spawned a field of inquiry that brings together mathematicians, computer scientists, and economists in studying stable matching problems and their variants. We will discuss many of their results in this section. To learn more about the body of literature surrounding stable matching see the excellent texts of Roth and Sotomayor [43], Gusfield and Irving [22], and Manlove [35].

Once Gale and Shapley demonstrated that stable matchings always exist the next natural questions involved what the set of stable matchings in stable marriage problems look like. McVitie and Wilson [37] gave an algorithm which found not just one stable matching, but the entire set of possible stable matchings for a given market. They also observed that in a stable matching where one side of the market are all matched with their best possible partner, like in deferred acceptance, the other side of the market are all matched with their worst possible partner. Another interesting structural result came from Gale and Sotomayor [19] when they demonstrated that if someone was matched in any stable matching they were matched in all stable matchings. An immediate corollary of this is that all stable matchings of a given market have the same size.

Following Conway's Lattice Theorem 2.3.6, Blair [8] demonstrates the amazing fact that any finite distributive lattice is isomorphic to the set of stable matchings of some stable marriage problem. In Blair's paper he gives a construction which takes a finite distributive lattice and constructs a stable marriage problem whose set of stable matchings is isomorphic to the given lattice. Blair notes that his construction is not the most efficient possible in terms of the number of vertices in the stable marriage graph. In particular his construction has potentially exponential growth in the number of vertices in the stable marriage instance in terms of the number of elements of the lattice. Later Gusfield et al. [21] show a more efficient construction which reduces the exponential growth to at worst polynomial growth.

One natural question which arises when first hearing about stable matching is: what happens when we remove sides from the market? That is, what happens when we allow all participants in the market to match with any other participant? Formally in stable marriage we were considering a constrained matching problem in bipartite graphs, but now we intend to lift the restriction that the graph is bipartite. This problem has been given the evocative name Stable Roommates. In Gale and Shapley's original work [18] they consider this problem and give an example which shows the existence of a problem instance where no stable matching is possible. Abeledo and Isaak [1] advance this further by proving that a matching market is two-sided if and only if every reordering of the participants' preferences results in a market that has a stable matching.

In the stable roommates problem we would like an algorithm like the deferred acceptance procedure of Gale and Shapley that provides stable matchings, but since not every problem instance has a stable matching the algorithm should also be able to decide when a stable matching does not exist. In 1985, Irving [23] provided such an algorithm. In collaboration with Gusfield [25], Irving showed as a consequence of his algorithm that for any stable roommates instance if a participant is matched in a stable matching they are matched in all stable matchings. As a corollary this implies that stable matchings in a given market all have the same size. While Irving has an algorithm to find a stable matching or report that none exist we need consider the question what sort of matching is desirable when no stable matching exists. A possible relaxation would be to ask for a matching with the least number of blocking pairs. It has been shown that it is NP-hard to find such a matching [3]. For those without a complexity background, an optimization problem being NP-hard is considered strong evidence that no efficient algorithm exists for that problem. By efficient we mean that worst case running time is bounded above by a polynomial function of the size of the input specified in some reasonable encoding, for instance binary. See chapters 3 and 4 of Introduction to Algorithms [12] for an introduction to time complexity analysis and chapter 34 for a discussion of NP-hardness.

Another natural question is to ask about what happens if indifference is allowed in stable marriage problems. By this we mean that we allow ties in participants' preferences. Once we allow ties we need to delineate between three different notions of stability. In any notion of stability a matching is considered stable if a blocking pair does not exist. A blocking pair with respect to a matching is a pair of vertices u and v that are not matched in the matching and that mutually prefer each other to their respective partners. In our original definition of stability in stable marriage both u and v strictly prefer each other to their matched partners. When ties are involved this notion of stability will be called weak-stability. Strong-stability says u must strictly prefer v to their partner but v may strictly prefer, or be indifferent to, u compared to their matched partner. Super-stability allows both u and v to either strictly prefer, or be indifferent to, each other compared to their matched partner.

Irving found that for stable marriage with ties, and where all possible pairings from both sides of market are acceptable to the participants, there always exists a weakly-stable matching [24]. It is also known that strong-stable and super-stable matchings need not always exist, but that polynomial time algorithms for deciding if they do and finding one if so exist. Irving resolves the super-stable case, while Kavitha et al. [29] resolved the strongly-stable case. The work of Kavitha also covers the possibility of unacceptable matches, while an algorithm for finding strongly-stable matchings with unacceptable partners was given by Manlove [34]. For weakly-stable matchings with unacceptable matches Manlove et al. [36] demonstrated that weakly-stable matchings always exist, but they may have different sizes, which for previously mentioned variants was not the case. They showed that the problem of finding a largest weakly-stable matching is NP-hard.

We mentioned in the previous section (1.1) that the problem of finding a stable matching of optimal weight in a stable marriage problem is solvable in polynomial time. While this is true, the weighted stable roommates problem is considerably harder. Feder has shown that the problem of solving weighted stable roommates is NP-hard [17]. Interestingly though the linear programming formulation Rothblum gave for weighted stable marriage can be used to find $\frac{1}{2}$-integral weighted stable roommates solutions. In 1994 he proved this result with Abeledo [2]. By $\frac{1}{2}$-integral we mean extreme point solutions of Rothblum's formulation take variable values in the set $\{0, \frac{1}{2}, 1\}$. While not solving the problem entirely, if the reader has an eye for applications we could imagine that a $\frac{1}{2}$ matching can have reasonable interpretations. For example splitting time in half between two contracts, or having a 50 percent chance of being matched with a $\frac{1}{2}$ matched partner.

1.3 Our Contributions and Outline

Our first significant contribution to this body of theory is a simpler view of the linear programming formulation of weighted stable matching studied by Vande Vate and Rothblum. We present a short proof that Rothblum's formulation has integer-valued vectors for extreme points which uses only some elementary theory of polytopes and basic structural results on stable matchings. Our proof is inspired by the common paradigm of iterative rounding [33], yet takes a slightly different direction as one will see in Chapter 3. We are hopeful that our proof technique can be used in the investigation of linear programming formulations for other stable matching type problems in the future since its simplicity leaves it wide open for extension.

Our other contributions are in the study of cyclic complete three-sided stable matchings as proposed by Knuth. We began our study in hopes of applying modern computing power to push higher the best known size for which it was known there is always a stable matching. As we previously mentioned it currently stands at 4 and that result belongs to Eriksson and his collaborators. We propose a computer search framework for testing all instances of the problem for a given size. The framework is interesting because it allows for the cutting off of many instances simultaneously by not needing to explore their full preferences. It does this through the use of lemmas which are sufficient for proving existence of stable matchings, and eliminating from consideration instances that are equivalent to previously checked instances.

We provide some new lemmas which are sufficient for concluding that an instance of the three-sided problem has stable matchings. Many of these lemmas are about finding a particular partial matching and applying "induction" to get a stable matching in a smaller instance. Other lemmas try to satisfy all agents of a given side in a way that they are unwilling to prevent the matching from being stable. In the study of symmetry, we provide a definition of when two instances of the three-sided problem are symmetric and study the structure of equivalence classes under this symmetry.

In the next chapter we will give the reader the necessary background to understand our contributions, taking them through many of the the famous results previously mentioned along the way. They will also learn about linear programming and the methods of iterative rounding, and previous structural results in stable matching theory. The two chapters following preliminaries will detail our contributions to firstly polyhedral theory in stable matching, and secondly to three-sided stable matching problems. After that we conclude with a summary of what was studied, and provide many open avenues for the interested reader to begin exploring research problems in the theory of stable matchings based on our contributions in this thesis.

Chapter 2

Preliminaries

This chapter gives background on the tools used in the main body of work. Most of this material, with possible exceptions being iterative rounding and the structure of stable matching instances, should be familiar to a student with comparable background to a Combinatorics and Optimization undergraduate at the University of Waterloo. We will emphasize the results in the covered areas that we will need later, but it is important to point out that these areas go much deeper than what is written here and so the end of each section includes suggested texts for further reading.

2.1 Matching Theory

2.1.1 Graphs and Matchings

A *graph* G is an ordered pair (V, E) where V is called the vertex set and $E \subseteq \{\{a, b\} : a, b \in V, a \neq b\}$ is called the edge set. The clause $a \neq b$ forbids self-loops and insisting that E is a proper set forbids parallel edges. Some authors choose to work with more generality but for ease of exposition we will not. As defined above, G is an undirected graph, but if one were to change E to a set of ordered pairs in $V \times V$ then G would be called a directed graph. We will work here with undirected graphs. When V and E are not explicit we can use $V(G)$ and $E(G)$ respectively to refer to the vertex and edge sets of G.

Since it is somewhat cumbersome to write $\{a, b\} \in E(G)$ every time we wish to refer to an edge of the graph G we will use throughout this text the shorthand ab to denote the edge $\{a, b\}$.

In a graph G, two vertices $a, b \in V(G)$ are said to be *adjacent*, denoted $a \sim b$, if $ab \in E$. For any vertex $a \in V(G)$ we denote by $\delta(a) = \{e \in E(G) : a \in e\}$ the set of edges incident upon a. The *: degree* of a, $d(a)$,is defined as $d(a) = |\delta(a)|$.

A *path* P is a graph where $V(P) = \{v_0, v_1, \ldots, v_n\}$ and

$$E(P) = \{v_0v_1, v_1v_2, \ldots, v_{n-1}v_n\}.$$

where all v_i are distinct for $i \in \{0, 1, \ldots, x_n\}$. A *cycle* is a path with the exception that $v_0 = v_n$.

A graph $G = (V, E)$ is said to be *bipartite* if there exists a partition of V into V_0, V_1 such that for every edge $ab \in E$ either $a \in V_0$ and $b \in V_1$, or $b \in V_0$ and $a \in V_1$.

Given a graph $G = (V, E)$, a *matching* on G is any $M \subseteq E$ satisfying that for all $e_1, e_2 \in M$ we have $e_1 \cap e_2 = \emptyset$. We use $V(M) = \{v \in V : \exists e \in M, v \in e\}$ to denote the set of vertices matched in M. Intuitively our definition of matching means that each vertex matched in M (those in $V(M)$) is matched to exactly one "partner". That is to say $|\delta(v) \cap M| = 1$. The *partner* of vertex v, denoted $M(v)$, is the vertex such that $vM(v) \in M$. When it is clear from context we may also use M to refer to the graph "induced" by M, by which we mean the graph $(V(M), M)$.

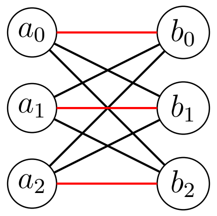

Figure 2.1: A complete bipartite graph and matching.

Graph has vertex set $\{a_0, a_1, a_2\} \cup \{b_0, b_1, b_2\}$. The red edges indicate matching the matching $\{a_0b_0, a_1b_1, a_2b_2\}$.

2.1.2 Maximum Cardinality Matching

Problem One classical problem in Matching Theory is the question of finding a maximum cardinality matching given a graph. This question was first investigated by Berge in [5]. He gave a characterization of when a matching has maximum cardinality.

Let M be a matching in graph G. Let P be a path contained in G with $E(P) = \{v_0v_1, \ldots, v_{n-1}v_n\}$. We say that P is M-*alternating* if the sequence of edges $E(P)$ alternate being in M and not being in M. More precisely P is M-alternating provided that for any $i \in \{0, \ldots, n-2\}$, edge $v_iv_{i+1} \in M$ if and only if $v_{i+1}v_{i+2} \in E(G)\backslash M$. An M-alternating path P is called M-*augmenting* if v_0 and v_n are not in $V(M)$ and $n \geq 1$.

The name M-augmenting path purposefully evokes the image of flow augmenting paths in network flow theory as there is a way to increase the cardinality of a matching M by augmenting it with an M-augmenting path. This augmenting operation is called symmetric difference, which we define below.

For any sets $S, T \subseteq U$ the *symmetric difference* of S and T, denoted $S \triangle T$, is given by $S \triangle T = (S \cup T)\backslash(S \cap T)$.

The definition of symmetric difference tells us that for any $x \in S \triangle T$ either $x \in S$ or $x \in T$ but not both. Formally we could say $S \triangle T = (S\backslash T) \cup (T\backslash S)$, which is immediate from our definition of $\triangle$ and basic set theory.

Theorem 2.1.1 *Augmenting Path Theorem (Berge [5]): Let G be a graph and let $M \subseteq E(G)$ be a matching. Then M is of maximum cardinality over the set of matchings in G if and only if there does not exist any M-augmenting path contained in G.*

Proof: For the ($\Longleftarrow$) direction suppose that there exists M-augmenting path P in G. We denote the edge set of P by $E(P) = \{v_0v_1, \ldots, v_{n-1}v_n\}$. We claim that $M' = M \triangle E(P)$ is a matching of greater cardinality than M. To see that M' is a matching we will show that for all $v \in V(M')$, $|\delta(v) \cap M'| = 1$. Let $v \in V(M')$. Since $v \in V(M')$, $|\delta(v) \cap M'| > 0$. Suppose for a contradiction that $|\delta(v) \cap M'| \geq 2$. Then there exist distinct $e_1, e_2 \in \delta(v)$ such that $e_1, e_2 \in M'$. By the nature of symmetric difference either $e_1 \in M\backslash E(P)$ or $e_1 \in E(P)\backslash M$, and similarly either $e_2 \in M\backslash E(P)$ or $e_2 \in E(P)\backslash M$. We cannot have both $e_1, e_2 \in M\backslash E(P)$ since M is a matching, and we cannot have both $e_1, e_2 \in E(P)\backslash M$ since P

is an M-alternating path. So we may assume without loss of generality that $e_1 \in M \backslash E(P)$ and $e_2 \in E(P) \backslash M$. Since $e_1 \in M$, $v \neq v_0$ and $v \neq v_n$. Since P is an M-augmenting path, $v \neq v_0$, $v \neq v_n$, and $e_2 \in E(P) \backslash M$ there exists $e \in M \cap E(P)$ such that $e \in \delta(v)$. Since $e_1 \notin E(P)$ we have that $e \neq e_1$. But then $e, e_1 \in M$ with $e \cap e_1 = \{v\} \neq \emptyset$ contradicting that M is a matching. Therefore $|\delta(v) \cap M'| < 2$, implying that $|\delta(v) \cap M'| = 1$ and thus that M' is a matching. Now since $V(M') = V(M) \cup \{v_0, v_n\}$ we observe that $|M'| = |M| + 1 > |M|$. This implies that M is not a maximum cardinality matching in G.

Now suppose for the ($\Longrightarrow$) direction that M is a matching in G which is not of maximum cardinality. That is there exists some matching M' such that $|M'| > |M|$. Consider their symmetric difference $J = M' \triangle M$. Observe that each vertex in the graph $(V(G), J)$ has degree at most two. This follows from observing that if there was a vertex v whose degree in $(V(G), J)$ was at least three then either two edges incident upon v are in M or M' contradicting that M and M' are matchings. Therefore $(V(G), J)$ consists only of vertex disjoint paths and cycles. The edges of said paths and cycles alternate belonging to M' and to M. Otherwise there would be a vertex with two edges incident upon it in the same matching, a contradiction. So the cycles are even in number of edges and contain the same number of edges from each of M' and M. But since $|M'| > |M|$ there is a path, P, with more edges in M' than in M. This follows from counting edges in M and M' noticing that cycles contribute the same number to each. The path P is M-augmenting. ■

We cover this proof not only because it is intrinsically interesting, but also because the structure of J, the symmetric difference of two matchings, will arise in the future when we study the structure of stable matchings.

Further Reading This problem is very well understood. For instance Tutte's classic min-max theorem [48], and Edmond's Blossom algorithm [14]. A textbook appropriate for advanced undergraduate or beginner graduate students is Combinatorial Optimization by Cook, Cunningham, Pulleybank, and Schrijver [11] which contains a chapter covering the results mentioned here.

2.1.3 Maximum Weight Matching

Problem Suppose we are given a bipartite graph G and a weight function $w : E(G) \to \mathbb{R}$. The Maximum Weight Matching problem is to find a matching M which maximizes

$\sum_{e \in M} w(e)$. This problem and its solution via the Hungarian Algorithm attributed to Kuhn and Munkres [32][38] is one of the earliest success stories of Combinatorial Optimization. Another approach which gives some flavour of the work to come is to model the problem via a linear program. It is this approach we will explain here.

Linear Program Linear programming has proven to be a powerful and unifying tool in combinatorial optimization. If the reader is uncomfortable with linear programming please see section 2.2.1 for a primer. Consider the following linear program:

$$\begin{aligned}
\max \sum_{e \in E(G)} w(e) x_e & & \\
\text{s.t.} \sum_{e \in \delta(v)} x_e \leq 1 & & \text{for all } v \in V(G) \\
x_e \geq 0 & & \text{for all } e \in E(G).
\end{aligned}$$

Let E be a set of discrete elements. Let $S \subseteq E$. We define the *incidence vector* of S to be $\chi(S) \in \mathbb{Z}^{|E|}$ where

$$\chi(S)_e = \begin{cases} 1, & \text{if } e \in S \\ 0, & \text{otherwise.} \end{cases}$$

Let $M \subseteq E(G)$ be a matching in G. Then $\chi(M)$ is feasible for the above linear program. Hence the optimal solution to the linear program is an upper bound on the maximum weight of a matching in G. In fact Birkhoff [6] showed that all extreme point solutions (see 2.2.1 for definition) to this program are integral. This implies the optimal extreme point solutions are incidence vectors of matchings and that the optimal value of this linear program is equal to the maximum weight of a matching in G. In the next section we will define the terms of Birkhoff's Theorem and discuss a proof, not the original proof of Birkhoff, but a proof using the techniques of iterative rounding.

2.2 Iterative Rounding

The technique of iterative rounding was originally inspired by Jain's work on survivable network design [27]. Since that time it has proven to be a versatile technique seeing application across a wide variety of branches of Combinatorial Optimization including

Matchings, Spanning Trees, Flows, and Network Design [33] to name a few. Iterative Rounding was initially studied as a procedure for obtaining approximation algorithms, but has since been adapted to reprove many classical results. The earliest known use of iterative rounding for exact optimization problems was given by Steinitz in his study of rearrangements [47]. It is this latter application to proofs of integrality that we will focus on, but in the context of matching. We will begin with some necessary background from the theory of linear programming, then proceed to discuss the general form the iterative rounding technique takes in the context we are studying, and finish by demonstrating its application to maximum weight bipartite matching.

2.2.1 Linear Programming Tools

In this section we will review the fundamentals of linear programming theory and discuss results necessary for application to iterative rounding. For a comprehensive treatment of linear programming consider the classic text of Chvatal [9].

Let A be an $m \times n$ matrix, let $b \in \mathbb{R}^m$ and let $c \in \mathbb{R}^n$. The goal of a *linear programming problem* is to find $x^* \in \{x \in \mathbb{R}^n : Ax \leq b, x \geq 0\}$ which maximizes $c^T x^*$. Here $x \geq 0$ is a shorthand for $x_i \geq 0$ for all $i \in \{1, \dots, n\}$. This can be written in the compact form $\max\{c^T x : Ax \leq b, x \geq 0\}$ or displayed as

$$\begin{aligned} \max \quad & c^T x \\ Ax &\leq b \\ x &\geq 0. \end{aligned} \tag{2.1}$$

The linear function $c^T x$ is referred to as the *objective function* in variables x. Given any row a_i of A and the corresponding b_i entry of b, $a_i^T x \leq b_i$ is a *constraint*. The constraints $x \geq 0$ are called *non-negativity constraints*.

The choice of the standard form (2.1) eases exposition, but it is by no means rigid. If one wishes to apply a constraint of the form $a^T x = b$ or $a^T x \geq b$, or even wishes to be free of non-negativity constraints then such a linear program could be written down and converted to an equivalent program (possibly with more variables) in our standard form. To see how to do this refer to Chvatal [9], or try it as an easy exercise.

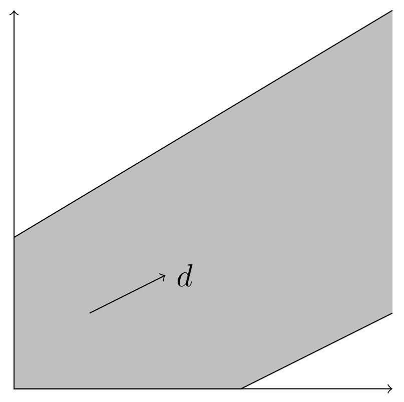

Figure 2.2: An unbounded polyhedron

Given a linear program in the form 2.1, $P = \{x \in \mathbb{R}^n : Ax \leq b, x \geq 0\}$ is referred to as the *feasible region*. It describes the space of vectors which satisfy the constraints of the linear program. Such P is a geometric object called a *polyhedron*, which can be defined as the space formed by the intersection of a finite number of half-spaces.

A polyhedron, P, is *unbounded* if there exists $d \in \mathbb{R}^n \backslash \{0\}$ such that for all $\alpha \in \mathbb{R}$ with $\alpha \geq 0$ and $x \in P$, $x + \alpha d \in P$. Such d is called a *direction*. A polyhedron which is not unbounded is called *bounded*, and such polyhedra are called *polytopes*.

Observe that if P is unbounded then the corresponding linear program may not have a finite optimal value. In particular this happens when translating along a direction d can increase the objective value That is to say $P \neq \emptyset$, and there exists direction d and objective function c for which $c^T d > 0$. By choosing successively larger α we may make the objective value of $x + \alpha d$ arbitrarily large. Linear programs whose feasible regions are polytopes always have finite optimal values. For the critical reader we refer you to Chvatal [9] for a proof.

Let P denote the feasible region of a linear program of the form 2.1 and let x be a vector in P. Then x is a *vertex* of P if there exist h and δ such that $h^T x = \delta$ and for all $y \in P$ such that $y \neq x$ we have $h^T y < \delta$. The hyperplane specified by (h, δ) is called a *supporting hyperplane* for x.

In the same setting as the previous definition, we say x is an *extreme point* if there do not exist distinct $y^1, y^2 \in P$ such that

$$x = \frac{1}{2}y^1 + \frac{1}{2}y^2.$$

Again in the same setting, for ease of exposition let

$$A' = \begin{bmatrix} A \\ -I \end{bmatrix} \quad b' = \begin{bmatrix} b \\ 0 \end{bmatrix} \tag{2.2}$$

and thus $P = \{x : A'x \leq b'\}$. The vector $x \in P$ is called a *basic feasible solution* if there exist n linearly independent rows of A', say $a_1, \ldots, a_n$ (not necessarily the first n rows of A), with corresponding entries of b': $b_1, \ldots, b_n$ satisfying $a_i x = b_i$ for all $i \in \{1, \ldots, n\}$.

It is not immediately obvious that the three concepts x being a vertex, an extreme point, or a basic feasible solution are equivalent but in the next lemma we will show the fact that they are indeed.

Lemma 2.2.1 *Let P be the polyhedron corresponding to the feasible region of 2.1, and let $x \in P$. Then the following are equivalent:*

1. *x is a vertex of P,*
2. *x is an extreme point of P,*
3. *and x is a basic feasible solution of P.*

Proof: ((1) $\implies$ (2)) First suppose that x is a vertex of P with supporting hyperplane $h^T x = \delta$. Then $h^T y < \delta$ for all $y \in P$ such that $y \neq x$. Suppose for a contradiction that x is not an extreme point of P. Then there exist $y^1, y^2 \in P$ with $y_1 \neq y_2$, such that

$$x = \frac{1}{2}y^1 + \frac{1}{2}y^2.$$

So then

$$\delta = h^T x = h^T(\frac{1}{2}y^1 + \frac{1}{2}y^2) = \frac{1}{2}h^T y^1 + \frac{1}{2}h^T y^2 < \frac{1}{2}\delta + \frac{1}{2}\delta = \delta,$$

yielding $\delta < \delta$, a contradiction. Thus x is a vertex implies x is an extreme point.

((2) $\implies$ (3)) For ease of exposition let A' and b' be as in equations 2.2. We proceed by contraposition. Suppose that x is not a basic feasible solution. Then there are not n linearly independent constraints tight at x. Let k be the number of constraints satisfied by x at equality. We may assume that the first k rows of A', $a_1, \dots, a_k$ along with the first k entries of b' describe the constraints x satisfies at equality. Notice that either $k < n$, or $k \geq n$ and $a_1, \dots, a_k$ are not linearly independent. In either case the system (in variables y):

$$a_1^T y = 0$$
$$\dots$$
$$a_k^T y = 0$$

has a non-trivial solution. Let $d \neq 0 \in \mathbb{R}^n$ be such a solution. Since adding or subtracting multiples of d from x does not affect satisfaction of tight constraints, and all other constraints satisfied by x are not tight (that is, $a_i x < b_i$ for such constraints), there exists $\epsilon \neq 0$ such that $x + \epsilon d, x - \epsilon d \in P$. Thus letting $y^1 = x + \epsilon d$ and $y^2 = x - \epsilon d$ we have

$$\frac{1}{2}(y^1 + y^2) = \frac{1}{2}(2x) = x.$$

Since $y^1, y^2 \in P$ with $y_1 \neq y_2$, and $x = \frac{1}{2}y^1 + \frac{1}{2}y^2$, x is not an extreme point. Thus we have shown that x is extreme point implies that x is a basic feasible solution.

((3) $\implies$ (1)) Now suppose that x is a basic feasible solution. Suppose that rows of A', $a_1, \dots, a_n$, along with $b_1, \dots, b_n$ describe the linearly independent constraints satisfied by x at equality. By linear independence, x is the unique solution to

$$\begin{bmatrix} a_1 \\ \vdots \\ a_n \end{bmatrix} x = \begin{bmatrix} b_1 \\ \vdots \\ b_n \end{bmatrix}.$$

Let $h^T = \sum_{i=1}^n a_i$ and let $\delta = \sum_{i=1}^n b_i$. We claim that (h, δ) describe a supporting hyperplane for x. Indeed let $y \in P$. Then

$$h^T y = \sum_{i=1}^n a_i y \leq \sum_{i=1}^n b_i$$

with the last inequality following since $y \in P$ implies $a_i y \leq b_i$ for all $i \in \{1, \dots, n\}$. Observe that equality holds above if and only if $a_i y = b_i$ for all $i \in \{1, \dots, n\}$. But since x

is the unique solution to the system of equations $a_i y = b_i$ for all $i \in \{1, \dots, n\}$ this implies that $y = x$. Hence if $y \in P$ with $y \neq x$ then $h^T y < \delta$. Therefore x is a basic feasible solution implies that x is a vertex, and hence the lemma holds. ■

Lemma 2.2.2 *(Rank Lemma [33]) Let x be an extreme point of P. If $x > 0$ then there are n linearly independent constraints tight at x of the form $a_i x = b_i$, which is the maximum number possible. Here a_i denotes a row of A and b_i is the entry of b corresponding to a_i.*

Proof: By Lemma 2.2.1 we have that x is a basic feasible solution. Hence there exist n linearly independent constraints tight at x. Since $x > 0$ each tight constraint is of the form $a_i x = b_i$ for some i, as no non-negativity constraints are tight. Therefore the row rank of A is at least n. But A is an $m \times n$ matrix, and thus the column rank of A is at most n. Since it is a ubiquitous fact from linear algebra that column rank and row rank are equal for any matrix this implies that the row rank of A is equal to n. Therefore n is the maximal number of linearly independent constraints of the form, $a_i x = b_i$, tight at x.
■

Lemma 2.2.3 *Let x be an optimal solution to a linear program of the form 2.1 with feasible region denoted by P. We claim there exists $x^* \in P$ such that x^* is an extreme point and $c^T x = c^T x^*$.*

Proof: Let x be an optimal solution with the maximum number of constraints tight at x. Suppose for a contradiction that x is not an extreme point. We will demonstrate that we can always find another optimal solution with more tight constraints than x (possibly non-negativity constraints, which manifest as 0-entries of x) contradicting our choice of x. Since x is not an extreme point there exists $y^1, y^2 \in P$ with $y_1 \neq y_2$, such that

$$x = \frac{1}{2}y^1 + \frac{1}{2}y^2.$$

We define a direction $d \neq 0$ along which we will translate x without ruining optimality. Let

$$d = \frac{1}{2}y^2 - \frac{1}{2}y^1.$$

Then $x + d = y^2 \in P$ and $x - d = y^1 \in P$. We have the following inequalities

$$\begin{aligned} A(x+d) &\leq b \\ x + d &\geq 0 \\ A(x-d) &\leq b \\ x - d &\geq 0. \end{aligned}$$

Let $A^=$ be the collection of rows of A along with corresponding entries of b, $b^=$, that satisfy $A^= x = b^=$. So by the above inequalities $A^=(-d) \leq 0$ and $A^= d \leq 0$. Therefore $A^= d = 0$. Further, by the optimality of x,

$$\begin{aligned} c^T x &\geq c^T(x+d) \\ \text{and } c^T x &\geq c^T(x-d) \\ \text{which implies } c^T d &= 0. \end{aligned}$$

Thus for any ϵ, $x + \epsilon d$ satisfies $A^=(x+\epsilon d) = b^=$ and $c^T(x+\epsilon d) = c^T x$. Since $d \neq 0$, we may assume that there exists i such that $d_i < 0$ (otherwise we may use $-d$ instead of d). We define $\epsilon > 0$, which is finite as $d_i < 0$, as follows:

$$\epsilon = \min\{\min\{\frac{x_j}{-d_j} : d_j < 0\}, \min\{\frac{b_i - a_i x}{-a_i d} : a_i x < b_i, a_i d < 0\}\}.$$

where a_i are the rows A and b_i is the corresponding entry of b. By our choice of ϵ either $x + \epsilon d$ has one more 0 entry (in the $\frac{x_j}{-d_j}$ case) or one more tight constraint (in the $\frac{b_i - a_i x}{-a_i d}$ case) than x. Further by the minimality of ϵ the non-tight constraints remain feasible, and as previously argued the tight constraints are not violated so $x + \epsilon d$ is feasible. Since $c^T d = 0$ we thus have $x + \epsilon d$ is an optimal solution with more tight constraints than x. ■

Integrality The previous lemma tells us that an algorithm which solves linear programs need only concern itself with extreme point solutions. So extreme point solutions are of particular interest in applications. We call a polyhedron P integral if every extreme point x of P is contained in $\mathbb{Z}^n$. We will call a linear program integral if its feasible region is integral. Integrality is important when using linear programs to model discrete optimization problems since fractional solutions rarely have an interpretation in this context. Later in this section we will demonstrate how iterative rounding can be used to prove that the feasible region of a linear program is integral.

Algorithms for Linear Programming The most widely taught algorithm for solving linear programs is the simplex method[13]. The algorithm works by starting at an extreme point and traveling along edges of the polyhedron given by the feasible region to adjacent extreme points which improve the objective value until the optimal solution is found. The order it chooses extreme points to visit is based on what is called a pivot rule. There is no known pivot rule which ensures that the simplex method examines a polynomial number of extreme points in the worst case. While this indicates a poor worst case performance for

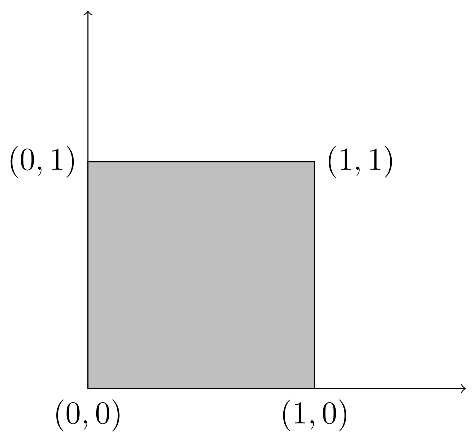

Figure 2.3: An integral polytope

This integral polytope has extreme points $(0, 0)$, $(0, 1)$, $(1, 1)$, and $(1, 0)$ which are all in $\{0, 1\}^2 \subseteq \mathbb{Z}^2$.

known implementations of the simplex method, it has been observed to work efficiently in practice possibly due to its average case polynomial time performance [46]. The ellipsoid method came later, initially as an approximation algorithm for real convex minimization, and later shown to be able to solve linear programs in polynomial time [20]. While not thought to be faster than simplex in practice, the polynomial time worst case bound of the ellipsoid method is important for discrete optimization as it allows the use of linear programming for giving polynomial time algorithms for combinatorial problems,

Further Reading The theory of linear programming is rich and very well understood. One particularly important topic not mentioned here that one needs to know to truly understand linear programming is duality theory and complementary slackness. It studies the phenomenon that linear programs come in primal-dual pairs that are solved simultaneously, and the consequences surrounding this. Unfortunately this theory is outside the scope of this thesis, but we encourage the interested reader to see Chvatal [9] for a treatment of this topic. Its study is worthwhile for combinatorial optimizers as it has led to, among other ideas, the famous class of algorithms known as primal-dual methods.

2.2.2 Strategy

After our brief detour into the theory of linear programming, we are now ready to describe the method through which iterative rounding can be used to prove that a linear programming relaxation of a combinatorial optimization problem is integral. In this subsection we will describe the general approach at a high level, and in the following section we will apply this approach to bipartite matching to demonstrate the technical details. See the excellent monograph of Lau, Ravi, and Singh[33] for other applications, and ways to extend this approach to giving approximation algorithms.

Since we wish to describe iterative rounding in general, we are going to need some terminology to formalize what we mean in general by a problem, a combinatorial optimization problem, residual problem instances, and linear programming relaxations. This may seem somewhat esoteric, but the goal is to capture the general principles common to iterative rounding approaches. For a more concrete treatment of iterative rounding as it pertains to maximum weight matching in bipartite graphs see the next subsection: 2.2.3

We formalize the notion of a *problem* as a relation $R \subseteq \mathcal{I} \times \mathcal{O}$ where $\mathcal{I}$ is some set of possible inputs, and $\mathcal{O}$ is some set of possible outputs. The pair $(I, O) \in \mathcal{I} \times \mathcal{O}$ are related in R if O is a solution when given input instance I. We say that algorithm $\mathcal{A}$ *solves* problem R if $\mathcal{A}$ takes inputs from $\mathcal{I}$ and for every $I \in \mathcal{I}$, running $\mathcal{A}$ with input I yields output O such that $(I, O) \in R$.

Example For example consider the maximum cardinality matching problem from subsection 2.1.2. In this problem we could say that $\mathcal{I}$ is the set of all graphs, and $\mathcal{O}$ is the set of all matchings of a graph. So if we let R denote the maximum cardinality matching problem, G denote a graph and M denote a matching of G then we can say

$$(G, M) \in R \text{ if and only if for all matchings } M' \text{ of } G, \text{ we have } |M| \geq |M'|.$$

Moreover we have an algorithm, namely Edmonds Blossom algorithm [14], which solves problem R.

Combinatorial Optimization Problems Generally speaking combinatorial optimization problems are problems wherein either implicitly or explicitly one is tasked with choosing an optimal subset $S \in \mathcal{S}$ where $\mathcal{S}$ is a family of feasible sets formed from elements of a discrete ground set E. In our previous example, the maximum cardinality matching problem satisfies this definitions of a combinatorial optimization problem. Namely given a graph G as input then our ground set is $E(G)$, and our family of feasible subsets can be described implicitly as $\mathcal{S} = \{M \subseteq E : |M \cap \delta(v)| \leq 1 \text{ for all } v \in V(G)\}$. Our optimal $M \in \mathcal{S}$ is the matching which satisfies $|M| \geq |M'|$ for all $M' \in \mathcal{S}$.

Step 1 - Linear Programming Formulation We are given some combinatorial optimization problem CP with ground set E as an input instance to start. From this optimization problem instance we attempt to write a linear programming relaxation $LP(E)$. For $LP(E)$ to be a relaxation of this instance of CP, there is necessarily a means of translating between solutions of this instance of CP and integral solutions of $LP(E)$. Classically this is achieved via the mapping of subsets of E to their corresponding incidence vectors (see χ in max weight matching section 2.1.3). Such $LP(E)$ is called a relaxation because the optimal value of $CP(E)$ is bounded by the optimal value of $LP(E)$ and in the case that $LP(E)$ is integral the optimal values are equal.

In some applications the linear programming relaxation given in step 1 will have an exponential number of constraints. In this situation it is not immediate that $LP(E)$ is solvable in polynomial time with respect to size of the input to CP. While this problem does not arise in the applications to matching we describe in this thesis, there are methods which may circumvent it. The ellipsoid method operates with a separation oracle [20], which is a procedure that decides if a given point is feasible for the linear program, and in the case of infeasibility provides a hyperplane which separates the point from the feasible region. It is normal that the inequalities of $LP(E)$ have some implicit description in terms of the combinatorics of CP, and so it may be possible to give a polynomial time separation oracle despite having an exponential number of constraints needed to define $LP(E)$. If one can give such a separation oracle then $LP(E)$ is polynomial time solvable despite its exponential size.

In our upcoming description of a general iterative rounding procedure, we will refer to the concept of residual problem instances. What we are aiming to capture is the idea of being able to use solutions to smaller instances of a problem to solve a larger instance of a problem, for some appropriate notion of smaller and larger.

Let $P \subseteq \mathcal{I} \times \mathcal{O}$ be a problem. Suppose that $\mathcal{I}$ is equipped with some partial order $\leq$. Let $I, I' \in \mathcal{I}$. If $I' \leq I$ then we say that I' is a *residual problem instance* of problem instance

I.

Example Consider once more the example of the maximum cardinality matching problem. Let G, G' be graphs. We say G' is a subgraph of graph G if $V(G') \subseteq V(G)$ and $E(G') \subseteq E(G)$. The subgraph relation is a partial order on the space of all graphs. Hence we can say that G' is a residual problem instance of G.

Step 2 - Iterative Algorithm Let CP be a combinatorial optimization problem on ground set E. Let $LP(E)$ be a linear programming relaxation of $CP(E)$ for which the variables of our linear program are bounded between 0 and 1. The assumption that the feasible region of our linear programs is contained in $[0,1]^{|E|}$ is reasonable because such a case occurs naturally when using χ to map between solutions of CP and integral solutions of $LP(E)$.

In this step we describe an algorithm which constructs an integral optimal solution to $LP(E)$, which is equivalently a solution to $CP(E)$. The algorithm constructs integral optimal solution x^* as follows:

1. Let x be an optimal extreme point solution of $LP(E)$.
2. For each $e \in E$ for which $x_e \in \{0,1\}$ set $x_e^* = x_e$ and remove e from E.
3. If E is now empty return x^* and terminate.
4. Otherwise repeat from step 1.

Observe that if the algorithm always finds $x_e \in \{0,1\}$ then $|E|$ decreases at every iteration and thus the algorithm terminates. What the algorithm is doing intuitively is finding variables which are integral in an extreme point solution, fixing them to their corresponding values in the solution it will return, then removing those variables from the ground set resulting in a residual problem. The algorithm then iterates, considering an optimal solution for the residual problem.

The following lemma explains how an algorithm can be used to prove the integrality of a polytope. This lemma indicates the properties we will need to prove our algorithm satisfies in order for its existence to imply that $LP(E)$ is integral.

Lemma 2.2.4 *If there exists an algorithm $\mathcal{A}$ that always returns an integral optimal solution to $LP(E)$ for any objective function c, and terminates then the feasible region of $LP(E)$ is integral.*

Proof: We may assume without loss of generality that $LP(E)$ is a maximization problem. Let x be an extreme point of P, the feasible region of $LP(E)$. Then x is equivalently a vertex of P by Lemma 2.2.1. So there exists a supporting hyperplane described by (h, δ) for which $h^T x = \delta$ and for all $x' \in P$ with $x' \neq x$, we have $h^T x' < \delta$. If we consider the linear program $LP(E)$ with objective function h then x is the unique optimal solution of this linear program. Therefore if we run algorithm $\mathcal{A}$ with objective function h as input then the integral solution returned is equal to x by uniqueness. Thus x is integral, and so any extreme of P is integral. ■

Step 3 - Algorithm Analysis We need to establish two things in this phase: that the algorithm terminates in finite time giving an integral vector and that it indeed returns an optimal solution.

Termination Recall that we are assuming that the feasible region of $LP(E)$ is contained in $[0,1]^{|E|}$. To show that the algorithm terminates we demonstrate that there always exists an e for which $x_e \in \{0,1\}$ for every iteration. Doing so proves inductively that the algorithm terminates with a $\{0,1\}$ valued vector. There is a standard approach to showing the existence of $x_e \in \{0,1\}$ which proceeds by contradiction. If we suppose that every x_e lies in the interval $(0,1)$ then we may obtain a contradiction as follows. Since $x > 0$ we may invoke Lemma 2.2.2 saying there exists a maximal set of $|E|$ linearly independent constraints not drawn from non-negativity constraints which are tight at x. If we can then show an upper bound on the number of linearly independent constraints tight at x that is smaller than $|E|$, then we have obtained a contradiction as desired. This will become more concrete in the next section when we demonstrate an example.

Correctness It remains to show that the algorithm returns an optimal solution. For iterative algorithms this typically takes the form of an inductive argument on $|E|$. We show two things:

1. the vector x restricted to the variables in the residual problem instance is feasible for the linear programming relaxation of the residual problem instance,

2. and the integral optimal solution to the linear programming relaxation of the residual problem instance, along with the variables which were previously fixed to 0 or 1 in step 2 of the current iteration is a feasible solution to $LP(E)$.

Together this allows one to conclude that the objective value at x is at most that of the integral solution returned by the algorithm. Since x is optimal this implies that the integral solution returned has the same objective value as x and is thus optimal.

2.2.3 Application to Matching

Recall the linear programming relaxation of maximum weight matching in bipartite graphs which we will denote $LP(G)$ for a given bipartite graph G:

$$\begin{aligned}
\max \sum_{e\in E(G)} w(e)x_e & \\
\text{s.t.} \sum_{e\in\delta(v)} x_e \le 1 & \qquad \text{for all } v \in V(G)\\
x_e \ge 0 & \qquad \text{for all } e \in E(G).
\end{aligned}$$

We will use iterative rounding to prove that this linear program is integral. Consider an iterative rounding algorithm for bipartite matching with weighted input graph G.

```
1. Set $M \leftarrow \emptyset$
2. While $|E(G)| > 0$
   (a) Let $x$ be an extreme point optimal solution to $LP(G)$.
   (b) For each edge $e$ with $x_e = 1$: $M \leftarrow M \cup \{e\}$.
   (c) For each edge $e$ with $x_e \in \{0,1\}$ : $G \leftarrow G - e$.
3. return $M$
```

It is important to note that the linear program in step $2(a)$ needs only be solved once, during the first iteration, to obtain x. This is because the original extreme point x remains an extreme point for each subsequent iteration's linear programming relaxation when restricted to the appropriate variables.

To prove that $LP(G)$ is integral for any weighted bipartite graph G we need to show that our iterative algorithm always terminates in finite time and returns an integral solution of optimal value.

Since our graph G is finite, demonstrating termination is simply a matter of verifying that for any extreme point solution x of $LP(G)$ there exists an edge e with $x_e \in \{0, 1\}$. In doing so we will need the following lemma about extreme points of $LP(G)$.

Lemma 2.2.5 *Let x be an extreme point optimal solution to $LP(G)$. Suppose that for all $e \in E(G)$, $x_e > 0$. Then there exists $W \subseteq V(G)$ satisfying*

- $\sum_{e\in\delta(v)} x_e = 1$ *for all* $v \in W$
- $\{\chi(\delta(v)) : v \in W\}$ *is linearly independent (recall that $\chi : E(G) \to \{0,1\}^{|E(G)|}$ maps edge sets to incidence vectors)*
- $|W| = |E(G)|$.

Proof: The proof is immediate from Lemma 2.2.2. ■

We will now prove that our iterative rounding algorithm always terminates returning a matching (equivalently an integral solution). We will do this via a contradiction argument proving that a $0, 1$ variable can always be found.

Lemma 2.2.6 *Let x be an optimal extreme point solution to $LP(G)$. Then there exists $e \in E(G)$ such that $x_e \in \{0, 1\}$.*

Proof: Suppose for a contradiction that for all $e \in E(G)$, $0 < x_e < 1$. By Lemma 2.2.5 there exists $W \subseteq V(G)$ containing vertices corresponding to $|E(G)|$ linearly independent tight vertex constraints. Let $v \in W$. Since $\sum_{e\in\delta(v)} x_e = 1$ and $x_e < 1$ for all e, we have $d(v) \geq 2$. So then

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

The inequalities above thus hold as equalities. Since $\sum_{v\in V(G)} d(v) = \sum_{v\in W} d(v)$, $d(v) = 0$ for all $v \notin W$. Since $\sum_{v\in W} d(v) = 2|W|$ we have $d(v) = 2$ for all $v \in W$. Therefore

$E(G)$ is a set of cycles covering the vertices of W, which is to say each vertex is contained in an edge of $E(G)$ and the edges of $E(G)$ form cycles. Let $C \subseteq E(G)$ be a cycle with $V(C) \subseteq W$. Let V_1, V_2 be a bipartition of the vertex set of G (which exists since G is bipartite). Further since G is bipartite for each edge $vw \in E(C)$ one of v, w is in V_1 and the other is in V_2. Hence we have

$$\sum_{v \in V(C) \cap V_1} \chi(\delta(v)) = \sum_{v \in V(C) \cap V_2} \chi(\delta(v))$$

which contradicts the linear independence of $\{\chi(\delta(v)) : v \in W\}$. ■

It remains to verify that our algorithm returns an optimal solution. We show that via an inductive argument in the following lemma.

Lemma 2.2.7 *The iterative rounding algorithm returns a matching of weight at least that of an optimal solution $LP(G)$.*

Proof: We proceed by induction on the number of iterations the algorithm runs to solve the problem. In the base case if the algorithm runs for a single iteration then the returned solution is exactly the incidence vector of an extreme point optimal solution to $LP(G)$. That is, the lemma holds trivially. So for induction suppose the algorithm needs k iterations to return an answer for input graph G and for any graph G' such that the algorithm needs less than k iterations to solve the problem, the algorithm returns an optimal matching. Let x be the extreme point of $LP(G)$ found in step $2(a)$. Let R be the set of edges removed in step $2(c)$ of the algorithm when run on G. Let G' be the residual graph (G restricted to edges $E(G) \backslash R$). The next iterations of the algorithm are equivalent to simply running the algorithm on G'. Let M' be the matching returned. By the inductive hypothesis $w(\chi(M')) \geq w(x')$ for any x' feasible for $LP(G')$. Let x' be the point x restricted variables corresponding to edges in $E(G')$. Then $w(x) = w(x') + w(\chi(R))$. Further, as x' is feasible for $LP(G')$, we have $w(\chi(M')) \geq w(x')$. Let M be the matching returned. Then $M = M' \cup R$ and thus

$$w(\chi(M)) = w(\chi(M')) + w(\chi(R)) \geq w(x') + w(\chi(R)) = w(x).$$

Therefore by induction the algorithm always returns a matching with weight at least that of an optimal solution to $LP(G)$. ■

As discussed in the previous section, the above lemmas imply that that $LP(G)$ is integral.

2.3 Stable Matching

2.3.1 Problem

Input The stable matching problem gives as input a bipartite graph $G = (V, E)$ with bipartition $V = A \cup B$, where each $v \in V$ has a strict order over $\delta(v)$ referred to as v's preferences. We will use $u <_v w$ to denote that edge vw is preferred by v to edge vu, and we will use $>_v$, $\leq_v$, $\geq_v$, and $=_v$ analogously. We say equivalently that v prefers w to u and can treat $<_v$ as an order on the vertices adjacent to v. It will usually be clear from context how we are viewing $<_v$.

Output Given a stable matching instance as input with underlying graph G one is to find a matching M in G that is stable. A matching is called stable if there is no blocking pair ab. The edge $ab \in E(G)$ is said to be a blocking pair with respect to M if

$$ab \notin M \text{ and } b >_a M(a) \text{ and } a >_b M(b).$$

Application The sets A, B are colloqially referred to as men and women respectively. This alludes to the toy application of stable matching in forming couples from single people together who have preferences over the opposite gender. In this context a blocking pair references to a pair who mutually prefer each other to their partner. One of the original applications of stable matching [42] was to pair medical students with hospitals at which they were to do their residency, and another was college admissions.

2.3.2 Gale Shapley Algorithm

In their original work, Gale and Shapley gave an algorithm, called deferred acceptance [18], for solving the stable matching problem and in doing so proved that every bipartite graph with strict preference orders has a stable matching.

Deferred Acceptance Algorithm In the following description of the algorithm we adopt the convention that vertices prefer to be matched than to be unmatched. For each vertex in $a \in A$ we maintain a set $P(a)$ of vertices in B that a has previously "proposed" to. Given an input graph $G = (A \cup B, E)$ with preferences do:

1. Assign $M \leftarrow \emptyset$

```
2. Assign $P(a) \leftarrow \emptyset$ for all $a \in A$
3. While there exists $a \in A$ such that $a \notin V(M)$ and $P(a) \neq \{b \in B : b \text{ is adjacent to } a\}$:
    (a) Let $b$ be such that $b \geq_a b'$ for all $b' \in \delta(v) \backslash P(a)$
    (b) Assign $P(a) \leftarrow P(a) \cup \{b\}$
    (c) If $a >_b M(b)$ then assign $M \leftarrow M \cup \{ab\} \backslash \{M(b)b\}$
4. Return $M$.
```

Lemma 2.3.1 *Deferred Acceptance returns a matching.*

Proof: From step $3(c)$ it is clear that each $b \in B$ is matched to at most one $a \in A$. Further any $a \in A$ matched in step $3(c)$ is unmatched by the While condition of step 3, and hence each $a \in A$ is matched to at most one $b \in B$. Therefore M returned is a matching. ■

Theorem 2.3.2 *(Gale-Shapley): The matching returned by Deferred Acceptance is stable.*

Proof: We claim that Deferred Acceptance maintains the following invariant: at any iteration for any $a \in A$ for all $b \in P(a)$, $M(b) \geq_b a$. We prove this claim by induction. In the base case, at the start of the algorithm $P(a) = \emptyset$ for all $a \in A$ and hence the claim is vacuously true. In the inductive case consider some iteration of the algorithm. Let $a \in A$ be the vertex chosen in step 3. Let $b \in B$ be as chosen in step $3(a)$. In step $3(b)$, b is added to $P(a)$. By induction the invariant holds for all $b' \in P(a)$ except b. In step $3(c)$ if $a \leq_b M(b)$ then the invariant holds for $b \in P(a)$ and no changes are made. If $a >_b M(b)$ then b is matched to a and hence after step $3(c)$ the invariant holds for b and thus the entirety of $P(a)$. For all other $a' \neq a$, $P(a')$ is unchanged and hence by induction the invariant holds for all $b' \in P(a')$ except b. So consider if $b \in P(a')$. Observe that a is preferred by b to her match prior to this iteration and hence by transitivity the invariant also holds for $b \in P(a')$. Therefore by induction the invariant holds. It is easy to see from step 3 that the algorithm also maintains the invariant that for all $b \in \delta(a) \backslash P(a)$, $M(a) \geq_a b$. Hence if $ab \in E$ with $ab \notin M$ at termination of the algorithm then either $b \in P(a)$ or $b \notin P(a)$. If $b \in P(a)$ then $M(b) >_b a$ (as $a \neq M(b)$) and so ab is not a blocking pair. If $b \notin P(a)$ then as $ab \in E$, $b \in \delta(a) \backslash P(a)$ and so $M(a) >_a b$ (as $b \neq M(a)$). Therefore in either case ab does not block M. Therefore M is stable. ■

2.3.3 Structure

Over time a rich body of literature around the structure of the set of stable matchings for certain families of bipartite graphs with preferences has been developed [43]. In this subsection we will explore some of these results that will come up in our chapter on stable matching polytopes.

We say a stable matching M is A-*optimal* with respect to our problem instance if for every stable matching M' in this problem instance and for every $a \in A$, $M(a) \geq_a M'(a)$. That is, every man in A likes M at least as well as they like any other stable matching. We can define B-optimal analogously. Gale and Shapley showed that not only do A-optimal and B-optimal stable matchings exist in any problem instance, but also that their Deferred Acceptance algorithm computes an A-optimal stable matching [18] (and could compute B-optimal matching by switching roles of A and B).

Theorem 2.3.3 *(Gale and Shapley): For any bipartite graph $G = (A \cup B, E)$ with strict preferences there always exists a unique A-optimal (B-optimal analogously) stable matching and it can be computed via the Deferred Acceptance algorithm.*

Proof: For any $v \in A \cup B$ let

$$C(v) = \{w \in V(G) : w \sim v \text{ and there exists stable matching } M, w = M(v)\}$$

be the set of partners v has a chance of being matched to in a stable matching. We will show that no $a \in A$ is ever rejected by any $b \in C(a)$ during the operation of Deferred Acceptance and hence since each $a \in A$ proposes to their most preferred choice in $C(a)$ before any other vertex in $C(a)$ the resulting matching is A-optimal. The uniqueness is achieved since the preferences are strict. We proceed by induction on the number of iterations Deferred Acceptance has run. In the base case no proposals and hence no rejections have yet happened. So consider the inductive case. Suppose that at the given iteration b rejects a (that is, either a is unsuccessful in proposing to b or b leaves a for a better proposer at this step). Let $a' \neq a$ be the vertex b is matched to at the end of step 3(c) of the iteration under consideration. Then $a' >_b a$. By our induction hypothesis $P(a') \cap C(a') = \{b\}$ as a' has not been rejected by any $b' \in C(a')$. Let M be a matching formed by matching a to b and matching every other v to a vertex in $C(v)$. Then $b >_{a'} M(a')$ as b is the most preferred vertex of a' in $C(a')$. Further $a' >_b a = M(b)$, and thus the pair $a'b$ blocks M. Since $a'b$ blocks any such matching M, $b \notin C(a)$ (as otherwise we could form a stable matching M matching a to b). Hence the claim holds by induction, and we have proven the theorem. ■

Knuth took the above idea a step further to show that the interests of A and B are opposed to each other.

Theorem 2.3.4 *(Knuth) [30]:Let M and M' be two stable matchings for some problem instance. Let $M \geq_A M'$ denote that for all $a \in A$, $M(a) \geq_a M'(a)$. Define $M \geq_B M'$ analogously. We have that $M \geq_A M'$ if and only if $M' \geq_B M$.*

Proof: Suppose $M \geq_A M'$. We will show that $M' \geq_B M$ since the other direction is symmetric about the roles of A and B. Suppose for a contradiction that there exists $b \in B$ for which $M(b) >_b M'(b)$. Let $a = M(b)$. Then $M'(b) \neq M(b)$ and thus $M'(a) \neq M(a)$ as otherwise b has two matches in M'. Since $M \geq_A M'$ and $M(a) \neq M'(a)$, $b = M(a) >_a M'(a)$. That is we have $a >_b M'(b)$ and $b >_a M'(a)$ and thus ab blocks M'. So M' is not a stable matching, a contradiction. ■

Corollary 2.3.5 *For each $b \in B$, let*

$$C(b) = \{a \in A : \text{there exists stable matching } M, a = M(b)\}.$$

Let $\ell(b) \in C(b)$ be the vertex satisfying that for all $a \in C(b)$, $a \geq_b \ell(b)$. Then the A-optimal stable matching matches each $b \in B$ to $\ell(b)$. Note that there is also a symmetric result obtained by exchanging the roles of A and B.

These theorems begin to point towards an interesting structure for the set of stable matchings of a given problem instance. It was first observed by Conway [30] that this set forms an algebraic structure called a lattice. For those unfamiliar with lattices in this context we will proceed by explaining all the terms necessary to continue.

Let L be a set endowed with a partial order $\geq$. The set L is said to be a *lattice* if for any $x, y \in L$ there exists a unique $sup(x, y) \in L$ such that $sup(x, y) \geq x, y$ and for every z with $z \geq x, y$ we have $z \geq sup(x, y)$. Similarly there exists a unique $inf(x, y) \in L$ such that $x, y \geq inf(x, y)$ and for every z with $x, y \geq z$ we have $inf(x, y) \geq z$. A lattice is said to be *complete* if each $X \subseteq L$ also has functions *sup* and *inf*.

Lattice functions for stable matchings We now describe functions that will behave as *sup* and *inf* for stable matchings. Let M, M' be any two stable matchings for a given problem instance. Let $M^* = sup(M, M')$ be given as follows. For all $a \in A$, $M^*(a) = M(a)$ if $M(a) >_a M'(a)$ and $M^*(a) = M'(a)$ otherwise. For all $b \in B$, $M^*(b) = M(b)$ if

$M(b) <_b M'(b)$ and $M^*(b) = M'(b)$ otherwise. We can define $inf(M, M')$ analogously by interchanging the roles of A and B in the define of $sup(M, M')$. Explicitly let $M^- = inf(M, M')$ be given as follows. For all $b \in B$, $M^-(b) = M(b)$ if $M(b) >_b M'(b)$ and $M^-(b) = M'(b)$ otherwise. For all $a \in A$, $M^-(a) = M(a)$ if $M(a) <_a M'(a)$ and $M^-(a) = M'(a)$ otherwise.

Intuitively *sup* presents each $a \in A$ with a choice between their partner in M and partner in M' and pairs them with their preferred choice. In the next theorem we need to show *sup* and *inf* map to stable matchings and respect the partial order $\geq_A$.

Theorem 2.3.6 *(Conway) [30]: The set of stable matchings for a given problem instance endowed with the partial order $\geq_A$ is a lattice via the functions sup and inf defined above.*

Proof: We will demonstrate that *sup* behaves as desired and note that the argument is symmetric about the roles of A and B for *inf*. Let M, M' be stable matchings. It is clear from the construction of $sup(M, M')$ that $sup(M, M')$ is unique and $sup(M, M') \geq_A M$ and $sup(M, M') \geq_A M'$ provided that $sup(M, M')$ is a stable matching. Further if Z is a stable matching such that $Z \geq_A M, M'$ then for any $a \in A$, $Z(a) \geq_a M(a), M'(a)$. So $Z(a) \geq_a sup(M, M')(a)$ (since $Z(a) \in \{M(a), M'(a)\}$) and thus $Z \geq_A sup(M, M')$. Therefore it remains to show that $sup(M, M')$ is a stable matching.

Let $M^* = sup(M, M')$. We first show that M^* is a matching and then verify stability. To show that M^* is a matching we will prove that $M^*(a) = b$ if and only if $M^*(b) = a$. First, for sufficiency, suppose that $M^*(a) = b$. We may assume without loss of generality that $b = M(a)$. Then $b \geq_a M'(a)$. Since M' is stable, $a \leq_b M'(b)$ and hence $M^*(b) = a$ as desired. Now to show necessity suppose that $M^*(b) = a$. We may assume that $a = M(b)$. So $a \leq_b M'(b)$. Again by the stability of M', $b \geq_a M'(a)$. So $M^*(a) = b$ as desired. Therefore M^* is a matching.

Now to see stability. Suppose for a contradiction that ab blocks M^*. Then $b >_a M^*(a)$ and thus $b >_a M(a)$ and $b >_a M'(a)$. Also $a >_b M^*(b)$. So if $M^*(b) = M(b)$ then ab blocks M and if $M^*(b) = M'(b)$ then ab blocks M'. Therefore in either case we have a contradiction. ■

Further Reading The set of stable matchings of a given graph with preferences has been widely studied in academic literature. We have covered here the results key to our work in the following chapter, but many more interesting results are known. For instance, another interesting result to study is the surprising equivalence between distributive lattices and sets of stable matchings that we only began to set the stage for here. To read more on these things we recommend the survey text of Roth and Sotomayor[43].

Chapter 3

Stable Matching Polytope

In this we focus on a polyhedral characterization of the set of incidence vectors of stable matchings. Vande Vate first provided such a description in [49] for the special case where G is a complete bipartite graph. Rothblum [44] later generalized Vande Vate's result to incomplete preference lists and simplified the proof of integrality via an extreme point argument.

We provide a simpler, more compact argument for the integrality of Rothblum's formulation. Our arguments are elementary and rely solely on some well-known results on stable matchings as well as some knowledge of the local structure of extreme points in our formulation to achieve the desired result. This proof operates in the spirit of iterative rounding as discussed in 2.2. We say it is iterative rounding in spirit because, while it does not follow the strict approach of iterative rounding, it proves that there is an integral variable in every extreme point and analyses the result of dropping that variable and applying induction to the smaller problem instance to obtain a contradiction. Since it is an argument by minimal counterexample it is not immediately clear how to extend this proof to an iterative rounding algorithm.

3.1 Linear Description

Notation We will use some notation to ease exposition. Suppose $G = (A \cup B, E)$ is a bipartite graph. Let $S \subseteq E$ and let $x \in \mathbb{R}^{|E|}$. Then we denote $\sum_{e \in S} x_e$ by $x(S)$. Suppose every vertex in $A \cup B$ has strict preference orders over its neighbours in G as in a stable matching instance. Then for any $a \in A$ and $b \in B$ we let

$$\delta^{>a}(b) = \{a'b \in E(G) : a' >_b a\}.$$

We can define $\delta^{>b}(a)$ analogously, and further replace $>$ with $<, \leq, \geq, =$ in the natural way.

Linear Description We will begin with a linear description of a polytope which we claim has incidence vectors of stable matchings as its extreme points. This polytope is the matching polytope described in 2.1.3 with added "stability" constraints. If we let $G = (A \cup B, E)$ be a bipartite graph with preferences then we will denote the so-called stable matching polytope of G by $P(G)$ and it has the following description:

$$x(\delta(v)) \leq 1, \qquad \text{for all } v \in A \cup B \qquad (3.1)$$

$$x_e \geq 0, \qquad \text{for all } e \in E \qquad (3.2)$$

$$x(\delta^{>a}(b)) + x(\delta^{>b}(a)) + x_{ab} \geq 1, \qquad \text{for all } ab \in E. \qquad (3.3)$$

Our major theorem of this chapter is to demonstrate that $P(G)$ is integral, but first we must verify that indeed the integral extreme points of $P(G)$ are exactly the incidence vectors of stable matchings in G.

Lemma 3.1.1 *Let $x \in \mathbb{R}^{|E|}$. Then x is an integral extreme point of $P(G)$ if and only if there exists a stable matching M of G such that $x = \chi(M)$.*

Proof: Suppose that x is an integral extreme point of $P(G)$. By constraints 3.1 and 3.2, x is the incidence vector of some matching M of G. Suppose that M is not stable, that is to say there exists $ab \in E$ which blocks M. But ab satisfies:

$$x(\delta^{>a}(b)) + x(\delta^{>b}(a)) + x_{ab} \geq 1.$$

Since x is integral this implies that either

$$x(\delta^{>a}(b)) = 1 \quad \text{or} \quad x(\delta^{>b}(a)) = 1 \quad \text{or} \quad x_{ab} = 1,$$

and as $ab \notin M$ we known $\chi(M)_{ab} = x_{ab} = 0$. Thus we have two cases:

$$x(\delta^{>a}(b)) = 1 \quad \text{or} \quad x(\delta^{>b}(a)) = 1.$$

In the first case there exists $ab' \in \delta^{>a}(b)$ for which $x_{ab'} = 1$ and hence $ab' \in M$ with $b' >_a b$. Thus in the first case ab does not block M. Similarly in the second case there exists $a' >_b a$ with $a'b \in M$ and hence ab does not block M in either case. Therefore M is stable.

Now let M be a stable matching in G, and suppose $x = \chi(M)$. Since M is a matching $x(\delta(v)) \leq 1$ for all $v \in A \cup B$ and $x_e \geq 0$ for all $e \in E$. Hence it remains to verify that x satisfies constraints 3.3. Indeed let $ab \in E$. If $ab \in M$ then $1 = \chi(M)_{ab} = x_{ab}$ and so 3.3 is satisfied for ab. Now if $ab \notin M$ then, since M is stable, (a, b) is not a blocking pair. So there exists $ab' \in M$ with $b' >_a b$ or there exists $a'b \in M$ with $a' >_b a$. In the former case $x(\delta^{>b}(a)) = 1$ and in the latter $x(\delta^{>a}(b)) = 1$. In either case 3.3 is satisfied for ab as desired. ■

3.2 Proof of Integrality

In this section we build the necessary theory towards a proof that $P(G)$ is integral.

3.2.1 Edges of Polytopes

Before discussing the proof proper we need to take a slight detour to review some more elementary polyhedral geometry.

Let $x^1, x^2 \in \mathbb{R}^n$. Then $x \in \mathbb{R}^n$ is a *convex combination* of x^1, x^2 provided there exists $\lambda \in \mathbb{R}$ with $0 \leq \lambda \leq 1$ and

$$x = \lambda x^1 + (1 - \lambda)x^2.$$

Let $P \subseteq \mathbb{R}^n$ be a polytope. Then the hyperplane given by (h, δ) is said to describe a *valid inequality* for P if for all $x \in P$ $h^T x \leq \delta$.

Let $P \subseteq \mathbb{R}^n$ be a polytope. For our purposes we define an *edge* of P to be a set F of the form

$$F = P \cap \{x \in \mathbb{R}^n : h^T x = \delta\}$$

where (h, δ) describes a valid inequality for P, and there exist $x^1, x^2 \in P$ such that x^1 and x^2 are vertices of P for which all $x \in F$ are a convex combination of x^1, x^2.

Lemma 3.2.1 *Let $P \subseteq \mathbb{R}^n$ be a polytope. Let F be an edge of P defined by valid inequality (h, δ) and extreme points x^1, x^2. Suppose there exist $y^1, y^2 \in \mathbb{R}^n$ neither of which is a convex combination of x^1, x^2 satisfying*

$$\frac{1}{2}x^1 + \frac{1}{2}x^2 = \frac{1}{2}y^1 + \frac{1}{2}y^2.$$

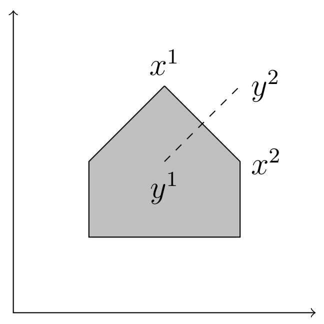

Figure 3.1: Visualizing Lemma 3.2.1

The extreme points x^1 and x^2 define an edge of this polytope. From the visual one can see that any line segment intersecting the edge between x^1 and x^2 cannot have both its endpoints y^1 and y^2 in the polytope.

Then at most one of y^1, y^2 lies in P.

Proof: Let

$$x = \frac{1}{2}x^1 + \frac{1}{2}x^2 = \frac{1}{2}y^1 + \frac{1}{2}y^2.$$

Suppose for a contradiction that $y^1, y^2 \in P$. Since $y^1, y^2 \notin F$ as y_1 and y_2 are not convex combinations of x_1 and x_2, we have

$$h^T y^1 < \delta \quad \text{and} \quad h^T y^2 < \delta.$$

So then,

$$\begin{aligned} h^T x &= h^T(\frac{1}{2}y^1 + \frac{1}{2}y^2) \\ &= \frac{1}{2}h^T y^1 + \frac{1}{2}h^T y^2 \\ &< \frac{1}{2}\delta + \frac{1}{2}\delta \\ &= \delta. \end{aligned}$$

That is $h^T x < \delta$. But since $x = \frac{1}{2}x^1 + \frac{1}{2}x^2$, x is a convex combination of x^1, x^2 and hence $x \in F$. That is $h^T x = \delta$, contradicting $h^T x < \delta$. ■

Lemma 3.2.1 says intuitively that line segments intersecting the midpoint of an edge of a polytope have at most one endpoint in P.

3.2.2 Structural Lemmas

In this section we describe some theory connecting the lattice structure of stable matchings described in 2.3.3 and the polytope $P(G)$. Going forward we assume that $E(G) \neq \emptyset$ as otherwise $P(G) = \emptyset$ is vacuously integral. Observe that this assumption implies that $0 \notin P(G)$ as all constraints 3.3 are violated by 0.

Lemma 3.2.2 *Let M and M' be distinct stable matchings in bipartite graph G with preferences. If $\chi(M)$ and $\chi(M')$ are extreme points of $P(G)$ which define an edge of $P(G)$ then either $sup(M, M') = M$ or $sup(M, M') = M'$.*

Before we begin the proof, we make a quick observation. We observe that $sup(M, M') = M$ if and only if $inf(M, M') = M'$. This follows since $sup(M, M') = M$ if and only if $M(a) \geq_a M'(a)$ for all $a \in A$ and hence $inf(M, M') = M'$. **Proof:** Let $S = sup(M, M')$ and $I = inf(M, M')$. Suppose for a contradiction that $S \neq M$ and $S \neq M'$. That is, by our preceding observation, S and I are distinct stable matchings from M and M'. We claim that

$$\chi(M) + \chi(M') = \chi(S) + \chi(I) \tag{3.4}$$

To see the claim, let $e \in E(G)$. If $e \in M \cap M'$ then the left hand side of 3.4 is 2, and further $e \in S \cap I$ and hence the right hand side of 3.4 is 2. If $e \notin M \cup M'$ then both the left and right hand side of 3.4 are 0. If $e \in M \backslash M'$ then either $e \in S$ or $e \in I$. In either case the left and right hand side of 3.4 is 1, and the same holds for $e \in M' \backslash M$. That is the claim holds.

By claim 3.4 we observe two things. First that $\chi(S)$ and $\chi(I)$ are not convex combinations of $\chi(M)$ and $\chi(M')$ since S and I are not equal to either of M or M'. Second we observe that

$$\frac{1}{2}\chi(M) + \frac{1}{2}\chi(M') = \frac{1}{2}\chi(S) + \frac{1}{2}\chi(I).$$

Therefore we are in a position to invoke Lemma 3.2.1, concluding that one of $\chi(S), \chi(I) \notin P(G)$. But this contradicts the Lattice Theorem 2.3.6 since S, I are stable matchings in G implies that $\chi(S), \chi(I) \in P$ by Lemma 3.1.1. ■

Corollary 3.2.3 *(Ratier) [40]: Let M and M' be distinct stable matchings in bipartite graph $G = (A \cup B, E)$ with preferences. If there exist $a \in A$ and $b \in B$ such that*

$$M(a) >_a M'(a) \quad \text{and} \quad M(b) >_b M'(b) \tag{3.5}$$

then $\chi(M)$ and $\chi(M')$ do not define an edge of the polytope $P(G)$.

Proof: Since $M(a) >_a M'(a)$, $aM(a) \in sup(M, M')$. That is

$$sup(M, M') \cap M \neq \emptyset$$

and since $M'(a) \neq M(a)$,

$$sup(M, M') \not\subseteq M'.$$

Since $M(b) >_b M'(b)$, $M'(b)b \in sup(M, M')$. That is

$$sup(M, M') \cap M' \neq \emptyset$$

and since $M(b) \neq M'(b)$,

$$sup(M, M') \not\subseteq M.$$

Taken together these observations prove that $sup(M, M')$ is a distinct stable matching from M and M'. Thus by Lemma 3.2.2, $\chi(M)$ and $\chi(M')$ do not define an edge of $P(G)$. ■

3.2.3 Integrality of $P(G)$

We are now ready to provide our proof that $P(G)$ is integral. The strategy is to first show that every extreme point x has some variable $e \in E$ with $x_e \in \{0, 1\}$, then, in a minimal counterexample, remove that variable and look at the reduced problem in the smaller variable set. By induction the vertices of the smaller polytope are integral. We will consider a particular edge of this polytope and obtain a contradiction via our structural lemmas of the previous section showing its end vertices cannot define an edge. The details will be made clear as we proceed.

Lemma 3.2.4 *Let $G = (A \cup B, E)$ be a bipartite graph with preferences. Let x be an extreme point of $P(G)$. Then there exists $e \in E(G)$ such that $x_e \in \{0, 1\}$.*

Proof: By constraints 3.2 and 3.1 we have that $0 \leq x_e \leq 1$. Suppose for a contradiction that for all $e \in E(G)$, $0 < x_e < 1$. Since x is an extreme point of $P(G)$ with $x > 0$ we may invoke Lemma 2.2.2, which says that x is defined by $|E|$ linearly independent constraints which are tight (satisfied with equality) at x and no such constraints are from constraints 3.2. Thus all such constraints are from 3.1 and 3.3. Let $V_x \subseteq V(G)$ and $E_x \subseteq E(G)$ be such that V_x and E_x respectively contain vertices and edges corresponding to a set of $|E|$ linearly independent constraints tight at x. That is V_x contains vertices corresponding to constraints from 3.1, E_x contains edges corresponding to constraints from 3.3, $|E| = |E_x| + |V_x|$, and

$$\{\chi(\delta(v)) : v \in V_x\} \cup \{\chi(\delta^{>a}(b)) + \chi(\delta^{>b}(a)) + \chi(\{ab\}) : ab \in E_x\}$$

is linearly independent. Choose V_x and E_x in such a way that $|V_x|$ is maximized.

First observe that $V_x \subsetneq V$. To see this, suppose that $V_x = V$. Then since G is bipartite

$$\sum_{a \in A} \chi(\delta(a)) = \sum_{b \in B} \chi(\delta(b))$$

and hence $\{\chi(\delta(v)) : v \in V = V_x\}$ is not linearly independent. Thus our observation that $V_x \subsetneq V$ holds.

For any $v \in V(G)$ denote by $N(v)$ the set $\{w \in V(G) : vw \in \delta(v)\}$ and let $N_{max}(v) \in N(v)$ be such that $N_{max}(v) \geq_v w$ for all $w \in N(v)$. Similarly let $N_{min}(v) \in N(v)$ be such that $N_{min}(v) \leq_v w$ for all $w \in N(v)$.

Let $v \in V(G)$. We claim that if $w = N_{max}(v)$ then $vw \notin E_x$. Indeed suppose for a contradiction that $vw \in E_x$. Since $w = N_{max}(v)$, $\delta^{>w}(v) = \emptyset$ and hence

$$1 \leq x(\delta^{>w}(v)) + x(\delta^{>v}(w)) + x_{vw} = x(\delta^{\geq v}(w)) \leq x(\delta(w)) \leq 1,$$

shows that $1 = x(\delta^{\geq v}(w))$ and thus $x(\delta^{<v}(w)) = 0$. Since $x_e > 0$ for all $e \in E(G)$ this implies that $\delta^{<v}(w)) = \emptyset$. Therefore

$$\chi(\delta(w)) = \chi(\delta^{\geq v}(w)) = \chi(\delta^{>w}(v)) + \chi(\delta^{>v}(w)) + \chi(\{vw\})$$

and hence by linear independence we cannot have both $vw \in E_x$ and $w \in V_x$. Since $vw \in E_x$ this implies $w \notin V_x$. But then the sets $V_x \cup \{w\}$ and $E_x \backslash \{vw\}$ describe the same vectors as V_x, E_x but $|V_x \cup \{w\}| > |V_x|$ contradicting our choice of V_x. Thus $vw \notin E_x$. Therefore for any $v \in V(G)$, $vN_{max}(v) \notin E_x$.

Let $v \in V_x$. Let $w = N_{min}(v)$. We claim analogously to the previous claim that $vw \notin E_x$. Indeed suppose for a contradiction that $vw \in E_x$. Since $v \in V_x$, by tightness,

$$x(\delta(v)) = 1$$

and since $vw \in E_x$, by tightness,

$$x(\delta^{>v}(w)) + x(\delta^{>w}(v)) + x_{vw} = 1.$$

Since $w = N_{min}(v)$, $\delta^{>w}(v) \cup \{vw\} = \delta(v)$ and hence

$$x(\delta^{>w}(v)) + x_{vw} = x(\delta(v))$$

and

$$\chi(\delta^{>w}(v)) + \chi(\{vw\}) = \chi(\delta(v)).$$

Thus

$$1 = x(\delta^{>v}(w)) + x(\delta^{>w}(v)) + x_{vw} = x(\delta^{>v}(w)) + x(\delta(v)) = x(\delta^{>v}(w)) + 1,$$

which implies that $x(\delta^{>v}(w)) = 0$. Again since $x_e > 0$ for all $e \in E(G)$, this implies that

$$\chi(\delta^{>v}(w)) = 0.$$

So we have that

$$\chi(\delta(v)) = \chi(\delta^{>w}(v)) + \chi(\{vw\}) = \chi(\delta^{>v}(w)) + \chi(\delta^{>w}(v)) + \chi(\{vw\}),$$

and hence by linear independence we cannot have both $vw \in E_x$ and $v \in V_x$. Since $v \in V_x$ this implies $vw \notin E_x$, a contradiction.

We now apply a token argument to demonstrate that $|E| > |E_x| + |V_x|$, a contradiction. The argument will proceed by assigning each $e \in E$ a token which it will distribute according to a pair of rules to E_x and V_x. We will argue that each $e \in E_x$ and $v \in V_x$ receives a token, and that some $e \in E_x$ has at least some token fraction leftover. In demonstrating this we have proven the desired claim.

Assign each $ab \in E$ a token. Redistribute the tokens according to the following rules:

1. if $ab \in E_x$ assign the token to $ab \in E_x$
2. otherwise
   (a) if $a \in V_x$ assign $\frac{1}{2}$ a token to $a \in V_x$
   (b) if $b \in V_x$ assign $\frac{1}{2}$ a token to $b \in V_x$.

First it should be clear that these rules are well-defined since no $ab \in E_x$ gives away more than the one token it receives. Let $ab \in E_x$. Obviously from rule 1, ab receives one token. Now consider $v \in V_x$. Since $x_e < 1$ for all e we know $N_{min}(v) \neq N_{max}(v)$, as otherwise $x(\delta(v)) = x_{vN_{min}(v)} < 1$ which contradicts $v \in V_x$ defining a tight constraint. Further by our previous observations, $vN_{min}(v) \notin E_x$ and $vN_{max}(v) \notin E_x$. So when the edges $vN_{min}(v)$ and $vN_{max}(v)$ are applying the rules to distribute their token they each give $\frac{1}{2}$ token to v by rule 2. So v receives at least one token. To complete the proof we need to find some $e \in E$ which did not give away its whole token. We have previously observed that $V_x \subsetneq V(G)$. So let $v \in V(G) \backslash V_x$. Consider the edge $vN_{max}(v)$. Since $vN_{max}(v) \notin E_x$ it does not give away its token by rule 1, and since $v \notin V_x$ it gives away at most half its token by rule 2. So the edge $vN_{max}(v)$ has at least $\frac{1}{2}$ a token remaining at the end of redistribution. Therefore $|E| > |E_x| + |V_x|$, contradicting that $|E| = |E_x| + |V_x|$. ■

Theorem 3.2.5 *Let $G = (A \cup B, E)$ be a bipartite graph with preferences. Then the polytope $P(G)$ is integral.*

Proof: Suppose for a contradiction the theorem does not hold. Let $G = (A \cup B, E)$ be a minimal counterexample in terms of $|E|$. It is clear from case analysis that $|E| > 2$. Let x be a non-integral extreme point of $P(G)$. By Lemma 3.2.4 there exists $ab \in E$ such that $x_{ab} \in \{0, 1\}$.

First we consider the case where $x_{ab} = 0$. Let $P' = P(G) \cap \{x \in \mathbb{R}^{|E(G)|} : x_{ab} = 0\}$. Then x is an extreme point of polytope P' since $P' \subseteq P(G)$. Let G' be the graph with $V(G') = V(G)$ and $E(G') = E(G) \backslash \{ab\}$. Observe that by dropping edge ab $P(G')$ differs from $P(G)$ only in not having constraint 3.3 for ab and constraints 3.1 for a and b not counting edge ab (which is equivalent to assigning $x_{ab} = 0$ always), hence we have

$$P' = P(G') \cap \{x \in \mathbb{R}^{|E(G')|} : x(\delta^{>a}(b)) + x(\delta^{>b}(a)) \geq 1\}.$$

Let H be the hyperplane $\{x \in \mathbb{R}^{|E(G')|} : x(\delta^{>a}(b)) + x(\delta^{>b}(a)) = 1\}$. Then every extreme point of P' is either an extreme point of $P(G')$ or the intersection of an edge of $P(G')$ with H. Since the extreme points of $P(G')$ are integral by our minimality assumption, x must be in the intersection of an edge F of $P(G')$ with H. Let M and M' be stable matchings such that $\chi(M)$ and $\chi(M')$ define F. Since x is not integral, $x \neq \chi(M)$ and $x \neq \chi(M')$. Also note that neither $\chi(M)$ nor $\chi(M')$ lies on the hyperplane H, since x is the unique and distinct intersection of H with the line segment of convex combinations of $\chi(M)$ and $\chi(M')$. Thus from our definition of H, since $\chi(M), \chi(M') \notin H$, we have $\left|M \cap (\delta^{>a}(b) \cup \delta^{>b}(a))\right| \neq 1$ and $\left|M' \cap (\delta^{>a}(b) \cup \delta^{>b}(a))\right| \neq 1$. On the other hand, the line segment of convex combinations of $\chi(M)$ and $\chi(M')$ has a non-empty intersection with H, namely x, and so without loss of generality we may assume that $\left|M \cap (\delta^{>a}(b) \cup \delta^{>b}(a))\right| = 2$ and $\left|M' \cap (\delta^{>a}(b) \cup \delta^{>b}(a))\right| = 0$ (notice the size of said intersections is at most 2 by 3.1).

Since $\left|M \cap (\delta^{>a}(b) \cup \delta^{>b}(a))\right| = 2$, $|M \cap \delta(a)| \leq 1$ and $|M \cap \delta(b)| \leq 1$, we have $M(a) >_a b$ and $M(b) >_b a$. Further since $\left|M' \cap (\delta^{>a}(b) \cup \delta^{>b}(a))\right| = 0$ we have that $b >_a M'(a)$ and $a >_b M'(b)$. Thus $M(a) >_a M'(a)$ and $M(b) >_b M'(b)$. Critically, this means M and M' satisfy 3.5 and so by corollary 3.2.3, $\chi(M)$ and $\chi(M')$ do not define an edge of $P(G')$. This contradicts that they do indeed define an edge of $P(G')$.

Finally we consider the case where $x_{ab} = 1$. By constraints 3.1 for a and b we see that $x_e = 0$ for all $e \in (\delta(a) \cup \delta(b)) \backslash \{ab\}$. Thus if there exists $e \in (\delta(a) \cup \delta(b)) \backslash \{ab\}$ then $x_e = 0$ and we may run the previous case analysis with edge e. So it remains to consider when $(\delta(a) \cup \delta(b)) \backslash \{ab\} = \emptyset$. If we let G' be the graph G with edge ab removed then $P(G')$ is described by precisely the same constraints as $P(G)$ with the exclusion of constraint 3.3 for edge ab. Let x' be the vector x restricted to variables $E(G) \backslash \{ab\}$. We claim that x' is an extreme point of $P(G')$. By the Rank Lemma 2.2.2, x is described by $|E(G)|$ linearly independent tight constraints. Since $\delta(a) = \delta(b) = \{ab\}$, the constraints 3.1 for a and b and the constraint 3.3 for ab are pairwise linearly dependent, and in fact equal. Hence at most one of them is in the description of x. Thus we can find $|E(G)| - 1 = |E(G')|$ linearly independent constraints tight at x' in $P(G')$ by excluding the constraint 3.3 for ab from those which describe x. Therefore x' is an extreme point of $P(G')$. So by minimality x' is integral, and thus x is integral. ■

This theorem is significant because it allows one to be able to solve maximum weight stable matching problems in bipartite graphs in polynomial time via the use of linear programming.

Chapter 4

Cyclic Stable Matching

This chapter describes the cyclic stable matching problem as posed by Knuth. We give some sufficient conditions for a matching to be stable in this context. Then we proceed to describe a computer search procedure for deciding if all preference systems of a certain size have a cyclic stable matching or finding a counterexample should one exist.

4.1 The Problem

In Knuth's writing on stable matching [31] he talks about some extensions of the classical stable matching problems as discussed so far. One such avenue for generalization is increasing the number of "genders", which is to say instead of matching pairs, we consider matching triples in tripartite graphs or more generally matching k-tuples in k-partite graphs. We will make this notion formal through the terminology of hypergraphs.

We say $G = (V, E)$ is a *hypergraph* if V is some discrete set and $E \subsetneq 2^V$ (2^V denotes the set of all subsets of elements in V). We can think of *adjacency* in this case as $v \sim w$ provided there exists $e \in E$ such that $v, w \in e$. The notions of *degree* d and $\delta(v)$, the edges incident upon a vertex, extend naturally. A *matching* in this context is a set $M \subseteq E$ such that for all $m_1, m_2 \in M$, $m_1 \cap m_2 = \emptyset$. We call G *k-uniform* if for all $e \in E$, $|e| = k$. We call k-uniform G *k-partite* if there exists $V_0, \ldots, V_{k-1}$ which partition V and for all $e = \{v_0, \ldots, v_{k-1}\} \in E$, $v_i \in V_i$ for $i = 0, \ldots, k-1$. Lastly we say k-partite graph G is *balanced* if $|V_0| = \cdots = |V_{k-1}|$.

With our description of hypergraphs complete we are ready to describe a generalization of stable matching.

In an instance of *cyclic k-gender stable matching* (*kGSM*), we are given a k-partite hypergraph $G = (V_0 \cup \cdots \cup V_{k-1}, E)$ and for each $i \in \{0, \ldots, k-1\}$, each $v_i \in V_i$ is equipped with a preference order over $V_{i+1} \cap V(\delta(v_i))$ where addition of indices is taken modulo k (so V_{k-1} vertices have preferences over V_0 vertices, hence the descriptor cyclic). That is each vertex in gender i has preferences over vertices acceptable to them in gender $i + 1$.

We say our instance of $kGSM$, and naturally the hypergraph G, is *complete* if for all $v_0 \in V_0, \ldots, v_{k-1} \in V_{k-1}$, we have $\{v_0, \ldots, v_{k-1}\} \in E$. Hence in complete instances, which we will shorthand to $CkGSM$ instances, all possible tuples are acceptable to use in matchings.

A *solution* to an instance of $kGSM$ is a matching $M \subseteq E$ that is stable in the sense that no blocking tuple exists. A tuple $e = \{v_0, \ldots, v_{k-1}\} \in E$ *blocks* M if $e \notin M$ and for each $i \in \{0, \ldots, k-1\}$, $v_{i+1} >_{v_i} M(v_i)$ where $M(v_i) \in V_{i+1}$ is the vertex satisfying that there exists an edge $m \in M$ for which $v_i, M(v_i) \in m$.

Note 4.1.1 *We may assume that instances of kGSM are balanced. To see this observe that adding dummy vertices does not affect the solutions. Thus if $|V_i| < |V_j|$ we can add $|V_j| - |V_i|$ dummy vertices to V_i to balance their sizes. If incomplete preferences are allowed then dummy vertices can simply have no incoming edges. If preferences are required to be complete then we may simply place dummy vertices as the least preferred vertices among the relevant preference lists and give the dummy vertices arbitrary orders over "real" vertices they face. We say that an instance of kGSM is of size n if it is balanced with each V_i having size n.*

We will focus on instances where $k = 3$ to simplify things somewhat. Knuth asked if instances of $3GSM$ always have stable solutions. To this day this question is unresolved for complete instances. Some, but not many, results are known and we will present them in the next section.

Conjecture 4.1.2 *(Knuth) For all sizes n, instances of $3GSM$ of size n have a stable matching.*

4.2 Known Results

First we will present a counterexample that demonstrates that when instances are not complete there need not always be a stable matching solution. This example is due to Biro et al. [7] and shows that when there are incomplete preferences there is a hypergraph with each V_i having size 6 and preference lists that admit no stable matching.

Note 4.2.1 *For our counterexample consider the tripartite hypergraph $R6 = (A \cup B \cup C, E)$ where*

$$A = \{a_0, a_1, a_2, a_0', a_1', a_2'\},\ B = \{b_0, b_1, b_2, b_0', b_1', b_2'\},\ \textit{and } C = \{c_0, c_1, c_2, c_0', c_1', c_2'\}.$$

Below we present the preference orders of each vertex, where vertices omitted from a list are deemed unacceptable and cannot form edges with the given vertex whose list is under consideration:

$$\begin{array}{lll}
a_0 : b_0 > b_0' & b_0 : c_0 > c_0' & c_0 : a_1 > a_1' \\
a_1 : b_1 > b_1' & b_1 : c_1 > c_1' & c_1 : a_2 > a_2' \\
a_2 : b_2 > b_2' & b_2 : c_2 > c_2' & c_2 : a_0 > a_0' \\
a_0' : b_2 & b_0' : c_2 & c_0' : a_0 \\
a_1' : b_0 & b_1' : c_0 & c_1' : a_1 \\
a_2' : b_1 & b_2' : c_1 & c_2' : a_2.
\end{array}$$

We will adopt the notation of the original authors calling the agents $\{a_i, b_i, c_i : 1 \leq i \leq 3\}$ the inner agents and $\{a_i', b_i', c_i' : 1 \leq i \leq 3\}$ the outer agents.

Lemma 4.2.2 *The problem instance of 3GSM described in Note 4.2.1 has no stable matching.*

Proof: Let M be a matching in $R6$. We observe that the possible triples in M have very particular forms. Let $m \in M$ be a triple. Suppose that inner vertex a_i is in m. Then by a_i's preferences either $b_i \in m$ or $b_i' \in m$. If $b_i \in m$ then as a_i is not acceptable to c_i, the triple m is of the form $a_i b_i c_i'$. If $b_i' \in m$ then $c_{i-1} \in m$ (recall index addition and subtraction is done modulo 3) and hence the triple m is of the form $a_i b_i' c_{i-1}$. Otherwise if outer vertex $a_i' \in m$ then the triple m is of the form $a_i' b_{i-1} c_{i-1}$. That is the only triples in M are those of the form

$$a_i b_i c_i' \quad \text{or} \quad a_i b_i' c_{i-1} \quad \text{or} \quad a_i' b_{i-1} c_{i-1}.$$

Then triples in M contain exactly two inner agents and one outer agent each. Since each agent can only be matched at most once, and inner agents are matched in pairs then M matches an even number of inner agents. But there are an odd number of inner agents (9 to be precise). Hence at least one inner agent is unmatched in M. By the symmetry of $R6$ we may assume without loss of generality that this inner agent is a_i. Consider the triple $T = \{a_i, b'_i, c_{i-1}\}$. If c_{i-1} is matched in M then, as a_i is unmatched, $M(c_{i-1}) = a'_i$. Hence $a_i >_{c_{i-1}} M(c_{i-1})$. Again since a_i is unmatched, b'_i is unmatched and thus prefers T to M. Hence the triple T blocks M, and since M was an arbitrary matching no matching of $R6$ is stable. ■

Since we have a counterexample to stability in the case where incomplete preferences are allowed this motivates us to restrict the problem to the situation where problem instances are complete. It is in this setting that we will work going forward. At the time of this writing we are not aware of any counterexample instance for complete $3GSM$ ($C3GSM$) that has no stable matching, nor of any proof that all instances admit a stable matching. Thus we consider a modified conjecture for $C3GSM$ instances as follows.

Conjecture 4.2.3 *(Knuth) For all sizes n, all instances of $C3GSM$ of size n have a stable matching.*

One natural angle towards a proof would be to attempt to extend the Deferred-Acceptance algorithm of Gale and Shapley to $C3GSM$. Farczadi et al. provide some insight into the feasibility of this route in [16]. Here they claim a certain step in the natural approach to extending Deferred-Acceptance is NP-complete. For readers without a complexity theory background this result gives strong evidence that there is no polynomial time algorithm for the concrete approach to extending Deferred-Acceptance examined in [16].

Let $G = (A \cup B \cup C, E)$ be a hypergraph underlying an instance of $C3GSM$. Let M' be a matching of A to B. We say that a matching M in G is an *extension* of M' if for all $a \in V(M') \cap A$, $M'(a) = b$ implies $M(a) = b$.

Theorem 4.2.4 *(Farczadi et al.): Consider an instance of $C3GSM$ with hypergraph $G = (A \cup B \cup C, E)$. Fix a perfect matching of A to B. The problem of deciding if this matching can be extended to a stable matching of G is NP-complete.*

So in lieu of algorithmic insight into the problem, researchers have presented ad-hoc methods to demonstrate when instances of $C3GSM$ are stable for fixed sizes n. It has been shown by Eriksson et al. [15] that complete instances of size $n \leq 4$ always have a stable matching. First observe that cases of size $n \leq 2$ are not interesting as blocking triples are impossible since a blocking triple must draw each of its vertices from a distinct triple in the matching (no vertices in instances of $C3GSM$ are unmatched in a stable matching, as there would be at least three and they prefer to be together than unmatched). The case for $n = 3$ is straightforward and can even be checked by hand. Eriksson et al. have shown something stronger for the $n = 3$ case: that they can actually fix the quality of match any arbitrary vertex receives in a stable matching. We will present below some notation to ease exposition then give their proof.

Let $G = (A \cup B \cup C, E)$ be the hypergraph for an instance of $C3GSM$ of size n. We will define the following functions for A but we will use analogous versions for B and C throughout. Let $a \in A$. Let $s \in \{1, \ldots, n\}$. We say that $f(G, a, s) = b$ provided that

$$|\{b' \in B : b' >_a b\}| = s - 1.$$

Intuitively $f(G, a, s)$ is a's s-th choice among vertices in B where $G = (A \cup B \cup C, E)$.

Let $G = (A \cup B \cup C, E)$ be the hypergraph for an instance of $C3GSM$. We say G has an *i, j, k triple* if there exist $a \in A$, $b \in B$, and $c \in C$ such that $f(G, a, i) = b$, $f(G, b, j) = c$, and $f(G, c, k) = a$. Of particular interest are $1, 1, 1$ triples. If a matching contains a $1, 1, 1$ triple then none of the vertices in the triple will participate in any blocking triple that prevents the matching from being stable.

Lemma 4.2.5 *(Eriksson et al.) [15]: Let $G = (A \cup B \cup C, E)$ be the hypergraph for an instance of $C3GSM$ of size 3. Let $a \in A$. Then G has a stable matching M where either $M(a) = f(G, a, 1)$ or, $M(a) = f(G, a, 2)$ and $M(b) = f(G, b, 1)$ where $b = f(G, a, 1)$.*

Proof: First consider the case where G has a $1, 1, 1$ triple (a', b', c'). Construct the matching M as follows. Add $a'b'c'$ to M. Now a', b', c' will not participate in any blocking triples against M. Hence there are only 6 vertices remaining, so regardless of how we choose the next two triples for M the resulting matching will be stable. If $a = a'$ then we are done, so we may assume $a \neq a'$. If $b = b'$ then complete M arbitrarily as long as $M(a) = f(G, a, 2)$. Otherwise, $b \neq b'$ and we may complete M arbitarily ensuring $M(a) = b$.

Now consider the case where G has no $1, 1, 1$ triple. Let $c = f(G, b, 1)$. Since there is no $1, 1, 1$ triple, $a \neq f(G, c, 1)$. Let $a' = f(G, c, 1)$. Again since there is no $1, 1, 1$ triple $b \neq f(G, a', 1)$. Let $b' = f(G, a', 1)$. Similarly $c \neq f(G, b', 1)$ and let $c' = f(G, b', 1)$. Let a'', b'', c'' be the remaining vertices in A, B, C respectively. Form the matching $M = \{abc, a'b'c', a''b''c''\}$. We claim M is stable. Since a and a' were matched to their first choices they will not participate in any blocking triple. Hence a'' is the only vertex in A willing to participate in a blocking triple. But a'' is only unmatched to b and b', both of which are matched to their first choices and hence are unwilling to participate in any blocking triple. Thus there are no blocking triples with respect to M and hence M is stable. Further since $M(a) = b$ we are done. ■

Using Lemma 4.2.5 Eriksson et al. show the following theorem. We omit the proof as, by the authors' own admission, it is a technical case analysis.

Theorem 4.2.6 *(Eriksson et al.) [15]: The conjecture 4.2.3 holds for $n \leq 4$.*

4.3 Our Contributions

In this section we will describe some work done towards resolving if $C3GSM$ instances always have solutions for higher sizes $n \geq 5$. We present first some sufficient conditions for problem instances to have stable matchings. Next we study the symmetry of problem instances and describe when problem instances are equivalent with respect to having a stable matching. All this work is building towards a proposal for a computer search procedure to decide if all instances of $C3GSM$ for a given size n have a stable matching, or present a counterexample if they do not.

4.3.1 Sufficient Conditions

Let G be the hypergraph for an instance of $C3GSM$, and let M be a matching in G. We denote by $\mathcal{B}(G, M)$ the set of blocking triples in G against M. Formally $\mathcal{B}(G, M) = \{abc \in E(G) : b >_a M(a), c >_b M(b), a >_c M(c)\}$. We denote the set of discontent vertices, those that participate in blocking triples, as $\mathcal{D}(G, M) = V(\mathcal{B}(G, M))$.

Note 4.3.1 *If we let $S \subseteq V(G)$ and denote by $G[S]$ the hypergraph G restricted to vertices in S and edges between vertices in S ($G[S]$ denotes the subgraph induced by S) then we* *may use $\mathcal{B}(G[S], M)$ to describe the blocking triples of M in G among vertices in S. In particular if we wish to study blocking triples present among agents matched in M we study the set $\mathcal{B}(G[V(M)], M)$.*

Note 4.3.2 *We mention a few things here that are immediately obvious from the definition of $\mathcal{D}$. Let $G = (A \cup B \cup C, E)$ be a hypergraph underlying an instance of $C3GSM$. Let M be a matching in G. Then we have:*

- *If M' is matching such that $M \subseteq M'$ then $\mathcal{D}(G, M') \subseteq \mathcal{D}(G, M)$.*
- *M is a stable matching in G if and only if $\mathcal{D}(G, M) = \emptyset$.*
- *If $v \in \mathcal{D}(G, M)$ then there exists $w >_v M(v)$ such that $w \in \mathcal{D}(G, M)$.*

Sometimes we want to talk about the vertices that some given vertices can form blocking triples with. The following definition defines a superset of such vertices which we will study in our subsequent lemmas.

Let G be the hypergraph underlying an instance of $C3GSM$. Let $W \subseteq V(G)$. We define $\mathcal{D}(G, M, W) = \{w \in \mathcal{D}(G, M) : \exists v \in W : w >_v M(v)\}$.

The following lemma underlies a common approach to finding stable matchings. In it we demonstrate how a partial matching can be used to construct a complete stable matching.

Lemma 4.3.3 *Let $G = (A \cup B \cup C, E)$ be a hypergraph underlying an instance of $C3GSM$ of size n, where $n - 1$ satisfies conjecture 4.2.3. Suppose there exists a non-empty matching M in G such that $V(M) \cap \mathcal{D}(G, M) = \emptyset$. Then G has a stable matching.*

Proof: Let $G' = G[V(G) \backslash V(M)]$ be the subgraph of G induced by vertices unmatched in M. Consider the instance of $C3GSM$ with underlying hypergraph G' and preference orders induced by G. Since $M \neq \emptyset$ the size of this new instance is at most $n - 1$. Hence by conjecture 4.2.3 there exists a stable matching M' in G'. Let $S = M \cup M'$. Then S is a perfect matching in G. We claim that S is stable. Indeed since $M \subseteq S$ and $V(M) \cap \mathcal{D}(G, M) = \emptyset$ we have that $V(M) \cap \mathcal{D}(G, S) = \emptyset$. So $\mathcal{D}(G, S) \subseteq V(M')$. Since $V(M') = V(G')$, if $\mathcal{D}(G, S)$ is non-empty then $\mathcal{D}(G', M')$ is non-empty, contradicting that M' is a stable matching in G'. Thus $\mathcal{D}(G, S) = \emptyset$ and therefore S is a stable matching in G. ■

Let G be a hypergraph underlying an instance of $C3GSM$ and let M be a matching in G. We define the set of vertices higher ranked than some partner of a matched vertex as

$$\mathcal{S}(G, M) = \{w \in V(G) : \exists v \in V(M), w >_v M(v)\}.$$

The condition in Lemma 4.3.3 that $V(M) \cap \mathcal{D}(G, M) = \emptyset$ is not very useful for our computer search to be described in 4.3.3 because we are operating with partial information about preferences in that context. We propose a stronger requirement which is easier to verify with partial information. The conditions $\mathcal{B}(G[V(M)], M) = \emptyset$ and $\mathcal{S}(G, M) \subseteq V(M)$ together imply that $V(M) \cap \mathcal{D}(G, M) = \emptyset$. In the proof of the following lemma this will be made clear.

Lemma 4.3.4 *Let $G = (A \cup B \cup C, E)$ be the hypergraph for an instance of $C3GSM$ of size n where $n-1$ satisfies conjecture 4.2.3. Suppose that G has a non-empty matching M for which $\mathcal{B}(G[V(M)], M) = \emptyset$. If*

$$\mathcal{S}(G, M) \subseteq V(M)$$

then G has a stable matching.

Proof: Let $a \in V(M)$. Suppose there exists $abc \in \mathcal{B}(G, M)$. Since $b >_a M(a)$ and $a \in V(M)$, we have that $b \in V(M)$. Similarly since $c >_b M(b)$ we have that $c \in V(M)$. Further since $a >_c M(a)$ we have that $abc \in \mathcal{B}(G[V(M)], M)$, contradicting that $\mathcal{B}(G[V(M)], M) = \emptyset$. Hence for all $e \in \mathcal{B}(G, M)$, $a \notin e$. Thus $a \notin \mathcal{D}(G, M)$ and therefore $V(M) \cap \mathcal{D}(G, M) = \emptyset$. So by Lemma 4.3.3 G has a stable matching. ■

Example Consider the following instance of $C3GSM$ with ... denoting arbitrary preferences:

$a_0 : \boldsymbol{b_1} > \boldsymbol{b_0} > \dots$	$b_0 : \boldsymbol{c_1} > \boldsymbol{c_0} > \dots$	$c_0 : \boldsymbol{a_1} > \boldsymbol{a_0} > \dots$
$a_1 : \boldsymbol{b_0} > \boldsymbol{b_1} > \dots$	$b_1 : \boldsymbol{c_0} > \boldsymbol{c_1} > \dots$	$c_1 : \boldsymbol{a_0} > \boldsymbol{a_1} > \dots$
$a_2 : \dots$	$b_2 : \dots$	$c_2 : \dots$
$a_3 : \dots$	$b_3 : \dots$	$c_3 : \dots$
$a_4 : \dots$	$b_4 : \dots$	$c_4 : \dots$

Matching $M = \{a_0b_0c_0, a_1b_1c_1\}$ satisfies Lemma 4.3.4. Therefore M and any stable matching of $a_2, a_3, a_4, b_2, b_3, b_4, c_2, c_3, c_4$ together form a stable matching in this problem instance.

Corollary 4.3.5 *Let G be the hypergraph for an instance of C3GSM and n be as in Lemma 4.3.4. If there exists $a, b, c \in V(G)$ such that $b = f(G, a, 1)$, $c = f(G, b, 1)$, and $a = f(G, c, 1)$ (a $1, 1, 1$ triple) then G has a stable matching.*

Proof: Invoke Lemma 4.3.4 with $M = \{abc\}$. ■

Corollary 4.3.6 *Let $G = (A \cup B \cup C, E)$ be the hypergraph for an instance of C3GSM with n as in Lemma 4.3.4. If one of A, B, C, say without loss of generality A, is such that there exists $b \in B$, such that for all $a \in A$, $f(G, a, 1) = b$ then G has a stable matching.*

Proof: Let $a_1, \ldots, a_n \in A$. Let $c = f(G, b, 1)$. Then there exists some a_k such that $a_k = f(G, c, 1)$. Since $a_k \in A$ we may invoke corollary 4.3.5 with (a_k, b, c) to complete the proof. ■

The following lemmas study situations where we nearly have a matching which satisfies $\mathcal{D}(G, M) = \emptyset$ except some part, without loss of generality A, has matched vertices which participate in some blocking triples of a particular form. The first lemma looks to build a matching containing M and using Lemma 4.3.3 to find a stable matching. The second lemma shows that if the matching M is large enough it implies the existence of a stable matching directly.

Lemma 4.3.7 *Let $G = (A \cup B \cup C, E)$ be the hypergraph underlying an instance of C3GSM of size n where $n - 1$ satisfies conjecture 4.2.3. Let M be a matching in G. Suppose that*

$$V(M) \cap (B \cup C) \cap \mathcal{D}(G, M) = \emptyset$$

and that there exists $b \in B \backslash V(M)$ satisfying

$$\mathcal{D}(G, M, V(M) \cap A) = \{b\}$$

and for all $a \in A \backslash V(M)$, $f(G[V(G) \backslash V(M)], a, 1) = b$. Then G has a stable matching. See figure 4.3.1 for a visualization.

Proof: Let $G' = G[V(G) \backslash V(M)]$ be the subgraph of G induced by vertices unmatched in M. Let $c = f(G', b, 1)$ and $a = f(G', c, 1)$. By our hypothesis, $b = f(G', a, 1)$. Let $T = \{abc\}$. Consider the matching $M' = M \cup T$. We will show that $V(M') \cap \mathcal{D}(G, M') = \emptyset$ to invoke Lemma 4.3.3 to complete the proof.

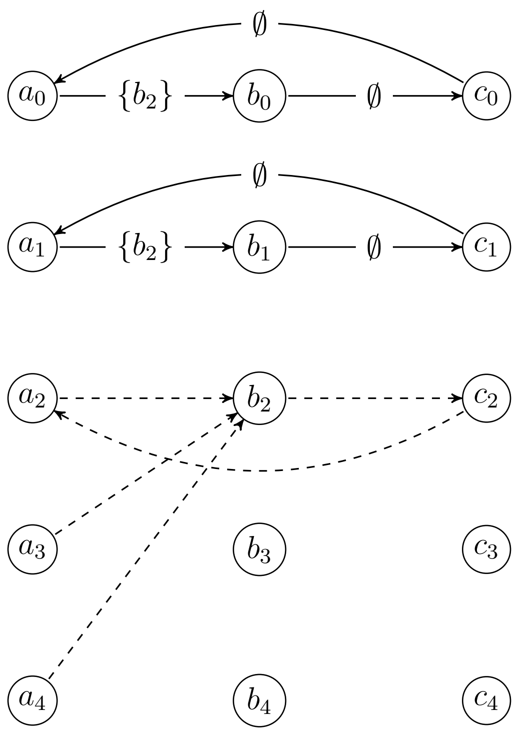

Figure 4.1: Visualizing Lemma 4.3.7

Solid arcs indicate matching M, and are labelled with the vertices preferred by the tail vertex of the arc to their match. Dashed arcs indicate first choice of tail vertex outside $V(M)$. Here a_2, b_2 and c_2 are analogous to a, b and c in the lemma statement.

Let $v \in V(M')$. First consider the case where $v \in V(M) \cap (B \cup C)$. Since $v \notin \mathcal{D}(G, M)$ and $M \subseteq M'$, we have that $v \notin \mathcal{D}(G, M')$. Second consider the case where $v \in \{a, b\}$. If $w >_v M'(v) = T(v)$ then $w \in V(M) \cap (B \cup C)$ by our construction of T. Since such $w \notin \mathcal{D}(G, M')$ we have $v \notin \mathcal{D}(G, M')$. Third consider the case where $v \in V(M) \cap A$. By our hypothesis if $w >_v M'(v) = M(v)$ then $w = b$. Since $b \notin \mathcal{D}(G, M')$ we have $v \notin \mathcal{D}(G, M')$. Finally consider the case where $v = c$. Since $v \in T$, by our construction of T, if $w >_v M'(v)$ then $w \in V(M) \cap A$. Since such $w \notin \mathcal{D}(G, M')$ we have $v \notin \mathcal{D}(G, M')$. Therefore $V(M') \cap \mathcal{D}(G, M') = \emptyset$, and since $T \subseteq M'$, we have $M' \neq \emptyset$ so we may apply Lemma 4.3.3 to conclude that G has a stable matching. ∎

Lemma 4.3.8 *Let $G = (A \cup B \cup C, E)$ be the hypergraph underlying an instance of C3GSM* *of size n. Let M be a matching in G such that $|M| \geq n-3$. Suppose that*

$$V(M) \cap C \cap \mathcal{D}(G, M) = \emptyset$$

and there exists $b \in B \backslash V(M)$ satisfying

$$\mathcal{D}(G, M, V(M) \cap A) = \{b\}$$

and for all $a \in A \backslash V(M)$, $f(G[V(G) \backslash V(M)], a, 1) = b$. Suppose also that there exists $c \in C \backslash V(M)$ satisfying

$$\mathcal{D}(G, M, V(M) \cap B) = \{c\}.$$

Then G has a stable matching.

Proof: Let $G' = G[V(G) \backslash V(M)]$ be the subgraph of G induced by vertices unmatched in M. Let $c' = f(G', b, 1)$ be the first choice of b among vertices unmatched in M. We consider two cases: when $c' = c$ and when $c' \neq c$.

First consider when $c' = c$. Let $a' = f(G', c', 1)$ be the first choice of c' unmatched in M. Let $T = \{a'bc\}$. Let $M' = M \cup T$. We will show $V(M') \cap \mathcal{D}(G, M') = \emptyset$. Let $v \in V(M')$. We give the following case analysis on v:

- If $v \in V(M) \cap C$ then $v \notin \mathcal{D}(G, M')$ since $M \subseteq M'$.
- If $v = b$ then $w >_v M'(v) = c$ implies $w \in V(M') \cap C$ by our construction of T. Thus $w \notin \mathcal{D}(G, M')$ and so $v \notin \mathcal{D}(G, M')$.
- If $v \in V(M) \cap A$ then $w >_v M'(v) = M(v)$ implies $w = b$ by our hypothesis. Since $b \notin \mathcal{D}(G, M')$ we thus have $v \notin \mathcal{D}(G, M')$.
- If $v = c$ then $w >_v M'(v) = T(v)$ implies $w \in V(M') \cap A$ and thus $w \notin \mathcal{D}(G, M')$, so $v \notin \mathcal{D}(G, M')$.
- If $v \in V(M) \cap B$ then $w >_v M'(v) = M(v)$ implies $w = c$ and thus $w \notin \mathcal{D}(G, M')$, so $v \notin \mathcal{D}(G, M')$.
- If $v = a'$ then $w >_v M'(v) = T(v)$ implies that $w \in V(M') \cap B$ by our construction of T. Thus $w \notin \mathcal{D}(G, M')$ and so $v \notin \mathcal{D}(G, M')$.

Therefore $V(M') \cap \mathcal{D}(G, M') = \emptyset$. Let S be an arbitrary perfect matching in G such that $M' \subseteq S$. Since $|M'| \geq n-2$, $|S \backslash M'| \leq 2$. Since $V(M') \cap \mathcal{D}(G, M')$ any blocking triple against S must use one vertex from each of three edges in $S \backslash M'$, but since $|S \backslash M'| \leq 2$ this implies it is impossible to form a blocking triple against S.

Now consider the case when $c' \neq c$. Let $a = f(G', c, 1)$. Let $a' = f(G'[V(G')\backslash a], c', 1)$ be the first choice of c' among vertices unmatched in M and excluding c's first choice, a. Let $b' = f(G'[V(G')\backslash b], a, 1)$. Let $T = \{ab'c\}$ and $T' = \{a'bc'\}$. Form the matching $M' = M \cup T \cup T'$. Again we study $V(M') \cap \mathcal{D}(G, M')$. Let $v \in V(M')$. In the same manner as the analysis for the previous case, we observe $V(M) \cap C \cap \mathcal{D}(G, M') = \emptyset$, $b \notin \mathcal{D}(G, M')$, $V(M) \cap C \cap \mathcal{D}(G, M') = \emptyset$, $c \notin \mathcal{D}(G, M')$, and $V(M) \cap B \cap \mathcal{D}(G, M') = \emptyset$, and $a' \notin \mathcal{D}(G, M')$. So it remains to consider $v \in \{a, b', c'\}$:

- If $v = a$ then $w >_a M'(a) = b'$ implies that $w \in V(M) \cap B$ or $w = b$. In either case w is not in $\mathcal{D}(G, M')$ and thus $v \notin \mathcal{D}(G, M')$.
- If $v = c'$ then $w >_{c'} M'(c') = a$ implies that $w \in V(M) \cap A$ or $w = a$. In either case w is not in $\mathcal{D}(G, M')$ and thus $v \notin \mathcal{D}(G, M')$.

Now we did not explicitly choose the partner of b' based on her preference, so we cannot perform the above line of reasoning to conclude that $b' \notin \mathcal{D}(G, M')$. Therefore $V(M') \cap \mathcal{D}(G, M') \subseteq \{b'\}$. Let S be an arbitrary perfect matching of G such that $M' \subseteq S$. Since $|M'| \geq n - 1$ and $V(M') \cap \mathcal{D}(G, M')$, there is exactly one triple $e \in S\backslash M'$ such that $V(e)$ and b' are the only vertices which could participate in any blocking triple against S in G. This is insufficient to form a blocking triple and thus S is a stable matching in G. ■

Just as we did with Lemma 4.3.3 and Lemma 4.3.4, in Lemmas 4.3.7 and 4.3.8 we want to replace conditions regarding the set of vertices in blocking triples with ones easier to check in our computer search. The following lemmas do so in a manner analogous to our approach to transforming Lemma 4.3.3 into Lemma 4.3.4.

Lemma 4.3.9 *Let $G = (A \cup B \cup C, E)$ be the hypergraph underlying an instance of C3GSM of size n where $n - 1$ satisfies conjecture 4.2.3. Let M be a matching in G. Suppose that $\mathcal{B}(G[V(M)], M) = \emptyset$, and*

$$\mathcal{S}(G, M) \cap (A \cup C) \subseteq V(M)$$

and that there exists $b \in B\backslash V(M)$ satisfying

$$\mathcal{S}(G, M) \cap B \subseteq \{b\} \cup V(M)$$

and for all $a \in A\backslash V(M)$, $f(G[V(G)\backslash V(M)], a, 1) = b$. Then G has a stable matching.

Proof: Similarly to the proof of Lemma 4.3.4 the conditions

$$\mathcal{B}(G[V(M)], M) = \emptyset$$

and

$$\mathcal{S}(G,M)\cap(A\cup C)\subseteq V(M)$$

imply that

$$V(M)\cap(B\cup C)\cap\mathcal{D}(G,M)=\emptyset.$$

Further

$$\mathcal{S}(G,M)\cap B\subseteq\{b\}\cup V(M)$$

implies similarly that

$$\mathcal{D}(G,M,V(M)\cap A)=\{b\}.$$

Thus by Lemma 4.3.7 G has a stable matching. ■

Lemma 4.3.10 *Let $G=(A\cup B\cup C,E)$ be the hypergraph underlying an instance of C3GSM of size n. Let M be a matching in G such that $|M|\geq n-3$. Suppose that $\mathcal{B}(G[V(M)],M)=\emptyset$, and*

$$\mathcal{S}(G,M)\cap A\subseteq V(M)$$

and there exists $b\in B\backslash V(M)$ satisfying

$$\mathcal{S}(G,M)\cap B\subseteq\{b\}\cup V(M)$$

and for all $a\in A\backslash V(M)$, $f(G[V(G)\backslash V(M)],a,1)=b$. Suppose also that there exists $c\in C\backslash V(M)$ satisfying

$$\mathcal{S}(G,M)\cap C\subseteq\{c\}\cup V(M).$$

Then G has a stable matching.

Proof: Similar to the proof of 4.3.9. ■

Another idea in finding stable matchings is to consider vertices left out of matchings. The following lemma shows that if you have a large enough matching where a dissatisfied vertex is not matched to their last choice then that matching can be extended to a stable matching.

Lemma 4.3.11 *Let $G=(A\cup B\cup C,E)$ be the hypergraph underlying an instance of C3GSM of size n. Let M be a matching in G with $|M|=n-2$. Suppose there exists $a\in A\cap V(M)$ such that*

$$V(M)\cap\mathcal{D}(G,M)=\{a\}.$$

If $f(G,a,n)$ is not in $V(M)$ then G has a stable matching.

In figure 4.3.1 we see that vertex a_2 is the only vertex in $V(M)$ blocking the matching. It prefers only b_3 to its partner and b_4 is the last choice of a_2. Since b_4 is unmatched in M and $|M| = 3 = 5 - 2$ we see that this example satisfies the conditions of Lemma 4.3.11.

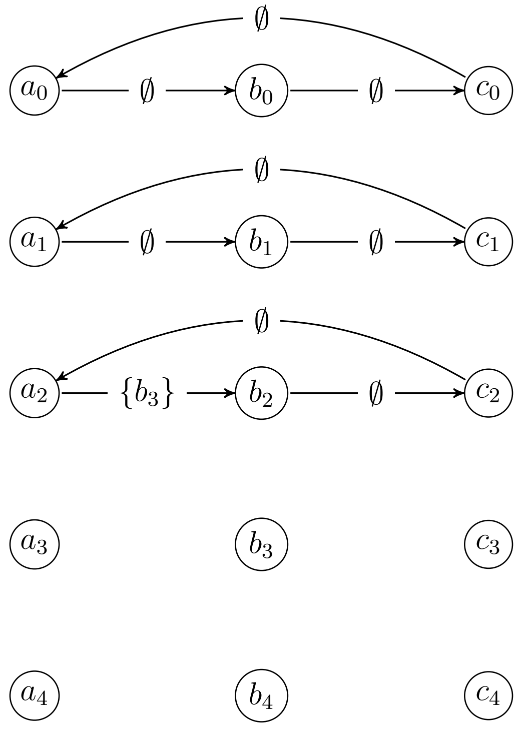

Figure 4.2: Visualizing Lemma 4.3.11

Solid arcs indicate matching M, and are labelled with $\mathcal{D}(G, M, \{t\})$ where t is the tail vertex of the arc. In this figure b_4 is the last choice of a_2.

Proof: By the size of M there exist $b, b' \in B$ such that $\{b, b'\} = B \backslash V(M)$. Without loss of generality say that $b = f(G, a, n)$. Let $G' = G[V(G) \backslash V(M)]$ and let $c' = f(G', b', 1)$. Let $a' \in A \backslash V(M)$. Let $T = \{a'b'c'\}$ and let $T' = \{a''bc''\}$ where a'' and c'' the remaining two vertices unmatched in M or T. Then $S = M \cup T \cup T'$ is a perfect matching of G. We claim that S is stable.

Let $v \in V(G)$. We will show that $v \notin \mathcal{D}(G,S)$ and thus that S is stable. If $v \in V(M) \cap (A \backslash \{a\} \cup B \cup C)$ then $v \notin \mathcal{D}(G,M)$ and thus $v \notin \mathcal{D}(G,S)$ as $M \subseteq S$. If $v = b'$ then $w >_v S(v)$ implies that $w \in V(M) \cap C$ by our construction of T. Since such $w \notin \mathcal{D}(G,S)$ we have $v = b' \notin \mathcal{D}(G,S)$. Now if $v = a$ then $w >_v S(v)$ implies that $w \in V(M) \cap B$ or $w = b'$. In either case $w \notin \mathcal{D}(G,S)$ and thus $a \notin \mathcal{D}(G,S)$. Now we have $V(M) \cap A \cap \mathcal{D}(G,S) = \emptyset$. So $\mathcal{D}(G,S) \subseteq \{a', a'', b, c', c''\}$. But $\{a', a'', b, c', c''\}$ consists only of vertices matched together in either the triple of T or the triple of T', hence they cannot form a blocking triple, as any triple from $\{a', a'', b, c', c''\}$ would contain a vertex and their match in S. Therefore $\mathcal{D}(G,S) = \emptyset$ and thus S is stable. ∎

Stabilizing A, B, C: The previous lemmas have the common theme of building matchings that have no blocking triples in the hypergraph among the matched vertices then using induction to make the matching perfect. The following lemma and corollary take a different strategy. Whereas before we tried to create a matching where matched vertices have no blocking triples even with unmatched vertices, alternatively we can attempt to pair all the vertices in one of A, B, C with a partner in such a way that none of them will participate in a blocking triple.

A motivation to consider this different approach comes from the context we intend to use these lemmas. In our computer search procedure to be defined in 4.3.3 we use these lemmas to eliminate instances of $C3GSM$ from consideration without actually finding a stable matching. A desirable feature in these lemmas is that they can eliminate instances with minimal knowledge of the preferences of the vertices. That is a lemma that can prove a stable matching exists only considering $f(G, v, 1)$ for all v is preferable to one that also needs to check $f(G, v, 2)$.

With the above motivation in mind, the intuitive difference between the lemmas previously described and those to be described is that the previous lemmas capture a lot of instances where vertices of A or B or C largely agree on which vertices are high ranking in terms of preferences. For example when there is a $1, 1, 1$ triple corollary 4.3.5 proves the existence of a stable matching. A distinctly different situation, but still operating within the first choices of vertices occurs when there are no $1, 1, 1$ triples, but each vertex in A has a different first choice vertex in B. Then one of the corollaries to follow, corollary 4.3.13, says that there is a stable matching in this instance.

Lemma 4.3.12 *Let $G = (A \cup B \cup C, E)$ be the hypergraph underlying an instance of $C3GSM$. Let M be a perfect A-B matching (that is a perfect matching of the underlying* *complete bipartite graph between A and B). Let $B' = \{b \in B : \exists a \in A : b >_a M(a)\}$. Hence B' is the set of vertices through which agents of A can form blocking triples. If for all $b, b' \in B'$, $f(G, b, 1) \neq f(G, b', 1)$ then G has a stable matching.*

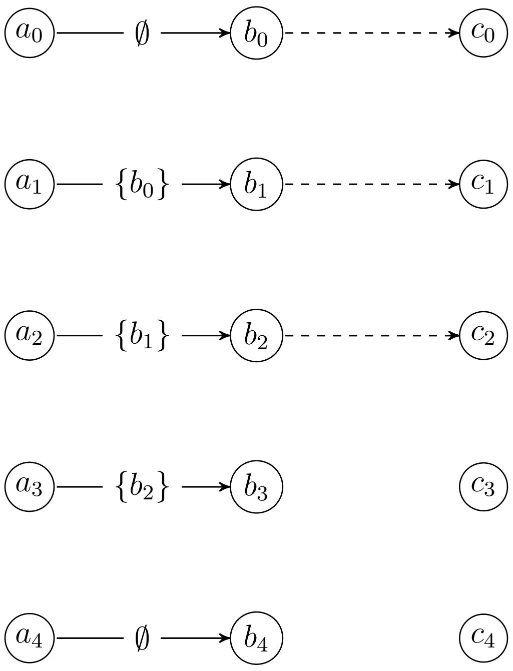

Figure 4.3: Visualizing Lemma 4.3.12

Solid arcs indicate matching M, and are labelled with $\mathcal{D}(G, M, \{t\})$ where t is the tail vertex of the arc. Dashed arcs indicate first choice of tail vertex.

Proof: Let $C' = \{c \in C : \exists b \in B', c = f(G, b, 1)\}$. Since $f(G, \cdot, 1)$ is a bijection between B' and C' we can form the matching $S_1 = \{abc \in E : a = M(b) \text{ and } c = f(G, b, 1)\}$. Let S_2 be an arbitrary perfect matching of $G[V(G) \backslash V(S_1)]$, the remaining vertices unmatched in S_1. Let $S = S_1 \cup S_2$. Then S is a complete matching on G. We claim that S is stable. Suppose for a contradiction there exists blocking triple a, b, c. Then $b >_a M(a)$ and hence $b \in B'$. Thus $M(b) = f(G, b, 1)$ and so $M(b) \geq_b c$ by definition of $f(G, \cdot, 1)$, but this contradicts that a, b, c is a blocking triple since that requires $c >_b M(b)$. ■

Corollary 4.3.13 *If $f(G, a, 1) \neq f(G, a', 1)$ for all $a, a' \in A$ in a hypergraph $G = (A \cup B \cup C, E)$ underlying a C3GSM instance then G has a stable matching.*

Proof: Observe that $M = \{ab : b = f(G, a, 1)\}$ is a perfect A-B matching since $f(G, \cdot, 1)$ is a bijection between A and B. In this case we have $B' = \emptyset$ by the definition of $f(G, \cdot, 1)$. Thus we may invoke Lemma 4.3.12 to obtain a stable matching of G. ■

4.3.2 Symmetry in Problem Instances

We will now turn our attention towards formulating an equivalence relation on instances of $C3GSM$. In doing so we hope to answer the question "when does knowledge that one instance of $C3GSM$ has a stable matching translate to knowing that another instance does?". This question is motivated by our efforts to design a computer search protocol in the next subsection. In checking if all $C3GSM$ instances of a certain size have a stable matching it is desirable to avoid repeated work by not checking instances "symmetric" to one already checked.

Let G and H be two hypergraphs underlying distinct instances of $C3GSM$. Let $>_v$ for $v \in V(G)$ denote the orders associated with vertices in G. Let $>'_v$ for $v \in V(H)$ denote the orders associated with vertices in H. We say that G with $>$ and H with $>'$ are *equivalent* in the sense that they denote symmetric problem instances, written $G \equiv H$, if there exists a bijection $\phi : V(G) \to V(H)$ such that for all $v \in V(G)$,

$$w >_v u \text{ if and only if } \phi(w) >'_{\phi(v)} \phi(u). \tag{4.1}$$

Note 4.3.14 *It is clear that $\equiv$ is an equivalence relation. Through the identity bijection we have reflexivity, through the inverse bijection we have symmetry, and through composition of bijections we have transitivity. Thus $\equiv$ is an equivalence relation as desired.*

Lemma 4.3.15 *If $G \equiv H$ then G has a stable matching if and only if H has a stable matching*

Proof:

By symmetry it suffices to prove the sufficiency direction. Let M be a stable matching of G. Define the matching $\phi(M)$ in H as

$$\phi(M) = \{\phi(a)\phi(b)\phi(c) : abc \in M\}.$$

It is not hard to see that since M is a matching and ϕ is a bijection that $\phi(M)$ is a matching. Suppose for a contradiction that $\phi(M)$ is not stable for H. Let xyz be a blocking triple of $\phi(M)$ against H. Let $abc = \phi^{-1}(x)\phi^{-1}(y)\phi^{-1}(z)$. By our definition of $\phi(M)$, $abc \notin M$. Further since $y >_x \phi(M)(x)$, by the definition of ϕ we have $b >_a M(a)$. Similarly $c >_b M(b)$ and $a >_c M(c)$. Thus abc is a blocking triple of M, a contradiction. ■

Note 4.3.16 *Since we require that ϕ is a bijection it is necessary that G and H underlie instances of $C3GSM$ of the same size, say n. Let $V(G)$ be partitioned as $V(G)_0 \cup V(G)_1 \cup V(G)_2$ since G is tripartite. Let $i \in \{0,1,2\}$. Since each $V(G)_i$ is of size n we may label the elements of $V(G)_i$ as $(i,0),\ldots,(i,n-1)$. We may use the same labelling convention for $V(H)$. Then ϕ is a bijection from $Z_3 \times Z_n$ to itself which preserves each vertex's order of its neighbours (and consequently the graph incidence structure). So we can say a few things about ϕ from this perspective.*

Observations Firstly by 4.1 we must preserve the graph structure. Hence for any $i \in \{0,1,2\}$ there exists $j \in \{0,1,2\}$ such that $\phi(V_i) = V_j$ otherwise the graph H is not tripartite. Furthermore if $\phi(V_i) = V_j$ then $\phi(V_{i+1}) = V_{j+1}$ with addition taken modulo 3. Again this follows from condition 4.1.

Together the previous observations imply that there exists $r \in \{0,1,2\}$ such that $\phi(V_i) = V_{i+r}$ for any $i \in \{0,1,2\}$. Hence we can specify ϕ by (r,Π) where $\Pi = \{\pi_0,\pi_1,\pi_2\}$ is a family of three permutations π_i on $\mathbb{Z}_n$, one for each part $V(G)_i$. In specifying $\phi = (r,\Pi)$ we require that (r,Π) satisfies a translation of condition 4.1 for all $(i,a) \in V(G)$; that is,

$$(i{+}1,b) >_{(i,a)} (i{+}1,b') \text{ if and only if } (i{+}1{+}r,\pi_{i+1}(b)) >_{(i+r,\pi_i(a))} (i{+}1{+}r,\pi_{i+1}(b')). \quad (4.2)$$

Of interest to us are the equivalence classes of a given problem instance under $\equiv$. For a hypergraph G underlying an instance of $C3GSM$ we will denote the equivalence class containing G by $[G]$. In a procedure to test whether instances of $C3GSM$ of a certain size have stable matchings one can hope to avoid unnecessary repeated work by only testing one representative from $[G]$ instead of all instances in $[G]$. We will now define a strict total order on instances of $C3GSM$ of size n, with the goal of obtaining a method to test only the minimum element with respect to this order for each equivalence class $[G]$.

To define an order on instances of $C3GSM$ of size n we will translate instances into finite sequences then apply a natural lexicographic order to said sequences. The next sequence of definitions will make this formal.

Let G be a hypergraph underlying an instance of $C3GSM$ of size n. Let $i \in \{0, 1, 2\}$ and let $j \in \{0, \ldots, n-1\}$. We define

$$seq(G, i, j) = f(G, (i, j), 1) f(G, (i, j), 2) \ldots f(G, (i, j), n)$$

to be the sequence of first through to last choices for vertex (i, j).We sometimes will refer to $seq(G, i, j)$ as (i, j)'s preference list.

Let G be a hypergraph underlying an instance of $C3GSM$ of size n. Let $i \in \{0, 1, 2\}$. We define a sequence for each vertex partition, $seq(G, i)$, by concatenating $seq(G, i, j)$ for each $j \in \{0, \ldots, n-1\}$. Formally we say

$$seq(G, i) = seq(G, i, 0) \ldots seq(G, i, n-1).$$

Let G be a hypergraph underlying an instance of $C3GSM$ of size n. We define our sequence for G, $seq(G)$, by concatenating the sequences corresponding to each vertex partition. Formally we have

$$seq(G) = seq(G, 0) seq(G, 1) seq(G, 2).$$

So $seq(G)$ consists of concatenating the preference lists of vertices 0 through $n-1$ of $V(G)_0$ followed by concatenating the same for $V(G)_1$ and finally $V(G)_2$.

We will use the symbol $>_{lex}$ to denote our lexicographical order of instances of $C3GSM$ of a given size n. Let G and H be hypergraphs underlying size n instances of $C3GSM$. We say $G >_{lex} H$ if $seq(G)$ is lexicographically larger than $seq(H)$. That is there exists some k such that

$$seq(G)_k > seq(H)_k$$

and for all $i < k$

$$seq(G)_i = seq(H)_i.$$

We compare vertices in $seq(G)_k$ and $seq(H)_k$ as if they are natural numbers corresponding to their labels. That is if $seq(G)_k = (i, a)$ and $seq(H)_k = (j, b)$ then we simply say $seq(G)_k > seq(H)_k$ if and only if $a > b$.

Unfortunately we do not have a full characterization of the lexicographic minimum of a given equivalence class under the order $>_{lex}$, which would be desirable. Instead we do have some necessary conditions which we collect in the following lemma. These are useful for finding instances of $C3GSM$ that we can avoid testing in a computer search since we are certain we are at least testing the lexicographic minimum of each equivalence class.

The following lemma presents four necessary conditions which we will summarize intuitively before presenting in rigorous formality. The first condition says that vertex $(0,0)$ has preference list $0, 1, \ldots, n-1$, which means that $(1,0) >_{(0,0)} (1,1) >_{(0,0)} (1,2) >_{(0,0)} \cdots >_{(0,0)} (1, n-1)$. The second condition says the same thing regarding $(1,0)$. The third condition says that preference lists of vertices in A are sorted lexicographically. The last condition says that you cannot rotate the roles of A, B, C to obtain a lexicographically smaller instance than the lexicographic minimum.

Lemma 4.3.17 *Let G be the hypergraph underlying an instance of $C3GSM$ of size n. If G is the lexicographic minimum of $[G]$ then the following are satisfied:*

1. $\mathit{seq}(G, 0, 0) = 0, 1, \ldots, n-1$,
2. $\mathit{seq}(G, 1, 0) = 0, 1, \ldots, n-1$,
3. *let* $a, b \in \{0, 1, \ldots, n-1\}$ *and let* $k \in \{1, \ldots, n\}$*. If* $f(G, (0,a), k) < f(G, (0,b), k)$ *and for all* $j \in \{0, \ldots, k-1\}$, $f(G, (0,a), j) = f(G, (0,b), k)$ *then* $a < b$,
4. *We have that*
$$\mathit{seq}(G) \leq_{lex} \mathit{seq}(G,1)\mathit{seq}(G,2)\mathit{seq}(G,0)$$
and
$$\mathit{seq}(G) \leq_{lex} \mathit{seq}(G,2)\mathit{seq}(G,0)\mathit{seq}(G,1).$$

Proof: We first prove (1). Let $\phi : V(G) \to V(G)$ be described by $(0, e, \pi_1, e)$ where e denotes the identity permutation and π_1 is the permutation which sends $\mathit{seq}(G, 0, 0)$ to $0, 1, \ldots, n-1$. Let H be the hypergraph underlying an instance of $C3GSM$ such that $G \equiv H$ with respect to ϕ. Since G is the lexicographic minimum of $[G]$ and $H \in [G]$, $\mathit{seq}(G, 0, 0) \leq_{lex} \mathit{seq}(H, 0, 0) = 0, 1, \ldots, n-1$. Since $0, 1, \ldots, n-1$ is the lexicographic minimum sequence possible we have $\mathit{seq}(G, 0, 0) = 0, 1, \ldots, n-1$.

To prove (2) we do something similar to (1). Let ϕ be described by $(0, e, e, \pi_2)$ where π_2 is the permutation which sends $\mathit{seq}(G, 1, 0)$ to $0, 1, \ldots, n-1$. Let H be the hypergraph underlying an instance of $C3GSM$ such that $G \equiv H$ with respect to ϕ. Observe that since $r = 0$ and $\pi_0 = \pi_1 = e$, $\mathit{seq}(G, 0) = \mathit{seq}(H, 0)$. Since G is the lexicographic minimum of $[G]$ and $H \in [G]$, $\mathit{seq}(G, 1, 0) \leq_{lex} \mathit{seq}(H, 1, 0) = 0, 1, \ldots, n-1$. Since $0, 1, \ldots, n-1$ is the lexicographic minimum sequence possible we have $\mathit{seq}(G, 1, 0) = 0, 1, \ldots, n-1$.

We now turn our attention to proving (3). Observe that $seq(G,0,a) <_{lex} seq(G,0,b)$. Let ϕ be described by $(0,\pi_0,e,e)$ where π_0 is the permutation which transposes a and b. Let H be the hypergraph underlying an instance of $C3GSM$ such that $G \equiv H$ with respect to ϕ. Suppose for a contradiction that $b < a$. Since $r = 0$, $\pi_1 = \pi_2 = e$, and π_0 simply transposes a and b, this yields

$$\begin{aligned} seq(H,0) &= seq(G,0,0)\dots seq(G,0,b-1)seq(G,0,a)seq(G,0,b+1) \\ &\quad \dots seq(G,0,a-1)seq(G,0,b)seq(G,0,a+1)\dots seq(G,0,n-1) \\ &\quad <_{lex} seq(G,0). \end{aligned}$$

This contradicts the lexicographic minimality of G.

Finally we prove (4). Let ϕ be described by $(1,e,e,e)$. We will prove $seq(G)$ is lexicographically at most $seq(G,1)seq(G,2)seq(G,0)$. The proof that G is lexicographically smaller that $seq(G,2)seq(G,0)seq(G,1)$ follows similarly using $(2,e,e,e)$. Let H be the hypergraph underlying an instance of $C3GSM$ such that $G \equiv H$ with respect to ϕ. Since $\pi_0 = \pi_1 = \pi_2 = e$ and $r = 1$ we have that

$$seq(H) = seq(G,1)seq(G,2)seq(G,0)$$

and thus by the lexicographic minimality of G, $seq(G) \leq_{lex} seq(H)$ and the result follows. ■

4.3.3 Computer Search

Our goal now is to describe a computer search procedure to test all instances of $C3GSM$ for a given size n under the assumption that for $k < n$ it is known that all instances of $C3GSM$ have a stable matching. In our case we study $n = 5$. A major challenge towards solving this problem computationally is that enumerating all instances of $C3GSM$ would be infeasible. For each vertex there is one possible preference order or each permutation of $\{1,\dots,n\}$. There are $3n$ such vertices, and thus there are $(5!)^{15}$ possible problem instances for $C3GSM$ with size 5.

We resolve to overcome the sheer number of possible $C3GSM$ instances by eliminating instances from consideration using our symmetry and sufficiency checks theory from the previous subsections. Our approach is based on the idea that if we relax the total orders to partial orders giving only the first few preferences of each vertex and can find a perfect stable matching then any $C3GSM$ instance that results from completing the partial order will also have said stable matching. The following theory formalizes this idea. Herein and henceforth we use the ordered pair labeling for vertices of G as outlined in Note 4.3.16.

A *partially specified instance* of $C3GSM$ of size n provides a complete 3-partite hypergraph G, as in an instance of $C3GSM$, but not the total orders for each of the vertices of G. In our partially specified instance there exists a vertex (g, v) called the *extender*, and an integer $\ell \in \{1, \dots, n+1\}$ called the *length* satisfying that:

1. For all (h, w) where $h < g$, or $h = g$ and $w < v$ we have that

   $$f(G, (h, w), 1), \dots, f(G, (h, w), \ell)$$

   are specified in the partial order of (h, w), and

   $$f(G, (h, w), \ell + 1), \dots, f(G, (h, w), n)$$

   are not specified in the partial order of (h, w). That is, (h, w) specifies their top-ranked ℓ preferences in their partial order.

2. For all (h, w) where $h = g$ and $w \geq v$, or $h > g$ we have that

   $$f(G, (h, w), 1), \dots, f(G, (h, w), \ell - 1)$$

   are specified in the partial order of (h, w) and

   $$f(G, (h, w), \ell), \dots, f(G, (h, w), n)$$

   are not specified in the partial order of (h, w). That is, (h, w) specifies their top-ranked $\ell - 1$ preferences in their partial order.

The idea of the definition is that every vertex lexicographically smaller than the extender has only specified their first ℓ choices, where ℓ is the length, and every vertex lexicographically at least the extender has only specified their first $\ell - 1$ choices. We give a few quick examples:

- A partially specified instance of $C3GSM$ with no vertices specifying any of their orders has extender $(0, 0)$ and length 1. This is the unique “empty” instance.

- A partially specified instance of $C3GSM$ of size 5 that is in fact an instance of $C3GSM$ proper has extender $(0,0)$ and length 6. Observe that since 6 is one greater than the size of the instance this is the only situation where the length is 6.
- The two previous examples can be generalized. A partially specified instance of $C3GSM$ of size n where each vertex has specified their first k choices, for $k \in \{0, \dots, n\}$ has extender $(0,0)$ and length $k+1$.
- A partially specified instance of $C3GSM$ that has only every vertex of $V(G)_0$ specifying their first choice and no other choices specified has extender $(1,0)$ and length 1.
- A partially specified instance of $C3GSM$ of size 5 with extender $(2,2)$ and length 3 has every vertex of $V(G)_0$ and $V(G)_1$ specifying their first 3 choices. It also $(2,0)$ and $(2,1)$ specifying their first 3 choices. The vertices $(2,2)$, $(2,3)$, and $(2,4)$ specify only their first 2 choices.

Let I and I' be two partially specified instances of $C3GSM$ of size n with the same underlying hypergraph G. Let (g,v) be the extender of I and ℓ be the length of I. Let (g',v') be the extender of I' and ℓ' be the length of I'. We say that I' is an *extension* of I provided that for all $v \in V(G)$ the preference order specified by v in I is contained in the preference order specified by v in I', and the following cases are satisfied:

1. If $(g,v) = (2, n-1)$ then $(g',v') = (0,0)$ and $\ell' = \ell + 1$.
2. If $g < 2$ and $v = n-1$ then $(g',v') = (g+1, 0)$ and $\ell' = \ell$.
3. If $g < 2$ and $v < n-1$ then $(g',v') = (g, v+1)$ and $\ell' = \ell$.

For example consider the partially specified instance of $C3GSM$ of size 5 where the first four vertices of $V(G)_0$ specify their first choice to be $(1,0)$, and the remaining vertices specify no choices. This instance has extender $(0,4)$ and length 1. The partially specified instances of $C3GSM$ where the first four vertices of $V(G)_0$ specify their first choice to be $(1,0)$ and the vertex $(0,4)$ specifies $f(G,(0,4),1) = k$ where $k \in \{(1,0), \dots, (1,4)\}$ are extensions of our original partially specified instance. In this example the indicator of our original instance satisfies the third case in our definition of extension.

As another example consider an instance of $C3GSM$ where each vertex has specified their first choice, and each vertex of $V(G)_0$ has specified their second choice. Each extension would have the same preferences but additionally have $(1, 0)$ specify their second choice. This example covers the second case in our definition of extension.

Any instance of $C3GSM$ where the first k preferences have been specified for each vertex provides an example for the first case in our definition of extension. In any such instance the extensions have vertex $(0, 0)$ specifying their $k + 1$-th preference.

Let I and I' be partially specified instances of $C3GSM$ with the same underlying graph. We say that I' is a *descendant* of I if there exists a sequence of partially specified $C3GSM$ instances

$$I = I_0, I_1, \ldots, I_k = I'$$

such that for all $i \in \{0, \ldots, k-1\}$ we have that I_{i+1} is an extension of I_i.

Let I be a partially specified instance of $C3GSM$. We say that I is *stable* if every instance of $C3GSM$ which is a descendant of I has a stable matching.

Computer Search Algorithm We now describe our procedure for deciding if each instance of $C3GSM$ of size $n = 5$ has a stable matching. For any partially specified instance of $C3GSM$, I, let $\mathcal{E}(I)$ denote the list of partially specified instances of $C3GSM$ which are extensions of I and let $\mathcal{P}(I)$ denote the unique instance such that I is an extension of $\mathcal{P}(I)$. For convention we will say $\mathcal{P}(I) = I$ for the empty instance with extender $(0, 0)$ and length 1. The algorithm is as follows:

1. Let I be the partially specified instance of $C3GSM$ of size 5 with extender $(0, 0)$ and length 1.
2. While $\mathcal{E}(I) \neq \emptyset$ or $\mathcal{P}(I) \neq I$
   (a) If I is stable via a lemma of 4.3.1 or the graph underlying I does not satisfy a conclusion of Lemma 4.3.17 then set $\mathcal{E}(I) \leftarrow \emptyset$ and $I \leftarrow \mathcal{P}(I)$, and continue to the next iteration of step 2.
   (b) Else If $\mathcal{E}(I) = \emptyset$ then return I as a counterexample to conjecture 4.2.3
   (c) Otherwise let $E \in \mathcal{E}(I)$. Remove E from $\mathcal{E}(I)$ and set $I \leftarrow E$.

3. If While loop terminates with no counterexample found then conjecture 4.2.3 is proven for $n = 5$ and return that it is so.

On using lemmas of subsection 4.3.1 In step 2(a) it may not be clear how to use the lemmas of subsection 4.3.1 to conclude that I is stable. This is actually easier to verify in many cases than is immediately apparent. If one can form a matching in I that only matches vertices with those that they have specified a ranking for, then it is easy to see which vertices they prefer to the matching. When the k-th choice of a vertex in an extension is specified all their choices of rank less than k have previously been specified. Hence if we have a matching M valid in a partially specified instance I, by which we mean for each $v \in V(M)$, $M(v)$ is ranked in v's partial order with respect to I, then the set of vertices preferred by some $v \in V(M)$ to $M(v)$ is completely known at I and does not change for any descendant of I. Therefore Lemmas 4.3.4, 4.3.9, 4.3.10, and 4.3.12 can be checked for instance I without worry that any descendant instances would violate their hypotheses. This follows since descendant instances are generated by a sequence of extensions which do not alter the set of vertices preferred by any matched vertex to their partner. There is a notable exception in Lemma 4.3.11 which we use in a slightly different way.

Fixing Choices With Lemma 4.3.11 Lemma 4.3.11 requires knowledge of a vertex's last choice in order to conclude that the given problem instance has a stable matching. Unfortunately lemmas which rely on last choices don't naturally lend themselves to cutting instances from consideration early on via step 2(a). There is an alternative approach to using them though. Suppose there is a partially specified instance I and matching M for which each vertex except for a that is matched in I is matched to a partner that they have already specified their ranking for in I, and all vertices a has specified a ranking for are matched in M. Also suppose that M satisfies all the requirements of lemma 4.3.11 except for the requirement that a's last choice is unmatched in M (as a consequence of not knowing a's last choice). Then by lemma 4.3.11 any fully specified instance of $C3GSM$ extended from I wherein $M(a)$ is not the last choice of a is one where a's last choice is unmatched in M and hence has a stable matching. Thus we know we need only consider extensions where $M(a)$ is the last choice of a. If one can find another such matching M' with $M'(a) \neq M(a)$ then we know I is stable since the last choice of a cannot simultaneously be $M'(a)$ and $M(a)$.

Example of Fixing Choices Consider the following partially specified instance of $C3GSM$:

$a_0 : b_0 > b_1$	$b_0 : c_0 > c_1$	$c_0 : a_4 > a_3$
$a_1 : b_1 > b_0$	$b_1 : c_1 > c_0$	$c_1 : a_0 > a_1$
$a_2 : b_0 > b_2$	$b_2 : c_2 > c_3$	$c_2 : a_2$
$a_3 : b_0 > b_3$	$b_3 : c_0 > c_3$	$c_3 : a_0$
$a_4 : b_0 > b_2$	$b_4 : c_0 > c_3$	$c_4 : a_0$

If we consider matching $M_1 = \{a_0b_0c_0, a_1b_1c_1, a_2b_2c_3\}$ then the matching is only violated with respect to the partner of c_0. Further there are no blocking triples internal to M_1 except possibly with c_0 and $|M_1| = 3 = 5 - 2$. By the previous discussion the only descendants of this instance which could possibly not be stable are those where c_0 has a_0 as their last choice. But if we consider $M_2 = \{a_0b_1c_1, a_1b_0c_0, a_2b_2c_2\}$ this matching similarly says the only descendants of this instance which could possibly not be stable are those where c_0 has a_1 as their last choice. Since c_0 cannot possibly have both a_0 and a_1 as their last choice all descendants of this instance are stable.

Correctness of Algorithm With the prior discussion in mind it becomes clear that the computer search procedure works. If the graph underlying some partially specified instance I does not satisfy a conclusion of lemma 4.3.17 then no instance of $C3GSM$ which is a descendant of I will satisfy that same conclusion and thus is not the lexicographical minimum instance of its equivalence class. Therefore step $2(a)$ only eliminates instances of $C3GSM$ which not lexicographical minima or have a stable matching by a lemma of subsection 4.3.1. In the former case, by lemma 4.3.15, we need not consider this instance as it has a stable matching if and only if the lexicographical minimum of its equivalence class, which we do consider, has a stable matching. In the latter case the sufficient condition lemmas tell us directly we need not consider these instances as they have a stable matching. Hence all removed instances during the execution of the computer search algorithm have a stable matching or are equivalent to an instance considered by the algorithm. Furthermore all instances of $C3GSM$ are considered since they all are descendants of the "empty" instance the algorithm starts running with.

The intuition behind our computer search procedure is that we can often conclude that a problem instance has a stable matching without looking at all the preferences of each vertex. Recall, for instance that if there is a cycle of vertices, all respective first choices of each other, then there is a stable matching. The goal is to develop enough theory of lemmas that imply stable matchings as in subsection 4.3.1 that we can eliminate many instances in step $2(a)$ early on in execution. To see the benefit of this suppose we eliminate in step $2(a)$ a partially specified instance that has only specified each vertex's first choice. There are $(4!)^{15}$ possible fully specified instances of $C3GSM$ which are descendants of the eliminated instance that we do not need to check now.

Results of Computational Experiments We implemented our computer search algorithm in Java (see github.com/tothw/cyclicstablematching for our source code). We were unsuccessful in getting the algorithm to terminate but we were able to eliminate many preferences. We are confident that our computations are close to proving that all $C3GSM$ instances of size 5 admit a stable matching or finding a counterexample to 4.2.3. Our confidence comes from the observation that the current implementation very rarely needs to go deeper than specifying every vertices' second choice to decide that the instance has a stable matching. However our computations have not terminated yet, so nothing can be said about the conjecture for $n = 5$ at this time. On the other hand for considering instances of size $n \geq 6$ we will need considerably more ideas characterizing when an instance has a stable matching in order to hope that the computational approach will go through.

Chapter 5

Conclusion

In this chapter we summarize our work and suggest some open problems for further investigation.

5.1 Summary

The focus of this thesis has been on understanding the nature of stable matching problems. In Chapter 2 we gave the necessary background to read this work, and also gave some interesting previously known results about the structure of the classical stable matching problem. This culminated in a lattice theorem for stable matchings: theorem 2.3.6.

In Chapter 3 we formulated an instance of the stable matching problem as the feasible region of a linear program. We then proved that the extreme points of this feasible region were integral using a technique inspired by the methods of iterative rounding. In our proof the structure given by the lattice theorem was instrumental in obtaining the result.

In Chapter 4 we turn our attention to stable matching problems with more than two parties. We investigated the cyclic stable matching problem with 3 "genders" and complete preference lists. It is conjectured (4.2.3) that all such instances of $C3GSM$ have a stable matching. We considered the case where instance sizes were $n = 5$, one greater than the best known. We offered some sufficient conditions for existence of a stable matching, and some theory about symmetry in problem instances. This new understanding was applied in a computational search framework to attempt to find counterexamples to the conjecture or prove that none exist. We implemented and ran experiments with this procedure, but could not find a counterexample or exhaust all possible cases.

It is my hope for the reader that through studying this work the rich structure of stable matchings has become apparent. Indeed this work is exciting because stable matching problems are very simple to state, and can arise so naturally in applications, yet they have such beautiful theoretical structure. It is this pairing of theory and application that makes these problems such a rewarding domain to work in. In the following section we will describe some open questions for the reader who has been excited by this work and hopes to jump in and solve problems.

5.2 Open Problems

To conclude we mention some problems that remain open. We will draw some questions from our work in Chapter 3 and Chapter 4.

5.2.1 Questions from Chapter 3

Convert Our Integrality Proof to an Iterative Rounding Procedure In spirit our short proof that the stable matching polytope is integral in chapter 3 is very close to the iterative rounding procedures discussed in [33]. If one were able to convert the proof into an iterative rounding algorithm and give a proof of correctness that would convert our approach into a more useful form. This is because there are natural extensions of those algorithms to more difficult problems, as can also be seen in [33]. In fact, that motivates the next question.

Approximation Algorithms via Our Integrality Proof It is well-known that the methods of iterative rounding are not simply for proofs of integrality, but are commonly used in designing approximation algorithms [33]. Since our proof of integrality for the stable matching polytope as given in chapter 3 is so close to an iterative rounding approach, can these methods be extended to given good approximations for hard variants of stable matching problems such as the introduction of non-strict preferences or relaxing the need for a bipartite graph [26]?

Consider the Popular Matching Polytope Besides stability there are types of desirable properties for matching under preferences, such as popularity [4]. A $\frac{1}{2}$-integral formulation of the popular matching polytope has recently been given [28], but an integral formulation is not known. To our knowledge no one has looked at approaching this problem from an iterative rounding perspective. Perhaps such a perspective could inspire an integral formulation and lead to simple proof as it did for stable matchings?

5.2.2 Questions from Chapter 4

Fully Characterize Symmetry In subsection 4.3.2 our discussion of symmetry in instances of $C3GSM$ left some questions open. We gave some necessary conditions for an instance to be the lexicographic minimum of its equivalence class. One opportunity would be to extend these conditions and give a full set of necessary and sufficient conditions for being the lexicographic minimum. Another opportunity once that is done is to give an algorithm that computes the lexicographic minimum instance in the equivalence class of a given problem instance of $C3GSM$. This would lead to being able to decide if two instances are symmetric since we could compute their lexicographic minimums and check if they are the same or not.

Solve the size n case for small n From our computational results, the size 5 case of conjecture 4.2.3 is close at hand, but still eluding us. It is my opinion that it is true, and that we just need more lemmas that yield stable matchings early on in running to make the computer search procedure go through. The size 5 case has been fruitful in leading to better understanding of sufficient conditions for stable matchings and so once resolved it may be worthwhile to consider size 6 and higher. Naturally of course at some point we must jump to the next question.

Resolve Conjecture 4.2.3 The big looming question is to resolve the conjecture that all instances of $C3GSM$ have a stable matching. We feel that our approach to studying the problem computationally may in fact lead to resolving the conjecture. Either with enough sophistication the computer search can be used to find a counterexample, or the theory built up to run the procedure for higher sizes reaches a critical mass where we understand enough to prove that conjecture 4.2.3 is true.

References

[1] Hernán Abeledo and Garth Isaak. A characterization of graphs that ensure the existence of stable matchings. *Mathematical social sciences*, 22(1):93–96, 1991.

[2] Hernán G Abeledo and Uriel G Rothblum. Stable matchings and linear inequalities. *Discrete Applied Mathematics*, 54(1):1–27, 1994.

[3] David J Abraham, Péter Biró, and David F Manlove. Almost stable matchings in the roommates problem. In *International Workshop on Approximation and Online Algorithms*, pages 1–14. Springer, 2005.

[4] David J Abraham, Robert W Irving, Telikepalli Kavitha, and Kurt Mehlhorn. Popular matchings. *SIAM Journal on Computing*, 37(4):1030–1045, 2007.

[5] Claude Berge. Two theorems in graph theory. *Proceedings of the National Academy of Sciences*, 43(9):842–844, 1957.

[6] Garrett Birkhoff. Tres observaciones sobre el algebra lineal. *Univ. Nac. Tucumán Rev. Ser. A*, 5:147–151, 1946.

[7] Péter Biró and Eric McDermid. Three-sided stable matchings with cyclic preferences. *Algorithmica*, 58(1):5–18, 2010.

[8] Charles Blair. Every finite distributive lattice is a set of stable matchings. *Journal of Combinatorial Theory, Series A*, 37(3):353–356, 1984.

[9] Vasek Chvatal. *Linear programming.* Macmillan, 1983.

[10] Economic Sciences Prize Committee et al. Stable matching: Theory, evidence, and practical design-the Sveriges Riksbank Prize in economic sciences in memory of Alfred Nobel 2012: Scientific background. Technical report, Tech. rep. The Nobel Foundation, Stockholm, Sweden, 2012.

[11] William J Cook, WH Cunningham, WR Pulleyblank, and A Schrijver. *Combinatorial optimization*. Springer, 2009.

[12] Thomas H Cormen, Charles E Leiserson, Ronald L Rivest, and Clifford Stein. *Introduction to algorithms*. MIT press, 2009.

[13] George B Dantzig, Alex Orden, Philip Wolfe, et al. The generalized simplex method for minimizing a linear form under linear inequality restraints. *Pacific Journal of Mathematics*, 5(2):183–195, 1955.

[14] Jack Edmonds. Paths, trees, and flowers. *Canadian Journal of Mathematics*, 17(3):449–467, 1965.

[15] Kimmo Eriksson, Jonas Sjöstrand, and Pontus Strimling. Three-dimensional stable matching with cyclic preferences. *Mathematical Social Sciences*, 52(1):77–87, 2006.

[16] Linda Farczadi, Konstantinos Georgiou, and Jochen Könemann. Stable marriage with general preferences. In *Algorithmic Game Theory*, pages 25–36. Springer, 2014.

[17] Tomás Feder. A new fixed point approach for stable networks and stable marriages. *Journal of Computer and System Sciences*, 45(2):233–284, 1992.

[18] David Gale and Lloyd S Shapley. College admissions and the stability of marriage. *The American Mathematical Monthly*, 69(1):9–15, 1962.

[19] David Gale and Marilda Sotomayor. Some remarks on the stable matching problem. *Discrete Applied Mathematics*, 11(3):223–232, 1985.

[20] Martin Grötschel, László Lovász, and Alexander Schrijver. The ellipsoid method and its consequences in combinatorial optimization. *Combinatorica*, 1(2):169–197, 1981.

[21] Dan Gusfield, Robert Irving, Paul Leather, and Michael Saks. Every finite distributive lattice is a set of stable matchings for a small stable marriage instance. *Journal of Combinatorial Theory, Series A*, 44(2):304–309, 1987.

[22] Dan Gusfield and Robert W Irving. *The stable marriage problem: structure and algorithms*. MIT press, 1989.

[23] Robert W Irving. An efficient algorithm for the stable roommates problem. *Journal of Algorithms*, 6(4):577–595, 1985.

[24] Robert W Irving. Stable marriage and indifference. *Discrete Applied Mathematics*, 48(3):261–272, 1994.

[25] Robert W Irving, Paul Leather, and Dan Gusfield. An efficient algorithm for the optimal stable marriage. *Journal of the ACM (JACM)*, 34(3):532–543, 1987.

[26] Kazuo Iwama and Shuichi Miyazaki. A survey of the stable marriage problem and its variants. In *International Conference on Informatics Education and Research for Knowledge-Circulating Society, 2008. ICKS 2008.*, pages 131–136. IEEE, 2008.

[27] Kamal Jain. A factor 2 approximation algorithm for the generalized steiner network problem. *Combinatorica*, 21(1):39–60, 2001.

[28] Telikepalli Kavitha. Popular half-integral matchings. In *LIPIcs-Leibniz International Proceedings in Informatics*, volume 55. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2016.

[29] Telikepalli Kavitha, Kurt Mehlhorn, Dimitrios Michail, and Katarzyna Paluch. Strongly stable matchings in time o (nm) and extension to the hospitals-residents problem. In *Annual Symposium on Theoretical Aspects of Computer Science*, pages 222–233. Springer, 2004.

[30] DE Knuth. Mariages stables. *Les Presses de l'Universite de Montreal, Montreal*, 1976.

[31] Donald Ervin Knuth. *Stable marriage and its relation to other combinatorial problems: An introduction to the mathematical analysis of algorithms*, volume 10. American Mathematical Soc., 1997.

[32] Harold W Kuhn. The Hungarian method for the assignment problem. *Naval Research Logistics Quarterly*, 2(1-2):83–97, 1955.

[33] Lap Chi Lau, Ramamoorthi Ravi, and Mohit Singh. *Iterative methods in combinatorial optimization*, volume 46. Cambridge University Press, 2011.

[34] David F Manlove. Stable marriage with ties and unacceptable partners. *Technical ReportTR-1999-29 University of Glasgow, Department of Computing Science*, 1999.

[35] David F Manlove. *Algorithmics of matching under preferences*, volume 2. World Scientific, 2013.

[36] David F Manlove, Robert W Irving, Kazuo Iwama, Shuichi Miyazaki, and Yasufumi Morita. Hard variants of stable marriage. *Theoretical Computer Science*, 276(1):261–279, 2002.

[37] David G McVitie and Leslie B Wilson. The stable marriage problem. *Communications of the ACM*, 14(7):486–490, 1971.

[38] James Munkres. Algorithms for the assignment and transportation problems. *Journal of the Society for Industrial and Applied Mathematics*, 5(1):32–38, 1957.

[39] Cheng Ng and Daniel S Hirschberg. Three-dimensional stable matching problems. *SIAM Journal on Discrete Mathematics*, 4(2):245–252, 1991.

[40] Guillaume Ratier. On the stable marriage polytope. *Discrete Mathematics*, 148(1):141–159, 1996.

[41] Alvin Roth. The nrmp as a labor market: understanding the current study of the match. *Journal of the American Medical Association*, 275:1054–1056, 1996.

[42] Alvin E Roth. The evolution of the labor market for medical interns and residents: a case study in game theory. *The Journal of Political Economy*, pages 991–1016, 1984.

[43] Alvin E Roth and Marilda A Oliveira Sotomayor. *Two-sided matching: A study in game-theoretic modeling and analysis*, chapter 2, page 37. Cambridge University Press, 1992.

[44] Uriel G Rothblum. Characterization of stable matchings as extreme points of a polytope. *Mathematical Programming*, 54(1-3):57–67, 1992.

[45] Susan L Saidman, Alvin E Roth, Tayfun Sönmez, M Utku Ünver, and Francis L Delmonico. Increasing the opportunity of live kidney donation by matching for two- and three-way exchanges. *Transplantation*, 81(5):773–782, 2006.

[46] Steve Smale. On the average number of steps of the simplex method of linear programming. *Mathematical Programming*, 27(3):241–262, 1983.

[47] Ernst Steinitz. Bedingt konvergente reihen und konvexe systeme. *Journal für die Reine und Angewandte Mathematik*, 143:128–176, 1913.

[48] William T Tutte. The factorization of linear graphs. *Journal of the London Mathematical Society*, 1(2):107–111, 1947.

[49] John H Vande Vate. Linear programming brings marital bliss. *Operations Research Letters*, 8(3):147–153, 1989.