An Algorithm for Stable Matching with Approximation up to the Integrality Gap

by

Natig Tofigzade

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, 2020

Author's Declaration

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Abstract

In the stable matching problem we are given a bipartite graph $G = (A \cup B, E)$ where A and B represent disjoint groups of agents, each of whom has ordinal preferences over the members of the opposite group. The goal is to find an assignment of agents in one group to those in the other such that no pair of agents prefer each other to their assignees.

In this thesis we study the stable matching problem with ties and incomplete preferences. If agents are allowed to have ties and incomplete preferences, computing a stable matching of maximum cardinality is known to be NP-hard. Furthermore, it is known to be NP-hard to achieve a performance guarantee of $33/29 - \varepsilon$ (≈ 1.1379) and UGC-hard to attain that of $4/3 - \varepsilon$ (≈ 1.3333). We present a polynomial-time approximation algorithm with a performance guarantee of $(3L - 2)/(2L - 1)$ where L is the maximum tie length. Our result matches the known lower bound on the integrality gap for the associated LP formulation.

Acknowledgements

I would like to express my deep gratitude to my primary supervisor, Prof. Jochen Koenemann, for his guidance and patience during my master's studies. He provided me with the freedom to pursue various ideas over these years. His invaluable feedbacks significantly increased the quality of the thesis.

I would also like to thank my second supervisor, Prof. Kanstantsin Pashkovich, for his encouragement and support. He gave me the courage to accomplish things that I would not have attempted otherwise.

Dedication

This is dedicated to my wife Asmar for her love, support, and encouragement in all my endeavours.

Table of Contents

List of Figures **ix**

List of Algorithms **x**

1 Introduction **1**

1.1 Motivation 1

1.2 Stable Matching 2

1.2.1 The Classical Stable Matching Problem 2

1.2.2 Stable Matching with Incomplete Preferences 3

1.2.3 Stable Matching with Ties 4

1.2.4 Stable Matching with Ties and Incomplete Preferences 5

1.3 Related Work 6

1.3.1 Maximum Stable Matching with One-Sided Ties 7

1.3.2 Maximum Stable Matching with Two-Sided Ties 8

1.4 Our Contribution 9

2 Background **11**

2.1 Matchings in Graphs 11

2.1.1 Basic Graph Concepts 11

2.1.2 Matchings 12

2.1.3 Maximum-Cardinality Matching 13

2.2 Maximum Stable Matching 14

2.2.1 The Gale-Shapley Algorithm 15

2.2.2 The Huang-Kavitha Algorithm for Ties of Length 2 17

2.2.3 The Improved Analysis of The Huang-Kavitha Algorithm by Chiang and Pashkovich 20

3 The Algorithm and Analysis **22**

3.1 Algorithm for Two-Sided Ties of Length up to L 22

3.1.1 How Men Propose 22

3.1.2 How Women Decide 23

3.1.3 The Algorithm 24

3.1.4 Stability of Output Matching 26

3.1.5 Running Time 27

3.2 Tight Analysis 27

3.2.1 Analytical Techniques 29

3.2.2 Inputs and Outputs 30

3.2.3 Cost 32

3.2.4 The Approximation Ratio 33

3.2.5 Costs of Connected Components in $M + \mathsf{OPT}$ 35

4 Concluding Remarks **50**

References **51**

List of Figures

1.1 Weakly stable matchings of different sizes 6

3.1 Illustrations of M-augmenting paths in $M + \mathsf{OPT}$ of length 3 on the left and of length 5 on the right. Dashed lines represent the edges in OPT and solid lines represent those in M. 39

3.2 An instance with ties of length at most L, $L \geq 2$ for which the algorithm outputs a stable matching M with $|\mathsf{OPT}|/|M| = (3L-2)/(2L-1)$ 49

List of Algorithms

1 The Gale-Shapley Algorithm 16
2 Phase 1 of the Huang-Kavitha algorithm (part 1) 18
3 Phase 1 of the Huang-Kavitha algorithm (part 2) 19
4 Stage 1 of the algorithm (part 1) 24
5 Stage 1 of the algorithm (part 2) 25

Chapter 1

Introduction

1.1 Motivation

In practice many computational problems are concerned with an assignment of agents in one group to those in the other. The agents usually constitute two disjoint sets where each agent in one has ordinal preferences over a subset of the agents in the other. Since many applications in practice are large-scale, centralized matching schemes are employed to feasibly compute optimal assignments. Perhaps the best-known example is the National Resident Matching Program (NRMP) that facilitates the placement of newly-graduated medical students into residency programs in the United States every year. In 2018, 43,909 new graduates applied through the NRMP for 33,167 positions offered [1]. Another reason for practicing centralized matching schemes is to achieve obligatory outcomes, that may be broken by the agents otherwise. To ensure this in the context of the NRMP, no medical graduate and hospital that are not assigned to each other should have a motivation to refuse their assignees and become matched together.

The abstract mathematical model of the above phenomena is called stable matching. The central and simplest model in the class of stable matchings is the classical stable marriage problem (commonly referred to as the stable matching problem), that was first studied by Gale and Shapley [8]. An instance of this problem involves a set of men and women, each of whom has strict preferences over the agents of the opposite gender. The objective is to find a group of man-woman pairs, namely a matching, such that no man and woman prefer each other to their assigned partners (a constraint called stability).

In the same paper [8], the authors also described a generalization of the classical stable marriage problem that fits to the context of the college admissions. The participants in the college admissions problem are called students on one side and colleges on the other side. Each agent again has preferences over a subset of the agents in the opposite group. The generalization is due to the capacity of each college representing the maximum number of students that can be assigned to the college, whereas a student can only be matched to at most one college. Although the stable matching problem is the special case of the college admissions problem in which the capacity of each college is one, Gale and Sotomayor [9] showed that a stable assignment for the latter can be computed by reducing it to the former using a natural cloning technique. The college admissions model is also known as the hospitals/residents problem, where the students and colleges are replaced by the residents and hospitals, respectively.

The challenge that the NRMP faces every year is the classical example of the hospitals/residents problem. The Canadian Resident Matching Service [2] is a similar centralized matching scheme in Canada for assigning medical students to training programs in hospitals.

1.2 Stable Matching

Finding a stable matching is one of the central problems in algorithmic game theory. It was first introduced by Gale and Shapley in their celebrated work [8] and has been extensively studied by mathematicians, computer scientists, and economists since. In this section we discuss the classical stable matching problem and its natural extensions. To find out more about the body of literature surrounding stable matching, we encourage the reader to see the outstanding monographs of Gusfield and Irving [10], Roth and Sotomayor [32], and Manlove [28].

1.2.1 The Classical Stable Matching Problem

An instance of the classical stable matching problem, introduced by Gale and Shapley [8], involves a set of n men and a set of n women such that each person specifies a *preference list* that ranks all the members of the opposite gender in strict order.

A matching M is a set of man-woman pairs such that each person belongs to at most one pair. If a man a and a woman b is a pair in M, we write $M(b) = a$ and $M(a) = b$. A person is said to be *matched* in M if he/she belongs to some pair in M, and *unmatched* otherwise. M is said to be a perfect matching if each individual belongs to exactly one pair.

In a given instance of the classical stable matching problem, a matching M is said to be *stable* if there are no man a and woman b such that a prefers b to $M(a)$, and b prefers a to $M(b)$. If M admits a pair (a, b) satisfying the above conditions, then M is said to be *unstable*; such a pair (a, b) is called a *blocking pair* for M.

Any stable matching M in an instance of the classical stable matching problem must be a perfect matching. To see this, suppose that M is not a perfect matching. Since the numbers of men and women are equal in the instance, there exists at least one unmatched man and one unmatched woman. Since all men are acceptable to every woman and vice versa in the instance, the unmatched man and woman together is a blocking pair contradicting that M is stable.

Gale and Shapley provided an algorithm [8], the Gale-Shapley algorithm, for finding a stable matching in $O(n^2)$ time. The Gale-Shapley algorithm provides a constructive proof that there always exists a stable matching for every instance of the classical stable matching problem. Later on, Irving and Leather [17] showed that for an instance of the classical stable matching problem the number of stable matchings may be exponentially many in the size of the instance.

1.2.2 Stable Matching with Incomplete Preferences

Since the introduction of the stable matching problem, several natural extensions of its classical form have been studied. A man a and a woman b are called *acceptable* to each other if each appears on the preference list of the other, and *unacceptable* otherwise. One generalization arises when the classical stable matching problem incorporates incomplete preference lists (SMI). In this case one or more individuals may find some members of the opposite gender unacceptable. Consequently, the definition of stability is also generalized accordingly. Given an instance of SMI, a matching M in the instance is said to be *stable* if there are no man a and woman b such that a and b are acceptable to each other, a is either unmatched or prefers b to $M(a)$, and b is either unmatched or prefers a to $M(b)$.

In an instance of SMI, while the number of men need not be equal to the number of women, it can be assumed without loss of generality that they are equal. Otherwise, one can easily add men and women, each of whom having an empty preference list, until the numbers of men and women are equal.

In contrast to the classical form, a stable matching in an instance of SMI may not be a perfect matching. A trivial example of this might be an instance involving an individual with an empty preference list. However, as in the case of the classical form, there always exists at least one stable matching in an instance of SMI, and it is straightforward to extend the Gale-Shapley algorithm to cope with incomplete preference lists and find such a matching in polynomial time (see [10, Section 1.4.2]).

While there may be numerous stable matchings in a given SMI instance, Gale and Sotomayor [9] showed that the set of all men and women can be partitioned into two subsets such that the members of one subset are matched in all stable matchings in the instance, and those of the other set are not matched in any. By doing so, they effectively proved that all stable matchings in the instance have the same size.

1.2.3 Stable Matching with Ties

An alternative extension of the classical stable matching problem emerges by relaxing the requirement that each individual ranks every member of the opposite gender in a strict order. In the stable matching problem with ties (SMT), each individual has a complete preference list, that is a partial order over the members of the opposite gender. Also, one can alternatively view the preference list of an individual as a sequence of ties, each of size at least one, in a strictly decreasing order of preference. In other words, an individual is indifferent between the members of any particular tie in his/her list, whereas he/she prefers each member of the tie to everyone in any subsequent tie.

Given that a' and a'' are men on the preference list of a woman b, she is said to *strongly prefer* a' to a'' if she strictly prefers a' to a''. b is *indifferent* between a' and a'' if she equally prefers a' and a''. If b is either indifferent between a' and a'' or strongly prefers a' to a'', she is said to *weakly prefer* a' to a''. Similarly, the notions of *indifference*, *strong preference*, and *weak preference* are defined for men.

Irving and Leather [17] introduced three distinct definitions of stability for SMT. Given an instance of SMT, a matching M is said to be *weakly stable* if there is no man a and woman b such that a strongly prefers b to $M(a)$, and b strongly prefers a to $M(b)$. A matching M is *strongly stable* if there is no man a and woman b such that either a strongly prefers b to $M(a)$ and b weakly prefers a to $M(b)$, or a weakly prefers b to $M(a)$ and b strongly prefers a to $M(b)$. Finally, a matching M is *super-stable* if there is no man a and woman b such that a weakly prefers b to $M(a)$, and b weakly prefers a to $M(b)$.

Given an instance $\mathcal{I}$ of SMT, breaking all ties arbitrarily gives rise to an instance $\mathcal{I}'$ of the classical stable matching problem. It is straightforward to see that a stable matching for $\mathcal{I}'$ is a weakly stable matching for $\mathcal{I}$. So one can strictly order all the members of each tie in an arbitrary way, and then apply the Gale-Shapley algorithm to find a weakly stable matching in a polynomial time [29]. This also implies that a weakly stable matching always exists. Indeed, Irving and Leather [17] showed that amongst all three different notions of stability it is only weak stability that always exists.

1.2.4 Stable Matching with Ties and Incomplete Preferences

In real-world problems ties and incomplete preference lists often arise simultaneously. In this thesis we focus on the stable matching problem with ties and incomplete preference lists (SMTI). Obviously, a preference list in an instance of SMTI may be incomplete and/or involve ties.

Within the setting of SMTI, the notions of stability given in [17] need to be generalized further [27]. A matching M for a given instance of SMTI is said to be *weakly stable* if there are no man a and woman b such that a and b are acceptable to each other, a is either unmatched or a strongly prefers b to $M(a)$, and b is either unmatched or b strongly prefers a to $M(b)$. Although the concepts of strong stability and super-stability are similarly generalized, we omit them. In this thesis we only focus on weak stability since weakly stable matchings always exist.

Given an instance of SMTI, it is again straightforward to find a weakly stable matching by breaking ties arbitrarily and invoking the Gale-Shapley algorithm (see [29]). We previously mentioned that all stable matchings in an instance of SMI are of the same size, and all weakly stable matchings in an instance of SMT are complete (and thus of the same size). However, in the case of SMTI, arbitrary tie-breaking technique leads to stable matchings of various sizes. To see this, consider the following example.

Example 1.1. *In Figure 1.1, there are two men a_0 and a_1, and two women b_0 and b_1, where a_0 finds only b_0 acceptable represented by the edge (a_0, b_0), and likewise, a_1 finds only*

b_1 acceptable represented by the edge (a_1, b_1). Furthermore, a_1 is indifferent between b_0 and b_1, and b_0 strictly prefers a_1 to a_0. In this setting the reader can verify that matchings $M = \{(a_0, b_0), (a_1, b_1)\}$ and $M' = \{(a_1, b_0)\}$, illustrated by the blue edges and the red edge, respectively, are both weakly stable.

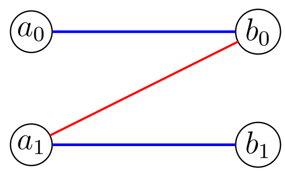

Figure 1.1: Weakly stable matchings of different sizes

As a consequence, it is reasonable to consider finding a maximum-cardinality weakly stable matching (hereafter a *maximum stable matching*) where the goal is to find a weakly stable matching of maximum size. Unfortunately, the problem of computing a maximum stable matching is known to be NP-hard [19]. Manlove et al. [29] showed that NP-hardness holds even for very restricted cases such as each tie is of length 2, ties appear on only one gender's preference lists, or at most one tie per list. Subsequently, a number of approximation algorithms have been designed to solve the maximum stable matching problem.

An algorithm for the maximum stable matching problem is said to be α-approximating if it always computes a stable (weakly) matching M with size $|M| \geq (1/\alpha) \cdot |\mathsf{OPT}|$ where OPT is an optimal stable matching. Various restricted cases of the maximum stable matching problem have been widely studied in the literature. Perhaps the most natural one is that ties appear on preference lists of only one gender, and it is called the maximum stable matching problem with one-sided ties. Within the context of restricted cases, we may emphasize the general case, that is the maximum stable matching problem, as the maximum stable matching problem with two-sided ties.

1.3 Related Work

In this section we review the previous work on the maximum stable matching problem. In Section 1.3.1, we concentrate on the maximum stable matching problem with one-sided ties. In Section 1.3.2, we review the prior results related to the maximum stable matching problem with two-sided ties that is the main focus of this thesis.

1.3.1 Maximum Stable Matching with One-Sided Ties

On the inapproximability side, Halldórsson et al. [12] showed for the case of one-sided ties that achieving an approximation ratio of $21/19 - \varepsilon$ (≈ 1.1053) is NP-hard, and that obtaining $5/4 - \varepsilon$ (≈ 1.25) is UGC-hard. These APX-hardness results still hold when each tie is of length at most 2.

On the positive side, Irving and Manlove [18] gave a $5/3$ (≈ 1.6666)-approximation algorithm for the special case where, in addition to being one-sided, ties are allowed only at the ends of preference lists. For the particular case of one-sided ties in which the length of each tie is at most 2, Halldórsson et al. [11] described a randomized algorithm that attains an expected ratio of $10/7$ (≈ 1.4286).

For the case of one-sided ties, Halldórsson et al. [12] provides an algorithm with an approximation ratio of $2/(1 + L^{-2})$ where L is the maximum tie length. This was later improved by Király [22] to an approximation ratio of $3/2$ ($= 1.5$) with an algorithm using the natural and effective idea of "promotion". The next important result was due to Iwama et al. [21] who presented an algorithm with an approximation guarantee of $25/17$ (≈ 1.47). Their algorithm relies on solving the linear programming relaxation of the associated IP formulation and uses a fractional optimal solution to break ties. Subsequently, Dean and Jalasutram [7] improved their algorithm and analysis to yield an approximation guarantee of $19/13$ (≈ 1.4615). Meanwhile, Huang and Kavitha [15] provided an approximation ratio of $22/15$ (≈ 1.4667) devising a linear-time algorithm that uses an approach of rounding half-integral stable matchings. Later on, Bauckholt et al. [3] gave a refined analysis of their algorithm establishing an approximation factor of $13/9$ (≈ 1.4444).

For the case of one-sided ties, Iwama et al. [21] demonstrated that the integrality gap for the corresponding linear programming formulation is at least $1 + (1 - \frac{1}{L})^L$ where L is the maximum tie length. Recently, Lam and Plaxton [24, 26] designed a polynomial-time algorithm with a $1 + (1 - \frac{1}{L})^L$-approximation factor that corresponds to the known lower bound on the integrality gap [21]. For the case of one-sided ties with unbounded tie length, their result implies an approximation guarantee of $1 + \frac{1}{e}$ as L approaches infinity in the limit [25]. When the maximum length of ties is 2, their result also implies an approximation guarantee of $5/4$ ($= 1.25$) that matches the known UGC-hardness result [12]. It is unknown whether hardness results, assuming unique games conjecture, can be improved for $L > 2$.

1.3.2 Maximum Stable Matching with Two-Sided Ties

On the negative side, Iwama et al. [19] were the first to show that the problem of finding a maximum stable matching (for the case with two-sided ties) is NP-hard, as referred to earlier. Furthermore, Halldórsson et al. [13] proved that there is a constant ε such that approximating the maximum stable matching within a factor of $1+\varepsilon$ is NP-hard. Later on, it was shown by Yanagisawa [33] that getting an approximation ratio of $33/29-\varepsilon$ (≈ 1.1379) is NP-hard, and that of $4/3-\varepsilon$ (≈ 1.3333) is UGC-hard. As in the case of one-sided ties, these APX-hardness results hold even when each tie is of length at most 2.

The simple extension of the Gale-Shapley algorithm due to [29] gives a 2-approximation solution for the case of two-sided ties. An important progress was achieved by Iwama et al. [20] who gave a $15/8$ ($= 1.875$)-approximation algorithm using a local search technique. The next breakthrough was due to Király [22] who developed a $5/3$ (≈ 1.6667)-approximation algorithm by coming up the ingenious idea of "promotion", that is increasing priorities of unmatched men to guide the tie-breaking process in a modification of the Gale-Shapley algorithm. Afterwards, McDermid [30] improved the approximation guarantee to $3/2$ ($= 1.5$) for the general case of two-sided ties using some ideas from the polynomial-time $3/2$-approximation algorithm by Király [22] that was designed for the case of one-sided ties. Thereafter, Paluch [31] and Király [23] gave linear-time $3/2$-approximation algorithms for the general case.

For the special case of two-sided ties where each tie is of length at most 2, Halldórsson et al. [12] were the first to give a non-trivial approximation algorithm, that is with a performance guarantee of $13/7$ (≈ 1.8571). For the identical particular case, the same authors [11] gave a randomized algorithm attaining an expected approximation factor of $7/4$ ($= 1.75$). Later on, Huang and Kavitha [15] designed a linear-time algorithm obtaining an approximation ratio of $10/7$ (≈ 1.4286) for the same special case. Their algorithm is a considerable modification of the Gale-Shapley algorithm [8], first computing a half-integral stable matching, and then rounding it to an integral stable matching. Subsequently, Chiang and Pashkovich [5] gave an effective and tight analysis of the algorithm by Huang and Kavitha [15], improving the approximation guarantee to $4/3$ (≈ 1.3333) that matches the UGC-hardness result [33] and the lower bound on the integrality gap given that each tie is of length at most two [21].

For the general case, Iwama et al. [21] demonstrated that the integrality gap for the corresponding natural linear programming formulation is at least $(3L-2)/(2L-1)$ where L is the maximum tie length. When L approaches infinity in the limit, their result simply implies a lower bound of 3/2 on the integrality gap that matches the best-known approximation guarantees in [23, 30, 31].

1.4 Our Contribution

In this thesis we focus on the maximum stable matching problem with two-sided ties of length up to L and obtain a polynomial-time $(3L-2)/(2L-1)$-approximation algorithm. This matches the known lower bound on the integrality gap of the natural LP [21]. Our main result is captured in the following theorem.

Theorem 1.2. *Given an instance of the maximum stable matching problem with incomplete preferences and ties of length at most L; the polynomial-time algorithm described in Section 3.1 finds a stable matching M with*

$$|M| \geq \frac{2L-1}{3L-2}\,|\mathsf{OPT}|,$$

where OPT is an optimal stable matching.

Our algorithm is an extension of that by Huang and Kavitha [15] for the special case of the maximum stable matching problem where ties are of length 2: every man has L proposals where each proposal goes to the acceptable women. Women can accept or reject these proposals under the condition that no woman holds more than L proposals at any point during the algorithm. Similar to the algorithm in [15], we use the concept of "promotion" introduced by Király [22] to grant men repeat chances in proposing to women. In comparison to [15], the larger number of proposals in our algorithm leads to subtle changes to the forward and rejection mechanisms of women, and to further modifications to the way we obtain the output matching.

Our analysis is inspired by the analyses of both Chiang and Pashkovich [5] and Huang and Kavitha [15], but requires several new ideas to extend it to the setting with larger ties. In both [15] and [5], the analyses are based on *charging schemes*: some objects are first assigned some values, called charges, and then charges are redistributed to nodes by a cost function. After a charging scheme is determined, relations between the generated total charges and the sizes of both output and optimal matchings are established, respectively, that lead to an approximation ratio. The analysis in [15] employs a complex charging scheme that acts globally, possibly distributing charges over the entire graph. In contrast, the charging scheme in [5] is local in nature and exploits only the local structure of the output and optimal matchings, respectively.

We do not know of a direct way to extend the local cost-based analysis of [5] to obtain an approximation algorithm whose performance beats the best known 3/2-approximation for the general case. Indeed, we believe that any such improvement must involve a non-trivial change in the charging scheme employed. As a result, we propose a new analysis that combines local and global aspects from [5, 15]. The central technical novelty in the analysis is captured by Lemma 3.14 that provides an improved lower bound on the "cost" of components whereas Corollary 3.17 bounds the "cost" from below by a simple multiple of the number of edges that are contained both in an optimal matching and in the components. As we will see in Chapter 3, our new charging scheme allows for a more fine-grained accounting of augmenting paths for the output matching of our algorithm.

Chapter 2

Background

In this chapter we provide basic definitions, fundamental structural and algorithmic results that will be useful in Chapter 3. The sections of this chapter are organized as follows. In Section 2.1, we begin with basic graph definitions followed by broad matching concepts in graphs. Then we give a characterization of maximum-cardinality matchings due to Berge [4]. In Section 2.2, we review fundamental definitions as well as key algorithmic results concerning maximum stable matchings.

2.1 Matchings in Graphs

2.1.1 Basic Graph Concepts

A *graph* G is a pair (V, E) where V is a finite set and E is a set involving two-element subsets of V. The elements of V are called *nodes* or *vertices*. An element $e = \{v, w\}$ of E is called an *edge* with the endpoints v and w, and may shortly be written as $e = vw$. A graph G is called *directed* if E is a set of ordered pairs, and *undirected* otherwise. We will be treating only undirected graphs in this paper.

If vw is an edge, then it is said that vw is *incident* to v and w, and that v and w are *adjacent* to or *neighbours* of each other. For a node $v \in V$, we denote by $N(v) = \{e \in E : v \in e\}$ the set of edges incident to v. The *degree* of a node v, denoted by $deg(v)$, is the number of edges incident to v, i.e. $deg(v) \coloneqq |N(v)|$.

An edge is called a *loop* if both endpoints of the edge are the same. Two or more edges having the same endpoints are called *parallel edges* or *multiple edges*. A graph G is called *simple* if it has neither loops nor parallel edges.

To avoid ambiguity, we write $V(G)$ and $E(G)$ to refer to the node and edge sets of G, respectively. Given G, a graph $H = (V(H), E(H))$ with $V(H) \subseteq V(G)$ and $E(H) \subseteq E(G)$ is called a *subgraph* of G. Note that given an edge set E, we may write $V(E)$ to refer to the set of all nodes that are incident to some edge on E.

A *path* P in a graph G is a sequence $v_0, e_1, v_1, \ldots, e_k, v_k$ of nodes v_i with no repeated node, and edges e_i such that $e_i = v_{i-1}v_i$ for every $i = 1, 2, \ldots, k$. Thus we say that P is a (v_0, v_k)-path. A *cycle* is a path P with the exception that $v_0 = v_k$ where $k \geq 1$ and all the edges are distinct. A graph G is called *connected* if there is a (v, w)-path for all $v, w \in V(G)$. Any graph G can be partitioned into connected subgraphs that are also called *connected components* of G. A connected component that contains only one node is called an *isolated node*.

A graph G is called *bipartite* if its node set $V(G)$ can be partitioned into two subsets V_1 and V_2 such that for every edge $vw \in E(G)$ either $v \in V_1$ and $w \in V_2$, or $v \in V_2$ and $w \in V_1$.

2.1.2 Matchings

Given a graph $G = (V, E)$, a set $M \subseteq E$ is a *matching* in G if every node in V is incident to at most one edge in M. A matching M in G is said to be *maximal* if it is not a subset of any other matching in G.

Given a matching M in G, a node $v \in V$ is said to be *matched* in M if v is incident to some member of M, i.e. $v \in V(M)$; otherwise we say that v is *M-exposed*. If v is matched in M, then the *partner* of v, denoted by $M(v)$, is the node such that $\{v, M(v)\} \in M$. Note that $M(v)$ is undefined if v is M-exposed. It follows from the above definitions that the number of nodes matched in M is exactly $2|M|$, and that the number of M-exposed nodes is $|V| - 2|M|$.

Let M be a matching in a graph G. Let P be a path $v_0, e_1, v_1, \ldots, e_k, v_k$ in G. P is said to be an *alternating path* with respect to M, also called an *M-alternating path*, if the sequence of edges $e_1, e_2, \ldots, e_k$ alternate being contained in M and not in M. More formally, P is an M-alternating path if and only if either $e_i \in M$ and $e_{i-1}, e_{i+1} \notin M$, or $e_i \notin M$ and $e_{i-1}, e_{i+1} \in M$ for every $i = 2, 3, \ldots, k-1$. P is an *M-alternating cycle* if P is an M-alternating path and a cycle. M-alternating path P is called *M-augmenting* if the endpoints of P, namely v_0 and v_k, are M-exposed.

2.1.3 Maximum-Cardinality Matching

In this section we need some notation and terminology as follows. For sets S and T, we let $S \oplus T$ denote their *symmetric difference* given by $S \oplus T = (S \setminus T) \cup (T \setminus S)$, that is the set of all those elements belonging either to S or to T but not to both.

Let $\mathcal{M}$ denote the set of all matchings in G. A *maximum-cardinality matching* in G is a matching $M \in \mathcal{M}$ that has the largest number of edges. The following is a fundamental structural result that gives a characterization of maximum-cardinality matchings.

Theorem 2.1 (**Augmenting Path Theorem** [4]). *Let G be a graph, $\mathcal{M}$ be the set of all matchings in G, and $M \in \mathcal{M}$. Then M has maximum cardinality over the set $\mathcal{M}$ if and only if there is not M-augmenting path in G.*

Proof. First consider for direction ($\Longleftarrow$) that there exists an M-augmenting path P in G. Let $E(P) = \{v_0v_1, v_1v_2, \ldots, v_{k-1}v_k\}$ denote the edge set of P. We claim that $M' = M \oplus E(P)$ is a matching of greater size than M. To see that M' is a matching, we show that $|N(v) \cap M'| = 1$ holds for all $v \in V(M')$. Let $v \in V(M')$. Since v is matched in M', $|N(v) \cap M'| > 0$ holds. Now assume for a contradiction that $|N(v) \cap M'| \geq 2$. Then this implies that there exists two distinct edges $e_1, e_2 \in N(v)$ such that $e_1, e_2 \in M'$. Since M' is the symmetric difference of M and $E(P)$, it holds by definition that either $e_1 \in M$ or $e_1 \in E(P)$, and similarly either $e_2 \in M$ or $e_2 \in E(P)$. We cannot have $e_1, e_2 \in M$ since M is a matching, and $e_1, e_2 \in P$ since P is an M-alternating path. Thus, without loss of generality, we may assume that $e_1 \in M \setminus E(P)$ and $e_2 \in E(P) \setminus M$. Since P is an M-augmenting path, we deduce that $v_0v_1, v_{k-1}v_k \notin M$ holds, and since $e_1 \in M$, we conclude that $v \neq v_0$ and $v \neq v_k$. Since P is an M-alternating path, $e_2 \in E(P) \setminus M$, $v \neq v_0$ and $v \neq v_k$, there exists $e \in M \cap E(P)$ such that $e \in N(v)$. Since $e_1 \notin E(P)$, we have that $e \neq e_1$. But then $e, e_1 \in M$ with $e, e_1 \in N(v)$ implying $|N(v) \cap M| \geq 2$ that contradicts the fact that M is a matching. Hence $0 < |N(v) \cap M'| \leq 2$ implying that $|N(v) \cap M'| = 1$, and so M' is a matching. Since P is an M-augmenting path, $|E(P) \setminus M| = |M \cap E(P)| + 1$ holds. By basic set theory, $|M'| = |M| - |M \cap E(P)| + |E(P) \setminus M|$ holds. Thus we conclude that $|M'| = |M| + 1 > |M|$ implying M is not a maximum-cardinality matching in G.

Now consider for direction ($\Longrightarrow$) that M is not a maximum cardinality matching in G. Then there is some matching M' in G that satisfies $|M'| > |M|$. Consider the symmetric difference $J = M' \oplus M$. Let $G' = (V(G), J)$ be a subgraph of G. Notice that each node of G' has degree at most two. If, conversely, there was a node in G' incident to at least three edges of J, then either M or M' would contain at least two edges incident to v. This contradicts the fact that M and M' are matchings. Therefore any connected component of G' is a path or cycle. The sequence of edges in such components alternate belonging to M and to M'. Otherwise there would be a node in G' incident to two edges from the same matching, a contradiction. Since the cycles are alternating, they contain even number of edges, and contain the same number of edges of M and M'. Since $|M'| > |M|$, there must be at least one path, P, with more edges of M' than M. This follows from observing that each cycle contributes the same number to each of $|M'|$ and $|M|$. Thus P is an M-augmenting path. $\square$

We note that this problem is well-understood. A recommended textbook relevant for advanced undergraduate and new graduate students is Combinatorial Optimization by Cook, Cunninghum, Pulleyblank, and Schrijver [6] that covers the result stated above and much more.

2.2 Maximum Stable Matching

In an instance of the maximum stable matching problem, we are given a bipartite graph $G = (A \cup B, E)$ where, following standard terminology, the nodes in A will be referred to as *men*, and the nodes in B represent *women*. Each man $a \in A$ possesses a partial preference order over women in B, and similarly, every woman in B has a partial preference order over men in A.

We restate some terminology and introduce some notational conventions. Let $a' \in A$ and $a'' \in A$ be on the preference list of $b \in B$. If b equally prefers a' and a'', we say that b is *indifferent* between a' and a'', denoted by $a' \simeq_b a''$. Additionally, if a' is ranked strictly higher than a'' on the list of b, we say that b *strongly prefers* a' to a'', denoted by $a' >_b a''$. We say that b *weakly prefers* a' to a'', denoted by $a' \geq_b a''$, if b is either indifferent between a' and a'' or strongly prefers a' to a''. Analogously, we define *indifference*, *strong preference* and *weak preference* for men over women on their list. Adopting the notational convention, we use (a', b) or (b, a') to refer to an edge $\{a', b\} \in E$. For $c \in A \cup B$, recall that $N(c)$ is the set of nodes adjacent to c in G.

Given a graph instance $G = (A \cup B, E)$ of the maximum stable matching problem, a matching M in G is said to be *stable* if there are no $a \in A$ and $b \in B$ such that $(a, b) \in E \backslash M$, a is either unmatched or $b >_a M(a)$, and b is either unmatched or $a >_b M(b)$. If M admits a pair (a, b) satisfying the above conditions, then M is said to be *unstable*; such a pair (a, b) is called a *blocking pair* for M. In their seminal work [8], Gale and Shapley proposed an efficient algorithm for finding a stable matching, providing constructive proof that stable matchings *always* exist.

2.2.1 The Gale-Shapley Algorithm

In this section we give the straightforward extension of the Gale-Shapley algorithm due to Manlove et al. [29] for the maximum stable matching problem as follows. Let $G = (A \cup B, E)$ be an input bipartite graph with two-sided ties and incomplete preferences.

Initially, every man $a \in A$ and woman $b \in B$ is unmatched. Each man $a \in A$ maintains a rejection list $R(a)$ to record the women who reject him. While there exists an unmatched man $a \in A$ who has not been rejected by all women in his preference list, he makes a proposal to a most preferred woman in his list who is not a member of $R(a)$ (If there are several such women, a breaks the tie arbitrarily).

If a woman $b \in B$ has not received any proposal yet and gets a new proposal, she accepts the proposal. On the other hand, if b holds a proposal and receives another one, she keeps the proposal from a weakly preferred man and rejects the other (if both are weakly preferred, b breaks the tie arbitrarily).

When a woman $b \in B$ accepts the proposal from a man $a \in A$, they become matched. In contrast, if b rejects a, he remains unmatched. After the algorithm terminates, the matched pairs make up the output matching M.

Theorem 2.2 (**Gale-Shapley** [8]). *The Gale-Shapley algorithm always returns a stable matching.*

Proof. Let M be the matching returned by the Gale-Shapley algorithm. Suppose for a contradiction that M is not stable, i.e. suppose there exists an edge $(a, b) \in E, a \in A, b \in B$ that blocks M. In the following, we first consider the case that b rejected a proposal from a at some point during the algorithm, and then we consider the opposite.

If b rejected a proposal from a during the algorithm, then b holds a proposal, when the algorithm terminates, from a man $M(b)$ who is not less preferred than a by b. Hence we get a contradiction to the statement that (a, b) blocks M.

Contrarily, if b did not reject a proposal from a during the algorithm, then the proposal of a was accepted by some woman who is not less preferred than b by a. Again, we get a contradiction to the statement that (a, b) blocks M. □

Algorithm 1 The Gale-Shapley Algorithm

let $G = (A \cup B, E)$ be an instance graph
let $N(c)$ be the set of nodes adjacent to $c \in A \cup B$ in G
for all $a \in A$ **do**
    $R(a) \coloneqq \varnothing$ ▷ $R(a)$ is the rejection history of man a
end for
initialize M to the empty set of matching, i.e. $M \coloneqq \varnothing$
while $\exists a \in A$ s.t. a is unmatched in M and $R(a) \neq N(a)$ **do**
    let $b \in N(a) \setminus R(a)$ be a woman such that $b \geq_a b'$ for all $b' \in N(a) \setminus R(a)$
    if b is unmatched in M **then**
        $M \coloneqq M \cup \{(a, b)\}$
    else
        if $a >_b M(b)$ **then**
            $M \coloneqq M \cup \{(a, b)\} \setminus \{(M(b), b)\}$
            $R(M(b)) \coloneqq R(M(b)) \cup \{b\}$
        else
            $R(a) \coloneqq R(a) \cup \{b\}$
        end if
    end if
end while
return M

We note that the Gale-Shapley algorithm gives a 2-approximate solution for the maximum stable matching problem. To see this, let M be the output of Algorithm 1 and M' be an arbitrary stable matching. Clearly, M and M' are both maximal in the underlying graph G. It is well-known that any two maximal matchings differ in size with at most a factor of 2.

2.2.2 The Huang-Kavitha Algorithm for Ties of Length 2

Our algorithm in Chapter 3 is an extension of the algorithm by Huang and Kavitha [15] for the special case of the maximum stable matching problem where ties are of length ≤ 2. To present the origin of some ideas that are fundamental to our algorithm, we give a detailed description of the Huang-Kavitha algorithm in this section. We note that the original technique of "promotion" that both the Huang-Kavitha and our algorithms use is due to Király [22]. The same technique was also used by McDermid [30] and Iwama et al. [21].

Let $G = (A \cup B, E)$ be an input bipartite graph with two-sided ties of length 2 and incomplete preferences. The Huang-Kavitha algorithm consists of two phases. In the first one each man can make two proposals and each woman can accept two proposals. The outcome of the first phase is a subgraph G' of G that is built by keeping only the edges that correspond to the accepted proposals at the end of this phase. The output of the second phase and so of the algorithm is a maximum-cardinality matching in G' where all nodes of degree two are matched.

The two-proposal approach of the algorithm, in contrast to the one-proposal one of the Gale-Shapley algorithm, effectively deals with the lack of certainty in the following situation. In the Gale-Shapley algorithm, when a woman is to decide between two proposals from men who are tied on her list, she is unable to make the best decision in the sense that which proposal leads to a larger stable matching. Thus the strategy of letting women accept two proposals is an effective way of coping with their uncertainty. Since a woman can ultimately be matched with only one man, the men are also allowed to make multiple proposals to increase their chance of being matched. Below we describe how proposals are made by men and accepted by women in the first phase of the algorithm.

How men propose. Each man $a \in A$ has two proposals p_a^1 and p_a^2. A man starts out as *basic* and later becomes *1-promoted* before finally being raised to *2-promoted* status. Each man $a \in A$ keeps a *rejection history* $R(a)$ which records the women who reject a proposal from a during his current promotion status. At the start of the algorithm, $R(a)$ is initialized to the empty set for all $a \in A$.

Each proposal p_a^i for $a \in A$ and $i = 1, 2$ goes to a woman in $N(a) \setminus R(a)$ most preferred by a. If there are two most preferred women in $N(a) \setminus R(a)$, the man a breaks the tie arbitrarily. If a proposal p_a^i for $a \in A$ and $i = 1, 2$ is rejected by a woman $b \in B$, she is added to the rejection history of a, and afterwards, p_a^i is sent to a most preferred remaining woman in $N(a) \setminus R(a)$.

Suppose now that every woman in $N(a)$ rejected a proposal from a. If a is basic, then the status of a changes to 1-promoted, and the rejection history of a is cleared, i.e. $R(a)$ is reset to $\varnothing$. After that, when $R(a)$ becomes equal to $N(a)$ again, the status of a is elevated to 2-promoted, and the rejection history of a is emptied once more. Eventually, when the next time $R(a) = N(a)$ holds, a stops making proposals.

Algorithm 2 Phase 1 of the Huang-Kavitha algorithm (part 1)

let $G = (A \cup B, E)$ be an instance graph, and $N(c)$ denote the set of nodes adjacent to $c \in A \cup B$ in G
let $G' = (A \cup B, E')$ be a multigraph with E' initialized to the empty multiset of edges
let $\deg_{G'}(u)$ denote the degree of node u in G'
for all $a \in A$ **do**
    $R(a) \coloneqq \varnothing$ ▷ $R(a)$ is the rejection history of man a
    $stat_a \coloneqq 0$ ▷ $stat_a$ is the promotion status of man a
end for
while $\exists a \in A$ s.t. $\deg_{G'}(a) < 2$ and $R(a) \neq N(a)$ **do**
    let $b \in N(a) \setminus R(a)$ be a woman s.t. $b \geq_a b'$ for all $b' \in N(a) \setminus R(a)$
    PROPOSE(a, b)
end while
return E'

How women decide. Each woman $b \in B$ can hold up to 2 proposals, and among these more than one can come from the same man. Whenever she holds less than 2 proposals, newly received proposals are automatically accepted. Otherwise, b first tries to *bounce* one of her proposals, and if that fails, she will try to *forward* one of her proposals. If b can neither bounce nor forward a proposal, then b rejects a proposal. We continue describing the details below. Suppose that b currently holds two proposals $p_{a'}^{i'}$ and $p_{a''}^{i''}$, and receives a new proposal p_a^i for some $a, a', a'' \in A$ and $i, i', i'' = 1, 2$.

Bounce step. If there is a man $\alpha \in \{a, a', a''\}$ and a woman $\beta \in B$ such that $\beta \simeq_\alpha b$, and β currently holds less than 2 proposals, then a proposal of α is transferred from b to β, and the bounce step is called *successful*.

Forward step. If there is a man $\alpha \in \{a, a', a''\}$ and a woman $\beta \in B \setminus \{b\}$ such that $\beta \simeq_\alpha b$, two of the proposals in $\{p_a^i, p_{a'}^{i'}, p_{a''}^{i''}\}$ are from α, and β has not rejected α in his current status, then a proposal p_α^1 is forwarded from b to β, and the forward step is called *successful*. As a consequence of a successful forward step, α makes the proposal p_α^1 to β.

Algorithm 3 Phase 1 of the Huang-Kavitha algorithm (part 2)

{The following subroutine describes how b accepts the proposal from a, or bounces, forwards, or rejects a proposal}

procedure PROPOSE(a, b)
    if $\deg_{G'}(b) < 2$ **then**
        $E' := E' \cup \{(a, b)\}$
    else
        let a' and a'' be the nodes adjacent to b in G', and $A(b)$ denote $\{a, a', a''\}$
        if $\exists \alpha \in A(b)$ and $\exists \beta \in N(\alpha)$ s.t. $\beta \simeq_\alpha b$ and $\deg_{G'} \beta < 2$ **then**
            $E' := E' \cup \{(a, b), (\alpha, \beta)\} \setminus \{(\alpha, b)\}$ ▷ bounce
        else if $\exists \alpha \in A(b)$ and $\exists \beta \in N(\alpha) \setminus R(\alpha)$ s.t. $\beta \simeq_\alpha b$, and $|(E' \cup \{(a, b)\}) \cap \{(\alpha, b)\}| = 2$ **then**
            $E' := E' \cup \{(a, b)\} \setminus \{(\alpha, b)\}$
            PROPOSE(α, β) ▷ forward
        else
            if $a \simeq_b a' \simeq_b a''$ and $stat_a = stat_{a'} = stat_{a''}$ **then**
                let $\alpha_0 \in A(b)$ be a man s.t. $|(E' \cup \{(a, b)\}) \cap \{(\alpha_0, b)\}| = 2$
                {two of those in $\{a, a', a''\}$ are the same man since ties are of length 2}
            else
                let $\alpha_0 \in A(b)$ be a man s.t. $\alpha_0 \leq_b \alpha$ for all $\alpha \in A(b)$, and $stat_{\alpha_0} \leq stat_\alpha$ if $\alpha_0 \simeq_b \alpha$
            end if
            $E' := E' \cup \{(a, b)\} \setminus \{(\alpha_0, b)\}$ ▷ reject
            $R(\alpha_0) := R(\alpha_0) \cup \{b\}$
            if $R(\alpha_0) = N(\alpha_0)$ **then**
                if $stat_{\alpha_0} < 2$ **then**
                    $stat_{\alpha_0} := stat_{\alpha_0} + 1$
                    $R(\alpha_0) := \varnothing$
                end if
            end if
        end if
    end if
end procedure

Note that in both bounce and forward steps, b does not reject α, and so the rejection history of α is not updated. To describe the rejection step, the following concepts are required. For a woman $b \in B$, proposal $p^{i'}_{a'}$ is called *superior* to $p^{i''}_{a''}$ for $a', a'' \in A$ and $i', i'' = 1, 2$ if b strongly prefers a' to a'', or if b is indifferent between a' and a'', and a' has higher promotion status than a''. A proposal p^i_a for $a \in A$ and $i = 1, 2$ is called *least desirable* if p^i_a is not superior to either of the other two proposals.

Rejection step. If neither bounce nor forward steps are successful, then b rejects any of the least desirable proposals in $\{p^i_a, p^{i'}_{a'}, p^{i''}_{a''}\}$ in all cases except the following. Observe that if $a \simeq_b a' \simeq_b a''$ holds, then two of these proposals are from the same man since ties are of length 2; in this case if all three proposals are least desirable, b rejects one of two proposals from this man. Consequently, b is added to the rejection history of the man whose proposal is rejected.

The following result is due to Huang and Kavitha [15].

Theorem 2.3 (**Huang-Kavitha** [15])**.** *Let $G = (A \cup B, E)$ be an instance of the special case of the maximum stable matching problem where ties of length 2. The linear-time Huang-Kavitha algorithm computes a stable matching M such that $|\mathsf{OPT}|/|M| \leq 10/7 \approx 1.4286$ where OPT is an optimal stable matching.*

In the next section we discuss the improved analysis of the Huang-Kavitha algorithm by Chiang and Pashkovich [5]. The authors show that the real performance ratio of the algorithm is at most 4/3, and this result is tight under UGC [33].

2.2.3 The Improved Analysis of The Huang-Kavitha Algorithm by Chiang and Pashkovich

Our analysis in Chapter 3 is inspired by the analysis of Chiang and Pashkovich [5]. In this section we compare analytical techniques that are used in [5] and [15]. We first recall the following main result of Chiang and Pashkovich [5].

Theorem 2.4 (**Chiang-Pashkovich** [5])**.** *Let $G = (A \cup B, E)$ be an instance of the special case of the maximum stable matching problem where ties of length 2. The Huang-Kavitha algorithm computes a stable matching M such that $|\mathsf{OPT}|/|M| \leq 4/3 \approx 1.3333$ where OPT is an optimal stable matching.*

The analyses in both [5] and [15] are based on charging schemes. However, the charging scheme in [5] is considerably different from that in [15] as we will see below.

Naturally, both papers [5] and [15] study the connected components in the union (approximately) of output and optimal matchings for the purpose of comparing the sizes of these matchings. For ease of exposition, let M and OPT be output and optimal stable matchings, respectively. In [15], the authors first show that there are no M-augmenting paths with respect to OPT, that are of lengths 1 or 3. To obtain an approximation guarantee of at most 1.5, the charges are first assigned to M-augmenting paths with respect to OPT, that are of lengths 5. But in [5], the origin of charges are the proposals accepted at the end of the first phase.

In [15], the origin of charges together with their distribution mechanism forms a "global" charging scheme because of the following reason. After the charges are generated, they are first distributed to so-called "good paths", and then redistributed from "good paths" to nodes. "Good paths" may potentially transfer charges across the entire graph. In contrast, the charging scheme in [5] is "local" since the charges are generated by the proposals accepted at the end of the first phase and distributed only to the nodes participating in these proposals (the participants in a proposal are a man making the proposal and a woman holding that proposal).

The overall idea in [15] is that the total generated charges are distributed to other connected components so that the total charge of each component is not too large with respect to the component's size. In other words, other connected components pay for the cost caused by M-augmenting paths with respect to OPT, that are of lengths 5. However, the approach in [5] is rather simple. It is first shown that the total number of generated charges is at most $4|M|$. Later, it is demonstrated that every connected component receives a total charge of at least three times the number of the edges that are both in OPT and in the component. Thus they immediately get Theorem 2.4.

Chapter 3

The Algorithm and Analysis

In this chapter we study the maximum stable matching problem with two-sided ties. Our main result is a polynomial-time $(3L-2)/(2L-1)$-approximation algorithm where L is the maximum tie length.

In section 3.1, we present the description and the implementation of our algorithm. In section 3.2, we analyze the approximation ratio of our algorithm.

3.1 Algorithm for Two-Sided Ties of Length up to L

3.1.1 How Men Propose

Each man $a \in A$ has L proposals $p_a^1, p_a^2, \ldots, p_a^L$. A man starts out as *basic*, and later becomes *1-promoted* before he is eventually elevated to *2-promoted* status. Each man $a \in A$ has a *rejection history* $R(a)$ which records the women who reject a proposal from a during his current promotion status. Initially, we let $R(a) = \varnothing$ for all $a \in A$.

Each proposal p_a^i for $a \in A$ and $i = 1, 2, \ldots, L$ goes to a woman in $N(a) \setminus R(a)$ most preferred by a, and ties are broken arbitrarily. If a proposal p_a^i for $a \in A$ and $i = 1, 2, \ldots, L$ is rejected by a woman $b \in B$, b is added to the rejection history of a, and subsequently, p_a^i is sent to a most preferred remaining woman in $N(a) \setminus R(a)$.

Suppose now that $R(a)$ becomes equal to $N(a)$ for some man $a \in A$. If a is either basic or 1-promoted then a's rejection history is cleared, and a is promoted. Otherwise, if a is already 2-promoted, a stops making proposals.

3.1.2 How Women Decide

Each woman $b \in B$ can hold up to L proposals, and among these more than one can come from the same man. Whenever she holds less than L proposals, newly received proposals are automatically accepted. Otherwise, b first tries to *bounce* one of her proposals, and if that fails, she will try to *forward* one of her proposals. If b can neither bounce nor forward a proposal, then b rejects a proposal.

We continue describing the details. In the following, we let $P(b)$ and $A(b)$ denote the set of proposals currently held by $b \in B$ and the set of men corresponding to these, respectively. Suppose that $|P(b)| = L$, and that b receives a new proposal p_a^i for some $a \in A$ and $i = 1, \ldots, L$.

Bounce step. If there is a man $\alpha \in A(b) \cup \{a\}$ and a woman $\beta \in B$ such that $\beta \simeq_\alpha b$, and β currently holds less than L proposals, then we move one of α's proposals from b to β, and we call the bounce step *successful*.

Forward step. If there is a man $\alpha \in A(b) \cup \{a\}$ and a woman $\beta \in B \setminus \{b\}$ such that $\beta \simeq_\alpha b$, at least two proposals from α are present in $P(b)$, no proposal from α is present in $P(\beta)$ and β is not in $R(\alpha)$, then b *forwards* a proposal $p_\alpha^j \in P(b) \cup \{p_a^i\}$ for some $j = 1, \ldots, L$ to β and the forward step is called *successful*. As a consequence of a successful forward step, α makes the proposal p_α^j to β.

We point out that bounce and forward steps do not lead to an update to the rejection history of an involved man. To describe the rejection step, we introduce the following notions. For a woman $b \in B$, a proposal $p_{a'}^{i'}$ is called *more desirable* than $p_{a''}^{i''}$ for $a', a'' \in A$ and $i', i'' = 1, \ldots, L$ if b strongly prefers a' to a'', or if b is indifferent between a' and a'' and a' has higher promotion status than a''. A proposal $p_{a'}^{i'} \in P(b)$ is *least desirable* in $P(b)$ if $p_{a'}^{i'}$ is not more desirable than any proposal in $P(b)$. Whenever $b \in B$ receives a proposal p_a^i, $|P(b)| = L$, and neither *bounce* nor *forward* steps are successful, we execute a rejection step.

Rejection step. If there is s unique *least desirable proposal* in $P(b) \cup \{p_a^i\}$, then b rejects that proposal. Otherwise, if there is more than one least desirable proposal in $P(b)$,

b rejects a proposal from a man with the largest number of least desirable proposals in $P(b) \cup \{p_a^i\}$. If there are several such men, then we break ties arbitrarily. Consequently, b is added to the rejection history of the man whose proposal is rejected.

3.1.3 The Algorithm

An approximate maximum-cardinality stable matching for a given instance $G = (A \cup B, E)$ is computed in two stages.

Stage 1. Men propose in an arbitrary order and women bounce, forward or reject proposals as described above. The first stage finishes when, for each man $a \in A$, one of the following conditions is satisfied: all proposals of a are accepted; or $R(a)$ becomes equal to $N(a)$ for the third time.

We represent the outcome of the first stage as a bipartite graph $G' = (A \cup B, E')$ with the node set $A \cup B$ and the edge set E', where each edge $(a, b) \in E'$ denotes a proposal from a held by b at the end of the first stage. Note that G' may be a multigraph in which an edge of the form (a, b) appears with multiplicity equal to the number of proposals that b holds from a. Clearly, each node u in G' has degree at most L, denoted by $\deg_{G'}(u) \leq L$, since every man has at most L proposals that can be accepted and every woman can hold at most L proposals at any point in the first stage.

Algorithm 4 Stage 1 of the algorithm (part 1)

```
let G = (A ∪ B, E) be an instance graph, and N(c) denote the set of nodes adjacent
    to c ∈ A ∪ B in G
let G′ = (A ∪ B, E′) be a multigraph with E′ initialized to the empty multiset of edges
let deg_{G′}(u) denote the degree of node u in G′, and A(b) denote the set of nodes
    adjacent to b ∈ B in G′
for all a ∈ A do
    R(a) := ∅                                   ▷ R(a) is the rejection history of man a
    stat_a := 0                                 ▷ stat_a is the promotion status of man a
end for
while ∃a ∈ A s.t. deg_{G′}(a) < L and R(a) ≠ N(a) do
    let b ∈ N(a) \ R(a) be a woman s.t. b ≥_a b′ for all b′ ∈ N(a) \ R(a)
    PROPOSE(a, b)
end while
return E′
```

Algorithm 5 Stage 1 of the algorithm (part 2)

{The following subroutine describes how b accepts the proposal from a, or bounces, forwards, or rejects a proposal}

procedure PROPOSE(a, b)
    if $\deg_{G'}(b) < L$ **then**
        $E' := E' \cup \{(a, b)\}$
    else if $\exists \alpha \in A(b) \cup \{a\}$ and $\exists \beta \in N(\alpha)$ s.t. $\beta \simeq_\alpha b$ and $\deg_{G'} \beta < L$ **then**
        $E' := E' \cup \{(a, b), (\alpha, \beta)\} \setminus \{(\alpha, b)\}$ ▷ bounce
    else if $\exists \alpha \in A(b) \cup \{a\}$ and $\exists \beta \in N(\alpha) \setminus R(\alpha)$ s.t. $\beta \simeq_\alpha b$, $|(E' \cup \{(a, b)\}) \cap \{(\alpha, b)\}| \geq 2$ and $\alpha \notin A(\beta)$ **then**
        $E' := E' \cup \{(a, b)\} \setminus \{(\alpha, b)\}$
        PROPOSE(α, β) ▷ forward
    else
        let $\mathcal{A}$ denote $\{\alpha \in A(b) \cup \{a\}$: for all $a' \in A(b) \cup \{a\}$, $\alpha \leq_b a'$ and if $\alpha \simeq_b a'$, then $stat_\alpha \leq stat_{a'}\}$
        let α_0 be a man in $\arg\max_{\alpha \in \mathcal{A}} |(E' \cup \{(a, b)\}) \cap \{(\alpha, b)\}|$
        $E' := E' \cup \{(a, b)\} \setminus \{(\alpha_0, b)\}$ ▷ reject
        $R(\alpha_0) := R(\alpha_0) \cup \{b\}$
        if $R(\alpha_0) = N(\alpha_0)$ **then**
            if $stat_{\alpha_0} < 2$ **then**
                $stat_{\alpha_0} := stat_{\alpha_0} + 1$
                $R(\alpha_0) := \varnothing$
            end if
        end if
    end if
end procedure

Stage 2. We compute a maximum-cardinality matching M in G' such that all nodes of degree L in G' are matched. The existence of such a matching is guaranteed by Lemma 3.1. The result of the second stage is such a matching M, that is the output of the algorithm.

Lemma 3.1. *There exists a matching in the graph G' such that all nodes of degree L in G' are matched. Moreover, there is such a matching M, where all nodes of degree L in G' are matched and we have*

$$|M| \geq |E'|/L\,.$$

Proof. Consider the graph $G' = (A \cup B, E')$ and the following linear program

$$\begin{aligned} \max \quad & \sum_{e \in E'} x_e \\ \text{s.t.} \quad & \sum_{e \in \delta(u)} x_e \leq 1 \quad (u \in A \cup B) \\ & \sum_{e \in \delta(u)} x_e = 1 \quad (u \in A \cup B, \deg_{G'}(u) = L) \\ & x \geq 0. \end{aligned}$$

It is well-known that the feasible region of the above LP is an integral polyhedron. Moreover, the above LP is feasible as is easily seen by considering the point that assigns $1/L$ to each edge in E'. Hence there exists an integral point optimal for this linear program. Notice that every integral point feasible for this linear program is a characteristic vector of a matching in G', which matches all nodes of degree L in G'. To finish the proof, notice that the value of the objective function calculated at $x^\star$ equals $|E'|/L$. Thus the value of this linear program is at least $|E'|/L$, finishing the proof. □

3.1.4 Stability of Output Matching

Let the above algorithm terminate with a matching M. We first argue that it is stable.

Lemma 3.2. *The output matching M is stable in $G = (A \cup B, E)$.*

Proof. Suppose for a contradiction that M is not stable, i.e. suppose that there exists an edge $(a, b) \in E$ that blocks M. If b rejected a proposal from a during the algorithm, then b holds L proposals when the algorithm terminates and all these proposals are from men who are weakly preferred by b over a. Thus the degree of b in G' is L implying that b is matched in M with a man who is not less preferred than a by b. We get a contradiction to the statement that (a, b) blocks M.

Conversely, if b did not reject any proposal from a during the algorithm, then the algorithm terminates with all L proposals of a being accepted, particularly, by women who are weakly preferred by a over b. Therefore the degree of a in G' is L implying that a is matched in M with a woman who is not less preferred than b by a. Again, we get a contradiction to the statement that (a, b) is a blocking pair for M. □

3.1.5 Running Time

In this section our goal is to show that the running time of the algorithm is polynomial in the size of the graph $G = (A \cup B, E)$. We achieve our goal by demonstrating that each stage of the algorithm has a polynomial execution time. For the first stage, we illustrate that only a polynomial number of proposals are bounced, forwarded or rejected during this stage. For the second stage, the proof of Lemma 3.2 implies that it is sufficient to find an optimal extreme solution for a linear program of polynomial size.

First, we show that proposals are bounced only polynomially many times. For every $b \in B$, at most L proposals may be bounced to b. Indeed, with each proposal bounced to b, the number of proposals held by b increases; also, the number of proposals held by b never decreases or exceeds L during the algorithm. Hence at most $L|B|$ proposals are bounced during the first stage.

Second, we illustrate that proposals are forwarded only polynomially many times. For every $a \in A$, promotion status of a, and $b \in B$ such that $(a, b) \in E$, at most one proposal of a may be forwarded to b. To see this, let b' be a woman forwarding a proposal of a to b. Notice that b cannot bounce the proposal after b receives it because, otherwise, b' could bounce it by the transitivity of indifference. Observe also that b may forward a proposal from a only if she holds another proposal from him. Then it follows from the forward step that no woman can forward a proposal of a to b as long as b holds a proposal from him. If b rejects the proposal, then she is added to the rejection history of a, and so b does not receive any proposal from a unless the promotion status of a changes. Hence at most $3|A||B|$ proposals are forwarded during the first stage.

Finally, we demonstrate that proposals are rejected only polynomially many times. For every $a \in A$, promotion status of a, and $b \in B$ such that $(a, b) \in E$, b may reject at most L proposals from a. Indeed, b holds at most L proposals at any point in time, and since b is added to the rejection history of a after she rejected him, b does not receive any proposal from a unless the promotion status of a changes. Hence at most $3L|A||B|$ proposals are rejected during the first stage.

3.2 Tight Analysis

Recall that OPT is a maximum-cardinality stable matching in G, and let M be the output matching defined above. If $a \in A$ is matched with $b \in B$ in OPT, we write $\mathsf{OPT}(a) := b$

and $\mathsf{OPT}(b) := a$. Similarly, we use the notations $M(a) := b$ and $M(b) := a$ when $a \in A$ is matched with $b \in B$ in M. Note that our analysis is based on graph G' and therefore all graph-related objects will assume G'.

Definition 3.3. *A man $a \in A$ is called* successful *if the stage 1 of the algorithm terminates with all of his L proposals being accepted. Likewise, a woman b is called* successful *if she holds L proposals when the stage 1 of the algorithm stops. In other words, a person $c \in A \cup B$ is* successful *if the degree of c in G' is L, and* unsuccessful *otherwise.*

Definition 3.4. *A woman is called* popular *if she rejected a proposal during the algorithm, and* unpopular *otherwise.*

Remarks 3.5 and 3.6 below directly follow from the algorithm and are consequences of the bouncing and rejection steps, respectively.

Remark 3.5. *Let $a \in A$ and $b, b' \in B$ be such that b holds a proposal from a when the algorithm finishes, b' is unsuccessful, and $b' \simeq_a b$. Then b is unpopular.*

Proof. Suppose for a contradiction that b is popular. Then at some point she could not bounce or forward any one of her proposals, and so she was to reject a proposal. This implies that after b became popular, whenever she received a new proposal that could be bounced, that proposal would immediately be bounced. But then, when the algorithm terminates, b holds a proposal from a that could successfully be bounced to b', a contradiction. □

Remark 3.6. *Let $a, a' \in A$ and $b \in B$ be such that b holds at least two proposals from a when the algorithm finishes, b rejected a proposal from a' at some point, a is basic, and $a' \simeq_b a$. Then there is an edge (a', b) in G'.*

Proof. Suppose for a contradiction that $(a', b) \notin G'$ holds. Let t be the most recent point in time when b rejects a proposal from a'. Then it follows from the algorithm that, at t, $a'' \geq_b a'$ holds for all $a'' \in A(b)$. The rejection step also implies that, at t, there is no $a'' \in A$ such that $a' \simeq_b a''$, a'' is basic, and b holds more than one proposal from a''. Moreover, the algorithm implies that, after t, whenever she receives a new proposal from a man a'' such that $a'' <_b a'$, she will immediately reject it unless she successfully bounces or forwards it. Now, consider a point in time after t when there is a man a'' such that $a' \simeq_b a''$, b already holds a proposal from a'', and receives another proposal from a''. Then the rejection step implies that she will reject one of the proposals from a'' unless she successfully bounces or forwards it. But then, when the algorithm terminates, b holds at least two proposals from a, a contradiction. □

3.2.1 Analytical Techniques

In the following sections we define *inputs*, *outputs*, and *costs* that are used in our charging scheme, and so are central to our analysis. Before we take a closer look at these notions and define them formally, let us discuss phenomena captured by them.

We use two different objects, inputs and outputs, to differentiate between two different viewpoints on proposals accepted when the algorithm ends. In particular, inputs are associated with the viewpoint of women on the proposals whereas outputs are associated with the viewpoint of men. The choice of terms "inputs" and "outputs" is due to the analysis in [15] where the edges of G' are directed from men to women, and so each proposal becomes an "input" for the woman, and analogously becomes an "output" for the corresponding man.

Now we describe the ideas that motivated our definitions concerning outputs and inputs. Let $M + \mathsf{OPT}$ denote the multiset that contains the edges in M and the edges in OPT. To establish the approximation guarantee of our algorithm, we analyze each connected component in $M + \mathsf{OPT}$. In order to show that M-augmenting paths in $M + \mathsf{OPT}$ do not lead to a large approximation guarantee, we introduce the notions of *bad* and *good inputs* as well as *bad* and *good outputs*. For example, a certain number of bad inputs and bad outputs are generated by the edges incident to the endpoints of an M-augmenting path in $M + \mathsf{OPT}$. Indeed, as we will see later, if $a_0 - b_0 - a_1 - \ldots - a_k - b_k$ is an M-augmenting path in $M + \mathsf{OPT}$ of length $2k+1$, $k \geq 2$ where $a_0 \in A$, then b_0 has at least $L-2$ bad inputs and a_k has at least $L-2$ bad outputs. Then to show the approximation guarantee of $(3L-2)/(2L-1)$, we provide a way to obtain a lower bound on the number of bad inputs and bad outputs of men and women in each M-augmenting path; and later we provide an upper bound on the total number of bad inputs and bad outputs of all men and women.

To implement the above ideas, we use a charging scheme. Our charging scheme associates a cost with each man and each woman. These costs keep track of bad inputs and bad outputs: bad inputs lead to an increase of the corresponding woman's cost and bad outputs lead to an increase of the corresponding man's cost. We show that the total cost of all men and women is bounded above by $2L|M|$. On the other side, we provide a lower bound on the total cost by giving a lower bound on the cost of each connected component in $M + \mathsf{OPT}$. These upper and lower bounds lead to the desired approximation guarantee of $(3L-2)/(2L-1)$.

3.2.2 Inputs and Outputs

In our analysis inputs and outputs are fundamental edge-related objects for our charging scheme. Each edge in G' generates a certain number of charges. For example, as we will see in Section 3.2.3, if an edge (a,b) in G' belongs either to M or to OPT, two charges are generated by (a,b) so that one is carried to node a and one is carried to node b by cost function. To define similar charging mechanisms for the remaining types of edges in G', we first distinguish them as in the following definitions.

Definition 3.7. *Given an edge (a,b) in G', we say that (a,b) is an* output from $a \in A$ *and an* input to $b \in B$ *if (a,b) is not in $M + \mathsf{OPT}$.*

To illustrate how outputs and inputs are determined, for example, let $(a,b) \in M$, $a \in A$, $b \in B$ and $n_{(a,b)}$ be the number of edges of the form (a,b) in the multigraph G', then the edge (a,b) gives rise to the following number $s_{(a,b)}$ of inputs (and to the same number of outputs)

$$s_{(a,b)} := \begin{cases} n_{(a,b)} - 1 & \text{if } (a,b) \notin \mathsf{OPT} \\ 0 & \text{if } n_{(a,b)} = 1 \\ n_{(a,b)} - 2 & \text{otherwise} \end{cases} .$$

Definition 3.8. *An input (a,b) to $b \in B$ is called a* bad input *if one of the following is true:*

- *b is popular and $a >_b \mathsf{OPT}(b)$.*
- *b is popular, $a \simeq_b \mathsf{OPT}(b)$, but $\mathsf{OPT}(b)$ is unsuccessful.*
- *b is popular, a is 1-promoted, $\mathsf{OPT}(b)$ is successful, and $M(b) \simeq_b \mathsf{OPT}(b) \simeq_b a$.*

An input (a,b) to $b \in B$ is a good input *if it is not a bad input. In other words, an input (a,b) to $b \in B$ is a* good input *if one of the following is true:*

- *b is unpopular.*
- *b is popular and $\mathsf{OPT}(b) >_b a$.*
- *b is popular, $a \simeq_b \mathsf{OPT}(b)$, $\mathsf{OPT}(b)$ is successful, and a is not 1-promoted.*

- *b is popular, $a \simeq_b \mathsf{OPT}(b)$, $\mathsf{OPT}(b)$ is successful, but not $M(b) \simeq_b \mathsf{OPT}(b) \simeq_b a$.*

An output (a, b) from a man a is called a bad output *if one of the following is true:*

- *b is unpopular.*
- *b is popular, $b >_a \mathsf{OPT}(a)$, a is 1-promoted, but not $M(b) \simeq_b \mathsf{OPT}(b) \simeq_b a$.*
- *b is popular, $b >_a \mathsf{OPT}(a)$, and a is basic.*

An output from a man a is a good output *if that is not a bad output. In other words, an output (a, b) from a man $a \in A$ is a* good output *if one of the following is true:*

- *b is popular and $\mathsf{OPT}(a) \geq_a b$.*
- *b is popular, $b >_a \mathsf{OPT}(a)$, and a is 2-promoted.*
- *b is popular, $b >_a \mathsf{OPT}(a)$, a is 1-promoted, and $M(b) \simeq_b \mathsf{OPT}(b) \simeq_b a$.*

Lemma 3.9. *There is no edge which is both a bad input and a bad output.*

Proof. Assume that an edge (a, b), $a \in A$, $b \in B$ is both a bad input to b and a bad output from a. First, consider the first case from the definition of a bad output. It trivially contradicts all the cases from the definition of a bad input. Second, consider the first case from the definition of a bad input and either the second or the third case from the definition of a bad output. Then the case (1) below is implied. Third, consider the second case from the definition of a bad input and either the second or the third case from the definition of a bad output. Then the case (2) below is implied. Finally, consider the third case from the definition of a bad input. It trivially contradicts both the second and the third case from the definition of a bad output. Thus one of the following cases is true:

1. $a >_b \mathsf{OPT}(b)$; $b >_a \mathsf{OPT}(a)$.
2. $a \simeq_b \mathsf{OPT}(b)$ and $\mathsf{OPT}(b)$ is unsuccessful; a is not 2-promoted.

In case (1), the edge (a, b) is a blocking pair for OPT, contradicting the stability of OPT.

In case (2), since $\mathsf{OPT}(b)$ is unsuccessful, $\mathsf{OPT}(b)$ was rejected by b as a 2-promoted man. On the other hand, $a \simeq_b \mathsf{OPT}(b)$, a is not 2-promoted, and b holds a proposal from a when the algorithm terminates, contradicting the rejection step.

□

Corollary 3.10. *The number of good inputs is at least the number of bad outputs.*

Proof. Assume for a contradiction that the number of good inputs is smaller than the number of bad outputs. Then there is an edge in G' which is a bad output but not a good input. In other words, there is an edge in G' which is both a bad output and a bad input, contradicting Lemma 3.9.

□

3.2.3 Cost

In our charging scheme, cost is a function that assigns charges, that originate from the edges, to the nodes. More specifically, the cost of a man a is obtained by counting the edges in G' incident to a, where bad outputs contribute 2 and all other edges contribute 1. Similarly, the cost of a woman b is obtained by counting the edges in G' incident to b, to which good inputs contribute 0 and all other edges contribute 1.

In the following, let $\deg(u)$ be the degree of the node u in G'. For $a \in A$, we define his *cost* as follows:

$$\mathsf{cost}(a) \coloneqq \deg(a) + k\,, \quad \text{where } k \text{ is the number of bad outputs from } a;$$

for $b \in B$, we define her cost as follows:

$$\mathsf{cost}(b) \coloneqq \deg(b) - k\,, \quad \text{where } k \text{ is the number of good inputs to } b,$$

For a node set $S \subseteq A \cup B$, $\mathsf{cost}(S)$ is defined as the sum of costs of all the nodes in S.

The above definitions lead to next three remarks.

Remark 3.11. *Let $b \in B$ be matched in M and have at least k bad inputs. Then $\mathsf{cost}(b) \geq k + 1$.*

Proof. Let k' be the number of good inputs to b. Since b is matched in M, the edge $(M(b), b)$ is contained in G' and therefore it is not an input to b. Thus $\deg(b) \geq k + k' + 1$. Hence, by definition of cost, $\mathsf{cost}(b) = \deg(b) - k' \geq k + 1$ holds.

□

Remark 3.12. *Let $b \in B$ be matched in OPT, have at least k bad inputs, and $(\mathsf{OPT}(b), b) \in E'$ where E' is the edge set of G'. Then $\mathsf{cost}(b) \geq k + 1$.*

Proof. Let k' be the number of good inputs to b. Since the edge $(\mathsf{OPT}(b), b)$ is in G', it is not an input to b. Thus $\deg(b) \geq k+k'+1$. So, by definition of cost, $\mathsf{cost}(b) = \deg(b)-k' \geq k+1$ holds.

□

Remark 3.13. *Let $b \in B$ be matched in both OPT and M, $\mathsf{OPT}(b) \neq M(b)$, and $(\mathsf{OPT}(b), b) \in E'$ where E' is the edge set of G'. Then $\mathsf{cost}(b) \geq 2$.*

Proof. Let k and k' be the numbers of bad inputs and good inputs to b, respectively. Since the edges $(\mathsf{OPT}(b), b)$ and $(M(b), b)$ are contained in G', they are not inputs to b. Thus $\deg(b) \geq k + k' + 2$. So, by definition of cost, $\mathsf{cost}(b) = \deg(b) - k' \geq k + 2 \geq 2$ holds.

□

3.2.4 The Approximation Ratio

Let $\mathcal{C}(M+\mathsf{OPT})$ denote the set of connected components in a graph induced by the edge set $M + \mathsf{OPT}$. Lemma 3.14 below bounds the cost of $M + \mathsf{OPT}$ and is proven in Section 3.2.5.

Lemma 3.14. $\sum_{C \in \mathcal{C}(M+\mathsf{OPT})} \mathsf{cost}(C) \geq (L + 1)|\mathsf{OPT}| + (L - 2)(|\mathsf{OPT}| - |M|)$.

We are ready to prove our main theorem, and restate it here for completeness.

Theorem 1.2. *Given an instance of the maximum stable matching problem with incomplete preferences and ties of length at most L; the polynomial-time algorithm described in Section 3.1 finds a stable matching M with*

$$|M| \geq \frac{2L-1}{3L-2}|\mathit{OPT}|,$$

where OPT is an optimal stable matching.

Proof. By Lemma 3.1, we have

$$|M| \geq \frac{|E'|}{L} = \sum_{u \in A \cup B} \frac{\deg(u)}{2L}.$$

By definition of cost and by Corollary 3.10, we obtain

$$\sum_{u \in A \cup B} \deg(u) \geq \mathsf{cost}(A \cup B).$$

Combining the above inequalities, we get

$$2L|M| \geq \sum_{u \in A \cup B} \deg(u) \geq \mathsf{cost}(A \cup B) = \sum_{C \in \mathcal{C}(M+\mathsf{OPT})} \mathsf{cost}(C),$$

By Lemma 3.14, we obtain

$$2L|M| \geq \sum_{C \in \mathcal{C}(M+\mathsf{OPT})} \mathsf{cost}(C) \geq (L+1)|\mathsf{OPT}| + (L-2)(|\mathsf{OPT}| - |M|).$$

By rearranging the terms, we obtain

$$2L|M| + (L-2)|M| \geq (L+1)|\mathsf{OPT}| + (L-2)|\mathsf{OPT}|,$$

and so we obtain the desired inequality

$$(3L-2)|M| \geq (2L-1)|\mathsf{OPT}|.$$

□

3.2.5 Costs of Connected Components in $M + \mathsf{OPT}$

The purpose of this subsection is to prove Lemma 3.14. We call a connected component of $M + \mathsf{OPT}$ *trivial* if it is an isolated node. A component in $M + \mathsf{OPT}$ is called *alternating path* if the sequence of its edges alternate being contained in M and in OPT. An alternating path is called *alternating cycle* if its endpoints are the same. We call an alternating path OPT-augmenting if the edges incident to its endpoints are in M. Likewise, we call an alternating path M-augmenting if the edges incident to its endpoints are in OPT. For ease of exposition, henceforth, we will refer by alternating paths only to the components that are not alternating cycles, OPT-augmenting or M-augmenting paths.

We begin by studying costs of connected components in $M+\mathsf{OPT}$. For each connected component, we find an appropriate lower bound. The costs of components that are alternating paths, alternating cycles or OPT-augmenting paths, can be bounded from below by $L+1$ multiplied by the number of edges that are both in OPT and in the associated component. However, the costs of M-augmenting paths can be bounded from below in a stronger way. While the costs for trivial paths, alternating paths, alternating cycles or OPT-augmenting paths can be obtained in a straightforward way, those for M-augmenting paths are central to our analysis and require a detailed study. After we establish the lower bounds on the costs of all connected components in $M + \mathsf{OPT}$, we start proving Lemma 3.14.

The following lemma bounds costs of edges in OPT from below. Recall that $\deg(u)$ is the degree of the node u in G'.

Lemma 3.15. *Let $a \in A$ and $b \in B$ be such that $(a, b) \in \mathsf{OPT}$. Then $\mathsf{cost}(\{a, b\}) \geq L$ holds. Furthermore, if $\deg(a) \geq 1$, then $\mathsf{cost}(\{a, b\}) \geq L + 1$; if $\deg(b) \leq L - 1$, then $\mathsf{cost}(\{a, b\}) \geq 2L - 1$.*

Proof. We consider $\deg(a)$ and $\deg(b)$ simultaneously. Since both are integers between 0 and L, the following cover all possible cases for values of $\deg(a)$ and $\deg(b)$:

1. $\deg(a) = 0$ and $\deg(b) = L$.
2. $1 \leq \deg(a) \leq L - 1$ and $\deg(b) = L$.
3. $\deg(a) = L$ and $\deg(b) = L$.
4. $\deg(a) = L$ and $\deg(b) \leq L - 1$.

5. $\deg(a) \leq L-1$ and $\deg(b) \leq L-1$.

In cases (1) and (2), a is unsuccessful. Since (a,b) is an edge in G, b rejected a proposal from a, and so b is popular. Thus there are L separate edges $(a^1,b),(a^2,b),\ldots,(a^L,b)$ in G' such that $a \leq_b a^i$ for all $i=1,2,\ldots,L$. Moreover, none of the edges (a^1,b), (a^2,b), $\ldots$, (a^L,b) is a good input because b is popular, $\mathsf{OPT}(b) \leq_b a^i$ for all $i=1,2,\ldots,L$, and $\mathsf{OPT}(b)$ is unsuccessful. Thus $\mathsf{cost}(b)=L$.

Hence, for case (1),

$$\mathsf{cost}(\{a,b\}) \geq \mathsf{cost}(b) = L\,;$$

for case (2)

$$\mathsf{cost}(\{a,b\}) = \underbrace{\mathsf{cost}(a)}_{\geq \deg(a) \geq 1} + \underbrace{\mathsf{cost}(b)}_{=L} \geq L+1\,,$$

as required.

In case (3), b is matched in M since $\deg(b)=L$. Thus, by Remark 3.11, $\mathsf{cost}(b) \geq 1$ holds. Hence

$$\mathsf{cost}(\{a,b\}) = \underbrace{\mathsf{cost}(a)}_{\geq \deg(a) = L} + \underbrace{\mathsf{cost}(b)}_{\geq 1} \geq L+1\,,$$

as desired.

In case (4), b is unsuccessful, and so b did not reject any proposal from a. Thus, a is basic and for every edge $(a,b') \in G'$ with $b' \in B$, and so $b' \geq_a b$. Thus, by Remark 3.5 and Definition 3.8, each edge $(a,b') \in G'$ with $b' \in B$ is a bad output from a.

Since $\deg(a)=L$, a is matched in M. Thus if there is no edge (a,b) in G', then a has $L-1$ bad outputs implying the desired inequality

$$\mathsf{cost}(\{a,b\}) \geq \mathsf{cost}(a) = \deg(a) + L - 1 = 2L-1\,.$$

But if there is an edge (a,b) in G', then a has $L-2$ bad outputs. Also $\mathsf{cost}(b) \geq 1$ by Remark 3.12. Thus

$$\mathsf{cost}(\{a,b\}) = \underbrace{\mathsf{cost}(a)}_{\geq 2L-2} + \underbrace{\mathsf{cost}(b)}_{\geq 1} \geq 2L-2+1 = 2L-1\,,$$

as needed.

In case (5), both a and b are unsuccessful. Since (a, b) is an edge in G and a is unsuccessful, a proposal from a was rejected by b at some time during the algorithm. On the other hand, since b is unsuccessful, she did not reject any proposal during the algorithm, a contradiction.

□

For completeness, we state the following remark that is trivially true.

Remark 3.16. *Trivial components have cost at least* $(L + 1)|OPT \cap C|$.

Alternating Paths, Alternating Cycles and OPT-Augmenting Paths

Recall that despite the original definition of alternating paths, we merely mean by them the components that are not alternating cycles, OPT-augmenting or M-augmenting paths. The following corollary of Lemma 3.15 provides lower bounds on the costs of alternating paths, alternating cycles and OPT-augmenting paths.

Corollary 3.17. *Let C be a connected component of $M + \mathsf{OPT}$ such that it is an alternating path, alternating cycle, or OPT-augmenting path. Then* $\mathsf{cost}(C) \geq (L + 1)|\mathsf{OPT} \cap C|$.

Proof. First, we note that since the length of an alternating path is even, the endpoints of it are either both men or both women as in (3) and (4) below. In contrast, the length of an OPT-augmenting path is odd, and so its endpoints are a man and a woman as in (1) below. Last, alternating cycles have the general form as in (2) below, but it can be represented by various ways simply by shifting the nodes to the right or to the left. Assuming C is as stated above, one of the following is true:

1. C is an OPT-augmenting path of the form $a_0 - b_1 - \ldots - a_k - b_{k+1}$.
2. C is an alternating cycle of the form $a_1 - b_1 - \ldots - a_k - b_k - a_1$, where $(a_1, b_1) \in \mathsf{OPT}$.
3. C is an alternating path of the form $b_1 - a_1 - \ldots - b_k - a_k - b_{k+1}$, where $a_1 \in A$ and $(a_1, b_1) \in \mathsf{OPT}$.
4. C is an alternating path of the form $a_1 - b_1 - \ldots - a_k - b_k - a_{k+1}$, where $a_1 \in A$ and $(a_1, b_1) \in \mathsf{OPT}$.

For cases (1), (2) and (3), Lemma 3.15 implies that $\mathsf{cost}(\{a_i, b_i\}) \geq L+1$ for every $i = 1, \ldots, k$. Thus

$$\mathsf{cost}(C) \geq \sum_{i=1}^{k} \underbrace{\mathsf{cost}(\{a_i, b_i\})}_{\geq L+1} \geq (L+1)k = (L+1)|\mathsf{OPT} \cap C|\,,$$

as required.

For case (4), Lemma 3.15 implies that $\mathsf{cost}(\{a_i, b_i\}) \geq L+1$ for every $i = 2, \ldots, k$ and $\mathsf{cost}(\{a_1, b_1\}) \geq L$. Since a_{k+1} is matched in M, $\mathsf{cost}(a_{k+1}) \geq 1$ holds. Thus

$$\mathsf{cost}(C) \geq \underbrace{\mathsf{cost}(\{a_1, b_1\})}_{\geq L} + \underbrace{\mathsf{cost}(a_{k+1})}_{\geq 1} + \sum_{i=2}^{k} \underbrace{\mathsf{cost}(\{a_i, b_i\})}_{\geq L+1} \geq (L+1)k = (L+1)|\mathsf{OPT} \cap C|\,.$$

□

M-Augmenting Paths

In this section, we provide a lower bound on the cost of components in $M + \mathsf{OPT}$, that are M-augmenting paths of length at least 5. We call an edge in an M-augmenting path *terminal* if it is incident to either endpoint of the path, and *internal* otherwise. We start by showing that there are no M-augmenting paths in $M + \mathsf{OPT}$ of length 1 or 3.

Lemma 3.18. *There is no M-augmenting path in M + OPT, that is of length 1 or of length 3.*

Proof. First, suppose that there is an M-augmenting path in $M + \mathsf{OPT}$, that is of length 1. That is to say, there exists an edge (a, b) in OPT such that neither a nor b is matched in M. Since (a, b) is in G and none of a and b is matched in M, (a, b) is a blocking pair for M, that contradicts Lemma 3.2.

Second, suppose that there is an M-augmenting path in $M + \mathsf{OPT}$, that is of length 3 and of form $a_0 - b_0 - a_1 - b_1$ where $a_0 \in A$ (see Figure 3.1). Since a_0 and b_1 are unmatched in M, $\deg(a_0) < L$ and $\deg(b_1) < L$ hold, and hence both a_0 and b_1 are unsuccessful. Since a_0 is unsuccessful, he is 2-promoted and was rejected by every woman in his preference list as a 2-promoted man. Since b_0 is such a woman, she is popular. Also, we notice that (a_1, b_0) is in M, and hence b_0 holds a proposal from a_1 when the algorithm terminates. Thus $a_1 \geq_{b_0} a_0$.

Observe that (a_1, b_1) is in OPT, and hence (a_1, b_1) is in G. Since b_1 is unsuccessful, b_1 did not reject any proposal during the algorithm. Since no proposal from a_1 was rejected by b_1, he is basic. Also, $b_0 \geq_{a_1} b_1$ holds since b_0 holds a proposal from a_1 when the algorithm finishes and no proposal from a_1 was rejected by b_1. Thus $a_1 \geq_{b_0} a_0$, $b_0 \geq_{a_1} b_1$, a_0 is 2-promoted, b_0 is popular, a_1 is basic and b_1 is unsuccessful.

First, $a_1 \simeq_{b_0} a_0$ cannot hold because a_1 is basic, and b_0 rejected a_0 as a 2-promoted man, whereas b_0 holds a proposal from a_1 when the algorithm ends. Second, $b_0 \simeq_{a_1} b_1$ cannot hold, otherwise we get a contradiction to Remark 3.5 since b_0 is popular, b_1 is unsuccessful, and b_0 holds a proposal from a_1 when the algorithm terminates. Hence we conclude that $a_1 >_{b_0} a_0$ and $b_0 >_{a_1} b_1$ hold. Since $(a_0, b_0) \in \mathsf{OPT}$ and $(a_1, b_1) \in \mathsf{OPT}$, (a_1, b_0) is a blocking pair for OPT, contradicting the stability of OPT.

□

Figure 3.1: Illustrations of M-augmenting paths in $M + \mathsf{OPT}$ of length 3 on the left and of length 5 on the right. Dashed lines represent the edges in OPT and solid lines represent those in M.

Now, we consider M-augmenting paths in $M + \mathsf{OPT}$, that are of lengths at least 5 (see Figure 3.1). Since the length of an M-augmenting path is odd, its endpoints are a man and a woman. Note that, our next results assume the representation, where, without loss of generality, the leftmost node is a man. In the following definition, a woman in an M-augmenting path points right is the compact way to say that the woman weakly prefers the man on her right to the man on her left, where the weakly preferred man is promoted if she is indifferent between them.

Definition 3.19. *Let $a_0 - b_0 - a_1 - \ldots - a_k - b_k$ be an M-augmenting path of length at least 5, where $a_0 \in A$. For $i = 0, \ldots, k-1$, we say that b_i* points right *if one of the following is true:*

- $a_{i+1} >_{b_i} a_i$.
- $a_{i+1} \simeq_{b_i} a_i$, *and* a_{i+1} *is not basic.*

The desired lower bound on the cost of M-augmenting paths in $M + \mathsf{OPT}$ is demonstrated by partitioning an M-augmenting path into the pieces of the first terminal edge, internal edges, and the last terminal edge, and providing a lower bound on the cost of each piece.

Remarks 3.20 and 3.21 below provide bounds on the costs of the terminal edges of an M-augmenting path in $M + \mathsf{OPT}$.

Remark 3.20. *Let $a_0 - b_0 - a_1 - \ldots - a_k - b_k$ be an M-augmenting path in $M + \mathsf{OPT}$ of length $2k+1$, $k \geq 2$, where $a_0 \in A$. Then $\mathsf{cost}(\{a_0, b_0\}) \geq L$. Moreover, b_0 rejected a proposal from a_0 at some point, and b_0 points right.*

Proof. First, since $(a_0, b_0) \in \mathsf{OPT}$, Lemma 3.15 implies $\mathsf{cost}(\{a_0, b_0\}) \geq L$. Second, observe that a_0 is not matched in M, and hence a_0 is unsuccessful. Thus b_0 rejected a_0 as a 2-promoted man. On the other hand, since b_0 has a proposal from a_1 when the algorithm finishes, we deduce that $a_1 \geq_{b_0} a_0$ holds. Notice that if $a_1 \simeq_{b_0} a_0$ holds, then a_1 is not basic. Thus b_0 points right, that finishes the proof.

□

Remark 3.21. *Let $a_0 - b_0 - a_1 - \ldots - a_k - b_k$ be an M-augmenting path in $M + \mathsf{OPT}$ of length $2k+1$, $k \geq 2$, where $a_0 \in A$. Then $\mathsf{cost}(\{a_k, b_k\}) \geq 2L - 1$.*

Proof. Observe that b_k is not matched in M, and hence $\deg(b_k) \leq L - 1$ holds. Since $(a_k, b_k) \in \mathsf{OPT}$ and $\deg(b_k) \leq L - 1$, Lemma 3.15 implies the desired inequality that $\mathsf{cost}(\{a_k, b_k\}) \geq 2L - 1$.

□

Lemma 3.22 below is important for a better understanding of the internal edges in M-augmenting paths and can be considered as rather a technical result followed by a corollary that is of an essential use. The proof of Lemma 3.22 is presented after we establish the key result of this section in Lemma 3.25 and prove Lemma 3.14.

Lemma 3.22. *Let $a_0 - b_0 - a_1 - \ldots - a_k - b_k$ be an M-augmenting path in $M + \mathsf{OPT}$ of length $2k+1$, $k \geq 2$, where $a_0 \in A$. Then for every $i = 1, \ldots, k-1$, at least one of the following is true:*

1. *$\mathsf{cost}(\{a_i, b_i\}) \geq L + 2$.*
2. *b_i rejected a proposal from a_i at some point, and b_i points right.*
3. *a_i is basic and $b_{i-1} >_{a_i} b_i$.*

For an M-augmenting path in $M + \mathsf{OPT}$ of length $2k+1$, $k \geq 2$, Lemma 3.15 implies that each internal edge that is both in the same path and in OPT has cost at least $L+1$. The following corollary of Lemma 3.22 establishes an essential fact when the cost of such an internal edge is exactly $L+1$.

Corollary 3.23. *Let $a_0 - b_0 - a_1 - \ldots - a_k - b_k$ be an M-augmenting path in $M + \mathsf{OPT}$ of length $2k+1$, $k \geq 2$, where $a_0 \in A$. For every $i = 1, \ldots, k-1$ such that $\mathsf{cost}(\{a_i, b_i\}) = L+1$, if b_{i-1} rejected a proposal from a_{i-1} at some point and b_{i-1} points right, then b_i rejected a proposal from a_i at some point and b_i points right.*

Proof. By Lemma 3.22, for every $i = 1, \ldots, k-1$, at least one of the following is true:

1. $\mathsf{cost}(\{a_i, b_i\}) \geq L + 2$.
2. b_i rejected a proposal from a_i at some point, and b_i points right.
3. a_i is basic and $b_{i-1} >_{a_i} b_i$.

In case (1), that is an immediate contradiction to $\mathsf{cost}(\{a_i, b_i\}) = L + 1$.

In case (3), a_i is basic. Thus if b_{i-1} points right as stated, then $a_{i-1} <_{b_{i-1}} a_i$. Hence $a_{i-1} <_{b_{i-1}} a_i$ and $b_{i-1} >_{a_i} b_i$, showing that (a_i, b_{i-1}) is a blocking pair for OPT, a contradiction to the stability of OPT.

In case (2), we obtain the desired statement.

□

Lemma 3.24 below provides a bound on the cost of the rightmost internal edge of an M-augmenting path in $M+\mathsf{OPT}$ given the fact that is established by Corollary 3.23 occurs. The proof of Lemma 3.24 is presented after the proof of Lemma 3.22.

Lemma 3.24. *Let $a_0 - b_0 - a_1 - \ldots - a_k - b_k$ be an M-augmenting path in $M + \mathsf{OPT}$ of length $2k+1$, $k \geq 2$, where $a_0 \in A$. If b_{k-1} rejected a proposal from a_{k-1}, and b_{k-1} points right, then $\mathsf{cost}(\{a_{k-1}, b_{k-1}\}) \geq L+2$.*

Now, we have all the tools to bound the cost of M-augmenting paths of length at least 5.

Lemma 3.25. *Let C be a connected component of $M + \mathsf{OPT}$, that is an M-augmenting path of length at least 5. Then $\mathsf{cost}(C) \geq (L+1)|\mathsf{OPT} \cap C| + (L-2)$.*

Proof. Let C be an M-augmenting path in $M + \mathsf{OPT}$ of length $2k+1, k \geq 2$. Recall our assumption that, without loss of generality, C is of the form $a_0 - b_0 - a_1 - \ldots - a_k - b_k$, where $a_0 \in A$. Then

$$\begin{aligned}\mathsf{cost}(C) = \underbrace{\mathsf{cost}(\{a_0, b_0\})}_{\geq L \text{ by Remark 3.20}} + \sum_{i=1}^{k-1} \underbrace{\mathsf{cost}(\{a_i, b_i\})}_{\geq L+1 \text{ by Lemma 3.15}} + \underbrace{\mathsf{cost}(\{a_k, b_k\})}_{\geq 2L-1 \text{ by Remark 3.21}} \geq \\ L + (L+1)(k-1) + 2L - 1 = \\ (L+1)(k-1) + 2(L+1) + (L-3) = \\ (L+1)(k+1) + (L-3) = \\ (L+1)|OPT \cap C| + (L-3).\end{aligned}$$

By Remark 3.20, b_0 rejected a proposal from a_0 at some point, and b_0 points right. Suppose now that the above inequality is tight only. Then Corollary 3.23 implies that, for all $i = 0, \ldots, k-1$, b_i rejected a proposal from a_i, and b_i points right. But then, Lemma 3.24 implies that $\mathsf{cost}(\{a_{k-1}, b_{k-1}\}) \geq L+2$ holds, contradicting that the above inequality is tight. Thus we get the desired inequality $\mathsf{cost}(C) \geq (L+1)|\mathsf{OPT} \cap C| + (L-2)$.

□

Proof of Lemma 3.14. By Corollary 3.17 and Remark 3.16, for every connected component C in $M+\mathsf{OPT}$ that is not an M-augmenting path, $\mathsf{cost}(C) \geq (L+1)|\mathsf{OPT} \cap C|$ holds. Also, by Lemma 3.25, for each connected component C in $M + \mathsf{OPT}$ that is an M-augmenting path of length at least 5, $\mathsf{cost}(C) \geq (L+1)|\mathsf{OPT} \cap C| + (L-2)$ holds. Since there are at least $|\mathsf{OPT}| - |M|$ M-augmenting paths in $M + \mathsf{OPT}$, we obtain the desired inequality $\sum_{C \in \mathcal{C}(M+\mathsf{OPT})} \mathsf{cost}(C) \geq (L+1)|\mathsf{OPT}| + (L-2)(|\mathsf{OPT}| - |M|)$.

□

Proof of Lemma 3.22. Clearly, (a_i, b_{i-1}) and (a_{i+1}, b_i) are contained in G' since they are in M. Thus $\deg(a_i) \geq 1$ and $\deg(b_i) \geq 1$. Moreover, since (a_i, b_i) is included in G, at least one of the following is true: $\deg(a_i) = L$; and $\deg(b_i) = L$. Hence it is sufficient to consider the following cases:

I. $\deg(a_i) < L$ and $\deg(b_i) = L$.

II. $\deg(a_i) = L$.

- II.I. b_i rejected a proposal from a_i.
  - II.I.I. b_i has at most $L-2$ good inputs.
  - II.I.II. b_i has $L-1$ good inputs.
- II.II. b_i did not reject any proposal from a_i.
  - II.II.I. there is an edge (a_i, b_i) in G'.
  - II.II.II. there is not an edge (a_i, b_i) in G'.
    - II.II.II.I. a_i has at least one bad output.
    - II.II.II.II. a_i has $L-1$ good outputs.

In case (I.), a_i is unsuccessful. Thus b_i rejected a proposal from a_i as a 2-promoted man. Also, b_i has a proposal from a_{i+1} when the algorithm finishes, implying (2).

In case (II.I.I.), $\deg(b_i) = L$ holds since b_i rejected a proposal from a_i at some point during the algorithm. Since b_i has at most $L-2$ good inputs, $\mathsf{cost}(b_i) \geq \deg(b_i) - (L-2) = 2$ holds. Thus

$$\mathsf{cost}(\{a_i, b_i\}) = \underbrace{\mathsf{cost}(a_i)}_{\geq \deg(a_i) = L} + \underbrace{\mathsf{cost}(b_i)}_{\geq 2} \geq L + 2\,,$$

implying (1).

In case (II.I.II.), b_i is popular since she rejected a proposal at some point. Let (a^j, b_i) for all $j = 1, \dots, L-1$ be good inputs to b_i. Then, by definition of good inputs, $a_i \geq_{b_i} a^j$ for all $j = 1, \dots, L-1$. Also, $a^j \neq a_i$ for all $j = 1, \dots, L-1$ because $(a_i, b_i) \in \mathsf{OPT}$, $(a_{i+1}, b_i) \in M$ and (a^j, b_i) for all $j = 1, \dots, L-1$ are good inputs. Since b_i rejected a proposal from a_i at some point while she has proposals from a_{i+1} and a^j for $j = 1, \dots, L-1$ when the algorithm ends, we deduce that $a_i \leq_{b_i} a_{i+1}$ and $a_i \leq_{b_i} a^j$ for all $j = 1, \dots, L-1$. Since $a_i \geq_{b_i} a^j$ and $a_i \leq_{b_i} a^j$, we conclude that $a_i \simeq_{b_i} a^j$ for all $j = 1, \dots, L-1$.

Since $a_i \leq_{b_i} a_{i+1}$, $a_i \simeq_{b_i} a^j$, $a^j \neq a_i$ for all $j = 1, \dots, L-1$, and ties are of length at most L, at least one of the following is true:

i. $a_i <_{b_i} a_{i+1}$.

ii. $a_i \simeq_{b_i} a_{i+1}$.

  ii.i. there exist $j', j'' = 1, \dots, L-1$, $j' \neq j''$ such that $a^{j'} = a^{j''}$.

  ii.ii. there exists $j' = 1, \dots, L-1$ such that $a^{j'} = a_{i+1}$.

In case (i.), we immediately get (2).

In case (ii.i.), by definition of good inputs, $a^{j'}$ is either basic or 2-promoted. If $a^{j'}$ is basic, that is in contradiction to the rejection step since $a_i \simeq_{b_i} a^{j'}$, b_i rejected a proposal from a_i at some point, b_i holds no proposal from a_i while she holds two proposals from $a^{j'}$ when the algorithm terminates. If $a^{j'}$ is 2-promoted, then a_{i+1} is not basic because $a_{i+1} \simeq_{b_i} a^{j'}$, b_i rejected $a^{j'}$ as a 1-promoted man while she holds a proposal from a_{i+1} when the algorithm ends. Thus we conclude (2).

In case (ii.ii.), if a_{i+1} is basic, that is in contradiction to the rejection step because $a_i \simeq_{b_i} a_{i+1}$, b_i rejected a proposal from a_i at some point, b_i holds no proposal from a_i while she holds two proposals from a_{i+1} when the algorithm finishes. Thus a_{i+1} cannot be basic, implying that b_i points right. Hence we deduce (2).

In case (II.II.I.), $\mathsf{cost}(a_i) \geq L$ holds. Also, by Remark 3.13, $\mathsf{cost}(b_i) \geq 2$ holds since $(a_i, b_i) \in \mathsf{OPT}$, $(a_i, b_i) \in G'$, and $(a_{i+1}, b_i) \in M$. Thus $\mathsf{cost}(\{a_i, b_i\}) = \mathsf{cost}(a_i) + \mathsf{cost}(b_i) \geq L + 2$, implying (1).

In case (II.II.II.I.), since $(a_{i+1}, b_i) \in M$, $\mathsf{cost}(b_i) \geq 1$ holds. Also, because a_i has at least one bad output, $\mathsf{cost}(a_i) \geq \deg(a_i) + 1 = L + 1$ holds. Thus $\mathsf{cost}(\{a_i, b_i\}) = \mathsf{cost}(a_i) + \mathsf{cost}(b_i) \geq L + 2$, implying (1).

In case (II.II.II.II.), let (a_i, b^j) for $j = 1, \ldots, L-1$ be good outputs from a_i. Since b_i did not reject any proposal from a_i during the algorithm, $(a_i, b_{i-1}) \in M$, and (a_i, b^j) for $j = 1, \ldots, L-1$ are outputs, we deduce that a_i is basic, $b_i \leq_{a_i} b_{i-1}$, $b_i \leq_{a_i} b^j$ for all $j = 1, \ldots, L-1$. Since a_i is basic and (a_i, b^j) for all $j = 1, \ldots, L-1$ are good outputs from a_i, we deduce that, by definition of good outputs, $b_i \neq b^j$, $b_i \geq_{a_i} b^j$ for all $j = 1, \ldots, L-1$, and hence $b_i \simeq_{a_i} b^j$ for all $j = 1, \ldots, L-1$.

Because $b_i \leq_{a_i} b_{i-1}$, $b_i \simeq_{a_i} b^j$, $b_i \neq b^j$ for all $j = 1, \ldots, L-1$, and ties are of length at most L, at least one of the following is true:

i. $b_i <_{a_i} b_{i-1}$.

ii. $b_i \simeq_{a_i} b_{i-1}$.

  ii.i. there exist $j', j'' = 1, \ldots, L-1$, $j' \neq j''$ such that $b^{j'} = b^{j''}$.

  ii.ii. there exists $j' = 1, \ldots, L-1$ such that $b^{j'} = b_{i-1}$.

In case (i.), we immediately get (3).

In cases (ii.i.) and (ii.ii.), by definition of good outputs, $b^{j'}$ is popular and so she rejected a proposal at some point. On the other hand, $b_i \simeq_{a_i} b^{j'}$, $b^{j'}$ holds at least two proposals from a_i when the algorithm finishes, b_i did not reject a_i during the algorithm, and b_i does not hold any proposal from a_i when the algorithm terminates, a contradiction to the forward step for $b^{j'}$.

□

The following remark is used to simplify the proof of Lemma 3.24 below.

Remark 3.26. *Let $a_0 - b_0 - a_1 - \ldots - a_k - b_k$ be an M-augmenting path in $M + \mathsf{OPT}$ of length $2k+1$, $k \geq 2$, where $a_0 \in A$. Then $\deg(a_{k-1}) = L$.*

Proof. Suppose for a contradiction that $\deg(a_{k-1}) < L$, and so a_{k-1} is unsuccessful. Thus b_{k-1} rejected a_{k-1} as a 2-promoted man, and so b_{k-1} is popular. Since b_k is unmatched in M, $\deg(b_k) < L$ holds. So b_k is unsuccessful, and thus a_k is basic. Since b_{k-1} rejected a proposal from a_{k-1} at some point, and b_{k-1} has a proposal from a_k when the algorithm ends, we deduce that $a_{k-1} \leq_{b_{k-1}} a_k$.

First, if $a_{k-1} \simeq_{b_{k-1}} a_k$ holds, we deduce that a_k is 2-promoted, contradicting the fact that a_k is basic. Thus $a_{k-1} <_{b_{k-1}} a_k$. Since b_k did not reject any proposal from a_k during the algorithm, and b_{k-1} holds a proposal from a_k when the algorithm terminates, $b_{k-1} \geq_{a_k} b_k$ holds. If $b_{k-1} >_{a_k} b_k$ holds, then (a_k, b_{k-1}) is a blocking pair for OPT, contradicting the stability of OPT. We conclude that $b_{k-1} \simeq_{a_k} b_k$. But then, since b_{k-1} has a proposal from a_k when the algorithm finishes, b_k is unsuccessful, and $b_{k-1} \simeq_{a_k} b_k$, Remark 3.5 implies that b_{k-1} is unpopular, a contradiction.

□

Proof of Lemma 3.24. Observe that $(a_k, b_{k-1}) \in M$, and thus $(a_k, b_{k-1}) \in G'$. Therefore, one of the following cases is true:

I. there is at least one edge (a_{k-1}, b_{k-1}) in G'.

II. there is no edge (a_{k-1}, b_{k-1}) in G'.

  II.I. there are at least two parallel edges (a_k, b_{k-1}) in G'.

  II.II. there are exactly $L-1$ edges, (a^j, b_{k-1}) for $j = 1, \ldots, L-1$ in G', and $a^j \neq a_{k-1}$, $a^j \neq a_k$ for $j = 1, \ldots, L-1$.

In case (I.), Remark 3.26 implies $\mathsf{cost}(a_{k-1}) \geq \deg(a_{k-1}) = L$, and Remark 3.13 implies $\mathsf{cost}(b_{k-1}) \geq 2$ since $(a_{k-1}, b_{k-1}) \in \mathsf{OPT}$, $(a_{k-1}, b_{k-1}) \in G'$, and $(a_k, b_{k-1}) \in M$. Thus,

$$\mathsf{cost}(\{a_{k-1}, b_{k-1}\}) = \underbrace{\mathsf{cost}(a_{k-1})}_{\geq L} + \underbrace{\mathsf{cost}(b_{k-1})}_{\geq 2} \geq L + 2\,,$$

as desired.

In case (II.I.), since b_{k-1} rejected a proposal from a_{k-1} at some point, and b_{k-1} holds at least two proposals from a_k when the algorithm terminates, we deduce that b_{k-1} is popular and $a_k \geq_{b_{k-1}} a_{k-1}$ holds. Also, since b_k is unmatched in M, $\deg(b_k) < L$ holds. Thus b_k is unsuccessful and a_k is basic. If $a_k \simeq_{b_{k-1}} a_{k-1}$, then Remark 3.6 implies that there is an edge (a_{k-1}, b_{k-1}) in G', a contradiction. Thus $a_k >_{b_{k-1}} a_{k-1}$ holds.

We show that $a_k >_{b_{k-1}} a_{k-1}$ leads to a contradiction. Since b_k is unsuccessful and there is an edge (a_k, b_{k-1}) in G', $b_{k-1} \geq_{a_k} b_k$ holds. If $b_{k-1} >_{a_k} b_k$, then (a_k, b_{k-1}) is a blocking pair for OPT, contradicting the stability of OPT. Thus $b_{k-1} \simeq_{a_k} b_k$ holds. But then, since

b_{k-1} holds a proposal from a_k when the algorithm ends, b_k is unsuccessful, and $b_{k-1} \simeq_{a_k} b_k$, Remark 3.5 implies that b_{k-1} is unpopular, a contradiction.

In case (II.II.), since b_{k-1} holds proposals from a_k and a^j for all $j = 1, \ldots, L-1$, and b_{k-1} rejected a proposal from a_{k-1} at some point, we deduce that b_{k-1} is popular, $a_k \geq_{b_{k-1}} a_{k-1}$, $a^j \geq_{b_{k-1}} a_{k-1}$ for all $j = 1, \ldots, L-1$. Since b_k is unmatched in M, $\deg(b_k) < L$ holds and therefore b_k is unsuccessful.

Analogously to the proof of case (II.I.), it can be shown that $a_k >_{b_{k-1}} a_{k-1}$ leads to a contradiction. Thus $a_k \simeq_{b_{k-1}} a_{k-1}$, and $a^j \geq_{b_{k-1}} a_{k-1}$ for all $j = 1, \ldots, L-1$ hold. Since $a^j \neq a_{k-1}$, $a^j \neq a_k$ for all $j = 1, \ldots, L-1$, and ties are of length at most L, at least one of the following is true:

i. there exists $j' = 1, \ldots, L-1$ such that $a^{j'} >_{b_{k-1}} a_{k-1}$.

ii. $a^j \simeq_{b_{k-1}} a_{k-1}$ for all $j = 1, \ldots, L-1$.

ii.i. there exist $j', j'' = 1, \ldots, L-1$, $j' \neq j''$ such that $a^{j'} = a^{j''}$.

In case (i.), $(a^{j'}, b_{k-1})$ is a bad input to b_{k-1} by definition. Thus, by Remark 3.11, $\mathsf{cost}(b_{k-1}) \geq 2$ holds. Since $\mathsf{cost}(a_{k-1}) \geq \deg(a_{k-1}) = L$ holds by Remark 3.26, we obtain the desired inequality

$$\mathsf{cost}(\{a_{k-1}, b_{k-1}\}) = \underbrace{\mathsf{cost}(a_{k-1})}_{\geq L} + \underbrace{\mathsf{cost}(b_{k-1})}_{\geq 2} \geq L + 2\,.$$

In case (ii.i.), recall that there is no edge (a_{k-1}, b_{k-1}) in G'. Since b_{k-1} rejected a proposal from a_{k-1} during the algorithm, $a^{j'} \simeq_{b_{k-1}} a_{k-1}$, and $(a_{k-1}, b_{k-1}) \notin G'$, we deduce from Remark 3.6 that $a^{j'}$ is not basic. If $a^{j'}$ is 1-promoted, then $(a^{j'}, b_{k-1})$ and $(a^{j''}, b_{k-1})$ are bad inputs to b_{k-1} by definition. Thus, by Remark 3.11, $\mathsf{cost}(b_{k-1}) \geq 3$ holds. Since $\mathsf{cost}(a_{k-1}) \geq \deg(a_{k-1}) = L$ holds by Remark 3.26, we get the desired inequality

$$\mathsf{cost}(\{a_{k-1}, b_{k-1}\}) = \underbrace{\mathsf{cost}(a_{k-1})}_{\geq L} + \underbrace{\mathsf{cost}(b_{k-1})}_{\geq 3} \geq L + 2\,.$$

If $a^{j'}$ is 2-promoted, then b_{k-1} rejected $a^{j'}$ as a 1-promoted man. On the other hand, $a_k \simeq_{b_{k-1}} a^{j'}$, a_k is basic, and b_{k-1} holds a proposal from a_k when the algorithm ends, a contradiction to the rejection step.

□

The following example shows that the bound in Theorem 1.2 is tight.

Example 3.27. *In Figure 3.2, the preference list of each individual is ordered from a most preferred person to a least preferred one, where individuals within parentheses are tied. For example, a_1^β is indifferent between all the women in his preference list except b_1^β, who is less preferred than the others.*

It is straightforward to check that there exists a unique maximum-cardinality stable matching, namely $\mathsf{OPT} = \{(a_0, b_0)\} \cup \{(a_i^j, b_i^j) \mid i = 1, \ldots, L-1,\ j = \alpha, \beta, \gamma\}$. We show that there exists an execution of the algorithm which outputs the matching $M = \{(a_0, b_0)\} \cup \{(a_i^\alpha, b_i^\gamma) \mid i = 1, \ldots, L-1\} \cup \{(a_i^\beta, b_i^\alpha) \mid i = 1, \ldots, L-1\}$, leading to the ratio $|\mathsf{OPT}|/|M| = (3L-2)/(2L-1)$.

Proof. The following is an execution of the algorithm which leads either to the matching M or a matching with the size of M.

- a_0 makes one proposal to every woman in his list; the women accept.
- a_i^α for all $i = 1, \ldots, L-1$ makes one proposal to every woman in his list; the women accept.
- a_i^β for all $i = 1, \ldots, L-1$ makes one proposal to every woman except the last one in his list; the women accept.
- a_i^γ starts to propose b_i^γ for all $i = 1, \ldots, L-1$, but each time a_i^γ makes a proposal, the proposal is rejected; a_i^γ gives up.

□

Figure 3.2: An instance with ties of length at most L, $L \geq 2$ for which the algorithm outputs a stable matching M with $|\mathsf{OPT}|/|M| = (3L-2)/(2L-1)$

Men's preferences	Women's preferences
$a_0 : (b_0 \ \ b_1^\gamma \ \ldots \ b_{L-1}^\gamma)$	$b_0 : (a_0 \ \ a_1^\beta \ \ldots \ a_{L-1}^\beta)$
$a_1^\alpha : (b_1^\alpha \ \ b_1^\gamma \ \ldots \ b_{L-1}^\gamma)$	$b_1^\alpha : a_1^\alpha \ \ a_1^\beta \ \ldots \ a_{L-1}^\beta$
$\vdots$	$\vdots$
$a_{L-1}^\alpha : (b_{L-1}^\alpha \ \ b_1^\gamma \ \ldots \ b_{L-1}^\gamma)$	$b_{L-1}^\alpha : a_{L-1}^\alpha \ \ a_1^\beta \ \ldots \ a_{L-1}^\beta$
$a_1^\beta : (b_0 \ \ b_1^\alpha \ \ldots \ b_{L-1}^\alpha) \ \ b_1^\beta$	$b_1^\beta : a_1^\beta$
$\vdots$	$\vdots$
$a_{L-1}^\beta : (b_0 \ \ b_1^\alpha \ \ldots \ b_{L-1}^\alpha) \ \ b_{L-1}^\beta$	$b_{L-1}^\beta : a_{L-1}^\beta$
$a_1^\gamma : b_1^\gamma$	$b_1^\gamma : (a_0 \ \ a_1^\alpha \ \ldots \ a_{L-1}^\alpha) \ \ a_1^\gamma$
$\vdots$	$\vdots$
$a_{L-1}^\gamma : b_{L-1}^\gamma$	$b_{L-1}^\gamma : (a_0 \ \ a_1^\alpha \ \ldots \ a_{L-1}^\alpha) \ \ a_{L-1}^\gamma$

Chapter 4

Concluding Remarks

In this Chapter we conclude this thesis by summarizing our contribution and briefly referring to a problem for future investigation. In Chapter 3, we present a polynomial-time algorithm for the maximum stable matching problem with two-sided ties that attains an approximation guarantee of $(3L-2)/(2L-1)$ where L is the maximum tie length. Our result matches the known lower bound on the integrality gap [21], indicating a potential obstacle to further improvements. When $L = 2$, our algorithm achieves an approximation ratio of 4/3 that matches the known UGC-hardness result [33]. For $L > 2$, we conjecture that the hardness result can be improved to $(3L-2)/(2L-1)$ so that it matches both our result and the lower bound on the integrality gap.

References

[1] http://www.nrmp.org (National Resident Matching Program website). Accessed 6 April 2020.

[2] http://www.carms.ca (Canadian Resident Matching Service website). Accessed 6 April 2020.

[3] Felix Bauckholt, Kanstantsin Pashkovich, and Laura Sanità. On the approximability of the stable marriage problem with one-sided ties. *arXiv e-prints*, 2018. arXiv:1805.05391.

[4] Claude Berge. Two theorems in graph theory. *Proceedings of the National Academy of Sciences*.

[5] Robert Chiang and Kanstantsin Pashkovich. On the approximability of the stable matching problem with ties of size two. *arXiv e-prints*, 2018. arXiv:1808.04510.

[6] William J Cook, William H Cunningham, William R Pulleyblank, and Alexander Schrijver. *Combinatorial Optimization*, chapter 5, pages 127–198. John Wiley & Sons, Ltd, 2011.

[7] Brian Dean and Rommel Jalasutram. Factor revealing LPs and stable matching with ties and incomplete lists. In *Proceedings of 3rd International Workshop on Matching Under Preferences*, pages 42–53, 2015.

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

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

[10] Dan Gusfield and Robert W Irving. *The Stable Marriage Problem: Structure and Algorithms*. MIT Press, 1989.

[11] Magnús M Halldórsson, Kazuo Iwama, Shuichi Miyazaki, and Hiroki Yanagisawa. Randomized approximation of the stable marriage problem. *Theoretical Computer Science*, 325(3):439–465, 2004.

[12] Magnús M Halldórsson, Kazuo Iwama, Shuichi Miyazaki, and Hiroki Yanagisawa. Improved approximation results for the stable marriage problem. *ACM Transactions on Algorithms*, 3(3):30, 2007.

[13] Magnús M Halldórsson, Robert W Irving, Kazuo Iwama, David F Manlove, Shuichi Miyazaki, Yasufumi Morita, and Sandy Scott. Approximability results for stable marriage problems with ties. *Theoretical Computer Science*, 306(1):431–447, 2003.

[14] Chien-Chung Huang and Telikepalli Kavitha. An improved approximation algorithm for the stable marriage problem with one-sided ties. In *Proceedings of International Conference on Integer Programming and Combinatorial Optimization (IPCO)*, pages 297–308, 2014.

[15] Chien-Chung Huang and Telikepalli Kavitha. Improved approximation algorithms for two variants of the stable marriage problem with ties. *Mathematical Programming*, 154(1):353–380, 2015.

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

[17] Robert W Irving and Paul Leather. The complexity of counting stable marriages. *SIAM J. Comput.*, 15(3):655–667, 1986.

[18] Robert W Irving and David F Manlove. Approximation algorithms for hard variants of the stable marriage and hospitals/residents problems. *J. Comb. Optim.*, 16:279–292, 2008.

[19] Kazuo Iwama, Shuichi Miyazaki, David Manlove, and Yasufumi Morita. Stable marriage with incomplete lists and ties. In *Proceedings of the 26th International Colloquium on Automata, Languages and Programming*, page 443–452, 1999.

[20] Kazuo Iwama, Shuichi Miyazaki, and Naoya Yamauchi. A 1.875-approximation algorithm for the stable marriage problem. In *Proceedings of the 18th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 288–287, 2007.

[21] Kazuo Iwama, Shuichi Miyazaki, and Hiroki Yanagisawa. A 25/17-approximation algorithm for the stable marriage problem with one-sided ties. *Algorithmica*, 68(3):758–775, 2014.

[22] Zoltán Király. Better and simpler approximation algorithms for the stable marriage problem. *Algorithmica*, 60(1):3–20, 2011.

[23] Zoltán Király. Linear time local approximation algorithm for maximum stable marriage. *Algorithms*, 6(3):471–484, 2013.

[24] Chi-Kit Lam. *Algorithms for Stable Matching with Indifferences*. PhD thesis, University of Texas at Austin, 2019.

[25] Chi-Kit Lam and C Gregory Plaxton. A $(1 + 1/e)$-approximation algorithm for maximum stable matching with one-sided ties and incomplete lists. In *Proceedings of the 30th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 2823–2840, 2019.

[26] Chi-Kit Lam and C Gregory Plaxton. Maximum stable matching with one-sided ties of bounded length. In *Proceedings of the 12th International Symposium on Algorithmic Game Theory*, pages 343–356, 2019.

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

[28] David F Manlove. *Algorithmics of Matching Under Preferences*. World Scientific, 2013.

[29] 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.

[30] Eric McDermid. A 3/2-approximation algorithm for general stable marriage. In *Proceedings of the 36th International Colloquium on Automata, languages and programming*, pages 689–700, 2009.

[31] Katarzyna Paluch. Faster and simpler approximation of stable matchings. *Algorithms*, 7(2):189–202, 2014.

[32] Alvin E. Roth and Marilda A. Oliveira Sotomayor. *Two-Sided Matching: A Study in Game-Theoretic Modeling and Analysis*. Cambridge University Press, 1990.

[33] Hiroki Yanagisawa. *Approximation Algorithms for Stable Marriage Problems.* PhD thesis, Kyoto University, 2007.