論文 / 著書情報
Article / Book Information

題目(和文)	
Title(English)	Algorithms and Graph-Theoretic Characterizations of Problems in Matching Under Preferences
著者(和文)	Ruangwises Suthee
Author(English)	Suthee RUANGWISES
出典(和文)	学位:博士(理学), 学位授与機関:東京工業大学, 報告番号:甲第11573号, 授与年月日:2020年9月25日, 学位の種別:課程博士, 審査員:伊東 利哉,渡辺 治,田中 圭介,鹿島 亮,森 立平
Citation(English)	Degree:Doctor (Science), Conferring organization: Tokyo Institute of Technology, Report number:甲第11573号, Conferred date:2020/9/25, Degree Type:Course doctor, Examiner:,,,,
学位種別(和文)	博士論文
Type(English)	Doctoral Thesis

Tokyo Institute of Technology

Ph.D. Thesis

Algorithms and Graph-Theoretic Characterizations of Problems in Matching Under Preferences

Author:
Suthee Ruangwises

Advisor:
Prof. Toshiya Itoh
Secondary Advisor:
Prof. Osamu Watanabe

A thesis submitted in fulfillment of the requirements
for the degree of Ph.D. in Mathematical and Computing Science

in the

Department of Mathematical and Computing Science
School of Computing

August 2020

Abstract

Matching under preferences is one of the most actively studied problems in theoretical computer science. The general objective of this problem is to match people with other people or with items, while each person has a list that ranks other people or items in order of preference. Two measures of optimality have been widely studied: popularity and stability. A matching is *popular* if it does not lose in a head-to-head election against any other matching. A matching is *stable* if there is no pair of people that are not matched to each other but prefer each other to their own partners.

In this thesis, we investigate three open problems related to popular and stable matchings using graph-theoretic characterizations. In the first problem, we study a probability that a popular matching exists in a random instance. In the second problem, we present an algorithm to measure badness of a matching that is not popular. In the third problem, we develop an algorithm to find a matching that has a property close to that of a stable matching and does not cross itself geometrically.

Acknowledgements

I would like to express my gratitude to all the people who contributed in some way to the work presented in this thesis. First, I am deeply grateful to my academic advisor, Prof. Toshiya Itoh, for accepting me into his lab and continuously providing me his excellent guidance, patience, and motivation for doing research throughout my Master's and Ph.D. studies during the past five years. I would also like to thank my secondary advisor, Prof. Osamu Watanabe, for his initial support since 2014 when I visited Tokyo Institute of Technology as a research intern while I was a third-year undergraduate student. He warmly welcomed to his lab and engaged me in novel ideas which brought me into this area of study. Besides Prof. Itoh and Prof. Watanabe, I would like to thank the other members of my thesis committee: Prof. Keisuke Tanaka, Assoc. Prof. Ryo Kashima, and Asst. Prof. Ryuhei Mori, for their interest in my work.

I would like to acknowledge the Department of Mathematical and Computing Science and the former Department of Information Processing at Tokyo Institute of Technology. My experience benefited greatly from the high-quality courses I took and the seminars I participated. I would also like to thank the current and past members of Itoh Lab and Watanabe Lab for the fruitful discussions about my research.

I am grateful to the funding sources that allow me to pursue my Master's and Ph.D. studies: the Development and Promotion of Science and Technology Talents Project Scholarship from the Royal Thai Government and the Monbukagakusho Honors Scholarship from the Japan Student Services Organization. I also received financial support from Tokyo Institute of Technology for my attendance at various conferences, from which I gained new experience and ideas in many aspects.

Finally, I must express my profound gratitude to my family: my parents and my late grandmother, for providing me continuous support and encouragement with their best wishes throughout the years of my studies and research. This accomplishment would not have been possible without them.

Contents

1 Introduction

1.1 Background

Matching under preferences is a problem involving matching people with items or with other people while each person has a list that ranks items or other people in his/her order of preference. This problem models many important real-world situations such as assignment of medical residents to hospitals [44], graduates to training positions [26], students to schools [1, 2], and families to government-subsidized housing [53].

The main objective of this problem is to find an optimal matching in each situation. Various definitions of optimality have been proposed. The least restrictive one is *Pareto optimality* [3, 4, 45]. A matching M is Pareto optimal if there is no other matching M' such that at least one person prefers M' to M but no one prefers M to M'. Other stronger definitions include *rank-maximality* [29] (allocating maximum number of people to their first choices, then maximum number to their second choices, and so on). However, two most well-studied properties of matchings are *stability* [16] and *popularity* [5, 18].

The *Stable Marriage Problem* is one of the most actively studied problems in computer science, mathematics, and economics [21, 46]. In the original bipartite setting called *Marriage Problem* (MP), a set of $n/2$ men and a set of $n/2$ women are given. Each person has a *preference list* that ranks all people of the opposite gender in strict order of preference. A man and a woman are called a *blocking pair* w.r.t. a matching M if they are not matched with each other in M but prefer each other to their own partners in M. A matching is called *stable* if it does not admit any blocking pair. Gale and Shapley [16] proved that a stable matching always exists in any instance and developed an $O(n^2)$ time algorithm to find one. The *Stable Roommates Problem* is a generalization of the original Stable Marriage Problem to a non-bipartite setting called *Roommates Problem* (RP), where each person can be matched with anyone regardless of gender. Unlike in MP, a stable matching in RP does not always exist [27].

Apart from stability, another less restrictive property of a preferable matching is popularity. For a pair of matchings X and Y, let $\phi(X,Y)$ denote the number of people who prefer a person they get matched by X to a person they get matched by Y. A matching M is called *popular* if $\phi(M,M') \geq \phi(M',M)$ for any other matching M'. The concept of popularity of a matching was first introduced by Gärdenfors [18] in the context of a cognitive science problem. Besides MP and RP settings, popular matchings were also studied in a setting of one-sided preference lists (matching people with items, where each person has a list that ranks items but each item does not have a list that ranks people) called *House Allocation Problem* (HAP). Note that the relation $\phi(X,Y) \geq \phi(Y,X)$ is not transitive, so a popular matching may or may not exist depending on the preference lists of people. See Example 1.

Example 1. Consider the following HAP instance with three people a_1, a_2, a_3 and three items b_1, b_2, b_3, with everyone having the same preferences.

Preference Lists	
$\boldsymbol{a_1}$: b_1, b_2, b_3	$M_1 = \{\{a_1, b_1\}, \{a_2, b_2\}, \{a_3, b_3\}\}$
$\boldsymbol{a_2}$: b_1, b_2, b_3	$M_2 = \{\{a_1, b_2\}, \{a_2, b_3\}, \{a_3, b_1\}\}$
$\boldsymbol{a_3}$: b_1, b_2, b_3	$M_3 = \{\{a_1, b_3\}, \{a_2, b_1\}, \{a_3, b_2\}\}$

For the three above matchings, we have $\phi(M_1, M_2) = 2 > 1 = \phi(M_2, M_1)$. Similarly, we also have $\phi(M_2, M_3) = 2 > 1 = \phi(M_3, M_2)$ and $\phi(M_3, M_1) = 2 > 1 = \phi(M_1, M_3)$. In fact, a popular matching does not exist in this instance. □

While a popular matching may not exist in some instances, several measures of badness of a matching that is not popular have been introduced. McCutchen [40] introduced two such measures: *unpopularity factor* and *unpopularity margin*. The unpopularity factor $u(M)$ of a matching M is the maximum ratio $\phi(M', M)/\phi(M, M')$ among all other possible matchings M', while the unpopularity margin $g(M)$ is the maximum difference $\phi(M', M) - \phi(M, M')$ among all other possible matchings M'. These two measures apply to all of MP, RP, and HAP settings. Note that the two measures are not equivalent as $\phi(M', M)$ and $\phi(M, M')$ may not add up to the total number of people since some people may like M and M' equally, thus it is possible for a matching to have higher unpopularity factor but lower unpopularity margin than another matching. See Example 2.

Example 2. Consider the following RP instance. A set in a preference list means all people in that set are ranked equally, e.g. a_2 prefers a_1 and a_4 equally as his first choices over a_3.

Preference Lists	
$\boldsymbol{a_1}$: a_4, a_2, a_3	$M_0 = \{\{a_1, a_2\}, \{a_3, a_4\}\}$
$\boldsymbol{a_2}$: $\{a_1, a_4\}, a_3$	$M_1 = \{\{a_1, a_3\}, \{a_2, a_4\}\}$
$\boldsymbol{a_3}$: $\{a_1, a_4\}, a_2$	$M_2 = \{\{a_1, a_4\}, \{a_2, a_3\}\}$
$\boldsymbol{a_4}$: $\{a_2, a_3\}, a_1$	

In this example, $\phi(M_0, M_1) = 1$, $\phi(M_1, M_0) = 0$, $\phi(M_0, M_2) = 3$, $\phi(M_2, M_0) = 1$, $\phi(M_1, M_2) = 3$, and $\phi(M_2, M_1) = 1$. Therefore, M_0 is popular, while $u(M_1) = \infty$, $g(M_1) = 1 - 0 = 1$, $u(M_2) = 3/1 = 3$, and $g(M_2) = 3 - 1 = 2$. Observe that M_1 has higher unpopularity factor but lower unpopularity margin than M_2. □

Finally, besides constraints on the preferences, geometric constraints are also important factors to consider in many real-world situations involving matchings, such as in the VLSI layout design [31]. In general, the *Noncrossing Matching Problem* (NMP) deals with a set of vertices lying on two parallel lines, with some edges joining vertices on the opposite lines. The goal of NMP is to find a *noncrossing matching*, a matching whose edges do not cross one another, subject to different objectives such as maximum size, maximum weight, etc.

1.2 Known Results

1.2.1 Stable Matchings

In an MP instance with $n/2$ men and $n/2$ women, Gusfield and Irving [21] showed that the Gale–Shapley algorithm [16] can be adapted to the setting where each person's preference list may not contain all people of the opposite gender. The algorithm runs in $O(m)$ time in this setting, where m is the total length of people's preference lists (i.e. the number of edges). Gale and Sotomayor [17] proved that in this modified setting, a stable matching may have size less than $n/2$, but every stable matching must have equal size and match the same set of people.

Irving [28] later generalized the notion of a stable matching to a setting where ties are allowed in people's preference lists and presented three possible interpretations of stability.

- A matching is *weakly stable* if there is no pair that strictly prefer each other to their own partners.
- A matching is *strongly stable* if there is no pair (a, b) such that a strictly prefers b to a's partner and b likes a not less than b's partner.
- A matching is *super-stable* if there is no pair that like each other not less than their own partners.

He also proposed an $O(n^2)$ time algorithm to find a weakly stable matching (which always exists), and an $O(n^4)$ time (resp. $O(n^2)$ time) algorithm to find a strongly stable (resp. super-stable) matching or report that none exists. The running time of the algorithm for a strongly stable matching was later improved to $O(mn)$ by Kavitha et al. [34].

In an RP instance with n people, Irving [27] developed an $O(n^2)$ time algorithm to find a stable matching or report that none exists. Tan [51] studied the exact condition for the existence of a stable matching and discovered that a stable matching exists if and only if a structure called *odd party* does not exist.

1.2.2 Popular Matchings

Gärdenfors [18] proved that in an MP instance where each person's preference list is *strict* (containing no tie), every stable matching must be popular (but not vice-versa), hence a popular matching always exists. In fact, every stable matching is also a minimum size popular matching, as proved by Huang and Kavitha [24]. They also developed an algorithm to compute a maximum size popular matching in this exact setting in $O(m \min(n_M, n_W))$ time, where n_M and n_W are the number of men and the number of women, respectively. The running time of this algorithm was later improved to $O(m)$ by Kavitha [32]. Also, Kavitha [33] showed that the problem of determining whether there is a popular matching of a given size s in a given instance is NP-hard.

While a popular matching always exists in an MP instance with strict preference lists, the problem of determining whether a popular matching exists in a given instance, however, becomes more computationally challenged in other settings. Biró et al. [8] proved that when

	Strict Preference Lists	**Ties Allowed**
Unweighted Setting	$O(m+n)$ [37]	$O(m\sqrt{n})$ [37]
Weighted Setting	$O(m+n)$ [42]	$O(m\min(t\sqrt{n},n))$ [42]
CHAP setting	$O(m+\sqrt{C}n_1)$ [39]	$O(m(\sqrt{C}+n_1))$ [39]

Table 1.1: Best known deterministic algorithms to find a popular matching or report that none exists in HAP

ties in the preference lists are allowed, determining whether a popular matching exists in a given MP or RP instance is NP-hard. Cseh et al. [10] showed that this problem is NP-hard in MP even when ties are allowed on only one side. Very recently, Faenza et al. [13] and Gupta et al. [20] independently proved that this problem is still NP-hard in RP even when people's preference lists are strict. Cseh and Kavitha [11] showed that in a complete graph RP instance with n people where each person's preference list is strict and contains all other people, the problem of determining whether a popular matching exists is solvable in polynomial time for an odd n but is NP-hard for an even n.

Popular matchings were also extensively studied in HAP setting, where a set A of n_1 people and a set B of n_2 items are given. Abraham et al. [5] developed an algorithm to find a popular matching in a given HAP instance, or report that none exists. The algorithm runs in $O(m+n)$ time when people's preference lists are strict and in $O(m\sqrt{n})$ time when ties are allowed, where m is the total length of people's preference lists (i.e. the number of edges) and $n = n_1 + n_2$ is the total number of people and items (i.e. the number of vertices). Kavitha and Shah [36] proposed a randomized algorithm to solve the same problem when ties are allowed in $O(n^\omega)$ time, where $\omega < 2.376$ is the exponent of matrix multiplication. This algorithm performs slightly better than that of Abraham et al. in dense graphs where $m = \Theta(n^2)$.

Mestre [42] generalized the algorithm of Abraham et al. to a weighted setting where people are given different voting weights. The algorithm runs in $O(m+n)$ time when ties are not allowed and in $O(m\min(t\sqrt{n},n))$ time when ties are allowed, where t is the number of distinct weights. Manlove and Sng [39] also generalized their algorithm to a setting where each item is allowed to be matched with more than one person called *Capacitated House Allocation Problem* (CHAP). The algorithm runs in $O(m+\sqrt{C}n_1)$ time when ties are not allowed and in $O(m(\sqrt{C}+n_1))$ time when ties are allowed, where C is the total capacity of all items. Table 1.1 shows the running time of the best known deterministic algorithms to find a popular matching or report that none exists in different settings of HAP.

McDermid and Irving [41] constructed a structure called *switching graph* that can be used to solve several problems in HAP such as counting the number of popular matchings and enumerating all popular matchings. Abraham and Kavitha [6] proved that in any HAP instance with at least one popular matching, one can achieve a popular matching by conducting at most two majority votes to force a change in assignments, starting at any matching. Kavitha et al. [35] introduced the concept of a *mixed matching*, which is a probability distribution over a set of matchings, and proved that a mixed matching that is "popular" always exists.

The *Random Popular Matching Problem* (RPMP) is a probabilistic problem involving the probability of existence of a popular matching in a random instance. For RPMP in HAP, each person's preference list is defined independently by selecting the first item $b_1 \in B$ uniformly at random, the second item $b_2 \in B \setminus \{b_1\}$ uniformly at random, the third item $b_3 \in B \setminus \{b_1, b_2\}$ uniformly at random, and so on. Mahdian [37] proved that in a random HAP instance with strict and complete preference lists, if $\alpha = n_2/n_1 > \alpha_*$, where $\alpha_* \approx 1.42$ is the root of equation $x^2 = e^{1/x}$, then a popular matching exists with high probability ($1 - o(1)$ probability). On the other hand, if $\alpha < \alpha_*$, then a popular matching exists with low probability ($o(1)$ probability). The point $\alpha = \alpha_*$ can be regarded as a phase transition point, at which the probability rises from asymptotically zero to asymptotically one. Itoh and Watanabe [30] later studied the weighted setting where each person has weight either w_1 or w_2, with $w_1 \geq 2w_2$, and found a phase transition at $\alpha = \Theta(\sqrt[3]{n_1})$.

1.2.3 Unpopularity Measures

McCutchen [40] developed an algorithm to compute $u(M)$ and $g(M)$ of a given matching M of an HAP instance in $O(m\sqrt{n_2})$ and $O((g+1)m\sqrt{n})$ time, respectively, where $g = g(M)$ is the unpopularity margin of M. He also proved that the problem of finding a matching that minimizes either measure is NP-hard. Huang et al. [25] later developed an algorithm to find a matching with bounded values of these measures in HAP instances with certain properties.

The notions of unpopularity factor and unpopularity margin also apply to MP and RP settings. Biró et al. [8] developed an algorithm to determine whether a given matching M is popular in $O(m\sqrt{n})$ time for MP and in $O(m\sqrt{n}\log n)$ time for RP (when running with the recent fastest algorithm to find a maximum weight perfect matching [12]). Their algorithm also simultaneously computes the unpopularity margin of M during the run. Huang and Kavitha [23] showed that an RP instance with strict preference lists always admits a matching with unpopularity factor $O(\log n)$, and proved that it is NP-hard to find a matching with the lowest unpopularity factor, or even the one with less than 4/3 times of the optimum. Kavitha [32] showed that in an MP instance with strict preference lists, for any positive integer $u < \min(n_M, n_W)$, there is a matching of size at least $\frac{u+1}{u+2}|M_{\max}|$ with unpopularity factor at most u, where $M_{\max}$ is a maximum size matching. Tables 1.2 and 1.3 show the running time of the best known algorithms related to popularity in each setting with strict preference lists, and with ties allowed, respectively.

1.2.4 Noncrossing Matchings

There have been several results on NMP in a bipartite graph where $2n$ vertices lie on two parallel lines, each containing n vertices. In a special setting where each vertex is adjacent to exactly one vertex on the opposite line, Fredman [15] presented an $O(n \log n)$ time algorithm to find a maximum size noncrossing matching by computing the length of the longest increasing subsequence (LIS). Widmayer and Wong [52] developed another algorithm that runs in $O(s + (n-s)\log(s+1))$ time, where s is the size of the solution. This algorithm has the same worst-case running time as the algorithm of Fredman, but runs faster in average case.

In a general setting where each vertex can be adjacent to any number of vertices on the

<table>
<tr><th rowspan="2"></th><th colspan="2">Two-sided Lists</th><th>One-sided Lists</th></tr>
<tr><th>Marriage Problem (MP)</th><th>Roommates Problem (RP)</th><th>House Allocation Problem (HAP)</th></tr>
<tr><td>Determine if a popular matching exists</td><td rowspan="3">$O(m)$ [18]</td><td rowspan="2">NP-hard [13, 20]</td><td>$O(m+n)$ [5]</td></tr>
<tr><td>Find a matching M that minimizes $g(M)$</td><td rowspan="2">NP-hard [40]</td></tr>
<tr><td>Find a matching M that minimizes $u(M)$</td><td>NP-hard [23]</td></tr>
<tr><td>Test popularity of a given matching M</td><td rowspan="2">$O(m\sqrt{n})$ [8]</td><td rowspan="2">$O(m\sqrt{n}\log n)$ [8, 12]</td><td>$O(m+n)$ [5]</td></tr>
<tr><td>Compute $g(M)$ of a given matching M</td><td>$O((g+1)m\sqrt{n})$ [40]</td></tr>
<tr><td>Compute $u(M)$ of a given matching M</td><td></td><td></td><td>$O(m\sqrt{n_2})$ [40]</td></tr>
</table>

Table 1.2: Best known algorithms for an unweighted instance with strict preference lists

<table>
<tr><th rowspan="2"></th><th colspan="2">Two-sided Lists</th><th>One-sided Lists</th></tr>
<tr><th>Marriage Problem (MP)</th><th>Roommates Problem (RP)</th><th>House Allocation Problem (HAP)</th></tr>
<tr><td>Determine if a popular matching exists</td><td colspan="2" rowspan="3">NP-hard [8]</td><td>$O(m\sqrt{n})$ [5]</td></tr>
<tr><td>Find a matching M that minimizes $g(M)$</td><td rowspan="2">NP-hard [40]</td></tr>
<tr><td>Find a matching M that minimizes $u(M)$</td></tr>
<tr><td>Test popularity of a given matching M</td><td rowspan="2">$O(m\sqrt{n})$ [8]</td><td rowspan="2">$O(m\sqrt{n}\log n)$ [8, 12]</td><td>$O(m\sqrt{n_2})$ [40]</td></tr>
<tr><td>Compute $g(M)$ of a given matching M</td><td>$O((g+1)m\sqrt{n})$ [40]</td></tr>
<tr><td>Compute $u(M)$ of a given matching M</td><td></td><td></td><td>$O(m\sqrt{n_2})$ [40]</td></tr>
</table>

Table 1.3: Best known algorithms for an unweighted instance with ties allowed in the preference lists

opposite line, Malucelli et al. [38] developed an algorithm to find a maximum size noncrossing matching. The algorithm runs in either $O(m \log\log n)$ or $O(m + \min(ns, m \log s))$ time depending on implementation, where m is the number of edges. In a setting where each edge has a weight, they also showed that the algorithm can be adapted to find a maximum weight noncrossing matching with $O(m \log n)$ running time.

1.3 Motivation and Goals

RPMP in HAP has a practical importance as it helps us predict whether a popular matching exists in a situation where we know only the number of people and items, but not the preferences of people. For example, a DVD rental shop owners can predict how many DVDs they have to prepare in order to satisfy customers, given only the number of customers. The previous result of Mahdian [37] solved this problem in the case where people's preference lists are both strict and complete. However, in many real-world situations, people's preference lists may not be complete since people may regard some items as undesired at all. For an instance where the preference lists are strict but not complete, with every person's preference list having the same length of a constant k, this problem was simulated by Abraham et al. [5], and was conjectured by Mahdian [37] that the phase transition point would shift by an amount exponentially small in k. However, the exact phase transition point, or whether it exists at all, had not been found yet. This leads to the first open problem we aim to solve.

It is known that the problem of finding a matching with minimum unpopularity factor or unpopularity margin in a given instance is NP-hard in most settings. However, the problem of computing an unpopularity factor or unpopularity margin of a given matching is deemed to be computationally easier. This problem also has importance as the algorithm works in a sense that it measures how bad a given solution is. In HAP, there are polynomial-time algorithms to compute both measures [40]. However, in MP or RP, there is only such algorithm to compute the unpopularity margin [8] but not the unpopularity factor, as shown in Tables 1.2 and 1.3. This leads to the second open problem we aim to solve.

Finally, we aim to study matching problems with both preferential constraints and geometric constraints. In real-world situations, the geometric constraints may represent physical locations, e.g. in the construction of bridges between cities on the two sides of a river, or may represent non-physical concepts such as rank of people or time. Consider an NMP instance where each vertex has a preference list containing vertices on the opposite line. We are interested in whether there exists a noncrossing matching that is also stable, or at least has a property close to that of a stable matching. This leads to the third open problem we aim to solve.

1.4 Methodology

In this thesis, we analyze three open problems in matching under preferences using graph-theoretic characterizations.

In the first problem, for each HAP instance, we construct an auxiliary graph called *top-choice graph* with a property that a popular matching exists if and only if the top-choice graph contains a *complex component* (a component with more than one cycle). The top-choice graph is in turn approximated by another auxiliary random graph. To find a phase transition point of probability of existence of a popular matching, we prove the upper bound and the lower bound separately. For the upper bound, we bound the number of subgraphs with specific properties in order to bound the probability of existence of a complex component in a random graph. For the lower bound, we use the *Galton-Watson branching process* (shown in Section 2.4) to bound the probability of existence of a complex component in a random graph.

In the second problem, for each MP or RP instance and each macthing, we construct an auxiliary graph that has close relation to the unpopularity factor of that matching. Then, we reduce the problem of computing the unpopularity factor into the problem of detecting a positive weight directed cycle for MP, and detecting a positive weight perfect matching for RP.

In the third problem, each NMP instance is represented by points on a plane. We develop a computational geometric algorithm that will always find a noncrossing matching has a property close to that of a stable matching, which implicitly proves that such matching always exists.

1.5 Contribution

In Chapter 3, we study RPMP in an HAP instance where the preference lists are strict but not complete, with every person's preference list having the same length of a constant k, and discover a phase transition at $\alpha = \alpha_k$, where $\alpha_k \geq 1$ is the root of equation $xe^{-1/2x} = 1-(1-e^{-1/x})^{k-1}$. In particular, we prove that for $k \geq 4$, if $\alpha > \alpha_k$, then a popular matching exists with high probability; and if $\alpha < \alpha_k$, then a popular matching exists with low probability. For $k \leq 3$, where the equation does not have a solution in $[1, \infty)$, a popular matching always exists with high probability for any value of $\alpha \geq 1$ without a phase transition. We also perform a simulation to help illustrate and verify the discovered phase transition.

In Chapter 4, we develop the first polynomial-time algorithm to compute the unpopularity factor of a given matching in an MP or RP instance by employing an auxiliary graph similar to the one in [8]. The algorithm runs in $O(m\sqrt{n}\log n)$ time for MP and in $O(m\sqrt{n}\log^2 n)$ time for RP. We also generalize the notion of unpopularity factor to the weighted setting where people are given different voting weights, and show that our algorithm can be slightly modified to support that setting with the same running time.

In Chapter 5, we investigate an NMP instance with n men and n women represented by points lying on two parallel lines, each line containing n people of one gender. Each person has a strict preference list that ranks a subset of people of the opposite gender. A *noncrossing blocking pair* w.r.t. a matching M is a blocking pair w.r.t. M that does not cross any edge in M. Our goal is to find a noncrossing matching that does not admit any noncrossing blocking pair, called a *weakly stable noncrossing matching* (WSNM). We constructively prove that a WSNM always exists in any instance by developing an $O(n^2)$ time algorithm to find one in a given instance.

2 Preliminaries

2.1 Basic Graph Terminologies

- **directed graph:** a graph with each edge having a direction, from one endpoint to another endpoint, assigned to it
- **weighted graph:** a graph with each edge having a real number called *weight* assigned to it
- **bipartite graph:** a graph whose vertices can be partitioned into two disjoint sets such that there is no edge connecting two vertices in the same set
- **matching:** a set of edges such that any two of them do not share any vertex
- **perfect matching:** a set of edges such that every vertex in the graph belongs to exactly one of them
- **cycle:** a sequence $(v_0, e_1, v_1, e_2, v_2, \ldots, e_k, v_k)$ of vertices and edges such that an edge e_i has endpoints v_{i-1} and v_i for each $i = 1, 2, \ldots, k$, with the condition that $v_0 = v_k$ and $v_0, v_1, \ldots, v_{k-1}$ are all different
- **alternating cycle w.r.t. a matching M:** a cycle whose edges alternate between the edges in M and not in M

2.2 Popular Matchings and Unpopularity Factor

In MP and RP, we consider an instance I consisting of a set A of n people, where each person has a preference list that ranks a subset of A as his/her acceptable partners in order of preference. In RP there is no further restriction, while in MP people are classified into two genders, and each person's preference list can contain only people of the opposite gender. In HAP, we consider an instance I consisting of a set A of n_1 people and a set B of n_2 items, with $n = n_1 + n_2$, where each person has a preference list that ranks a subset of B as his/her acceptable items in order of preference.

Let m be the total length of people's preference lists. A preference list is called *strict* if it does not contain tie. In HAP, a preference list is called *complete* if it contains all items in B.

For a matching M and a person $a \in A$, let $M(a)$ denote the person/item matched with a in M (for convenience, let $M(a) = null$ if a is unmatched in M). Analogously, in HAP, let $M(b)$ denote the person matched with an item $b \in B$ in M. Also, for a person $b \in A$ in MP and RP, or an item $b \in B$ in HAP, let $r_a(b)$ be the rank of b in a's preference list, with the most preferred one(s) having rank 1, the second most preferred one(s) having rank 2, and so on (for convenience, let $r_a(null) = \infty$).

Let $\mathbb{M}$ be the set of all matchings of a given instance I. For any pair of matchings X and Y in $\mathbb{M}$, define $\phi(X,Y)$ to be the number of people who strictly prefer the person/item they get matched by X to the person/item they get matched by Y, i.e.

$$\phi(X,Y) = |\{a \in A | r_a(X(a)) < r_a(Y(a))\}|.$$

Definition 1. [5, 18] A matching $M \in \mathbb{M}$ is *popular* if $\phi(M,M') \geq \phi(M',M)$ for every matching $M' \in \mathbb{M} - \{M\}$.

Also, let

$$\Delta(X,Y) = \begin{cases} \phi(Y,X)/\phi(X,Y), & \text{if } \phi(X,Y) > 0; \\ 1, & \text{if } \phi(X,Y) = \phi(Y,X) = 0; \\ \infty, & \text{otherwise.} \end{cases}$$

Definition 2. [40] For a matching $M \in \mathbb{M}$, an *unpopularity factor* $u(M)$ is defined by

$$u(M) = \max_{M' \in \mathbb{M} - \{M\}} \Delta(M,M').$$

Definition 3. [40] For a matching $M \in \mathbb{M}$, an *unpopularity margin* $g(M)$ is defined by

$$u(M) = \max_{M' \in \mathbb{M} - \{M\}} (\phi(M',M) - \phi(M,M')).$$

From Definitions 1, 2, and 3, it follows that a matching M is popular if and only if $u(M) \leq 1$, and M is popular if and only if $g(M) \leq 0$.

2.2.1 Random Instances

Consider an HAP instance where every person's preference list has equal length of a constant $k \leq n_2$. Such instance is called an *instance with k-incomplete preference lists*.

Definition 4. For a positive integer $k \leq n_2$, a *random instance with strict and k-incomplete preference lists* is an instance with each person's preference list chosen independently and uniformly from the set of all $\frac{n_2!}{(n_2-k)!}$ possible k-permutations of the n_2 items in B at random.

2.3 Noncrossing Matching Problem

In NMP, we consider a set of n men $m_1, m_2, \ldots, m_n$ represented by points lying on a vertical line in this order from top to bottom, and a set of n women $w_1, w_2, \ldots, w_n$ represented by points lying on another parallel line in this order from top to bottom. Each person a has a strict preference list denoted by a sequence L_a of people of the opposite gender, with the i-th entry being the i-th most preferred person by a.

A pair of edges *cross* each other if they intersect in the interior of both segments. Formally, an edge (m_i, w_j) crosses an edge (m_x, w_y) if and only if $(i-x)(j-y) < 0$. A matching is called *noncrossing* if it does not contain any pair of edges that cross each other.

Definition 5. A *blocking pair* w.r.t. a matching M is a pair (m_i, w_j) of a man and a woman that are not matched with each other, but m_i prefers w_j to $M(m_i)$ and w_j prefers m_i to $M(w_j)$.

Definition 6. A *noncrossing blocking pair* w.r.t. a matching M is a blocking pair w.r.t. M that does not cross any edge in M.

Definition 7. A matching M is called a *weakly stable noncrossing matching* (WSNM) if M is noncrossing and does not admit any noncrossing blocking pair.

Definition 8. A matching M is called a *strongly stable noncrossing matching* (SSNM) if M is noncrossing and does not admit any blocking pair.

Remark 1. Although the real-world applications of this geometric problem are likely to involve immovable objects, we keep the terminologies of men and women used in the original Stable Marriage Problem in order to understand and relate to the original problem more easily.

2.4 Miscellaneous

We first introduce *Chebyshev's inequality*, which states that for a random variable X with expected value μ and variance σ^2, and for any real number $k > 0$, we have

$$\Pr\left(|X - \mu| \geq k\sigma\right) \leq \frac{1}{k^2}.$$

It is worth noting the following lemma proved by Mahdian [37] about independent and uniform selection of items at random, which will be used in this thesis.

Lemma 1. [37] Suppose that we pick y elements from the set $\{1, 2, \ldots, z\}$ independently and uniformly at random (with replacement). Let a random variable X be the number of elements in the set that are not picked. Then, $\mathbb{E}[X] = e^{-y/z}z - \Theta(1)$ and $\mathrm{Var}[X] < \mathbb{E}[X]$.

Finally, we introduce the *Galton-Watson branching process* [7, pp.182–184], which is a process that generates a random graph in a breadth-first search tree manner when given a starting vertex and a distribution of the degree of each vertex. The process begins when the starting vertex spawns a number of children which are put in the queue in some order. Then, the first vertex in the queue also spawns children which are put at the end of the queue by the same manner, and so on. The process may stop at some point when the queue becomes empty, or otherwise continues indefinitely.

3 Random Popular Matching Problem in HAP

Consider an HAP instance consisting of a set A of n_1 people and a set B of $n_2 \geq n_1$ items, with $\alpha = n_2/n_1 \geq 1$. Throughout this chapter, we consider a setting where every person's preference list is strict.

In this chapter, we will investigate the phase transition in a random instance with strict and k-incomplete preference lists. In particular, we will prove the following statements.

- For $k \geq 4$, if $\alpha > \alpha_k$, then a popular matching exists with high probability; and if $\alpha < \alpha_k$, then a popular matching exists with low probability, where $\alpha_k \geq 1$ is the root of equation $xe^{-1/2x} = 1 - (1 - e^{-1/x})^{k-1}$.
- For $k \leq 3$, a popular matching always exists with high probability for any $\alpha \geq 1$.

We will also perform a simulation to help illustrate and verify the discovered phase transition.

3.1 A-Perfect Matchings

For convenience, for each person $a \in A$ we append a unique auxiliary *last resort item* ℓ_a to the end of a's preference list (ℓ_a has lower preference than all other items in the list). By introducing the last resort items, we can assume that every person is matched because we can simply match any unmatched person a with ℓ_a. Note that these last resort items are not in B and thus do not count toward n_2. Also, let $L = \{\ell_a | a \in A\}$ be the set of all last resort items.

For each person $a \in A$, let $f(a)$ denote the item at the top of a's preference list. Let $F = \{f(a) | a \in A\}$ be the set of all first-choice items, and let $S = B - F$. Then, for each person $a \in A$, let $s(a)$ denote the highest ranked item in a's preference list that is not in F. Note that $s(a)$ is well-defined for every $a \in A$ because of the existence of last resort items.

We say that a matching M is *A-perfect* if every person $a \in A$ is matched with either $f(a)$ or $s(a)$. Abraham et al. [5] proved the following lemma, which holds for any instance with strict (not necessarily complete) preference lists.

Lemma 2. [5] In any instance with strict preference lists, a popular matching exists if and only if an A-perfect matching exists.

The proof of Lemma 2 first shows that a matching M is popular if and only if M is an A-perfect matching such that every item in F is matched in M. This equivalence implies the forward direction of the lemma. On the other hand, the proof also shows that for any A-perfect matching M, we can modify M to make every item in F matched, hence implying the backward direction of the lemma.

3.2 Complete Preference Lists Setting

First, consider a setting where every person's preference list is strict and complete. Note that when $n_2 > n_1$, the last resort items are not necessary.

From a given instance, we construct a *top-choice graph*, a bipartite graph with parts $B' = B$ and $S' = S$ such that each person $a \in A$ corresponds to an edge connecting $f(a) \in B'$ and $s(a) \in S'$. Note that multiple edges are allowed in this graph. Previously, Mahdian [37] proved the following lemma.

Lemma 3. [37] In any instance with strict and complete preference lists, an A-perfect matching exists if and only if its top-choice graph does not contain a *complex component*, i.e. a connected component with more than one cycle.

By Lemmas 2 and 3, the problem of determining whether a popular matching exists is equivalent to determining whether the top-choice graph contains a complex component. However, the difficulty is that the number of vertices in the randomly generated top-choice graph is not fixed. Therefore, a random bipartite graph $G(x,y,z)$ with fixed number of vertices is defined as follows to approximate the top-choice graph.

Definition 9. [37] For integers x,y,z, $G(x,y,z)$ is a bipartite graph with $V \cup U$ as a set of vertices, where $V = \{v_1, v_2, \ldots, v_x\}$ and $U = \{u_1, u_2, \ldots, u_y\}$. Each of the z edges of $G(x,y,z)$ is selected independently and uniformly at random (with replacement) from the set of all possible edges between a vertex in V and a vertex in U.

This auxiliary graph has properties closely related to the top-choice graph. Mahdian [37] then proved that if $\alpha > \alpha_* \approx 1.42$, then $G(n_2, h, n_1)$ contains a complex component with low probability for any integer $h \in [e^{-1/\alpha}n_2 - n_2^{2/3}, e^{-1/\alpha}n_2 + n_2^{2/3}]$, and used those properties to conclude that the top-choice graph also contains a complex component with low probability, hence a popular matching exists with high probability.

Theorem 1. [37] In a random instance with strict and complete preference lists, if $\alpha > \alpha_*$, where $\alpha_* \approx 1.42$ is the solution of the equation $x^2 e^{-1/x} = 1$, then a popular matching exists with probability $1 - o(1)$.

Theorem 1 serves as an upper bound of the phase transition point in the case of strict and complete preference lists. On the other hand, the following lower bound was also proposed by Mahdian [37] along with a sketch of the proof, although the fully detailed proof was not given.

Theorem 2. [37] In a random instance with strict and complete preference lists, if $\alpha < \alpha_*$, then a popular matching exists with probability $o(1)$.

3.3 Incomplete Preference Lists Setting

The previous section shows known results in the setting where every person's preference list is strict and complete. In this section, we consider a setting where every person's preference list is strict and has the same length of a constant k.

Recall that $F = \{f(a)|a \in A\}$, $S = B - F$, and for each person $a \in A$, $s(a)$ is the highest ranked item in a's preference list not in F. The main difference from the complete preference lists setting is that in the incomplete preference lists setting, $s(a)$ can be either a real item or a last resort item even when $n_2 > n_1$. For each person $a \in A$, let P_a be the set of items in a's preference list (not including the last resort item ℓ_a). We then define $A_1 = \{a \in A|P_a \subseteq F\}$ and $A_2 = \{a \in A|P_a \not\subseteq F\}$. Note that $s(a) = \ell_a$ if and only if $a \in A_1$.

3.3.1 Top-Choice Graph

Analogously to the complete preference lists setting, we define the top-choice graph of an instance with strict and k-incomplete preference lists to be a bipartite graph with parts $B' = B$ and $S' \cup L'$, where $S' = S$ and $L' = L$. Each person $a \in A_2$ corresponds to an edge connecting $f(a) \in B'$ and $s(a) \in S'$. We call these edges *normal edges*. Each person $a \in A_1$ corresponds to an edge connecting $f(a) \in B'$ and $s(a) = \ell_a \in L'$. We call these edges *last resort edges*.

Although the statement of Lemma 3 proved by Mahdian [37] is for the complete preference lists setting, exactly the same proof applies to the incomplete preference lists setting as well. The proof first shows that an A-perfect matching exists if and only if each edge in the top-choice graph can be oriented such that each vertex has at most one incoming edge (because if an A-perfect matching M exists, we can orient each edge corresponding to $a \in A$ toward the endpoint corresponding to $M(a)$, and vice versa). Then, the proof shows that for any undirected graph H, each edge of H can be oriented in such manner if and only if H does not contain a complex component. Thus we can conclude the following lemma.

Lemma 4. In any instance with strict and k-incomplete preference lists, an A-perfect matching exists if and only if its top-choice graph does not contain a complex component.

In contrast to the complete preference lists setting, the top-choice graph in the incomplete preference lists setting has two types of edges (normal edges and last resort edges) with different distributions, and thus cannot be approximated by $G(x,y,z)$ defined in the previous section. Therefore, we have to construct another auxiliary graph $G'(x,y,z_1,z_2)$ as follows.

Definition 10. For integers x,y,z_1,z_2, $G'(x,y,z_1,z_2)$ is a bipartite graph with $V \cup U \cup U'$ as a set of vertices, where $V = \{v_1,v_2,\ldots,v_x\}$, $U = \{u_1,u_2,\ldots,u_y\}$, and $U' = \{u'_1,u'_2,\ldots,u'_{z_1+z_2}\}$. This graph has z_1+z_2 edges. Each of the first z_1 edges is selected independently and uniformly at random (with replacement) from the set of all possible edges between a vertex in V and a vertex in U. Then, each of the next z_2 edges is constructed by the following procedures: Uniformly select a vertex v_i from V at random (with replacement); then, uniformly select a vertex u'_j that has not been selected before from U' at random (without replacement) and construct an edge (v_i,u'_j).

The intuition behind the construction of $G'(x,y,z_1,z_2)$ is that we imitate the distribution of the top-choice graph in the incomplete preference lists setting, with V, U, and U' correspond to B', S', and L', respectively, and the first z_1 edges and the next z_2 edges correspond to normal edges and last resort edges, respectively.

Similarly to the complete preference lists setting, this auxiliary graph has properties closely related to the top-choice graph in the incomplete preference lists setting, as shown in the following lemma.

Lemma 5. Suppose that the top-choice graph H has t normal edges and $n_1 - t$ last resort edges for a fixed integer $t \le n_1$, and E is an arbitrary event defined on graphs. If the probability of E on the random graph $G'(n_2,h,t,n_1-t)$ is at most $O(1/n_1)$ for every fixed integer $h \in [e^{-1/\alpha}n_2 - n_2^{2/3}, e^{-1/\alpha}n_2 + n_2^{2/3}]$, then the probability of E on the top-choice graph H is at most $O(n_1^{-1/3})$.

Proof. Using the same technique as in Mahdian's proof of [37, Lemma 3], let a random variable X be the number of isolated vertices (zero-degree vertices) in part V (the part that has n_2 vertices) of $G'(n_2,h,t,n_1-t)$. By the definition of $G'(n_2,h,t,n_1-t)$, for each fixed value of h, the distribution of H conditioned on $|S'| = h$ is the same as the distribution of $G'(n_2,h,t,n_1-t)$ conditioned on $X = h$ (because $|S| = |S'| = h$ means that part B' of H has exactly h isolated vertices which correspond to the vertices in S). Also, from Lemma 1 with $y = n_1$ and $z = n_2$, we have $\mathbb{E}[X] = e^{-1/\alpha}n_2 - \Theta(1)$ and $\mathrm{Var}[X] < \mathbb{E}[X]$. Let $\delta = \frac{1}{2}n_2^{2/3}$, and let $I = [E[X]-\delta, E[X]+\delta]$. We have $I \subseteq [e^{-1/\alpha}n_2 - n_2^{2/3}, e^{-1/\alpha}n_2 + n_2^{2/3}]$ for sufficiently large n_2. Therefore,

$$\begin{aligned}
\Pr_H[E] &= \sum_h \Pr_H\Big[E\Big||S| = h\Big] \cdot \Pr_H[|S| = h] \\
&= \sum_h \Pr_{G'(n_2,h,t,n_1-t)}[E|X = h] \cdot \Pr_{G'(n_2,h,t,n_1-t)}[X = h] \\
&= \sum_h \Pr_{G'(n_2,h,t,n_1-t)}[X = h|E] \cdot \Pr_{G'(n_2,h,t,n_1-t)}[E] \\
&\le \Pr[|X - \mathbb{E}[X]| > \delta] + \sum_{h\in I} \Pr_{G'(n_2,h,t,n_1-t)}[X = h|E] \cdot \Pr_{G'(n_2,h,t,n_1-t)}[E] \\
&\le \Pr[|X - \mathbb{E}[X]| > \delta] + \sum_{h\in I} \Pr_{G'(n_2,h,t,n_1-t)}[E].
\end{aligned}$$

From Chebyshev's inequality, we have

$$\begin{aligned}
\Pr_H[E] &\le \frac{\mathrm{Var}[X]}{\delta^2} + \sum_{h\in I} \Pr_{G'(n_2,h,t,n_1-t)}[E] \\
&\le \frac{\mathbb{E}[X]}{\delta^2} + 2\delta \max_{h\in I} \Pr_{G'(n_2,h,t,n_1-t)}[E] \\
&< \frac{O(n_2)}{n_2^{4/3}} + n_2^{2/3} O(1/n_1) \\
&= O(n_1^{-1/3})
\end{aligned}$$

as desired. $\square$

3.3.2 Size of A_2

Since our top-choice graph has two types of edges with different distributions, we first want to bound the number of each type of edges. Note that the top-choice graph has $|A_2|$ normal edges and $|A_1|$ last resort edges, so the problem is equivalent to bounding the size of A_2.

First, we will prove the next two lemmas, which will be used to bound the ratio $\frac{|A_2|}{n_1}$.

Lemma 6. In a random instance with strict and k-incomplete preference lists,

$$1 - e^{-1/\alpha} - c_1 < \frac{|F|}{n_2} < 1 - e^{-1/\alpha} + c_1$$

with probability $1 - o(1)$ for any constant $c_1 > 0$.

Proof. Let $c_1 > 0$ be any constant. From Lemma 1 with $y = n_1$ and $z = n_2$, we have

$$\mathbb{E}[|F|] = n_2 - \mathbb{E}[|S|] = (1 - e^{-1/\alpha})n_2 + \Theta(1); \tag{3.1}$$

$$\mathrm{Var}(|F|) = \mathrm{Var}(|S|) < \mathbb{E}[|S|] \le \frac{e^{-1/\alpha}}{1 - e^{-1/\alpha}}\mathbb{E}[|F|].$$

From Chebyshev's inequality, we have

$$\begin{aligned}\Pr\Big[\big||F| - \mathbb{E}[|F|]\big| \ge c_1 \cdot \mathbb{E}[|F|]\Big] &\le \frac{\mathrm{Var}[|F|]}{(c_1 \cdot \mathbb{E}[|F|])^2} \\ &< \frac{e^{-1/\alpha}}{c_1^2(1 - e^{-1/\alpha})\mathbb{E}[|F|]} = O(1/n_1).\end{aligned} \tag{3.2}$$

Therefore, from (3.1) and (3.2) we can conclude that

$$1 - e^{-1/\alpha} - c_1 < \frac{|F|}{n_2} < 1 - e^{-1/\alpha} + c_1$$

with probability $1 - o(1)$ for sufficiently large n_2. □

Lemma 7. In a random instance with strict and k-incomplete preference lists,

$$1 - (1 - e^{-1/\alpha})^{k-1} - c_2 < \Pr[a \in A_2] < 1 - (1 - e^{-1/\alpha})^{k-1} + c_2$$

holds for any $a \in A$ for sufficiently large n_2, given any constant $c_2 > 0$.

Proof. If $k = 1$, then we have $P_a \subseteq F$ for every $a \in A$, which means $\Pr[a \in A_2] = 0$ and thus the lemma holds. From now on, we will consider the case that $k \ge 2$.

Let $c_2 > 0$ be any constant. We can select a sufficiently small c_1 (e.g. $c_1 = \frac{c_2}{(k-1)(c_2+2)}$, where the proof is given in Subsection 3.3.3) such that

$$(1 - e^{-1/\alpha} - c_1)^{k-1} > (1 - e^{-1/\alpha})^{k-1} - \frac{c_2}{2}; \tag{3.3}$$

$$(1 - e^{-1/\alpha} + c_1)^{k-1} < (1 - e^{-1/\alpha})^{k-1} + \frac{c_2}{2},. \tag{3.4}$$

Let $I = [(1-e^{-1/\alpha}-c_1)n_2, (1-e^{-1/\alpha}+c_1)n_2]$. From Lemma 6, $|F| \in I$ with probability $1-o(1)$ for sufficiently large n_2.

Note that $a \in A_1$ if and only if $P_a - \{f(a)\} \subseteq F$. Consider the process that we first independently and uniformly select the first-choice item of every person in A from the set B at random, creating the set F. Suppose that $|F| = q$ for some fixed integer $q \in I$. Then, for each $a \in A$, we uniformly select the remaining $k-1$ items in a's preference list one by one from the remaining $n_2 - 1$ items in $B - \{f(a)\}$ at random. Among the $(k-1)!\binom{n_2-1}{k-1}$ possible ways of selection, there are $(k-1)!\binom{q-1}{k-1}$ ways such that $P_a - \{f(a)\} \subseteq F$, so

$$\begin{aligned}
\Pr\left[a \in A_1 \middle| |F| = q\right] &= \Pr\left[P_a - \{f(a)\} \subseteq F \middle| |F| = q\right] \\
&= \frac{(k-1)!\binom{q-1}{k-1}}{(k-1)!\binom{n_2-1}{k-1}} \\
&= \frac{\binom{q-1}{k-1}}{\binom{n_2-1}{k-1}}.
\end{aligned}$$

Since $\binom{q-1}{k-1}/\binom{n_2-1}{k-1}$ converges to $\left(\frac{q}{n_2}\right)^{k-1}$ when n_2 increases to infinity for every $q \in I$, it is sufficient to assume $\Pr\left[a \in A_1 \middle| |F| = q\right] = \left(\frac{q}{n_2}\right)^{k-1}$ for sufficiently large n_2.

Now consider

$$\begin{aligned}
\Pr[a \in A_1] &= \sum_q \Pr[|F| = q] \cdot \Pr\left[a \in A_1 \middle| |F| = q\right] \\
&= \sum_{q \in I} \Pr[|F| = q] \cdot \Pr\left[a \in A_1 \middle| |F| = q\right] \\
&\qquad + \sum_{q \notin I} \Pr[|F| = q] \cdot \Pr\left[a \in A_1 \middle| |F| = q\right].
\end{aligned}$$

For the lower bound of $\Pr[a \in A_1]$, we have

$$\begin{aligned}
\Pr[a \in A_1] &\geq \sum_{q \in I} \Pr[|F| = q] \cdot \Pr\left[a \in A_1 \middle| |F| = q\right] \\
&= \sum_{q \in I} \Pr[|F| = q] \cdot \left(\frac{q}{n_2}\right)^{k-1} \\
&\geq \sum_{q \in I} \Pr[|F| = q] \cdot (1 - e^{-1/\alpha} - c_1)^{k-1} \\
&= \Pr[|F| \in I] \cdot (1 - e^{-1/\alpha} - c_1)^{k-1} \\
&> (1 - o(1))\left((1 - e^{-1/\alpha})^{k-1} - \frac{c_2}{2}\right),
\end{aligned}$$

where the last inequality follows from (3.3). Thus, we can conclude that $\Pr[a \in A_1] > (1 - e^{-1/\alpha})^{k-1} - c_2$ for sufficiently large n_2. On the other hand, for the upper bound of $\Pr[a \in A_1]$, we have

$$\begin{aligned}
\Pr[a \in A_1] &\le \sum_{q \in I} \Pr[|F| = q] \cdot \Pr\Big[a \in A_1 \Big| |F| = q\Big] + \sum_{q \notin I} \Pr[|F| = q] \\
&= \sum_{q \in I} \Pr[|F| = q] \cdot \left(\frac{q}{n_2}\right)^{k-1} + o(1) \\
&\le \sum_{q \in I} \Pr[|F| = q] \cdot (1 - e^{-1/\alpha} + c_1)^{k-1} + o(1) \\
&= \Pr[|F| \in I] \cdot (1 - e^{-1/\alpha} + c_1)^{k-1} + o(1) \\
&< (1 - o(1))\Big((1 - e^{-1/\alpha})^{k-1} + \frac{c_2}{2}\Big) + o(1),
\end{aligned}$$

where the last inequality follows from (3.4). Thus, we can conclude that $\Pr[a \in A_1] < (1 - e^{-1/\alpha})^{k-1} + c_2$ for sufficiently large n_2.

Therefore,

$$(1 - e^{-1/\alpha})^{k-1} - c_2 < \Pr[a \in A_1] < (1 - e^{-1/\alpha})^{k-1} + c_2,$$

which is equivalent to

$$1 - (1 - e^{-1/\alpha})^{k-1} - c_2 < \Pr[a \in A_2] < 1 - (1 - e^{-1/\alpha})^{k-1} + c_2.$$

□

Finally, the following lemma shows that the ratio $\frac{|A_2|}{n_1}$ lies around a constant $1 - (1 - e^{-1/\alpha})^{k-1}$ with high probability.

Lemma 8. In a random instance with strict and k-incomplete preference lists,

$$1 - (1 - e^{-1/\alpha})^{k-1} - c_3 < \frac{|A_2|}{n_1} < 1 - (1 - e^{-1/\alpha})^{k-1} + c_3$$

with probability $1 - o(1)$ for any constant $c_3 > 0$.

Proof. If $k = 1$, then we have $P_a \subseteq F$ for every $a \in A$, which means $|A_2| = 0$ and thus the lemma holds. From now on, we will consider the case that $k \ge 2$.

Let $c_3 > 0$ be any constant. We can select a sufficiently small c_2 such that $c_2(1 + (1 - e^{-1/\alpha})^{k-1} + c_2) < c_3$ and thus

$$(1 - c_2)\Big((1 - e^{-1/\alpha})^{k-1} - c_2\Big) > (1 - e^{-1/\alpha})^{k-1} - c_3; \tag{3.5}$$

$$(1 + c_2)\Big((1 - e^{-1/\alpha})^{k-1} + c_2\Big) < (1 - e^{-1/\alpha})^{k-1} + c_3; \tag{3.6}$$

From Lemma 7, we have

$$1 - (1 - e^{-1/\alpha})^{k-1} - c_2 < \Pr[a \in A_2] < 1 - (1 - e^{-1/\alpha})^{k-1} + c_2 \tag{3.7}$$

for sufficiently large n_2.

For each $a \in A$, define an indicator random variable X_a such that

$$X_a = \begin{cases} 1, & \text{for } a \in A_2; \\ 0, & \text{for } a \notin A_2. \end{cases}$$

Note that $|A_2| = \sum_{a \in A} X_a$. From (3.7), we have

$$1 - (1 - e^{-1/\alpha})^{k-1} - c_2 < \mathbb{E}[X_a] < 1 - (1 - e^{-1/\alpha})^{k-1} + c_2$$

for each $a \in A$, and from the linearity of expectation we also have

$$\left(1 - (1 - e^{-1/\alpha})^{k-1} - c_2\right) n_1 < \mathbb{E}[|A_2|] < \left(1 - (1 - e^{-1/\alpha})^{k-1} + c_2\right) n_1. \tag{3.8}$$

Since X_a and $X_{a'}$ are independent for any pair of distinct $a, a' \in A$, we have

$$\mathrm{Var}[|A_2|] = \sum_{a \in A} \mathrm{Var}[X_a] = \sum_{a \in A} \left(\mathbb{E}[X_a^2] - \mathbb{E}[X_a]^2\right) \le \sum_{a \in A} \mathbb{E}[X_a^2] = \sum_{a \in A} \mathbb{E}[X_a] = \mathbb{E}[A_2].$$

Then, from Chebyshev's inequality and (3.8) we have

$$\Pr\left[\left||A_2| - \mathbb{E}[|A_2|]\right| \ge c_2 \cdot \mathbb{E}[|A_2|]\right] \le \frac{\mathrm{Var}[|A_2|]}{(c_2 \cdot \mathbb{E}[|A_2|])^2} \le \frac{1}{c_2^2 \cdot \mathbb{E}[|A_2|]} = O(1/n_1).$$

This implies $(1 - c_2)\mathbb{E}[|A_2|] \le |A_2| \le (1 + c_2)\mathbb{E}[|A_2|]$ with probability $1 - O(1/n_1) = 1 - o(1)$. Therefore, from (3.5), (3.6), and (3.8) we can conclude that

$$1 - (1 - e^{-1/\alpha})^{k-1} - c_3 < \frac{|A_2|}{n_1} < 1 - (1 - e^{-1/\alpha})^{k-1} + c_3$$

with probability $1 - o(1)$. □

3.3.3 Proof of Inequalities (3.3) and (3.4)

For $k \ge 2$, we will prove that $c_1 = \frac{c_2}{(k-1)(c_2+2)}$ satisfies inequalities (3.3) and (3.4).

Let $p = 1 - e^{-1/\alpha}$. We have $0 < p < 1$ and $0 < c_1 < 1$. So,

$$\begin{aligned}
(p - c_1)^{k-1} &= p^{k-1} - \binom{k-1}{1} p^{k-2} c_1 + \binom{k-1}{2} p^{k-3} c_1^2 - \cdots + (-1)^{k-1} \binom{k-1}{k-1} c_1^{k-1} \\
&\ge p^{k-1} - \left[(k-1)c_1 + (k-1)^2 c_1^2 + \cdots + (k-1)^{k-1} c_1^{k-1}\right] \\
&= p^{k-1} - \left[\frac{c_2}{c_2+2} + \left(\frac{c_2}{c_2+2}\right)^2 + \cdots + \left(\frac{c_2}{c_2+2}\right)^{k-1}\right] \\
&> p^{k-1} - \left[\frac{c_2}{c_2+2} + \left(\frac{c_2}{c_2+2}\right)^2 + \cdots\right] \\
&= p^{k-1} - \frac{\frac{c_2}{c_2+2}}{1 - \frac{c_2}{c_2+2}} \\
&= p^{k-1} - \frac{c_2}{2}.
\end{aligned}$$

Therefore $(1-e^{-1/\alpha}-c_1)^{k-1} > (1-e^{-1/\alpha})^{k-1} - \frac{c_2}{2}$. Also, we have

$$\begin{aligned}
(p+c_1)^{k-1} &= p^{k-1} + \binom{k-1}{1} p^{k-2} c_1 + \binom{k-1}{2} p^{k-3} c_1^2 + \cdots + \binom{k-1}{k-1} c_1^{k-1} \\
&\leq p^{k-1} + (k-1)c_1 + (k-1)^2 c_1^2 + \cdots + (k-1)^{k-1} c_1^{k-1} \\
&= p^{k-1} + \frac{c_2}{c_2+2} + \left(\frac{c_2}{c_2+2}\right)^2 + \cdots + \left(\frac{c_2}{c_2+2}\right)^{k-1} \\
&< p^{k-1} + \frac{c_2}{c_2+2} + \left(\frac{c_2}{c_2+2}\right)^2 + \cdots \\
&= p^{k-1} + \frac{\frac{c_2}{c_2+2}}{1-\frac{c_2}{c_2+2}} \\
&= p^{k-1} + \frac{c_2}{2}.
\end{aligned}$$

Therefore $(1-e^{-1/\alpha}+c_1)^{k-1} > (1-e^{-1/\alpha})^{k-1} + \frac{c_2}{2}$.

3.4 Phase Transition

For each value of $k \geq 1$, we want to find a phase transition point α_k such that if $\alpha > \alpha_k$, then a popular matching exists with high probability; and if $\alpha < \alpha_k$, then a popular matching exists with low probability. We do so by proving the upper bound and lower bound separately.

3.4.1 Upper Bound

Lemma 9. Suppose that $0 \leq \beta < \alpha e^{-1/2\alpha}$. Then, $G'(n_2, h, \beta n_1, (1-\beta)n_1)$ contains a complex component with probability $O(1/n_1)$ for every fixed integer $h \in [e^{-1/\alpha}n_2 - n_2^{2/3}, e^{-1/\alpha}n_2 + n_2^{2/3}]$.

Proof. By the definition of $G'(n_2, h, \beta n_1, (1-\beta)n_1)$, each vertex in U' has degree at most one, thus removing U' does not affect the existence of a complex component. Moreover, the graph $G'(n_2, h, \beta n_1, (1-\beta)n_1)$ with part U' removed has exactly the same distribution as $G(n_2, h, \beta n_1)$ given in Definition 9. Therefore, it is sufficient to consider the graph $G(n_2, h, \beta n_1)$ instead.

Using the same technique as in Mahdian's proof of [37, Lemma 4], define a *minimal bad graph* to be two vertices joined by three vertex-disjoint paths, or two vertex-disjoint cycles joined by a path which is also vertex-disjoint from the two cycles except at both endpoints (the path can be degenerate, which is the only exception that the two cycles share a vertex). Note that any proper subgraph of a minimal bad graph does not contain a complex component, and every graph that contains a complex component must contain a minimal bad graph as a subgraph.

Let X and Y be subsets of vertices of $G(n_2, h, \beta n_1)$ in V and U, respectively. Define $BAD_{X,Y}$ to be an event that $X \cup Y$ contains a minimal bad graph as a *spanning* subgraph. Then, let $p_1 = |X|$, $p_2 = |Y|$, and $p = p_1 + p_2$. Observe that $BAD_{X,Y}$ can occur only when $|p_1 - p_2| \leq 1$, so $p_1, p_2 \geq \frac{p-1}{2}$. Also, there are at most $2p^2$ non-isomorphic minimal bad graphs with p_1 vertices in V and p_2 vertices in U, with each of them having $p_1!p_2!$ ways to arrange the vertices, and there are at most $(p+1)!\binom{\beta n_1}{p+1}\left(\frac{1}{n_2 h}\right)^{p+1}$ probability that all $p+1$ edges of each graph are selected in our random procedure. By the union bound, the probability of $BAD_{X,Y}$ is at most

$$2p^2 p_1! p_2! (p+1)! \binom{\beta n_1}{p+1} \left(\frac{1}{n_2 h}\right)^{p+1} \leq 2p^2 p_1! p_2! \left(\frac{\beta n_1}{n_2 h}\right)^{p+1}.$$

Again, by the union bound, the probability that at least one $BAD_{X,Y}$ occurs is at most

$$\begin{aligned}
\Pr\left[\bigvee_{X,Y} BAD_{X,Y}\right] &\le \sum_{p_1,p_2} \binom{n_2}{p_1}\binom{h}{p_2} 2p^2 p_1! p_2! \left(\frac{\beta n_1}{n_2 h}\right)^{p+1} \\
&\le \sum_{p_1,p_2} \frac{n_2^{p_1}}{p_1!} \cdot \frac{h^{p_2}}{p_2!} \cdot 2p^2 p_1! p_2! \left(\frac{\beta}{\alpha h}\right)^{p+1} \\
&= \sum_{p_1,p_2} \frac{2p^2}{h} \left(\frac{\beta}{\alpha}\right)^{p+1} \left(\frac{n_2}{h}\right)^{p_1} \\
&\le \sum_{p=1}^{\infty} \frac{O(p^2)}{n_1} \left(\frac{\beta}{\alpha}\right)^{p} \left(e^{-1/\alpha} - n_2^{-1/3}\right)^{-p/2} \\
&= \frac{O(1)}{n_1} \sum_{p=1}^{\infty} p^2 \left(\frac{\alpha^2}{\beta^2}\left(e^{-1/\alpha} - n_2^{-1/3}\right)\right)^{-p/2}.
\end{aligned}$$

By the assumption, we have $\alpha^2 e^{-1/\alpha} > \beta^2$, so $\frac{\alpha^2}{\beta^2}(e^{-1/\alpha} - n_2^{-1/3}) > 1$ for sufficiently large n_2, hence the above sum converges. Therefore, the probability that at least one $BAD_{X,Y}$ happens is at most $O(1/n_1)$. □

We can now prove the following theorem, which serves as an upper bound of α_k.

Theorem 3. In a random instance with strict and k-incomplete preference lists, if $\alpha e^{-1/2\alpha} > 1 - (1 - e^{-1/\alpha})^{k-1}$, then a popular matching exists with probability $1 - o(1)$.

Proof. Since $\alpha e^{-1/2\alpha} > 1 - (1 - e^{-1/\alpha})^{k-1}$, we can select a sufficiently small $\delta_1 > 0$ such that $\alpha e^{-1/2\alpha} > 1 - (1 - e^{-1/\alpha})^{k-1} + \delta_1$. Let

$$J_1 = [(1 - (1 - e^{-1/\alpha})^{k-1} - \delta_1)n_1, (1 - (1 - e^{-1/\alpha})^{k-1} + \delta_1)n_1].$$

From Lemma 8, $|A_2| \in J_1$ with probability $1 - o(1)$. Moreover, we have $\beta = \frac{t}{n_1} < \alpha e^{-1/2\alpha}$ for any integer $t \in J_1$.

Define E_1 to be an event that a popular matching exists in a random instance. First, consider the probability of E_1 conditioned on $|A_2| = t$ for each fixed integer $t \in J_1$. By Lemmas 5 and 9, the top-choice graph contains a complex component with probability $O(n_1^{-1/3}) = o(1)$. Therefore, from Lemmas 2 and 4 we can conclude that a popular matching exists with probability $1 - o(1)$, i.e. $\Pr\left[E_1 \middle| |A_2| = t\right] = 1 - o(1)$ for every fixed integer $t \in J_1$. So

$$\begin{aligned}
\Pr[E_1] &= \sum_{t} \Pr[|A_2| = t] \cdot \Pr\left[E_1 \middle| |A_2| = t\right] \\
&\ge \sum_{t \in J_1} \Pr[|A_2| = t] \cdot \Pr\left[E_1 \middle| |A_2| = t\right] \\
&\ge \Pr[|A_2| \in J_1] \cdot (1 - o(1)) \\
&= (1 - o(1))(1 - o(1)) \\
&= 1 - o(1).
\end{aligned}$$

Hence, a popular matching exists with probability $1 - o(1)$. □

3.4.2 Lower Bound

Lemma 10. Suppose that $\alpha e^{-1/2\alpha} < \beta \le 1$. Then, $G'(n_2, h, \beta n_1, (1-\beta)n_1)$ does not contain a complex component with probability $O(1/n_1)$ for every fixed integer $h \in [e^{-1/\alpha} n_2 - n_2^{2/3}, e^{-1/\alpha} n_2 + n_2^{2/3}]$.

Proof. Again, by the same reasoning as in the proof of Lemma 9, we can consider the graph $G(n_2, h, \beta n_1)$ instead of $G'(n_2, h, \beta n_1, (1-\beta)n_1)$, but now we are interested in an event that $G(n_2, h, \beta n_1)$ does not contain a complex component.

Since $\alpha e^{-1/2\alpha} < \beta$, we have $\alpha e^{-1/2\alpha} < (1-\epsilon)^{3/2}\beta$ for a sufficiently small $\epsilon > 0$. Consider the random bipartite graph $G(n_2, h, (1-\epsilon)\beta n_1)$ with parts V having n_2 vertices and U having h vertices. For each vertex v, let a random variable r_v be the degree of v. Since there are $(1-\epsilon)\beta n_1$ edges in the graph, the expected value of r_v for each $v \in V$ is

$$c_1 = \frac{(1-\epsilon)\beta n_1}{n_2} = \frac{(1-\epsilon)\beta}{\alpha}.$$

Since $e^{-1/\alpha} n_2 + n_2^{2/3} < \frac{e^{-1/\alpha} n_2}{1-\epsilon}$ for sufficiently large n_2, the expected value of r_v for each $v \in U$ is

$$c_2 = \frac{(1-\epsilon)\beta n_1}{h} > \frac{(1-\epsilon)\beta n_1}{e^{-1/\alpha} n_2 + n_2^{2/3}} > \frac{(1-\epsilon)\beta n_1}{e^{-1/\alpha} n_2/(1-\epsilon)} = \frac{(1-\epsilon)^2\beta}{\alpha e^{-1/\alpha}}$$

for sufficiently large n_2. Furthermore, each r_v has a binomial distribution, which converges to Poisson distribution when n_2 increases to infinity. The graph can be viewed as a special case of an *inhomogeneous random graph* [9, 50], which is a generalization of an Erdős-Rényi graph, where vertices of the graph are divided into several (finite or infinite) types. Each vertex of type i has κ_{ij} expected neighbors of type j.

The bipartite graph $G(n_2, h, (1-\epsilon)\beta n_1)$ can be considered as a special case of the inhomogeneous random graph where there are two types of vertices, with $\kappa_{11} = 0$, $\kappa_{12} = c_1$, $\kappa_{21} = c_2$, and $\kappa_{22} = 0$. It has an *offspring matrix*

$$T_\kappa = \{\kappa_{ij}\}_{i,j=1}^2 = \begin{pmatrix} 0 & c_1 \\ c_2 & 0 \end{pmatrix},$$

which has the largest eigenvalue $\|T_\kappa\| = \sqrt{c_1 c_2} > 1$. This is a necessary and sufficient condition to conclude that $G(n_2, h, (1-\epsilon)\beta n_1)$ contains a *giant component* (a component containing a constant fraction of vertices of the entire graph) with $1 - o(1)$ probability [9, 50]. In fact, by giving a precise bound in each step of [9], it is possible to show that the probability is greater than $1 - O(1/n_1)$ as desired.

Alternatively, we hereby show a direct proof of the bipartite case by approximating the construction of the graph with the Galton-Watson branching process (shown in Section 2.4) similar to that in the proof of existence of a giant component in the Erdős-Rényi graph in [7, pp.182–192].

Consider the construction of $G(n_2, h, (1-\epsilon)\beta n_1)$ with parts V and U starting at a vertex and discovering new vertices in a breadth-first search tree manner. We approximate it with the Galton-Watson branching process. Let T be the size of the process ($T = \infty$ if the process continues forever). Let z_1 and z_2 be the probability that $T < \infty$ when starting the process at a vertex in V and U, respectively. Also, let Z_1 and Z_2 be the number of children the root has when starting the process at a vertex in V and U, respectively.

Given that the root has i children, in order for the branching process to be finite, all of the i branches must be finite, so we get the equations.

$$z_1 = \sum_{i=0}^{\infty} \Pr[Z_1 = i] z_2^i;$$
$$z_2 = \sum_{i=0}^{\infty} \Pr[Z_2 = i] z_1^i.$$

Therefore,

$$z_1 = \sum_{i=0}^{\infty} \frac{c_1^i e^{-c_1}}{i!} \left(\sum_{j=0}^{\infty} \frac{c_2^j e^{-c_2} z_1^j}{j!} \right)^i = \sum_{i=0}^{\infty} \frac{c_1^i e^{-c_1}}{i!} e^{c_2(z_1-1)i} = e^{c_1(e^{c_2(z_1-1)}-1)}.$$

Setting $y = 1 - z_1$ yields the equation

$$1 - y = e^{c_1(e^{-c_2 y}-1)}. \tag{3.9}$$

Define

$$g(y) = 1 - y - e^{c_1(e^{-c_2 y}-1)}.$$

We have $g(0) = 1 - 0 - 1 = 0$, $g(1) < 0$, and $g'(0) = c_1 c_2 - 1$. By the assumption that $c_1 c_2 > 1$, we have $g'(0) > 0$, so there must be $y \in (0,1)$ such that $g(y) = 0$, thus being a solution of (3.9). So, $\Pr[T = \infty] = y \in (0,1)$, when y is a solution of (3.9), meaning that there is a constant probability that the process continues indefinitely. Moreover, from the property of Poisson distribution we can show that $\Pr[x < T < \infty]$ is exponentially low in term of x. Therefore, we can select a constant k_1 such that $\Pr[k_1 \log n_1 < T < \infty] < O(1/n_1^2)$.

Finally, when we perform the Galton-Watson branching process at a vertex in $G(n_2, h, (1-\epsilon)\beta n_1)$, there is a constant probability that the process will continue indefinitely, thus creating a giant component. Otherwise, with probability $1 - O(1/n_1^2)$ we will create a component with size smaller than $k_1 \log n_1$, so we can remove that component from the graph and then repeatedly perform the process starting at a new vertex. After repeatedly performing this process for some logarithmic number of times, we only remove $O(\log^2 n_1)$ vertices from the graph, which does not affect the constant $y = \Pr[T = \infty]$, so the probability that we never end up with a giant component in every time is at most $O(1/n_1)$. Therefore, $G(n_2, h, (1-\epsilon)\beta n_1)$ contains a giant component with probability $1 - O(1/n_1)$. □

Remark 2. In the complete preference lists setting with $\alpha e^{-1/2\alpha} < (1-\epsilon)^{3/2}$, we have $c_1 = \frac{1-\epsilon}{\alpha}$ and $c_2 > \frac{(1-\epsilon)^2}{\alpha e^{-1/\alpha}}$, which we still get $c_1 c_2 = \frac{(1-\epsilon)^3}{\alpha^2 e^{-1/\alpha}} > 1$, which is a sufficient condition to reach the same conclusion.

We can now prove the following theorem, which serves as a lower bound of α_k.

Theorem 4. In a random instance with strict and k-incomplete preference lists, if $\alpha e^{-1/2\alpha} < 1-(1-e^{-1/\alpha})^{k-1}$, then a popular matching exists with probability $o(1)$.

Proof. Like in the proof of Theorem 3, we can select a sufficiently small $\delta_2 > 0$ such that $\alpha e^{-1/2\alpha} < 1-(1-e^{-1/\alpha})^{k-1}-\delta_2$. Let

$$J_2 = [(1-(1-e^{-1/\alpha})^{k-1}-\delta_2)n_1, (1-(1-e^{-1/\alpha})^{k-1}+\delta_2)n_1].$$

We have $|A_2| \in J_2$ with probability $1-o(1)$ and $\beta = \frac{t}{n_1} > \alpha e^{-1/2\alpha}$ for any integer $t \in J_2$.

Now we define E_2 to be an event that a popular matching does not exist in a random instance. By the same reasoning as in the proof of Theorem 3, we can prove that $\Pr\left[E_2 \middle| |A_2| = t\right] = 1-o(1)$ for every fixed $t \in J_2$ and reach an analogous conclusion that $\Pr[E_2] = 1-o(1)$. □

3.4.3 Phase Transition Point

Since $f(x) = xe^{-1/2x} - (1-(1-e^{-1/x})^{k-1})$ is a strictly increasing function in $[1,\infty)$ for every $k \geq 1$, $f(x) = 0$ can have at most one root in $[1,\infty)$. That root, if exists, will serve as a phase transition point α_k. In fact, for $k \geq 4$, $f(x) = 0$ has a unique solution in $[1,\infty)$; for $k \leq 3$, $f(x) = 0$ has no solution in $[1,\infty)$ and $\alpha e^{-1/2\alpha} > 1-(1-e^{-1/\alpha})^{k-1}$ for every $\alpha \geq 1$, so a popular matching always exists with high probability without a phase transition regardless of value of α. Therefore, from Theorems 3 and 4 we can conclude our main theorem below.

Theorem 5. In a random instance with strict and k-incomplete preference lists with $k \geq 4$, if $\alpha > \alpha_k$, where $\alpha_k \geq 1$ is the root of equation $xe^{-1/2x} = 1-(1-e^{-1/x})^{k-1}$, then a popular matching exists with probability $1-o(1)$; and if $\alpha < \alpha_k$, then a popular matching exists with probability $o(1)$. In such random instance with $k \leq 3$, a popular matching exists with probability $1-o(1)$ for any value of $\alpha \geq 1$.

For each value of $k \geq 4$, the phase transition occurs at the root $\alpha_k \geq 1$ of equation $xe^{-1/2x} = 1-(1-e^{-1/x})^{k-1}$ as shown in Table 3.1 and Figure 3.1. Note that as k increases, the right-hand side of the equation converges to 1, hence α_k converges to Mahdian's value of $\alpha_* \approx 1.42$ in the complete preference lists setting.

k	4	5	6	7	8	9	10	...
α_k	1.2428	1.3411	1.3835	1.4031	1.4124	1.4170	1.4193	...

Table 3.1: Approximate value of α_k for each integer $k \geq 4$

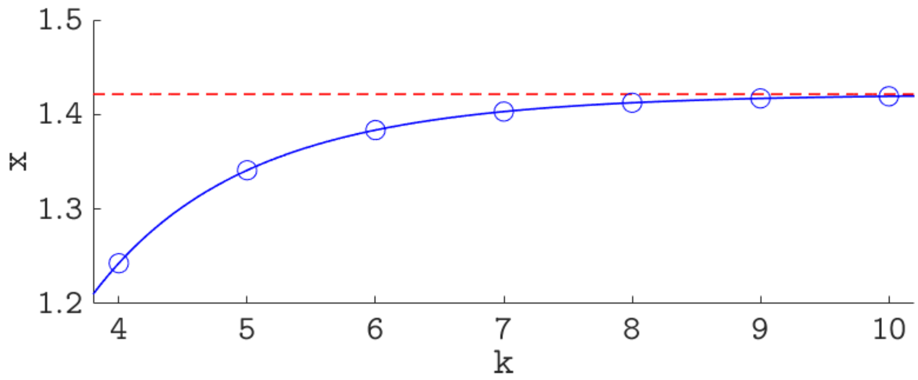


Figure 3.1: Solution in $[1, \infty)$ of the equation $xe^{-1/2x} = 1 - (1 - e^{-1/x})^{k-1}$ for each $k \geq 4$, with the dashed line plotting $x = \alpha_* \approx 1.42$

Remark 3. For each person a, as the size of P_a increases, the probability that $P_a \not\subseteq F$ increases and thus the probability that $a \in A_2$ also increases, and so do the expected size of A_2 and the phase transition point. Therefore, in the setting where the lengths of people's preference lists are fixed but not equal (e.g. half of the people have preference lists with length k_1, and another half have those with length k_2), the phase transition will occur between $\alpha_{k_{\min}}$ and $\alpha_{k_{\max}}$, where $k_{\min}$ and $k_{\max}$ are the shortest and longest lengths of people's preference lists, respectively.

3.5 Results from Simulation

We perform a simulation in the following procedure. For each integer k in the range from 2 to 8, we set α to be a sequence of values ranging from 1.0 to 1.6, with n_1 increasing exponentially from 10 to 1,000,000. For each combination of parameters, we generate 100 random instances and count the number of instances with a popular matching

Note that the case $k = 1$ is trivial as a popular matching always exists in every instance, since for each item $b \in F$ we can simply match b to any person $a \in A$ such that $f(a) = b$; it can be easily proved that this matching is popular. Hence, we start the simulation from the case $k = 2$.

To determine whether a popular matching exists in a given instance, we implement the following **DecideAPerfectMatching** algorithm developed by Abraham et al. [5] to determine the existence of an A-perfect matching (which is equivalent to the existence of a popular matching). This algorithm takes as input a graph G that has $A \cup B \cup L$ as a set of vertices and $E = \{(a, f(a)) | a \in A\} \cup \{(a, s(a)) | a \in A\}$ as a set of edges. It returns YES if an A-perfect matching exists and NO if an A-perfect matching does not exist.

DecideAPerfectMatching($G = (A \cup B \cup L, E)$)
    while some item $b \in B \cup L$ has degree 1
        $a :=$ unique person matched to b
        $G := G - \{a, b\}$
    while some item $b \in B \cup L$ has degree 0
        $G := G - \{b\}$
    if $|A| > |B \cup L|$ **then**
        return NO
    else
        return YES

3.5.1 Case $k = 2$

For $k = 2$, there is no phase transition point. Results from the simulation are shown in Table 3.2 and Figure 3.2. As n_1 grows up, the number of instances with a popular matching stays at or close to 100 for every value of α.

$k = 2$						
α	n_1					
	10^1	10^2	10^3	10^4	10^5	10^6
1.0	96	100	100	100	100	100
1.1	99	100	100	100	100	100
1.2	100	100	100	100	100	100
1.3	100	99	100	100	100	100
1.4	100	100	99	100	100	100
1.5	99	100	100	100	100	100
1.6	100	100	100	100	100	100

Table 3.2: Number of instances with a popular matching in the case $k = 2$

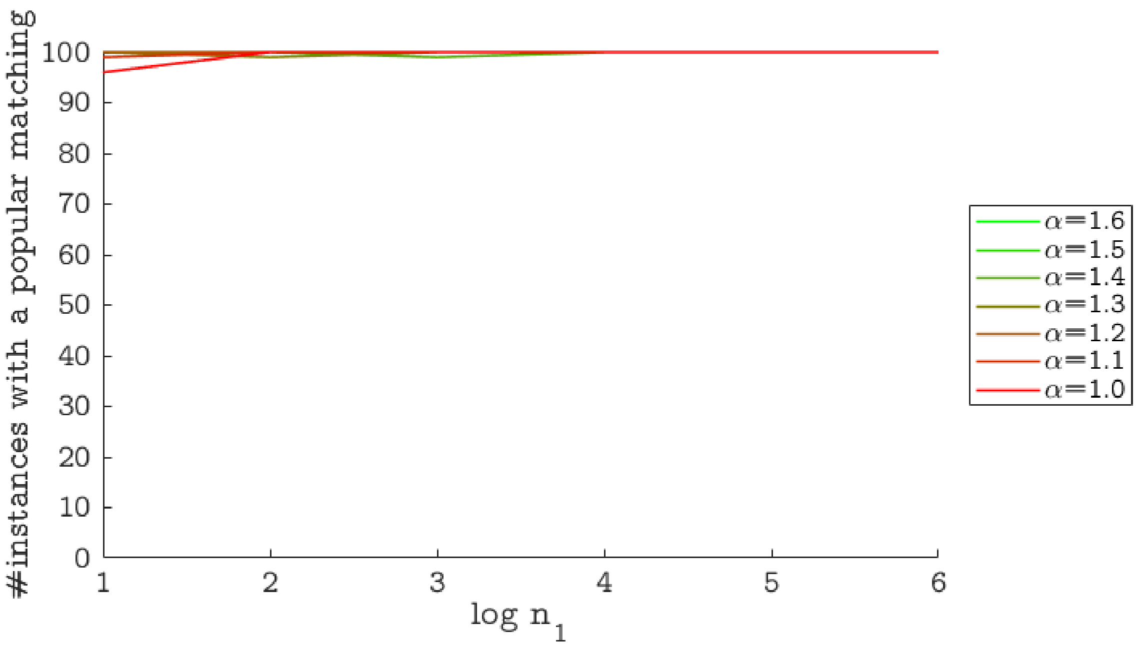


Figure 3.2: Number of instances with a popular matching in the case $k = 2$, with each line plotting the results from each value of α

3.5.2 Case $k = 3$

For $k = 3$, there is no phase transition point. Results from the simulation are shown in Table 3.3 and Figure 3.3. As n_1 grows up, the number of instances with a popular matching increases to near 100 for every value of α.

$k = 3$						
α	n_1					
	10^1	10^2	10^3	10^4	10^5	10^6
1.0	90	86	84	87	90	95
1.1	94	89	95	98	100	100
1.2	99	96	100	99	100	100
1.3	100	98	100	100	100	100
1.4	98	100	100	100	100	100
1.5	100	100	100	100	100	100
1.6	98	100	100	100	100	100

Table 3.3: Number of instances with a popular matching in the case $k = 3$

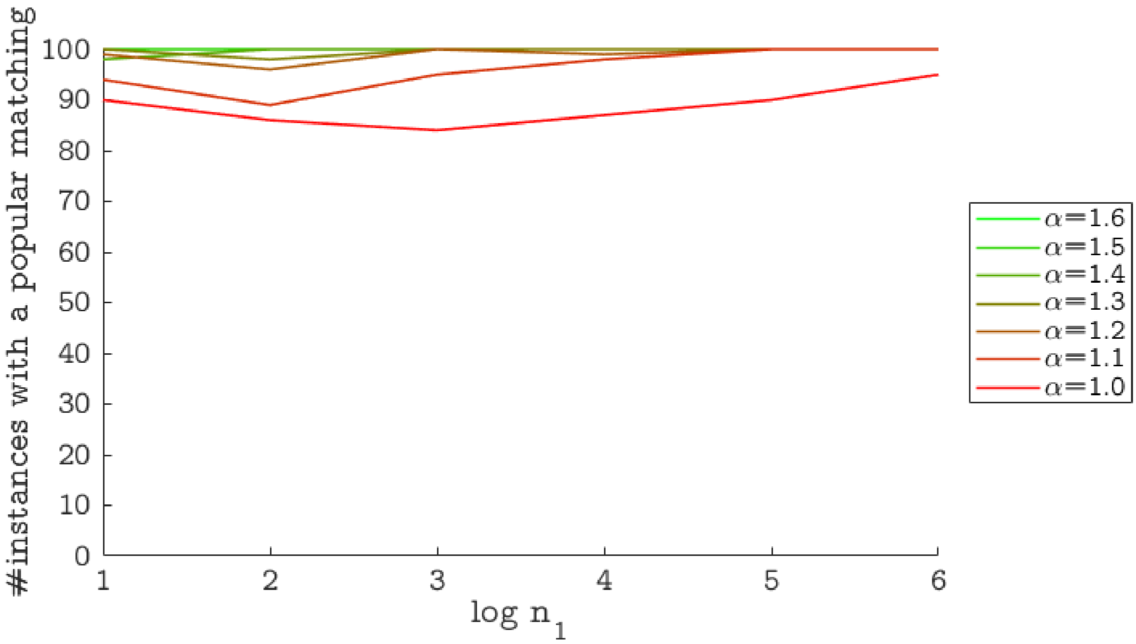


Figure 3.3: Number of instances with a popular matching in the case $k = 3$, with each line plotting the results from each value of α

3.5.3 Case $k = 4$

For $k = 4$, the phase transition point is $\alpha_4 \approx 1.2428$. Results from the simulation are shown in Table 3.4 and Figure 3.4. As n_1 grows up, the number of instances with a popular matching increases to 100 for $\alpha \geq 1.3$ and decreases to zero for $\alpha \leq 1.2$.

$k = 4, \alpha_k \approx 1.2428$						
α	n_1					
	10^1	10^2	10^3	10^4	10^5	10^6
1.0	79	52	5	0	0	0
1.1	85	65	32	0	0	0
1.2	91	84	69	53	11	0
1.3	96	92	92	94	97	100
1.4	98	95	95	100	100	100
1.5	98	98	99	100	100	100
1.6	98	98	100	100	100	100

Table 3.4: Number of instances with a popular matching in the case $k = 4$

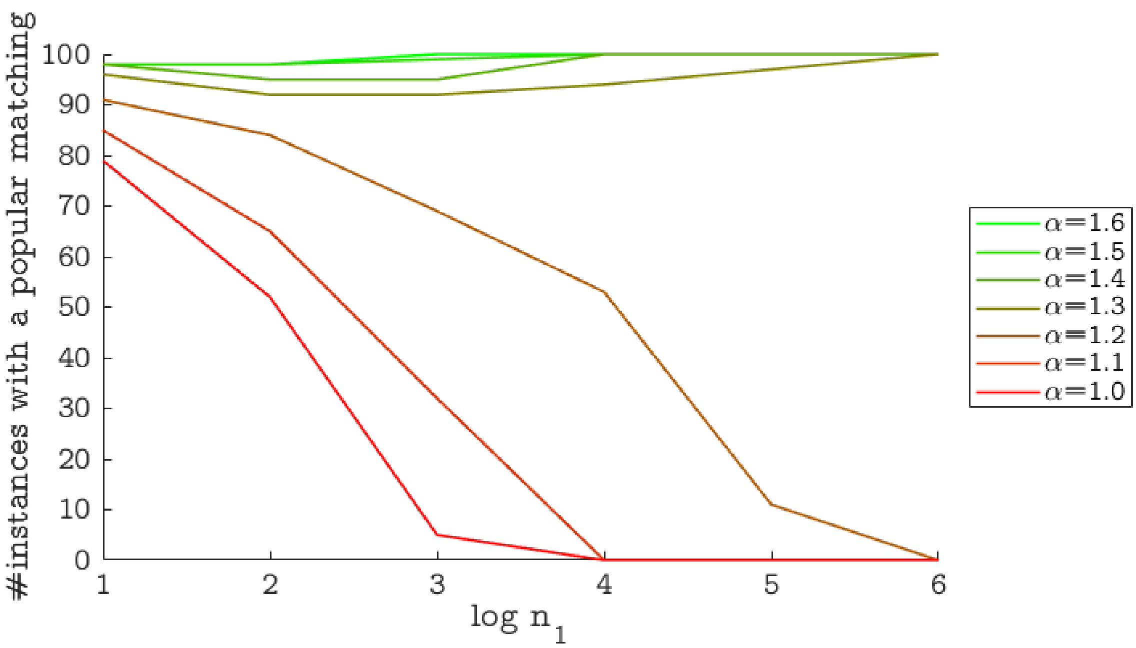


Figure 3.4: Number of instances with a popular matching in the case $k = 4$, with each line plotting the results from each value of α

Figure 3.5 shows comparison of the results from different values of α when $n_1 = 10^6$. The number rises from zero to 100 when α passes the phase transition point $\alpha_4 \approx 1.2428$.

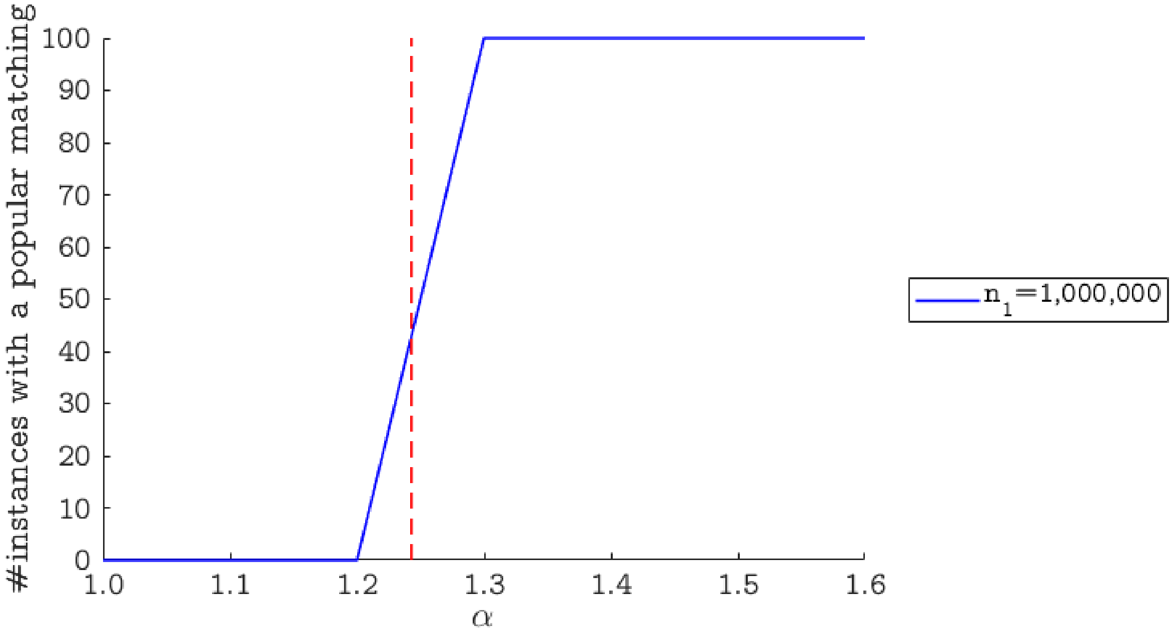


Figure 3.5: Number of instances with a popular matching in the case $k = 4$ for $n_1 = 10^6$, with the dashed line plotting $\alpha = \alpha_4 \approx 1.2428$

3.5.4 Case $k = 5$

For $k = 5$, the phase transition point is $\alpha_5 \approx 1.3411$. Results from the simulation are shown in Table 3.5 and Figure 3.6. As n_1 grows up, the number of instances with a popular matching increases to 100 for $\alpha \geq 1.4$ and decreases to zero for $\alpha \leq 1.3$.

$k = 5, \alpha_k \approx 1.3411$						
α	n_1					
	$\mathbf{10^1}$	$\mathbf{10^2}$	$\mathbf{10^3}$	$\mathbf{10^4}$	$\mathbf{10^5}$	$\mathbf{10^6}$
1.0	69	14	0	0	0	0
1.1	80	40	0	0	0	0
1.2	84	67	24	0	0	0
1.3	95	86	74	57	12	0
1.4	97	96	94	98	100	100
1.5	97	97	98	100	100	100
1.6	99	98	99	100	100	100

Table 3.5: Number of instances with a popular matching in the case $k = 5$

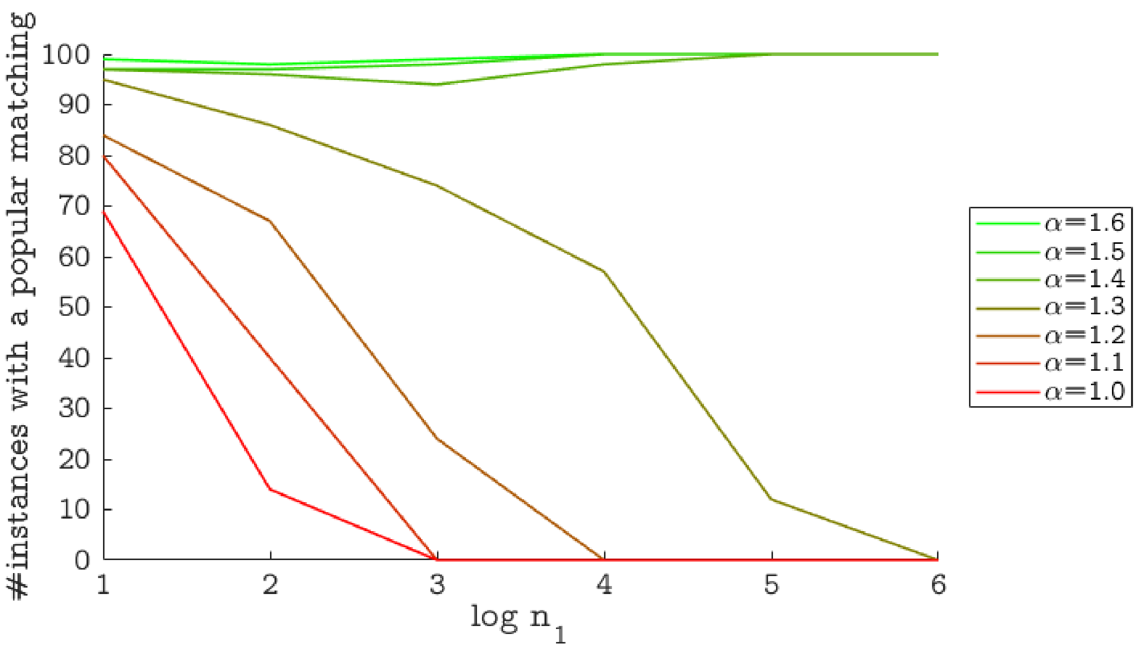


Figure 3.6: Number of instances with a popular matching in the case $k = 5$, with each line plotting the results from each value of α

Figure 3.7 shows comparison of the results from different values of α when $n_1 = 10^6$. The number rises from zero to 100 when α passes the phase transition point $\alpha_5 \approx 1.3411$.

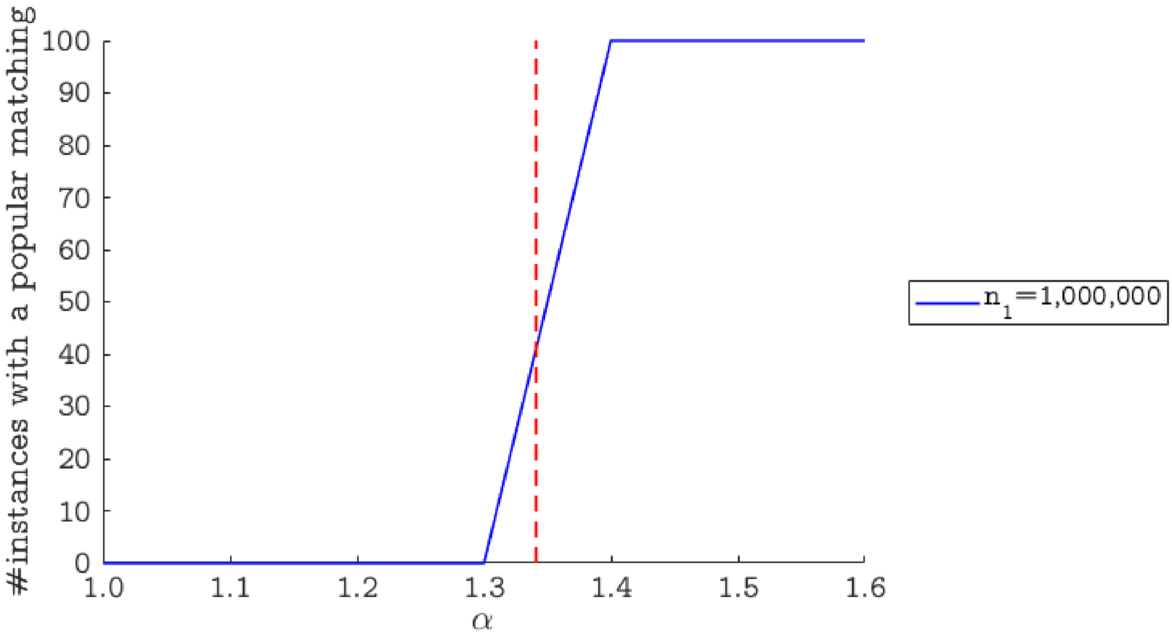


Figure 3.7: Number of instances with a popular matching in the case $k = 5$ for $n_1 = 10^6$, with the dashed line plotting $\alpha = \alpha_5 \approx 1.3411$

3.5.5 Case $k = 6$

For $k = 6$, the phase transition point is $\alpha_6 \approx 1.3835$. Results from the simulation are shown in Table 3.6 and Figure 3.8. As n_1 grows up, the number of instances with a popular matching increases to near 100 for $\alpha \geq 1.4$ and decreases to zero for $\alpha \leq 1.3$.

$k = 6, \alpha_k \approx 1.3835$						
α	n_1					
	$\mathbf{10^1}$	$\mathbf{10^2}$	$\mathbf{10^3}$	$\mathbf{10^4}$	$\mathbf{10^5}$	$\mathbf{10^6}$
1.0	57	6	0	0	0	0
1.1	75	29	0	0	0	0
1.2	84	55	12	0	0	0
1.3	95	76	52	8	0	0
1.4	97	95	92	90	91	99
1.5	98	96	98	100	100	100
1.6	98	99	100	100	100	100

Table 3.6: Number of instances with a popular matching in the case $k = 6$

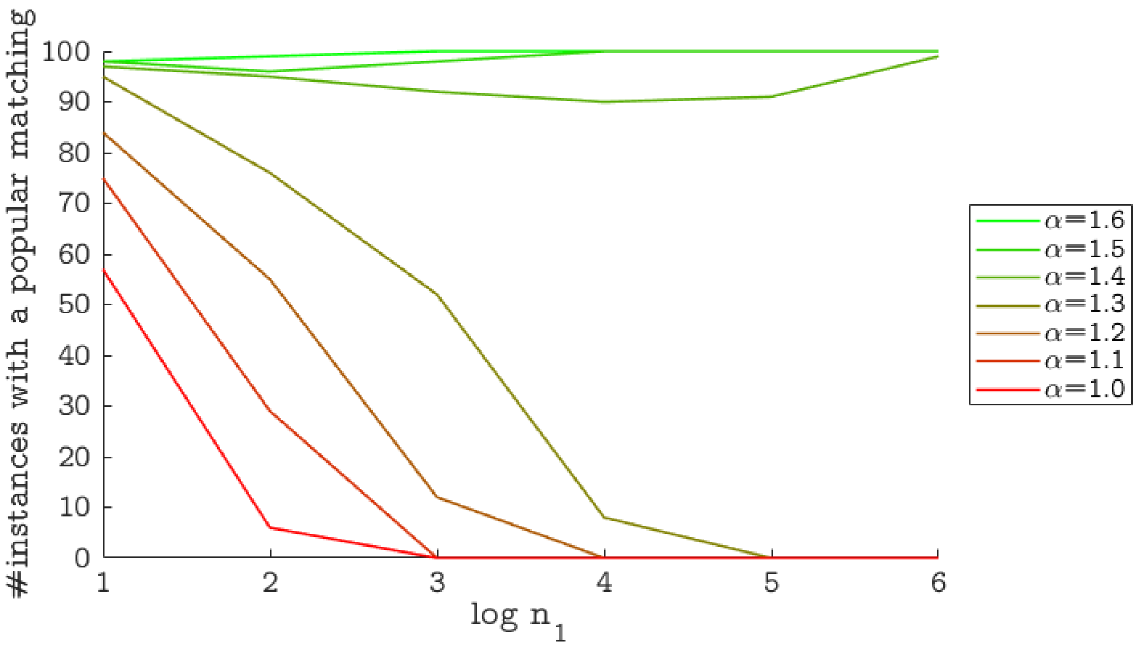


Figure 3.8: Number of instances with a popular matching in the case $k = 6$, with each line plotting the results from each value of α

Figure 3.9 shows comparison of the results from different values of α when $n_1 = 10^6$. The number rises from zero to 100 when α passes the phase transition point $\alpha_6 \approx 1.3835$.

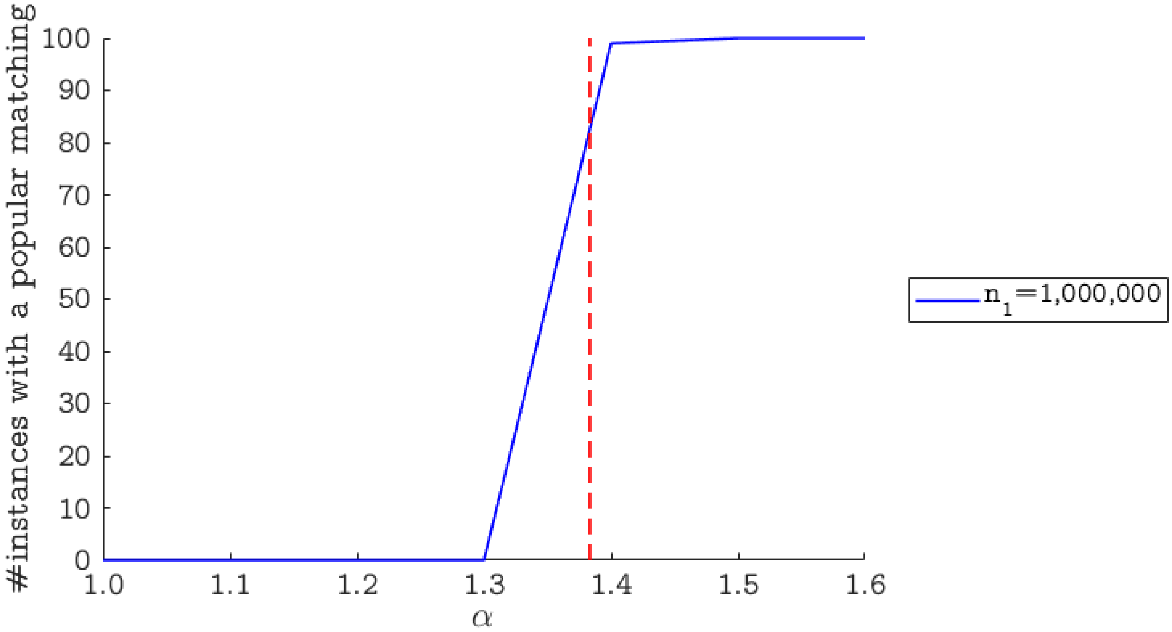


Figure 3.9: Number of instances with a popular matching in the case $k = 6$ for $n_1 = 10^6$, with the dashed line plotting $\alpha = \alpha_6 \approx 1.3835$

3.5.6 Case $k = 7$

For $k = 7$, the phase transition point is $\alpha_7 \approx 1.4031$. Results from the simulation are shown in Table 3.7 and Figure 3.10. As n_1 grows up, the number of instances with a popular matching increases to 100 for $\alpha \geq 1.5$ and decreases to zero for $\alpha \leq 1.3$. Note that $\alpha = 1.4$ lies very close to the phase transition point $\alpha_7 \approx 1.4031$, hence the number decreases relatively slowly as n_1 grows up.

$k = 7, \alpha_k \approx 1.4031$						
α	n_1					
	10^1	10^2	10^3	10^4	10^5	10^6
1.0	62	3	0	0	0	0
1.1	74	18	0	0	0	0
1.2	82	52	2	0	0	0
1.3	95	73	36	1	0	0
1.4	98	96	91	85	81	72
1.5	98	95	94	99	100	100
1.6	98	98	99	100	100	100

Table 3.7: Number of instances with a popular matching in the case $k = 7$

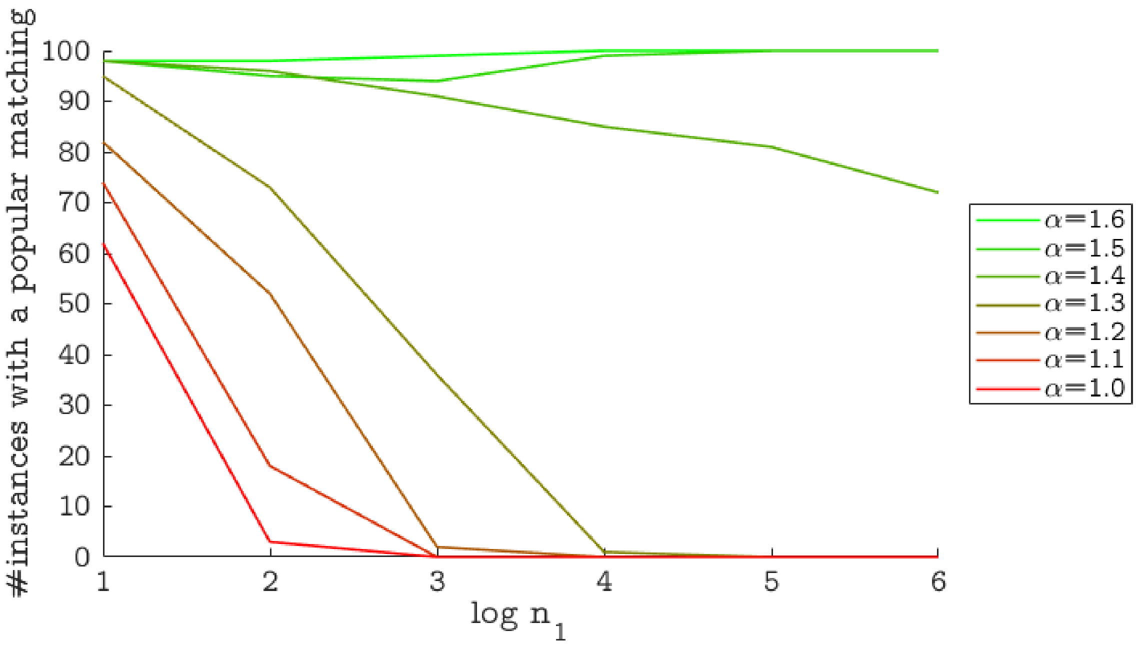


Figure 3.10: Number of instances with a popular matching in the case $k = 7$, with each line plotting the results from each value of α

Figure 3.11 shows comparison of the results from different values of α when $n_1 = 10^6$. The number rises from zero to 100 when α passes the phase transition point $\alpha_7 \approx 1.4031$.

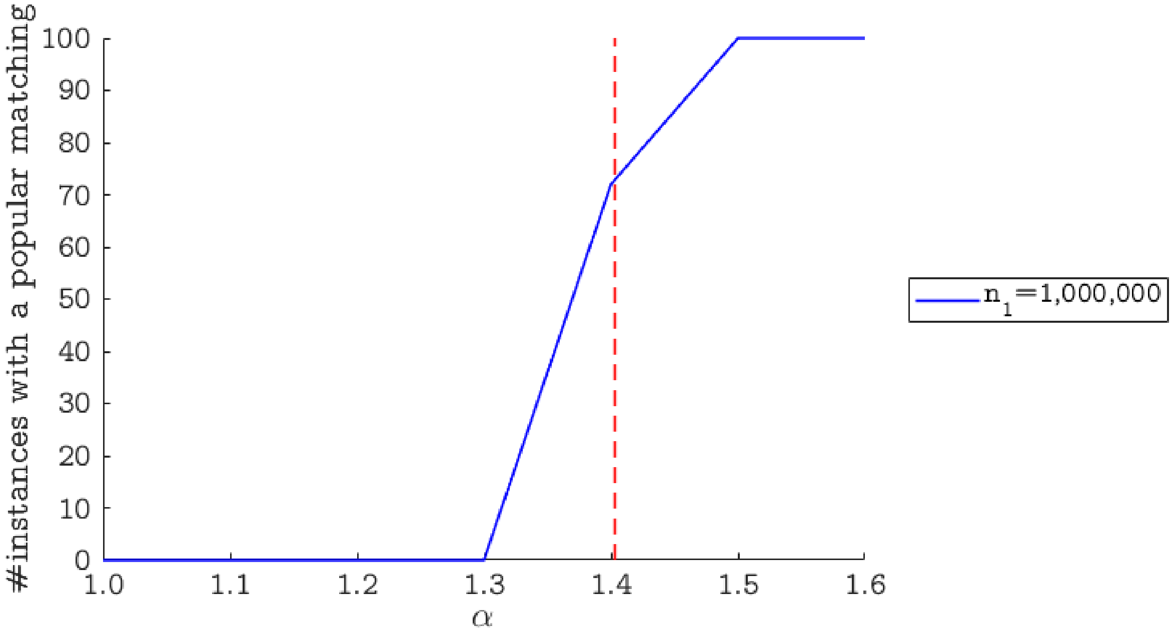


Figure 3.11: Number of instances with a popular matching in the case $k = 7$ for $n_1 = 10^6$, with the dashed line plotting $\alpha = \alpha_7 \approx 1.4031$

3.5.7 Case $k = 8$

For $k = 8$, the phase transition point is $\alpha_8 \approx 1.4124$. Results from the simulation are shown in Table 3.8 and Figure 3.12. As n_1 grows up, the number of instances with a popular matching increases to 100 for $\alpha \geq 1.5$ and decreases to near zero for $\alpha \leq 1.4$.

$k = 8, \alpha_k \approx 1.4124$						
α	n_1					
	$\mathbf{10^1}$	$\mathbf{10^2}$	$\mathbf{10^3}$	$\mathbf{10^4}$	$\mathbf{10^5}$	$\mathbf{10^6}$
1.0	52	0	0	0	0	0
1.1	77	15	0	0	0	0
1.2	89	42	0	0	0	0
1.3	96	77	28	0	0	0
1.4	97	92	83	70	58	28
1.5	97	91	95	99	99	100
1.6	98	96	99	100	100	100

Table 3.8: Number of instances with a popular matching in the case $k = 8$

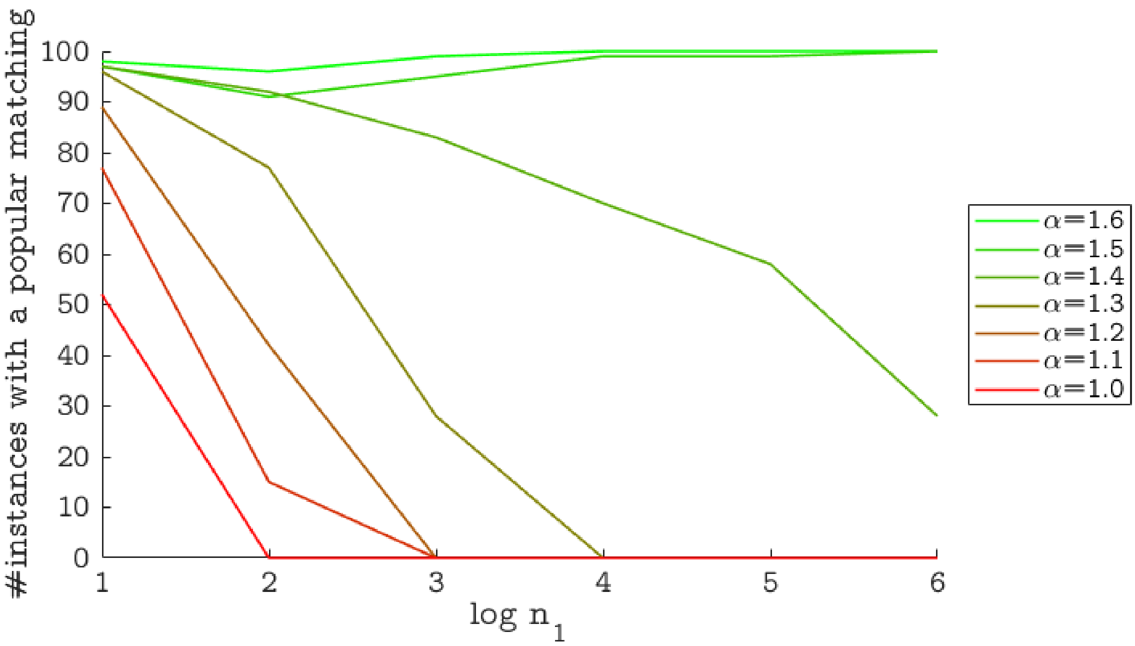


Figure 3.12: Number of instances with a popular matching in the case $k = 8$, with each line plotting the results from each value of α

Figure 3.13 shows comparison of the results from different values of α when $n_1 = 10^6$. The number rises from zero to 100 when α passes the phase transition point $\alpha_8 \approx 1.4124$.

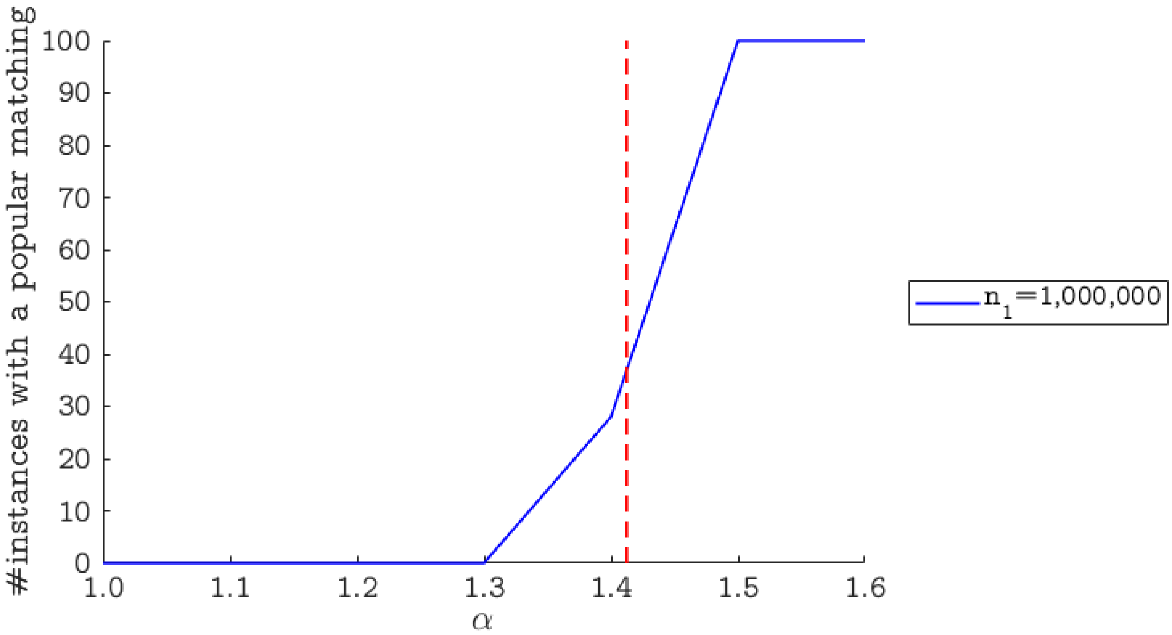


Figure 3.13: Number of instances with a popular matching in the case $k = 8$ for $n_1 = 10^6$, with the dashed line plotting $\alpha = \alpha_8 \approx 1.4124$

4 Computing Unpopularity Factor in MP and RP

Let I be an MP or RP instance consisting of a set $A = \{a_1, a_2, \ldots, a_n\}$ of n people. Throughout this chapter, we consider a more general setting where ties among two or more people in the preference lists are allowed.

In this chapter, we will develop an algorithm to compute the unpopularity factor of a given matching in I, which runs in $O(m\sqrt{n}\log n)$ time for MP and in $O(m\sqrt{n}\log^2 n)$ time for RP. We will also generalize the notion of unpopularity factor to the weighted setting where people are given different voting weights, and show that our algorithm can be slightly modified to support that setting with the same running time.

4.1 Unweighted Setting

We first consider an unweighted setting where every person has equal voting weight.

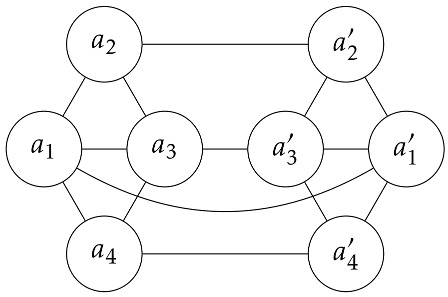
4.1.1 RP Instances

Let I be an RP instance, M be a matching of I, and k be an arbitrary nonnegative rational number. Beginning with a similar approach to [8], we construct an undirected graph $H_{(M,k)}$ with vertices $A \cup A'$, where $A' = \{a'_1, a'_2, \ldots, a'_n\}$ is a set of “copies” of people in A. An edge $\{a_i, a_j\}$ exists if and only if a_i is in a_j's preference list and a_j is in a_i's preference list; an edge $\{a'_i, a'_j\}$ exists if and only if $\{a_i, a_j\}$ exists; an edge $\{a_i, a'_j\}$ exists if and only if $i = j$. See Example 3.

Example 3. Consider the following matching M in an RP instance.

<u>Preference Lists</u>
$\boldsymbol{a_1}$ **:** a_2, a_3, a_4
$\boldsymbol{a_2}$ **:** a_3, a_1
$\boldsymbol{a_3}$ **:** a_1, a_2, a_4
$\boldsymbol{a_4}$ **:** a_1, a_3

$M = \{(a_1, a_2), (a_3, a_4)\}$



The auxiliary graph $H_{(M,k)}$ is shown on the right. □

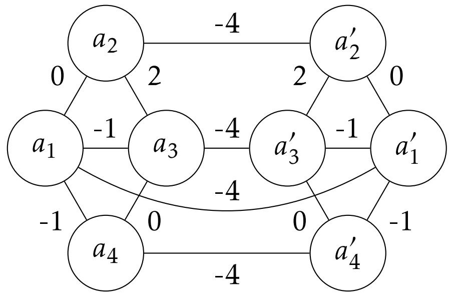
The major distinction of our algorithm is that we assign weights to edges of $H_{(M,k)}$ differently from [8]. For each pair of i and j with an edge $\{a_i, a_j\}$, define $\delta_{i,j}$ as follows.

$$\delta_{i,j} = \begin{cases} 1, & \text{if } a_i \text{ is unmatched in } M \text{ or } a_i \text{ prefers } a_j \text{ to } M(a_i); \\ -k, & \text{if } a_i \text{ prefers } M(a_i) \text{ to } a_j; \\ 0, & \text{if } \{a_i, a_j\} \in M \text{ or } a_i \text{ likes } a_j \text{ and } M(a_i) \text{ equally.} \end{cases}$$

For each pair of i and j, we set the weights of both $\{a_i, a_j\}$ and $\{a'_i, a'_j\}$ to be $\delta_{i,j} + \delta_{j,i}$. Finally, for each edge $\{a_i, a'_i\}$, we set its weight to be $-2k$ if a_i is matched in M, and 0 otherwise. See Example 4.

Example 4. Consider the matching M in Example 3, with $k = 2$.

$\delta_{i,j}$		j			
		1	2	3	4
i	1		0	-2	-2
	2	0		1	
	3	1	1		0
	4	1		0	



The values of all $\delta_{i,j}$ are shown in the left table, and the auxiliary graph $H_{(M,2)}$ is shown on the right. □

The intuition behind the construction of this auxiliary graph is that we want to check whether $u(M) > k$, i.e. whether there exists another matching M' that is more than k times more popular than M. Each matching M' is represented by a perfect matching of $H_{(M,k)}$ consisting of the edges of M' (joining vertices in A), the copies of these edges (joining vertices in A'), and the edges joining each unmatched person a_i with his own copy a_i'. We do so by adding two points for each person who prefers M' to M, subtracting $2k$ points for each one who prefers M to M', and finally checking whether the total score is positive. This is the reason why we set $\delta_{i,j}$ to be 1 if a_i prefers a_j to $M(a_i)$ and to be $-k$ if a_i prefers $M(a_i)$ to a_j. This is also the reason why we set the weight of $\{a_i, a_i'\}$ to be $-2k$ if a_i is matched in M.

The relation between $u(M)$ and the graph $H_{(M,k)}$ is formally shown in the following lemma.

Lemma 11. $u(M) > k$ if and only if $H_{(M,k)}$ contains a positive weight perfect matching.

Proof. For any matching M', define $A_1(M')$ to be a set of people in A that are matched in M', and $A_2(M')$ to be a set of people in A that are unmatched in M'. Also, define

$$A_1^+(M') = \{a_i \in A_1(M') | a_i \text{ is unmatched in } M \text{ or } a_i \text{ prefers } M'(a_i) \text{ to } M(a_i)\};$$
$$A_1^-(M') = \{a_i \in A_1(M') | a_i \text{ prefers } M(a_i) \text{ to } M'(a_i)\};$$
$$A_2^-(M') = \{a_i \in A_2(M') | a_i \text{ is matched in } M\}.$$

We have $\phi(M', M) = |A_1^+(M')|$ and $\phi(M, M') = |A_1^-(M')| + |A_2^-(M')|$.

Suppose that $u(M) > k$. From the definition of $u(M)$, there must be a matching M_0 such that $\phi(M_0, M) > k\phi(M, M_0)$. In the graph $H_{(M,k)}$, consider a perfect matching

$$S_0 = M_0 \cup \{\{a_i', a_j'\} | \{a_i, a_j\} \in M_0\} \cup \{\{a_i, a_i'\} | a_i \text{ is unmatched in } M_0\}$$

with weight W_0. From the definition, we have

$$\begin{aligned} W_0 &= 2\left(|A_1^+(M_0)| - k|A_1^-(M_0)|\right) - 2k|A_2^-(M_0)| \\ &= 2\left(|A_1^+(M_0)| - k\left(|A_1^-(M_0)| + |A_2^-(M_0)|\right)\right) \\ &= 2(\phi(M_0, M) - k\phi(M, M_0)) \\ &> 0, \end{aligned}$$

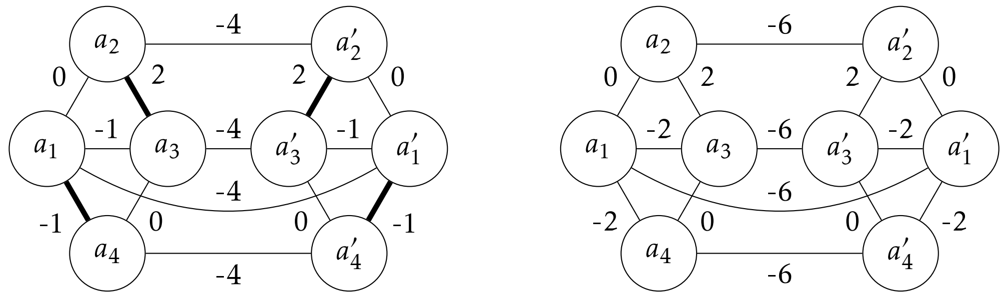
hence $H_{(M,k)}$ contains a positive weight perfect matching.

On the other hand, suppose there is a positive weight perfect matching S_1 of $H_{(M,k)}$ with weight W_1. See Example 5. Let $M_1 = \{\{a_i, a_j\} \in S_1\}$ and $M_2 = \{\{a_i, a_j\} | \{a_i', a_j'\} \in S_1\}$. Since S_1 is a perfect matching of $H_{(M,k)}$, we have $A_2(M_1) = A_2(M_2)$, and

$$\begin{aligned} &0 < W_1 \\ &= (|A_1^+(M_1)| - k|A_1^-(M_1)|) + (|A_1^+(M_2)| - k|A_1^-(M_2)|) - 2k|A_2^-(M_1)| \\ &= (|A_1^+(M_1)| - k|A_1^-(M_1)|) + (|A_1^+(M_2)| - k|A_1^-(M_2)|) - k|A_2^-(M_1)| - k|A_2^-(M_2)| \\ &= (\phi(M_1, M) - k\phi(M, M_1)) + (\phi(M_2, M) - k\phi(M, M_2)). \end{aligned}$$

Therefore, we have either $\phi(M_1, M) > k\phi(M, M_1)$ or $\phi(M_2, M) > k\phi(M, M_2)$, which implies $u(M) > k$. □

Example 5. Consider the auxiliary graphs $H_{(M,2)}$ and $H_{(M,3)}$ constructed from a matching M in Example 3.



On the left, $H_{(M,2)}$ has a positive weight perfect matching consisting of the bold-faced edges, but on the right, $H_{(M,3)}$ does not. This implies $2 < u(M) \leq 3$. □

For a given value of k, the problem of determining whether $u(M) > k$ is now transformed to detecting a positive weight perfect matching of $H_{(M,k)}$, which can be done by finding the maximum weight perfect matching of $H_{(M,k)}$.

Lemma 12. Given an RP instance I, a matching M of I, and a rational number $k = x/y$, where $x \in [0, n-1]$ and $y \in [1, n]$ are integers, there is an algorithm to determine whether $u(M) > k$ in $O(m\sqrt{n}\log n)$ time.

Proof. From Lemma 11, the problem of determining whether $u(M) > k$ is equivalent to determining whether $H_{(M,k)}$ has a positive weight perfect matching. Observe that $H_{(M,k)}$ has $O(n)$ vertices and $O(m)$ edges, and we can multiply the weights of all edges by y so that they are all integers with magnitude $O(n)$. Using the recent algorithm of Duan et al. [12], we can find a maximum weight perfect matching in a graph with integer weight edges of magnitude poly(n) in $O(m\sqrt{n}\log n)$ time, hence we can detect a positive weight perfect matching in $H_{(M,k)}$ in $O(m\sqrt{n}\log n)$ time. □

As the possible values of $u(M)$ are limited, we can perform a binary search for its value. This allows us to efficiently compute $u(M)$. To the best of our knowledge, this is the first approach on popular matchings that employs the binary search technique.

Theorem 6. Given an RP instance I and a matching M of I, there is an algorithm to compute $u(M)$ in $O(m\sqrt{n}\log^2 n)$ time.

Proof. Observe that if $u(M)$ is not ∞, it must be in the form of x/y, where $x \in [0, n-1]$ and $y \in [1, n]$ are integers, meaning that there are at most n^2 possible values of $u(M)$. By performing a binary search on the value of $k = x/y$ (if $u(M) > n-1$, then $u(M) = \infty$), we run the algorithm in Lemma 12 to determine whether $u(M) > k$ for $O(\log n^2) = O(\log n)$ times to find the exact value of $u(M)$, hence the total running time is $O(m\sqrt{n}\log^2 n)$. □

4.1.2 MP Instances

The running time of the algorithm in Theorem 6 is for a general RP instance. However, in an MP instance we can improve it using the following approach. For any matching M in an MP instance, we define a matching

$$S = M \cup \{\{a'_i, a'_j\} | \{a_i, a_j\} \in M\} \cup \{\{a_i, a'_i\} | a_i \text{ is unmatched in } M\}$$

in the graph $H_{(M,k)}$. Since S is a perfect matching, for any perfect matching S' of $H_{(M,k)}$, every edge of S' that is not in S must be a part of some cycle in which the edges alternate between S and S'. Moreover, from the definition of $\delta_{i,j}$, every edge of S has zero weight. Therefore, $H_{(M,k)}$ contains a positive weight perfect matching if and only if it contains a positive weight alternating cycle w.r.t. S. Hence, the problem becomes equivalent to detecting a positive weight alternating cycle (w.r.t. S) in $H_{(M,k)}$. Note that this property holds for every RP instance, not limited to only MP.

However, the special property of MP is that A is bipartite. Let A_M and A_W be the two parts of A with no edge between vertices in the same part (which correspond to the sets of men and women, respectively). Also, let $A'_M = \{a'_i | a_i \in A_M\}$ and $A'_W = \{a'_i | a_i \in A_W\}$. Observe that we can divide the vertices of $H_{(M,k)}$ into two parts $H_1 = A_M \cup A'_W$ and $H_2 = A_W \cup A'_M$ with no edge between vertices in the same part, so $H_{(M,k)}$ is also bipartite.

In $H_{(M,k)}$, we can orient the edges of S toward H_2 and all other edges toward H_1, hence the problem of detecting a positive weight alternating cycle becomes equivalent to detecting a positive weight directed cycle (see Example 6), which can be done in $O(m\sqrt{n})$ time using the shortest path algorithm of Goldberg [19]. Therefore, by performing a binary search on the value of $u(M)$ similar to in RP, the total running time for MP is $O(m\sqrt{n}\log n)$.

Example 6. Consider the following matching M in an MP instance with men a_1 and a_3, along with women a_2 and a_4.

$A_M = \{a_1, a_3\}$, $A_W = \{a_2, a_4\}$

Preference Lists

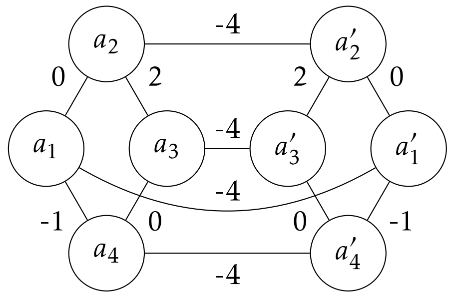
$\boldsymbol{a_1}$ **:** a_2, a_4

$\boldsymbol{a_2}$ **:** a_3, a_1

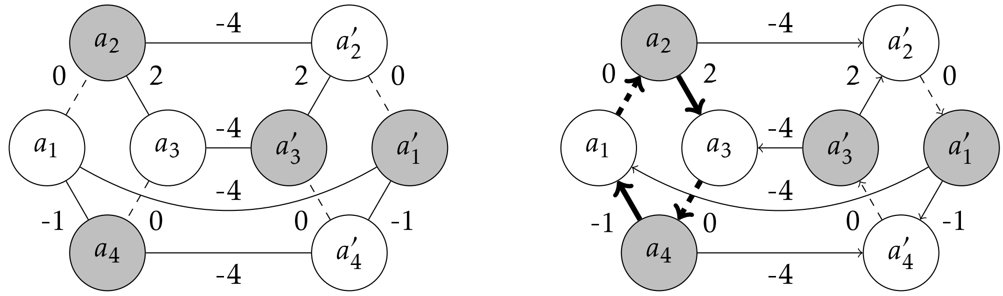
$\boldsymbol{a_3}$ **:** a_2, a_4

$\boldsymbol{a_4}$ **:** a_1, a_3

$M = \{\{a_1, a_2\}, \{a_3, a_4\}\}$



the auxiliary graph $H_{(M,2)}$ is shown on the top right.



On the left, observe that $H_{(M,2)}$ is a bipartite graph with parts $H_1 = \{a_1, a_3, a'_2, a'_4\}$ (colored in white) and $H_2 = \{a_2, a_4, a'_1, a'_3\}$ (colored in grey). Also, the edges of S are shown in dashed lines.

On the right, we orient the edges of S (dashed arrows) toward H_2, and the rest toward H_1. This directed graph has a positive weight directed cycle consisting of the bold-faced arrows, which implies $u(M) > 2$. □

In a way similar to RP, we have the following lemma and theorem for MP.

Lemma 13. Given an MP instance I, a matching M of I, and a number $k = x/y$, where $x \in [0, n-1]$ and $y \in [1, n]$ are integers, there is an algorithm to determine whether $u(M) > k$ in $O(m\sqrt{n})$ time.

Theorem 7. Given an MP instance I and a matching M of I, there is an algorithm to compute $u(M)$ in $O(m\sqrt{n}\log n)$ time.

4.2 Weighted Setting

The previous section shows the algorithm to compute an unpopularity factor of a given matching in an unweighted RP or MP instance where every person has equal voting weight. However, in many real-world situations, people may have different voting weights based on position, seniority, etc. Our algorithm can also be slightly modified to support a weighted instance with integer weights bounded by $N = \text{poly}(n)$ with the same running time in both RP and MP.

In the weighted setting, each person $a_i \in A$ has a weight $w(a_i)$. We analogously define $\phi(M,M')$ to be the sum of weights of people who strictly prefer a matching M to a matching M', i.e.

$$\phi(M,M') = \sum_{a \in A_{(M,M')}} w(a),$$

where $A_{(M,M')} = \{a \in A | r_a(M(a)) < r_a(M'(a))\}$. We also define $\Delta(M,M')$ and $u(M)$ the same way as in the unweighted setting. For each $a_i \in A$, we assume that $w(a_i)$ is a non-negative integer not exceeding $N = \text{poly}(n)$. Note that an unweighted instance can be viewed as a special case of a weighted instance where $w(a_i) = 1$ for all $a_i \in A$.

To support the weighted setting, we construct an auxiliary graph $H_{(M,k)}$ with the same set of vertices and edges as in the unweighted setting, but with slightly different weights of the edges. For each pair of i and j with an edge $\{a_i, a_j\}$, define

$$\delta_{i,j} = \begin{cases} w(a_i), & \text{if } a_i \text{ is unmatched in } M \text{ or } a_i \text{ prefers } a_j \text{ to } M(a_i); \\ -kw(a_i), & \text{if } a_i \text{ prefers } M(a_i) \text{ to } a_j; \\ 0, & \text{if } \{a_i, a_j\} \in M \text{ or } a_i \text{ likes } a_j \text{ and } M(a_i) \text{ equally.} \end{cases}$$

For each pair of i and j, the weights of $\{a_i, a_j\}$ and $\{a'_i, a'_j\}$ is $\delta_{i,j} + \delta_{j,i}$. Finally, for each edge $\{a_i, a'_i\}$, we set its weight to be $-2kw(a_i)$ if a_i is matched in M, and 0 otherwise.

The auxiliary graph $H_{(M,k)}$ still has the same relation with $u(M)$, as shown in the following lemma.

Lemma 14. In the weighted RP instance, $u(M) > k$ if and only if $H_{(M,k)}$ contains a positive weight perfect matching.

Proof. The proof of this lemma is very similar to that of Lemma 11. We define the sets $A_1(M')$, $A_2(M')$, $A_1^+(M')$, $A_1^-(M')$, and $A_2^-(M')$ by the same way as in the proof of Lemma 11. However, from now on we will compute the sum of weights of the elements in each set instead of counting the number of its elements.

For any set B, define $w(B) = \sum_{a \in B} w(a)$. We have $\phi(M',M) = w(A_1^+(M'))$ and $\phi(M,M') = w(A_1^-(M')) + w(A_2^-(M'))$.

Suppose that $u(M) > k$. There must exist a matching M_0 such that $\phi(M_0,M) > k\phi(M,M_0)$. Similarly to the proof of Lemma 11, in the graph $H_{(M,k)}$ consider a perfect matching

$$S_0 = M_0 \cup \{\{a'_i, a'_j\} | \{a_i, a_j\} \in M_0\} \cup \{\{a_i, a'_i\} | a_i \text{ is unmatched in } M_0\}$$

with weight W_0. From the definition, we have

$$\begin{aligned} W_0 &= 2\,(w(A_1^+(M_0)) - kw(A_1^-(M_0))) - 2kw(A_2^-(M_0)) \\ &= 2\,(w(A_1^+(M_0)) - k\,(w(A_1^-(M_0)) + w(A_2^-(M_0)))) \\ &= 2(\phi(M_0, M) - k\phi(M, M_0)) \\ &> 0, \end{aligned}$$

hence $H_{(M,k)}$ contains a positive weight perfect matching.

On the other hand, suppose there is a positive weight perfect matching S_1 of $H_{(M,k)}$ with weight W_1. Let $M_1 = \{\{a_i, a_j\} \in S_1\}$ and $M_2 = \{\{a_i, a_j\} | \{a'_i, a'_j\} \in S_1\}$. Similarly to the proof of Lemma 11, we have $A_2(M_1) = A_2(M_2)$, and

$$\begin{aligned} 0 &< W_1 \\ &= (w(A_1^+(M_1)) - kw(A_1^-(M_1))) + (w(A_1^+(M_2)) - kw(A_1^-(M_2))) - 2kw(A_2^-(M_1)) \\ &= (w(A_1^+(M_1)) - kw(A_1^-(M_1))) + (w(A_1^+(M_2)) - kw(A_1^-(M_2))) - kw(A_2^-(M_1)) - kw(A_2^-(M_2)) \\ &= (\phi(M_1, M) - k\phi(M, M_1)) + (\phi(M_2, M) - k\phi(M, M_2)). \end{aligned}$$

Therefore, we have either $\phi(M_1, M) > k\phi(M, M_1)$ or $\phi(M_2, M) > k\phi(M, M_2)$, which implies $u(M) > k$. □

Since the weights of people are bounded by $N = \text{poly}(n)$, the unpopularity factor $u(M)$ must be in the form $k = x/y$, where x and y are integers not exceeding Nn. For a given value of k, if we multiply the weights of all edges of $H_{(M,k)}$ by y, they will be integers with magnitude $O(Nn) = \text{poly}(n)$. Therefore, we can still use the algorithm of Duan et al. [12] to find a maximum weight perfect matching of $H_{(M,k)}$ with the same running time.

Moreover, there are at most $O(N^2 n^2)$ possible values of $u(M)$. By performing a binary search on the value of k, we have to run the above algorithm for $O(\log N^2 n^2) = O(\log n)$ times as in the unweighted setting, hence the total running time is still $O(m\sqrt{n}\log^2 n)$.

The argument for MP instances still works for the weighted setting as well since $H_{(M,k)}$ is still bipartite, hence we have the following theorems for the weighted setting RP and MP.

Theorem 8. Given a weighted RP instance I with integer weights bounded by $N = \text{poly}(n)$ and a matching M of I, there is an algorithm to compute $u(M)$ in $O(m\sqrt{n}\log^2 n)$ time.

Theorem 9. Given a weighted MP instance I with integer weights bounded by $N = \text{poly}(n)$ and a matching M of I, there is an algorithm to compute $u(M)$ in $O(m\sqrt{n}\log n)$ time.

5 Finding a Weakly Stable Noncrossing Matching

In NMP, we introduced the definitions of WSNM and SSNM in Definitions 7 and 8, respectively. An SSNM is a matching that is both noncrossing and stable, while a WSNM is "stable" in a weaker sense as it may admit a blocking pair, just not a noncrossing one.

Observe that an SSNM may not exist in some instances. For example, in an instance of two men and two women, with $L_{m_1} = (w_2, w_1), L_{m_2} = (w_1, w_2), L_{w_1} = (m_2, m_1)$, and $L_{w_2} = (m_1, m_2)$, the only stable matching is $\{(m_1, w_2), (m_2, w_1)\}$, and its two edges do cross each other. On the other hand, the above instance has two WSNMs: $\{(m_1, w_2)\}$ and $\{(m_2, w_1)\}$.

It also turns out that a WSNM always exists in every instance. In this chapter, we will constructively prove the existence of a WSNM by developing an $O(n^2)$ time algorithm to find one in a given instance.

5.1 Outline of Algorithm

Without loss of generality, for each man m_i and each woman w_j, we assume that w_j is in m_i's preference list if and only if m_i is also in w_j's preference list (otherwise we can simply remove the entries that are not mutual from the lists). Initially, every person is unmatched ($M = \emptyset$).

Our algorithm uses proposals from men to women similarly to the Gale–Shapley algorithm [16] in the original Stable Marriage Problem, but in a more constrained way. Let M be the current matching. When a woman w_j receives a proposal from a man m_i, if she prefers her current partner $M(w_j)$ to m_i, she rejects m_i; otherwise if she is currently unmatched or prefers m_i to $M(w_j)$, she dumps $M(w_j)$ and accepts m_i.

Consider a man m_i and a woman w_j not matched with each other. An entry w_j in L_{m_i} has the following possible states:

1. ***accessible*** (to m_i), if (m_i, w_j) does not cross any edge in M;
   - 1.1. ***available*** (to m_i), if w_j is accessible to m_i, and is currently unmatched or matched with a man she likes less than m_i, i.e. m_i is going to be accepted if he proposes to her (for convenience, if w_j is currently matched with m_i, we also call w_j accessible and available to m_i).
   - 1.2. ***unavailable*** (to m_i), if w_j is accessible to m_i, but is currently matched with a man she likes more than m_i, i.e. m_i is going to be rejected if he proposes to her;
2. ***inaccessible*** (to m_i), if w_j is not accessible to m_i;

For a man m_i, if every entry in L_{m_i} before $M(m_i)$ is either inaccessible or unavailable, then we say that m_i is *stable*; otherwise (there is at least one available entry before $M(m_i)$) we say that m_i is *unstable*.

The main idea of our algorithm is that, at any point, if there is at least one unstable man, we pick the topmost unstable man m_k (the unstable man m_k with least index k) and perform the following operations.

1. Let m_k *dump* his current partner $M(m_k)$ (if any), i.e. remove $(m_k, M(m_k))$ from M, and let him propose to the available woman w_l that he prefers most.
2. Let w_l *dump* her current partner $M(w_l)$ (if any), i.e. remove $(M(w_l), w_l)$ from M, and let her accept m_k's proposal.
3. Add the new pair (m_k, w_l) to M.

We repeatedly perform such operations until every man becomes stable. Note that throughout the algorithm, every proposal will result in acceptance and M will always be noncrossing since men propose only to women available to them.

5.2 Proof of Correctness

First, we will show that if our algorithm stops, then the matching M given by the algorithm must be a WSNM.

Assume, for the sake of contradiction, that M admits a noncrossing blocking pair (m_i, w_j). That means m_i prefers w_j to his current partner $M(m_i)$, w_j prefers m_i to her current partner $M(w_j)$, and (m_i, w_j) does not cross an edge in M, thus the entry w_j in L_{m_i} is available and is located before $M(m_i)$. However, by the description of our algorithm, the process stops when every man becomes stable, which means there cannot be an available entry before $M(m_i)$ in L_{m_i}, a contradiction. Therefore, we can conclude that our algorithm gives a WSNM as a result whenever it stops.

However, it is not trivial that our algorithm will eventually stop. In contrast to the Gale–Shapley algorithm [16] in the Stable Marriage Problem, in this problem a woman is not guaranteed to get increasingly better partners throughout the process because a man can dump a woman too if he later finds a better available woman previously inaccessible to him (due to having an edge obstructing them). In fact, it is actually the case where the process may not stop if at each step we pick an arbitrary unstable man instead of the topmost one. For example, in an instance of two men and two women with $L_{m_1} = (w_2, w_1), L_{m_2} = (w_1, w_2), L_{w_1} = (m_1, m_2), L_{w_2} = (m_2, m_1)$, the order of picking $m_1, m_2, m_2, m_1, m_1, m_2, m_2, m_1, \ldots$ results in the process continuing forever, with the matching M looping between $\{(m_1, w_2)\}$, $\{(m_2, w_2)\}$, $\{(m_2, w_1)\}$, and $\{(m_1, w_1)\}$ at each step.

Fortunately, it turns out that our algorithm always stop. We will prove that statement as well as evaluating the worst-case running time of our algorithm after we introduce the explicit implementation of the algorithm in the next section.

5.3 Implementation

To implement the above algorithm, we have to consider how to efficiently find the topmost unstable man at each step in order to perform the operations on him. Of course, a straightforward way to do this is to update the state of every entry in every man's preference list after each step, but that method will be very inefficient. Instead, we introduce the following scanning method.

Throughout the algorithm, we do not know exactly the set of all unstable men, but we instead keep a set S of men that are "possibly unstable". Initially, the set S contains all men, i.e. $S = \{m_1, m_2, \dots, m_n\}$, and at each step we maintain the set S of the form $\{m_i, m_{i+1}, \dots, m_n\}$ for some $i \in [n]$ (that means $m_1, m_2, \dots, m_{i-1}$ are guaranteed to be stable at that time). Note that in the actual implementation, we can store only the index of the topmost man in S instead of the whole set. At each step, we scan the topmost man m_i in S and check whether m_i is stable. If m_i is already stable, then we simply skip him by removing m_i from S and moving to scan the next man in S. If m_i is unstable, then m_i is indeed the topmost unstable man we want, so we perform some operations on m_i. Note that the operations may cause some men to become unstable, so after that we have to add all men that are possibly affected by the operations back to S. The details of the scanning and updating processes are as follows.

During the scan of m_i, let m_{prev} be the *matched* man closest to m_i that lies above him, and let $w_{\text{first}} = M(m_{\text{prev}})$ (we let $w_{\text{first}} = w_1$ if there is no m_{prev}). Also, let m_{next} be the *matched* man closest to m_i that lies below him, and let $w_{\text{last}} = M(m_{\text{next}})$ (we let $w_{\text{last}} = w_n$ if there is no m_{next}). Observe that matching m_i with anyone lying above w_{first} will cross the edge $(m_{\text{prev}}, w_{\text{first}})$, and matching m_i with anyone lying below w_{last} will cross the edge $(m_{\text{next}}, w_{\text{last}})$. Therefore, the range of all women accessible to m_i ranges exactly from w_{first} to w_{last}, hence the range of all women available to m_i ranges from either w_{first} or $w_{\text{first}+1}$ (depending on whether w_{first} prefers m_i to m_{prev}) to either w_{last} or $w_{\text{last}-1}$ (depending on whether w_{last} prefers m_i to m_{next}). See Fig. 1.

Then in the available range, m_i selects the woman w_j that he prefers most.

Case 1: w_j does not exist or m_i is currently matched with w_j.

That means m_i is currently stable, so we can skip him. We remove m_i from S and proceed to scan m_{i+1} in the next step (called a *downward jump*).

Case 2: w_j exists and m_i is not currently matched with w_j

That means m_i is indeed the topmost unstable man we want, so we perform the operations on him by letting m_i propose to w_j and dump his current partner (if any).

Case 2.1: m_{prev} exists and $w_j = w_{\text{first}}$.

That means w_{first} dumps m_{prev} to get matched with m_i, which leaves m_{prev} unmatched and he may possibly become unstable. Furthermore, $m_{\text{prev}+1}$, $m_{\text{prev}+2}, \dots, m_{i-1}$ as well as m_i himself may also possibly become unstable since they now gain access to women lying above w_{first} previously inaccessible to them (if $w_{\text{first}} \neq w_1$). On the other hand, $m_1, m_2, \dots, m_{\text{prev}-1}$ clearly remain stable, hence we add $m_{\text{prev}}, m_{\text{prev}+1}, \dots, m_{i-1}$ to S and proceed to scan m_{prev} in the next step (called an *upward jump*).

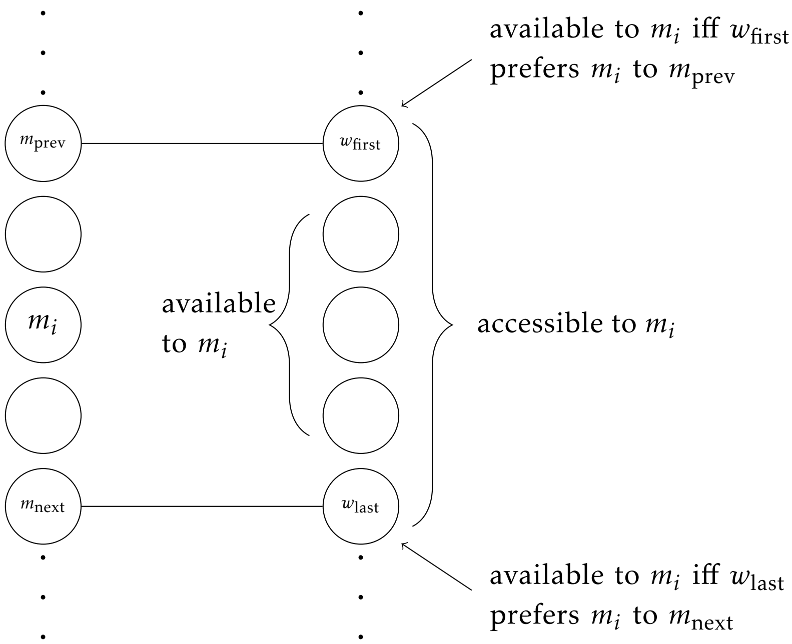


Figure 5.1: Accessible and available women to m_i

Case 2.2: m_{prev} does not exist or $w_j \neq w_{\text{first}}$.

Case 2.2.1: m_i is currently matched and w_j lies geometrically below $M(m_i)$.

Then, $m_{\text{prev}}, m_{\text{prev}+1}, \ldots, m_{i-1}$ (or $m_1, m_2, \ldots, m_{i-1}$ if m_{prev} does not exist) may possibly becomes unstable since they now gain access to women between $M(m_i)$ and w_j previously inaccessible to them. Therefore, we perform the upward jump to m_{prev} (or to m_1 if m_{prev} does not exist), adding $m_{\text{prev}}, m_{\text{prev}+1}, \ldots, m_{i-1}$ (or $m_1, m_2, \ldots, m_{i-1}$) to S and proceed to scan m_{prev} (or m_1) in the next step, except when $m_i = m_1$ that we perform the downward jump to m_2.

It turns out that this case is impossible, which we will prove in the next section.

Case 2.2.2: m_i is currently unmatched or w_j lies geometrically above $M(m_i)$.

Then all men lying above m_i clearly remain stable (because the sets of available women to $m_1, m_2, \ldots, m_{i-1}$ either remain the same or become smaller). Also, m_i now becomes stable as well (because m_i selects a woman he prefers most in the available range), except in the case where $w_j = w_{\text{last}}$ (because the edge $(m_{\text{next}}, w_{\text{last}})$ is removed and m_i now has access to women lying below w_{last} previously inaccessible to him). Therefore, we perform the downward jump, removing m_i from S and moving to scan m_{i+1} in the next step, except when $w_j = w_{\text{last}}$ that we have to scan m_i again in the next step (this exception, however, turns out to be impossible, which we will prove in the next section).

We scan the men in this way until S becomes empty (see Example 7). By the way we add all men that may possibly become unstable after each step back to S, at any step S is guaranteed to contain the topmost unstable man.

Example 7. Consider an instance of three men and three women with the following preference lists.

$$\begin{array}{ll} \boldsymbol{m_1}: w_3, w_1, w_2 & \qquad \boldsymbol{w_1}: m_3, m_2, m_1 \\ \boldsymbol{m_2}: w_2, w_3, w_1 & \qquad \boldsymbol{w_2}: m_3, m_2, m_1 \\ \boldsymbol{m_3}: w_2, w_1, w_3 & \qquad \boldsymbol{w_3}: m_3, m_2, m_1 \end{array}$$

Our algorithm will scan the men in the following order and output a matching $M = \{(m_2, w_1), (m_3, w_2)\}$, which is a WSNM.

Step	**Process**	M **at the end of step**	S **at the end of step**
0		$\emptyset$	$\{m_1, m_2, m_3\}$
1	scan m_1, add (m_1, w_3)	$\{(m_1, w_3)\}$	$\{m_2, m_3\}$
2	scan m_2, add (m_2, w_3), remove (m_1, w_3)	$\{(m_2, w_3)\}$	$\{m_1, m_2, m_3\}$
3	scan m_1, add (m_1, w_1)	$\{(m_1, w_1), (m_2, w_3)\}$	$\{m_2, m_3\}$
4	scan m_2, add (m_2, w_2), remove (m_2, w_3)	$\{(m_1, w_1), (m_2, w_2)\}$	$\{m_3\}$
5	scan m_3, add (m_3, w_2), remove (m_2, w_2)	$\{(m_1, w_1), (m_3, w_2)\}$	$\{m_2, m_3\}$
6	scan m_2, add (m_2, w_1), remove (m_1, w_1)	$\{(m_2, w_1), (m_3, w_2)\}$	$\{m_1, m_2, m_3\}$
7	scan m_1	$\{(m_2, w_1), (m_3, w_2)\}$	$\{m_2, m_3\}$
8	scan m_2	$\{(m_2, w_1), (m_3, w_2)\}$	$\{m_3\}$
9	scan m_3	$\{(m_2, w_1), (m_3, w_2)\}$	$\emptyset$

$\square$

5.4 Proof of Finiteness

First, we will prove the following lemma about the algorithm described in the previous section.

Lemma 15. During the scan of a man m_i, if m_i is currently matched, then m_i does not propose to any woman lying geometrically below $M(m_i)$.

Proof. We call a situation when a man m_i proposes to a woman lying geometrically below $M(m_i)$ a *downward switch*. Assume, for the sake of contradiction, that a downward switch occurs at least once during the whole algorithm. Suppose that the first downward switch occurs at step s, when a man m_i is matched to $w_k = M(m_i)$ and proposes to w_j with $j > k$. We have m_i prefers w_j to w_k.

Consider the step $t < s$ when m_i proposed to w_k (if m_i proposed to w_k multiple times, consider the most recent one). At step t, w_j must be inaccessible or unavailable to m_i (otherwise he would choose w_j instead of w_k), meaning that there must be an edge (m_p, w_q) with $p > i$ and $k < q < j$ obstructing them in the inaccessible case, or an edge (m_p, w_q) with $p > i$, $q = j$, and w_j preferring m_p to m_i in the unavailable case.

We define a *dynamic edge* e as follows. First, at step t we set $e = (m_p, w_q)$. Then, throughout the process we update e by the following method: whenever the endpoints of e cease to be partners of each other, we update e to be the edge joining the endpoint that dumps his/her partner with his/her new partner. Formally, suppose that e is currently (m_x, w_y). If m_x dumps w_y to get matched with $w_{y'}$, we update e to be $(m_x, w_{y'})$; if w_y dumps m_x to get matched with $m_{x'}$, we update e to be $(m_{x'}, w_y)$.

By this updating method, the edge e will always exist after step t, but may change over time. Observe that from step t to step s, we always have $x > i$ because of the existence of (m_i, w_k). Moreover, before step s, if m_x dumps w_y to get matched with $w_{y'}$, by the assumption that a downward switch did not occur before step s, we have $y' < y$, which means the index of the women's side of e's endpoints never increases. Consider the edge $e = (m_x, w_y)$ at step s, we must have $x > i$ and $y \leq q \leq j$. If $y < j$, then the edge e obstructs m_i and w_j, making w_j inaccessible to m_i. If $y = j$, that means w_j never got dumped since step t, so she got only increasingly better partners, thus w_j prefers m_x to m_i, making w_j unavailable to m_i. Therefore, in both cases m_i could not propose to w_j, a contradiction. Hence, a downward switch cannot occur in our algorithm. $\square$

Lemma 15 shows that a woman cannot get her partner stolen by any woman that lies below her, which is equivalent to the following corollary.

Corollary 1. If a man m_i dumps a woman w_j to propose to a woman w_k, then $k < j$.

It also implies that Case 2.2.1 in the previous section never occurs. Therefore, the only case where an upward jump occurs is Case 2.1 (m_{prev} exists and m_i proposes to w_{first}). We will now prove the following lemma.

Lemma 16. During the scan of a man m_i, if m_{next} exists, then m_i does not propose to w_{last}.

Proof. Assume, for the sake of contradiction, that m_i proposes to w_{last}. Since m_{next} exists, this proposal obviously cannot occur in the very first step of the algorithm. Consider a man m_k we scanned in the previous step right before scanning m_i.

Case 1: m_k lies below m_i, i.e. $k > i$.

In order for the upward jump from m_k to m_i to occur, m_i must have been matched with a woman but got her stolen by m_k in the previous step. However, $m_{i+1}, m_{i+2}, \ldots, m_{\text{next}-1}$ are all currently unmatched (by the definition of m_{next}), so the only possibility is that $m_k = m_{\text{next}}$, and thus his partner that got stolen was w_{last}. Therefore, we can conclude that w_{last} prefers m_{next} to m_i, which means w_{last} is currently unavailable to m_i, a contradiction.

Case 2: m_k lies above m_i, i.e. $k < i$.

The jump before the current step was a downward jump, but since m_{next} has been scanned before, an upward jump over m_i must have occurred at some point before the current step. Consider the most recent upward jump over m_i before the current step. Suppose than it occurred at the end of step t and was a jump from $m_{k'}$ to m_j, with $k' > i$ and $j < i$. In order for this jump to occur, m_j must have been matched with a woman but got her stolen by $m_{k'}$ at step t. However, $m_{i+1}, m_{i+2}, \ldots, m_{\text{next}-1}$ are all currently unmatched (by the definition of m_{next}), so the only possibility is that $m_{k'} = m_{\text{next}}$, and thus m_j's partner that got stolen was w_{last}. We also have $m_{j+1}, m_{j+2}, \ldots, m_{\text{next}-1}$ were all unmatched during step t (otherwise w_{last} would be inaccessible to $m_{k'}$), and w_{last} prefers m_{next} to m_j.

Now, consider the most recent step before step t in which we scanned m_i. Suppose it occurred at step s. During step s, m_j was matched with w_{last} and w_{last} was accessible to m_i. However, m_i was still left unmatched after step s (otherwise an upward jump over m_i at step t could not occur), meaning that w_{last} must be unavailable to him back then due to w_{last} preferring m_j to m_i. Therefore, we can conclude that w_{last} prefers m_{next} to m_i, thus w_{last} is currently unavailable to m_i, a contradiction. □

Lemma 16 shows that a man cannot get his partner stolen by any man lying above him, or equivalent to the following corollary.

Corollary 2. If a woman w_j dumps a man m_i to accept a man m_k, then $k > i$.

Now, we will show that the position of each woman's partner can only move downward throughout the process, which guarantees the finiteness of the number of steps in the entire process.

Lemma 17. After a woman w_j ceases to be a partner of a man m_i, she cannot be matched with any man $m_{i'}$ with $i' \leq i$ afterwards.

Proof. Suppose that w_j's next partner (if any) is m_a. It is sufficient to prove that $a > i$. First, consider the situation when m_i and w_j cease to be partners.

Case 1: w_j dumps m_i.

This means w_j dumps m_i to get matched with m_a right away. By Corollary 2, we have $a > i$ as required.

Case 2: m_i dumps w_j.

Suppose that m_i dumps w_j to get matched with w_k. By Corollary 1, we have $k < j$.

Case 2.1: m_i never gets dumped afterwards.

That means m_i will only get increasingly better partner, and the position of his partner can only move upwards (by Corollary 1), which means w_j cannot be matched with m_i again, or any man lying above m_i afterwards due to having an edge $(m_i, M(m_i))$ obstructing. Therefore, m_a must lie below m_i, i.e. $a > i$.

Case 2.2: m_i gets dumped afterwards.

Suppose that m_i first gets dumped by w_y at step s. By Corollary 1, we have $y \leq k < j$ (because m_i only gets increasingly better partners before getting dumped). Also suppose that w_y dumps m_i in order to get matched with m_x. By Corollary 2, we have $x > i$. Similarly to the proof of Lemma 1, consider a dynamic e first set to be (m_x, w_y) at step s. We have the index of the men's side of e's endpoints never decreases, and that of the women's side never increases. Therefore, since step s, there always exists an edge (m_x, w_y) with $x > i$ and $y < j$, obstructing w_j's access to m_i and all men lying above him. Therefore, m_a must lie below m_i, i.e. $a > i$. $\square$

5.5 Running Time Analysis

Consider any upward jump from m_i to m_k with $i > k$ that occurs right after m_i stole w_j from m_k. We call such a jump *associated to* w_j, and it has *size* $i - k$.

For any woman w_j, let U_j be the sum of the sizes of all upward jumps associated to w_j. From Lemma 17, we know that the position of w_j's partner can only move upward throughout the process, so we have $U_j \leq n - 1$. Therefore, the sum of the sizes of all upwards jumps is $\sum_{j=1}^{n} U_j \leq n(n-1) = O(n^2)$. Since the scan starts at m_1 and ends at m_n, the total number of downward jumps equals to the sum of the sizes of all upward jumps plus $n - 1$, hence the total number of steps in the whole algorithm is $O(n^2)$.

For each m_i, we keep an array of size n, with the j-th entry storing the rank of w_j in L_{m_i}. Each time we scan m_i, we query the minimum rank of available women, which is a consecutive range in the array. Using an appropriate range minimum query (RMQ) data structure such as the one introduced by Fischer [14], we can perform the scan with $O(n)$ preprocessing time per array and $O(1)$ query time. Therefore, the total running time of our algorithm is $O(n^2)$.

In conclusion, we proved that our developed algorithm is correct and terminates in $O(n^2)$ time, which also implicitly proves the existence of a WSNM in any instance.

Theorem 10. A weakly stable noncrossing matching exists in any instance with n men and n women with strict preference lists.

Theorem 11. There is an $O(n^2)$ time algorithm to find a weakly stable noncrossing matching in an instance with n men and n women with strict preference lists.

5.6 Generalization and Follow-Up Problems

In this chapter, we constructively proved that a WSNM always exists in any MP instance by developing an $O(n^2)$ time algorithm to find one. Note that our algorithm does not require the numbers of men and women to be equal. In the case that there are n_M men and n_W women, the algorithm works similarly with $O(n_M n_W)$ running time.

Our algorithm can also be generalized to the setting where people's preference lists are not strict. If we keep the definition of a blocking pair introduced in Definition 5 unchanged, similarly to the way to define a weakly stable matching in [28], then we can modify the instance by breaking ties in an arbitrary way. Clearly, a WSNM in the modified instance will also be a WSNM in the original one (because every noncrossing blocking pair in the original instance will also be a noncrossing blocking pair in the modified instance). Therefore, the algorithm still works with the same running time.

Note that the definition of a WSNM allows multiple answers with different sizes for an instance. For example, in an instance of three men and three women, with $L_{m_1} = (w_3, w_1, w_2), L_{m_2} = (w_1, w_2, w_3), L_{m_3} = (w_2, w_3, w_1), L_{w_1} = (m_2, m_3, m_1), L_{w_2} = (m_3, m_1, m_2)$, and $L_{w_3} = (m_1, m_2, m_3)$, both $\{(m_1, w_3)\}$ and $\{(m_2, w_1), (m_3, w_2)\}$ are WSNMs, but our algorithm only outputs the first one with smaller size. This naturally prompted a follow-up problem of finding a WSNM with maximum size in a given instance, which is deemed to be a practically better solution than other WSNMs as it satisfies more people. Also, as we showed that an SSNM does not always exist, another natural follow-up problem is to determine whether an SSNM exists in a given instance, and to find one if it does.

After our result was first published in [48], both problems were subsequently solved by Hamada et al. [22]. They developed algorithms to solve the first problem in $O(n^4)$ time using dynamic programming, and the second problem in $O(n^2)$ time using the rural hospitals theorem [17, 43, 44].

6 Conclusion

In this thesis, we analyzed three open problems in matching under preferences using graph-theoretic characterizations.

In Chapter 3, we studied RPMP in an HAP instance where the preference lists are strict but not complete, with every person's preference list having the same length of a constant k, and discovered a phase transition at $\alpha = \alpha_k$, where $\alpha_k \geq 1$ is the root of equation $xe^{-1/2x} = 1 - (1 - e^{-1/x})^{k-1}$. We also performed a simulation to help illustrate and verify the discovered phase transition.

In many real-world situations, ties can and are likely to occur among people's preference lists as people may like two or more items equally. RPMP in the case with ties allowed was also mentioned by Mahdian [37] and simulated by Abraham et al. [5] using a parameter t to denote the probability that each entry in a preference list is tied with previous entry. Intuitively, and also confirmed by the experimental results of [5], when ties are very likely to occur (t is very close to 1), a popular matching is likely to exist even when α is as low as 1. However, the exact phase transition point for each value of t, or whether it exists at all, has still not been found yet. This leaves a possible future work of studying the transition point in this setting for each value of t, both with complete and incomplete preference lists. Other future work includes studying RPMP in other settings such as RP and CHAP, e.g. the latter in the most basic case where every item has the same capacity c.

In Chapter 4, we developed an algorithm to compute the unpopularity factor of a given matching in $O(m\sqrt{n}\log n)$ time for MP and in $O(m\sqrt{n}\log^2 n)$ time for RP. We also generalized the notion of unpopularity factor to the weighted setting where people are given different voting weights, and show that our algorithm can be slightly modified to support that setting with the same running time. Our results also complete Tables 6.1 and 6.2, which show the updated running time of the best known algorithms related to popularity in each setting with strict preference lists, and with ties allowed, respectively.

While the problem of finding a matching that minimizes the unpopularity factor or the unpopularity margin in a given matching is NP-hard, the problem of approximating the optimum of either measure is still open. For the unpopularity factor in RP with strict preference lists, the current best algorithm is the one developed by Huang and Kavitha [23], which approximates it up to $O(\log n)$ factor. A possible future work is to investigate whether there is a better approximation algorithm for RP, or to develop one for HAP. For the unpopularity margin, however, there is currently no efficient algorithm to approximate the optimum, both in RP and HAP, which leaves a lot of rooms for future improvement.

<table>
<tr><th rowspan="2"></th><th colspan="2">Two-sided Lists</th><th>One-sided Lists</th></tr>
<tr><th>Marriage Problem (MP)</th><th>Roommates Problem (RP)</th><th>House Allocation Problem (HAP)</th></tr>
<tr><td>Determine if a popular matching exists</td><td rowspan="3">$O(m)$ [18]</td><td rowspan="2">NP-hard [13, 20]</td><td>$O(m+n)$ [5]</td></tr>
<tr><td>Find a matching M that minimizes $g(M)$</td><td rowspan="2">NP-hard [40]</td></tr>
<tr><td>Find a matching M that minimizes $u(M)$</td><td>NP-hard [23]</td></tr>
<tr><td>Test popularity of a given matching M</td><td rowspan="2">$O(m\sqrt{n})$ [8]</td><td rowspan="2">$O(m\sqrt{n}\log n)$ [8, 12]</td><td>$O(m+n)$ [5]</td></tr>
<tr><td>Compute $g(M)$ of a given matching M</td><td>$O((g+1)m\sqrt{n})$ [40]</td></tr>
<tr><td>Compute $u(M)$ of a given matching M</td><td>$\boldsymbol{O(m\sqrt{n}\log n)}$ [§4]</td><td>$\boldsymbol{O(m\sqrt{n}\log^2 n)}$ [§4]</td><td>$O(m\sqrt{n_2})$ [40]</td></tr>
</table>

Table 6.1: Updated best known algorithms for an unweighted instance with strict preference lists

<table>
<tr><th rowspan="2"></th><th colspan="2">Two-sided Lists</th><th>One-sided Lists</th></tr>
<tr><th>Marriage Problem (MP)</th><th>Roommates Problem (RP)</th><th>House Allocation Problem (HAP)</th></tr>
<tr><td>Determine if a popular matching exists</td><td rowspan="3" colspan="2">NP-hard [8]</td><td>$O(m\sqrt{n})$ [5]</td></tr>
<tr><td>Find a matching M that minimizes $g(M)$</td><td rowspan="2">NP-hard [40]</td></tr>
<tr><td>Find a matching M that minimizes $u(M)$</td></tr>
<tr><td>Test popularity of a given matching M</td><td rowspan="2">$O(m\sqrt{n})$ [8]</td><td rowspan="2">$O(m\sqrt{n}\log n)$ [8, 12]</td><td>$O(m\sqrt{n_2})$ [40]</td></tr>
<tr><td>Compute $g(M)$ of a given matching M</td><td>$O((g+1)m\sqrt{n})$ [40]</td></tr>
<tr><td>Compute $u(M)$ of a given matching M</td><td>$\boldsymbol{O(m\sqrt{n}\log n)}$ [§4]</td><td>$\boldsymbol{O(m\sqrt{n}\log^2 n)}$ [§4]</td><td>$O(m\sqrt{n_2})$ [40]</td></tr>
</table>

Table 6.2: Updated best known algorithms for an unweighted instance with ties allowed in the preference lists

In Chapter 5, we constructively proved that a WSNM always exists in any MP instance by developing an $O(n^2)$ time algorithm to find one. We also posed two follow-up open problems, finding a maximum size WSNM and determining whether an SSNM exists, which were both subsequently solved by Hamada et al. [22].

Other related open problems include investigating the noncrossing matching in the geometric version of the Stable Roommates Problem, where people can be matched regardless of gender. The most basic and natural setting of this problem is where people are represented by points arranged on a circle. This leads to a possible future work of developing an algorithm to determine whether a WSNM or an SSNM exists in a given instance, and to find one if it does.

Bibliography

[1] A. Abdulkadiroğlu, P. A. Pathak, and A. E. Roth. The New York City High School Match. *American Economic Review*, 95(2): 364–367 (2005). doi:10.1257/000282805774670167

[2] A. Abdulkadiroğlu, P. A. Pathak, A. E. Roth, and T. Sönmez. The Boston Public School Match. *American Economic Review*, 95(2): 368–371 (2005). doi:10.1257/000282805774669637

[3] A. Abdulkadiroğlu and T. Sönmez. Random Serial Dictatorship and the Core from Random Endowments in House Allocation Problems. *Econometrica*, 66(3): 689–701 (1998). doi:10.2307/2998580

[4] D. J. Abraham, K. Cechlárová, D. F. Manlove, and K. Mehlhorn. Pareto Optimality in House Allocation Problems. *Proceedings of 15th Annual International Symposium on Algorithms and Computation (ISAAC)*, pp. 3–15 (2004). doi:10.1007/978-3-540-30551-4_3

[5] D. J. Abraham, R. W. Irving, T. Kavitha, and K. Mehlhorn. Popular Matchings. *SIAM Journal on Computing*, 37(4): 1030–1045 (2007). doi:10.1137/06067328X

[6] D. J. Abraham and T. Kavitha. Dynamic Matching Markets and Voting Paths. In *Proceedings of the 10th Scandinavian Workshop on Algorithm Theory (SWAT)*, pp. 65–76 (2006). doi:10.1007/11785293_9

[7] N. Alon and J. Spencer. The Probabilistic Method. Third edition. John Wiley & Sons (2008). doi:10.1002/9780470277331

[8] P. Biró, R. W. Irving, and D. Manlove. Popular Matchings in the Marriage and Roommates Problems. In *Proceedings of the 7th International Conference on Algorithms and Complexity (CIAC)*, pp. 97–108 (2010). doi:10.1007/978-3-642-13073-1_10

[9] B. Bollobás, S. Janson, and O. Riordan. The phase transition in inhomogeneous random graphs. *Random Structures & Algorithms*, 31(1): 3–122, (2007). doi:10.1002/rsa.20168

[10] Á. Cseh, C.-C. Huang, and T. Kavitha. Popular Matchings with Two-Sided Preferences and One-Sided Ties. *SIAM Journal on Discrete Mathematics*, 31(4): 2348–2377 (2017). doi:10.1137/16M1076162

[11] Á. Cseh and T. Kavitha. Popular Matchings in Complete Graphs. In *Proceedings of the 38th IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS)*, pp. 17:1–17:14 (2018). doi:10.4230/LIPIcs.FSTTCS.2018.17

[12] R. Duan, S. Pettie, and H.-H. Su. Scaling Algorithms for Weighted Matching in General Graphs. *ACM Transactions on Algorithms*, 14(1): 8:1–8:35 (2018). doi:10.1145/3155301

[13] Y. Faenza, T. Kavitha, V. Powers, and X. Zhang. Popular Matchings and Limits to Tractability. In *Proceedings of the 30th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pp. 2790–2809 (2019). doi:10.1137/1.9781611975482.173

[14] J. Fischer. Optimal Succinctness for Range Minimum Queries. In *Proceedings of the 9th Latin American Symposium on Theoretical Informatics (LATIN)*, pp. 158–169 (2010). doi:10.1007/978-3-642-12200-2_16

[15] M. L. Fredman. On computing the length of longest increasing subsequences. *Discrete Applied Mathematics*, 11(1): 29–35 (1975). doi:10.1016/0012-365X(75)90103-X

[16] D. Gale and L. S. Shapley. College Admissions and the Stability of Marriage. *American Mathematical Monthly*, 69: 9–15 (1962). doi:10.2307/2312726

[17] D. Gale and M. Sotomayor. Some remarks on the stable matching problem. *Discrete Applied Mathematics*, 11(3): 223–232 (1985). doi:10.1016/0166-218X(85)90074-5

[18] P. Gärdenfors. Match making: Assignments based on bilateral preferences. *Behavioral Science*, 20: 166–173 (1975). doi:10.1002/bs.3830200304

[19] A. V. Goldberg. Scaling Algorithms for the Shortest Paths Problem. *SIAM Journal on Computing*, 24(3): 494–504 (1995). doi:10.1137/S0097539792231179

[20] S. Gupta, P. Misra, S. Saurabh, and M. Zehavi. Popular Matching in Roommates Setting is NP-hard. In *Proceedings of the 30th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pp. 2810–2822 (2019). doi:10.1137/1.9781611975482.174

[21] D. Gusfield and R. W. Irving. The Stable Marriage Problem: Structure and Algorithms. MIT Press (1989).

[22] K. Hamada, S. Miyazaki, and K. Okamoto. Strongly Stable and Maximum Weakly Stable Noncrossing Matchings. In *Proceedings of the 31st International Workshop on Combinatorial Algorithms (IWOCA)*, pp. 304–315 (2020). doi:10.1007/978-3-030-48966-3_23

[23] C.-C. Huang and T. Kavitha. Near-Popular Matchings in the Roommates Problem. *SIAM Journal on Discrete Mathematics*, 27(1): 43–62 (2013). doi:10.1137/110852838

[24] C.-C. Huang and T. Kavitha. Popular matchings in the stable marriage problem. *Information and Computation*, 222: 180–194 (2013). doi:10.1016/j.ic.2012.10.012

[25] C.-C. Huang, T. Kavitha, D. Michail, and M. Nasre. Bounded Unpopularity Matchings. *Algorithmica*, 61: 738–757 (2011). doi:10.1007/s00453-010-9434-9

[26] A. Hylland and R. Zeckhauser. The Efficient Allocation of Individuals to Positions. *Journal of Political Economy*, 87(22): 293–314 (1979).

[27] R. W. Irving. An efficient algorithm for the "stable roommates" problem. *Journal of Algorithms*, 6: 577–595 (1985). doi:10.1016/0196-6774(85)90033-1

[28] R. W. Irving. Stable marriage and indifference. *Discrete Applied Mathematics*, 48(3): 261–272 (1994). doi:10.1016/0166-218X(92)00179-P

[29] R. W. Irving, T. Kavitha, K. Mehlhorn, D. Michail, and K. Paluch. Rank-maximal matchings. *ACM Transactions on Algorithms*, 2(4): 602–610 (2006). doi:10.1145/1198513.1198520

[30] T. Itoh and O. Watanabe. Weighted random popular matchings. *Random Structures & Algorithms*, 37(4): 477–494 (2010). doi:10.1002/rsa.20316

[31] Y. Kajitami and T. Takahashi. The noncross matching and applications to the 3-side switch box routing in VLSI layout design. In *Proceedings of the IEEE International Symposium on Circuits and Systems*, pp. 776–779 (1986).

[32] T. Kavitha. A Size-Popularity Tradeoff in the Stable Marriage Problem. *SIAM Journal on Computing*, 43(1): 52–71 (2014). doi:10.1137/120902562

[33] T. Kavitha. Popular Matchings of Desired Size. In *Proceedings of the 44th International Workshop on Graph-Theoretic Concepts in Computer Science (WG)*, pp. 306–317 (2018). doi:10.1007/978-3-030-00256-5_25

[34] T. Kavitha, K. Mehlhorn, D. Michail, and K. E. Paluch. Strongly stable matchings in time $O(nm)$ and extension to the hospitals-residents problem. *ACM Transactions on Algorithms*, 3(2): 1–18 (2007). doi:10.1145/1240233.1240238

[35] T. Kavitha, J. Mestre, and M. Nasre. Popular mixed matchings. *Theoretical Computer Science*, 412(24): 2679–2690 (2011). doi:10.1016/j.tcs.2010.03.028

[36] T. Kavitha and C. D. Shah. Efficient Algorithms for Weighted Rank-Maximal Matchings and Related Problems. In *Proceedings of the 17th International Symposium on Algorithms and Computation (ISAAC)*, pp. 153–162 (2006). doi:10.1007/11940128_17

[37] M. Mahdian. Random popular matchings. In *Proceedings of the 7th ACM Conference on Electronic Commerce (EC)*, pp. 238–242 (2006). doi:10.1145/1134707.1134733

[38] F. Malucelli, T. Ottmann, and D. Pretolani. Efficient labelling algorithms for the maximum noncrossing matching problem. *Discrete Applied Mathematics*, 47(2): 175–179 (1993). doi:10.1016/0166-218X(93)90090-B

[39] D. Manlove and C. T. S. Sng. Popular Matchings in the Capacitated House Allocation Problem. In *Proceedings of the 14th Annual European Symposium on Algorithms (ESA)*, pp. 492–503 (2006). doi:10.1007/11841036_45

[40] R. M. McCutchen. The Least-Unpopularity-Factor and Least-Unpopularity-Margin Criteria for Matching Problems with One-Sided Preferences. In *Proceedings of the 15th Latin American Symposium on Theoretical Informatics (LATIN)*, pp. 593–604 (2008). doi:10.1007/978-3-540-78773-0_51

[41] E. McDermid and R. W. Irving. Popular matchings: structure and algorithms. *Journal of Combinatorial Optimization*, 22: 339–358 (2011). doi:10.1007/s10878-009-9287-9

[42] J. Mestre. Weighted popular matchings. *ACM Transactions on Algorithms*, 10(1): 2:1–2:16 (2014). doi:10.1145/2556951

[43] A. E. Roth. On the Allocation of Residents to Rural Hospitals: A General Property of Two-Sided Matching Markets. *Econometrica*, 54(2): 425–427 (1986). doi:10.2307/1913160

[44] A. E. Roth. The Evolution of the Labor Market for Medical Interns and Residents: A Case Study in Game Theory. *Journal of Political Economy*, 92(6): 991–1016 (1984). doi:10.1086/261272

[45] A. E. Roth and A. Postlewaite. Weak Versus Strong Domination in a Market with Indivisible Goods. *Journal of Mathematical Economics*, 4: 131–137 (1977).

[46] A. E. Roth and M. A. O. Sotomayor. Two-Sided Matching: A Study in Game-Theoretic Modeling and Analysis (Econometric Society Monographs). Cambridge University Press (1990). doi:10.1017/CCOL052139015X

[47] S. Ruangwises and T. Itoh. Random Popular Matchings with Incomplete Preference Lists. *Journal of Graph Algorithms and Applications*, 23(5): 815–835 (2019). doi:10.7155/jgaa.00513

[48] S. Ruangwises and T. Itoh. Stable Noncrossing Matchings. In *Proceedings of the 30th International Workshop on Combinatorial Algorithms (IWOCA)*, pp. 405–416 (2019). doi:10.1007/978-3-030-25005-8_33

[49] S. Ruangwises and T. Itoh. Unpopularity Factor in the Marriage and Roommates Problems. *Theory of Computing Systems* (2020). doi:10.1007/s00224-020-09978-5

[50] B. Söderberg. General formalism for inhomogeneous random graphs. *Physical Review E*, 66(6): 066121 (2002). doi:10.1103/PhysRevE.66.066121

[51] J. J. M. Tan. A necessary and sufficient condition for the existence of a complete stable matching. *Journal of Algorithms*, 12(1): 154–178 (1991). doi:10.1016/0196-6774(91)90028-W

[52] P. Widmayer and C. K. Wong. An optimal algorithm for the maximum alignment of terminals. *Information Processing Letters*, 10: 75–82 (1985). doi:10.1016/0020-0190(85)90067-5

[53] Y. Yuan. Residence exchange wanted: A stable residence exchange problem. *European Journal of Operational Research*, 90: 536–546 (1996). doi:10.1016/0377-2217(94)00358-0