Technische Universität Berlin
Electrical Engineering and Computer Science
Institute of Software Engineering and Theoretical Computer Science
Algorithmics and Computational Complexity (AKT)

Robustness of Stable Matchings: Complexity and Experiments

Kimon Böhmer

Thesis submitted in fulfillment of the requirements for the degree
"Bachelor of Science" (B. Sc.) in the field of Computer Science

August 2023

First reviewer: Prof. Dr. Mathias Weller
Second reviewer: Prof. Dr. Stefan Schmid
Supervisor: Niclas Boehmer

Hiermit erkläre ich, dass ich die vorliegende Arbeit selbstständig und eigenhändig sowie ohne unerlaubte fremde Hilfe und ausschließlich unter Verwendung der aufgeführten Quellen und Hilfsmittel angefertigt habe.

Die selbstständige und eigenhändige Ausfertigung versichert an Eides statt

Berlin, den 22.8.2023 Kimon Böhmer

Datum Unterschrift

Zusammenfassung

Wir untersuchen die Robustheit von stabilen Matchings gegen Vertauschungen in den Präferenzen der Agenten sowie gegen die Löschung von Agenten mit einem speziellen Fokus auf dem bipartiten STABLE MARRIAGE-Problem. Neben stabilen Matchings betrachten wir auch die Robustheit von stabilen Paaren und Agenten, also Paaren oder Agenten, die in einem stabilen Matching enthalten sind. Dabei analysieren wir die Komplexität, die kleinste Zahl an Änderungen zu berechnen, um Instabilität zu erreichen (*Worst-Case-Robustheit*) und finden ein komplexitätstheoretisch gemischtes Bild vor. Zudem analysieren wir die Komplexität, die Stabilitätswahrscheinlichkeit nach Anwendung von einer gewissen Zahl an zufälligen Änderungen (*Average-Case-Robustheit*) zu berechnen, und treffen dabei ausschließlich auf Schwere-Resultate. Genauer gesagt zeigen wir #P-Schwere für die zwei Zählprobleme #MATCHING-SWAP und #AGENT-DELETE, bei denen nach der Anzahl an möglichen Änderungen gesucht wird, sodass ein Matching bzw. Agent stabil bleibt. Außerdem führen wir ausführliche Experimente bezüglich der Average-Case-Robustheit auf synthetisch erzeugten Instanzen durch. Unsere Ergebnisse zeigen auf, dass stabile Matchings auf unseren synthetischen Daten höchst unrobust sind, zumal es meistens ausreicht, zwei zufällige Vertauschungen in jeder Präferenzliste auszuführen, um ein stabiles Matching instabil zu machen. Wir beobachten außerdem, dass sich verschiedene stabile Matchings derselben Instanz signifikant in ihrer Robustheit unterscheiden können. Zum Beispiel sind vom berühmten Gale-Shapley-Algorithmus produzierte Matchings meistens weniger robust als stabile Matchings, die auf Fairness zwischen den beiden Seiten achten. Wir führen zudem eine simple Robustheits-Heuristik namens „blocking pair proximity“ ein, die sehr stark mit der beobachteten Robustheit unserer synthetischen Daten korreliert. Von dieser Heuristik leiten wir eine neue Methode ab, um robuste stabile Matchings zu berechnen. Auf der anderen Seite stellen wir fest, dass Paare von Agenten einehöhere Wahrscheinlichkeit haben, nach einer Reihe von zufälligen Änderungen stabil zu bleiben, sich die Robustheit von stabilen Paaren derselben Instanz jedoch erheblich unterscheiden kann.

Abstract

Focusing on the bipartite Stable Marriage problem, we investigate the robustness of stable matchings against swaps in the agents' preferences and against deletions of agents. Besides stable matchings, we also consider the robustness of stable pairs and agents, i.e. pairs or agents that are contained in a stable matching. We analyse the complexity of computing the smallest number of changes that lead to instability (*worst-case robustness*) and find a mixed complexity picture. Furthermore, we analyse the complexity of computing the probability that a matching, pair or agent remains stable after some changes have been performed at random (*average-case robustness*), exclusively encountering hardness. More precisely, we show #P-hardness for two counting problems called #Matching-Swap and #Agent-Delete, where the task is to count the number of possible changes such that the matching or agent remains stable. Additionally, we conduct extensive experiments on synthetic instances regarding the average-case robustness of matchings and pairs. Our results reveal that stable matchings in our synthetic data are highly unrobust, as performing two random swaps in each preference list is usually sufficient to make a stable matching unstable. We find out that different stable matchings of the same instance can differ significantly in their robustness. For example, stable matchings produced by the popular Gale-Shapley algorithm tend to be less robust than stable matchings that consider fairness between the two sides. We also introduce a simple robustness measure called blocking pair proximity which correlates very strongly with the observed robustness of our synthetic data. From this measure, we derive a new method to compute robust stable matchings. On the other hand, we find that pairs of agents have a higher probability of remaining stable after some random changes, but the robustness of stable pairs of the same instance can differ significantly.

Contents

Chapter 1

Introduction

1.1 Motivation

In two-sided stable matching problems, we are given two sets of agents with each agent having preferences over the agents from the other set. The objective is to find a stable matching of agents from one side to agents from the other side, i.e. a matching where no pair of agents prefers each other to their current partner. Since their introduction by Gale and Shapley [GS62], such matching under preference problems have been studied extensively in economics and computer science (see the survey of Manlove [Man13]) and many real-world applications including assignment of students to schools [Abd+05], assignment of children to daycare places [KMT+11] or online dating [HHA10] have been identified.

Importantly, in many applications, agents remain matched for a longer period of time over which their preferences might change. Then, it could happen that an initially stable matching later in time contains some pair of agents that is not included in the matching and prefers each other to their matched agents, making the matching unstable. Additionally, the preferences that were submitted initially could turn out to be inaccurate, e.g. because some agent gains more information about the agents of the other set. Clearly, it is desirable that a matching remains stable when only few changes in the preferences of the agents occur. This observation has motivated different lines of research in the stable matching literature, for instance, the study of robust stable matchings, that are matchings which are likely to remain stable even if some changes are performed [CSS21; Gan+22; Gan+23; MV18] or that can be easily adapted to be stable again [Gaj+20; Gen+17a; Gen+17b; Gen+19].

In our work, we take a new view at robustness and stable matchings. Instead of computing different types of robust stable matchings, we quantify the robustness of a given stable matching. In addition to making the whole process more transparent and predictable, we see multiple possible use cases for our robustness measures. For instance, it can be used to make an informed decision between different proposed stable matchings or more generally might serve as an additional criterion to decide between different matching algorithms. Moreover, if a matching is detected to be very non-robust, the organisers of the matching market can initiate some countermeasures such as checking whether preferences were elicited correctly or ask agents to reevaluate their preferences after providing them with further information or guidance.

Naturally, the question arises how to measure and compute the robustness of a stable matching. We will address this question in the first part of our work. For real-world applications, different settings will lead to differently robust stable matchings. For instance, we will observe that if the agents prefer other agents that are similar to them (and, consequently, each agent has quite different preferences), the stable matchings tend to be less robust than in the case where all agents have similar preferences. Additionally, one may be interested in how robust a stable matching usually is (i.e. how many changes in the preferences of the agents can be allowed on average in order to still have a stable matching), and how to compute a robust stable matching. These questions will be the focus of the experimental part of our work.

1.2 Related work

Closest to our work are the papers of Aziz et al. [Azi+20; Azi+22] and of Boehmer et al. [Boe+21c]. Related to the worst-case robustness of stable matchings, Boehmer et al. [Boe+21c] initiated the study of (constructive) bribery problems in the stable matching literature, i.e., whether a given matching can be made stable using a limited number of changes. Following up on their work, Eiben et al. [Eib+23] and Bérczi, Csáji, and Király [BCK22] studied problems related to modifying an instance so that the existence of a stable matching fulfilling some desired property is ensured. Related to the average-case robustness of stable matchings, Aziz et al. [Azi+20; Azi+22] analysed different problems occurring when agent's preferences are uncertain. Closest to our work, they prove the computational intractability of computing the probability that a given matching is stable assuming that each agent provides us with a probability distribution over preference lists. This problem is a generalization of our #Matching-Swap problem (see Chapter 2 for a definition).
In addition to the study of finding robust stable matchings, there are also multiple lines of works in the stable matching literature motivated by the observation that agent's preferences change over time. For instance, in the study of dynamic stable matchings the goal is usually to adapt classic stability notions to dynamic settings [ALG20; BLY20; Boe+21c; DL05; Dov22; Liu21]. Moreover, various works have studied computational aspects of minimally adapting a given stable matching to re-establish stability after changes occurred [Bha+15; BHN22a; BHN22b; Bre+20; Fei+20; Gaj+20; KLM16].
While the study of bribery in stable matching instances is quite new, there are many works on this topic in the voting literature [FHH09; FR16; SYE13]. Similar to our approach, Baumeister and Hogrebe [BH23] and Boehmer et al. [Boe+21a; Boe+22] recently initiated the study of the robustness of election winners to random noise. In addition, the idea of using destructive swap bribery as a worst-case robustness measure has also been explored in the context of tournament solutions [BSS22; DP23] and group identification problems [Boe+23].

1.3 Overview

We contribute a new perspective on the study of the robustness of stable matchings to changes in the instance. In particular, we quantify the robustness of complete matchings, but also the robustness of different local configurations such as whether an agent pair can be included in a stable matching or whether an agent is assigned a partner. We focus on measuring the robustness against swaps in the agent's preferences and against the deletion of agents. In Chapter 3, we mainly focus on the complexity of computing the smallest amount of deletions to make a matching, pair or agent unstable (referred to as "destructive bribery" by Boehmer et al. [Boe+21c]). However, we also settle the complexity of two "constructive bribery" problems regarding pairs, i.e. computing the smallest number of swaps or deletions such that an agent is assigned in a stable matching. Moreover we generalise the stability criterion and consider the problem of computing the smallest number of swaps or deletions such that the given matching contains a certain number of blocking pairs.

In Chapter 4, we turn from the worst-case robustness to computing the average-case robustness of stable matchings, pairs and agents. For the average-case robustness, we want to compute the probability that a matching, pair or agent is stable after a certain number of changes have been performed. To compute this probability, we need to count the number of solutions to the corresponding decision problem and divide them by the total number of possible solutions. Therefore, computing the average-case robustness is computationally equivalent to solving the counting variant of the corresponding decision problem. Notably, our constructive and destructive decision problems both become equivalent in the counting setting.

The complete complexity picture is presented in Table 1.1. We find that the complexity decisively depends on the type of change and the object whose robustness we want to assess. Subsequently, for all problems for which we did not show that the decision variant is NP-hard, we show #P-hardness of the corresponding counting problem.[1] The presented reductions are quite involved and the #P-hardness results of the counting problems form a stark contrast with simple polynomial-time algorithms for the decision variant.

In the experimental part (Chapter 5), customizing the diverse synthetic dataset introduced by Boehmer, Heeger, and Szufa [BHS22], we perform extensive experiments related to the stability of stable matchings and pairs in case swaps in the agents preferences are performed. Among others, we find that almost all matchings in our synthetic instances can be made unstable by performing a single swap. Regarding robustness to random noise, motivated by the proven computational intractability of computing stability probabilities exactly, we explore a sampling-based approach building upon the Mallows model [Mal57] to analyse robustness to random changes. It turns out that stable matchings are generally still surprisingly non-robust, yet the degree of the non-robustness significantly varies between instances.

We also present a heuristic to measure the average-case robustness of a matching which builds upon the number of pairs that are close to being blocking, and show that it is

[1] #P is the counting analogue of NP. One consequence of this is that the existence of a polynomial time algorithm for a #P-hard problem implies that all problems in NP can be decided in polynomial time.

Change	Matching	Pair	Agent
Swap (Const.)	P (†)	NP-c. (†)	NP-c. (Theorem 3.11)
Swap (Dest.)	P (Theorem 3.3) #bps: NP-c. (Theorem 3.4)	NP-c. (Theorem 3.6)	NP-c. (Theorem 3.12)
Swap (Count.)	#P-c. (Theorem 4.3)	-	-
Delete (Const.)	NP-c. (†)	P (†)	P (Observation 3.7)
Delete (Dest.)	P (Theorem 3.1) #bps: NP-c. (Theorem 3.2)	?	P (Corollary 3.9)
Delete (Count.)	-	#P-c. (Corollary 4.17)	#P-c. (Theorem 4.8)

Table 1.1: Overview of our complexity results. Complexity results marked with (†) are due to Boehmer et al. [Boe+21c]. Counting problems marked with "-" are already intractable due to the corresponding decision problem. "NP-c" and "#P-c" stands for NP-complete and #P-complete. "#bps" stands for requiring a certain number of blocking pairs.

of excellent quality in practice. Notably, we also find that within an instance different initially stable matchings have a noticeably different robustness: For instance, matchings produced by the popular Gale-Shapley algorithm tend to be less robust than so-called summed-rank minimal stable matchings. We also propose a new way of computing a stable matching that we call robust stable matching and observe that it has an even higher average-case robustness than the summed-rank minimal stable matching.
Lastly, we conduct similar experiments on the robustness of stable pairs, i.e. pairs of agents that are included in at least one stable matching. We observe that stable pairs are in general much more robust than stable matchings, but the robustness between stable pair pairs of the same instance can vary a lot. Since the robustness of pairs depends on many factors, our robustness measure, called the *blocking score* of a pair, does not achieve an as strong correlation as the blocking pair proximity. Contrary to the matching setting, an initially unstable pair can have a quite high probability of being stable when random noise is added. A very rough overview of our main experimental findings is shown in Table 1.2.

	Matchings	Pairs
average-case robustness	• low (Finding 1)	• high (Finding 8)
correlation of simple robustness measures	• worst-case robustness: weak (Finding 2) • blocking pair proximity: very strong (Finding 3) • number of blocking pairs after some random noise is performed: medium (Finding 6)	• blocking score: medium (Finding 7)
difference in robustness for different stable matchings/pairs	• robust/summed-rank min matchings are more robust than the Gale-Shapley matching (Finding 4)	• robustness of stable pairs can vary a lot (Finding 9)
initially unstable	• very low stable matching probability (Finding 5)	• quite high stable pair probability possible (Finding 10)

Table 1.2: Overview of our experimental results.

Chapter 2

Preliminaries

We use the notation $[n] := \{1, ..., n\}$.

Stable Marriage. Let $\mathcal{I} = (U, W, \mathcal{P})$ denote an instance of the Stable Marriage (SM) problem, where $U = \{m_1, ..., m_n\}$ is the set of *men*, $W = \{w_1, ..., w_m\}$ is the set of *women* and each *agent* $a \in A := U \cup W$ has a *preference list* $\succ_a$, that is a complete and strict order over all agents of opposite gender. We denote this preference list of an agent $a \in A$ as $a : a_1 \succ a_2 \succ ... \succ a_n$ where a *prefers* agent x over y if $x \succ y$. We use the notation $x \succ ...$ to indicate that the order of the remaining agents after x is arbitrary. The *preference profile* of an instance is $\mathcal{P} := \{\succ_a \mid a \in A\}$. We write $rk_a(b)$ for the position that agent b has in the preference list of a, and define the inverse operation $p_a(i) := b \Leftrightarrow rk_a(b) = i$. For a preference list $\succ_x$ and two agents a, b we define the *distance* between a and b in $\succ_x$ as $|rk_x(a) - rk_x(b)|$.

A *matching* M is a set of pairs $\{u, w\}$ with $u \in U$ and $w \in W$ such that each agent is contained in at most one pair. Agents that are contained in a pair are *assigned* in M. Analogously, if an agent is not contained in any pair, they are *unassigned* in M. For a matching M and an agent a, we denote as $M(a)$ the agent a is matched to in M or $\perp$ if a is unassigned. A matching is *complete* if no agent is unassigned. For a matching M, a pair $\{u, w\}$ is called *blocking* if u prefers w to $M(u)$ or is unassigned and w prefers m to $M(w)$ or is unassigned. If a matching does not admit a blocking pair, it is called *stable*.

Decision and Counting complexity. P is the class of all problem that can be solved in polynomial time. NP is the class of all problems that can be solved by a nondeterministic Turing machine in polynomial time. A problem is called *NP-hard* if we can reduce each problem in NP to that problem in polynomial time. It is called *NP-complete* if it is NP-hard and contained in NP.
The complexity class #P contains all problems for which there is a nondeterministic Turing machine that contains exactly as many accepting branches as there are solutions to the problem. A problem is called *#P-hard* if we can Turing-reduce each problem in #P to that problem in polynomial time. It is called *#P-complete* if it is #P-hard and contained in #P.

Graphs and elections. We will briefly introduce graphs and elections since we use them for various proofs. An *undirected graph* is a pair $G = (V, E)$ of *vertices* V and *edges* E. An edge $e = \{u, v\}$ is a set of two vertices $u \neq v$. We say that a vertex v is *incident* to an edge e if $v \in e$ and v is *adjacent* to a vertex u if $\{v, u\} \in E$.
An *election* is a triple $\mathcal{E} = (V, C, \mathcal{P})$ of a set of *voters* V, a set of candidates C and a preference profile $\mathcal{P}$, which contains a *preference list* $\succ_v$, that is a complete and strict order over C, for each voter $v \in V$.

Types of Changes. We will consider two operations that change our SM instance. A *swap* operation swaps two neighbouring agents in the preference list of one agent.

Example 1. For a man $u \in U$ and four women $w_1, w_2, w_3, w_4 \in W$ let $u : w_2 \succ w_4 \succ w_3 \succ w_1$ be the preference list of u. There are three different swap operations, e.g. swapping w_4 and w_3 in u's preference list will result in the altered preference list $u : w_2 \succ w_3 \succ w_4 \succ w_1$.

A *delete* operation deletes an agent from the set of agents and thereby also from the preferences of all other agents.

Example 2. Let $\mathcal{I} = (U, W, \mathcal{P})$ be an SM instance with $U = \{u\}$ and $W = \{w_1, w_2, w_3, w_4\}$ and let the preference list of u be $u : w_2 \succ w_4 \succ w_3 \succ w_1$. There are five different delete operations, e.g. deleting woman w_3 will result in an altered instance where $W = \{w_1, w_2, w_4\}$ and $u : w_2 \succ w_4 \succ w_1$.

Worst-Case Robustness. Let $\mathcal{I} = (U, W, \mathcal{P})$ be an SM instance. The *swap distance* between two preference lists $\succ_a$ and $\succ'_a$, denoted by $\kappa(\succ_a, \succ'_a)$, is the minimum number of swap operations needed to obtain $\succ'_a$ from $\succ_a$. The *swap distance* between two preference profiles $\mathcal{P}$ and $\mathcal{P}'$ over the same set of agents A is $\kappa(\mathcal{P}, \mathcal{P}') := \sum_{a \in A} \kappa(\succ_a, \succ'_a)$. A matching M is *d-robust* if for all preference profiles $\mathcal{P}'$ at swap distance d from $\mathcal{P}$, M is stable. The *worst-case robustness* of M in $\mathcal{I}$ is the largest number d such that M is d-robust.

Example 3. Let $U = \{u_1, .., u_n\}$, $W = \{w_1, ..., w_n\}$ with preferences

$$u_i : w_i \succ w_{i+1} \succ ... \succ w_n \succ w_1 \succ ... \succ w_{i-1}$$

and

$$w_i : u_i \succ u_{i+1} \succ ... \succ u_n \succ u_1 \succ ... \succ u_{i-1}$$

for all $i \in [n]$. Then matching $M := \{\{u_i, w_i\} \mid i \in [n]\}$ is stable and $n-1$-robust, as for each blocking pair $\{u, w\}$, if b swap operations are needed to place w in front of $M(u)$ in u's preferences, then $n - b$ swap operations are needed to place u in front of $M(w)$ in w's preferences.

Theorem 2.1. *Let $\mathcal{I} = (U, W, \mathcal{P})$ be a* Stable Marriage*-instance with $|U| = |W| = n > 1$ and let M be a stable matching for that instance. Then, $n - 1$ is a tight upper bound for the worst-case robustness of M.*

Proof. For contradiction, suppose that there is an instance $\mathcal{I}$ that admits a d-robust stable matching M, where $d \geq n$. Since M is stable, there exists an agent $a \in A$ that does not rank $M(a)$ last. Let a' be the agent that directly follows $M(a)$ in the preference list of a. As the preference list of a' ranks n agents, it takes at most $n-1$ swaps to swap a in front of $M(a')$ in the preferences of a'. Thus, by additionally swapping a' and $M(a)$ in the preference list of a, we can create the blocking pair $\{a, a'\}$ with at most n swaps, contradicting the d-robustness of M. Tightness of the upper bound follows from Example 3. $\square$

Average-Case Robustness. Given an SM instance $\mathcal{I} = (U, W, \mathcal{P})$, a matching M and a budget ℓ, we define the *stable matching probability* of M as the probability of M being stable after ℓ swaps have been performed, i.e. the number of preference profiles at swap distance ℓ from $\mathcal{P}$ where M is stable divided by the total number of preference profiles at swap distance ℓ from $\mathcal{P}$.

Example 4. Let $U = \{u_1, u_2, u_3\}, W = \{w_1, w_2, w_3\}$ with preferences

$$u_1 : w_2 \succ w_1 \succ w_3, \qquad w_1 : u_2 \succ u_1 \succ u_3$$

$$u_2 : w_3 \succ w_2 \succ w_1, \qquad w_2 : u_3 \succ u_2 \succ u_1$$

$$u_3 : w_1 \succ w_3 \succ w_2, \qquad w_3 : u_1 \succ u_3 \succ u_2$$

Then matching $M := \{\{u_1, w_1\}, \{u_2, w_2\}, \{u_3, w_3\}\}$ is stable but has a low average-case robustness of $\frac{6}{12} = 0.5$ for budget $\ell = 1$: While the six operations that improve $M(a)$ in the preferences of a do not affect the stability, any of the six operations that weaken $M(a)$ in the preferences of a and thereby improve another agent a' will create the blocking pair $\{a, a'\}$. For $\ell = 2$ the average-case robustness decreases to $\frac{3}{8}$.

A matching may have a low worst-case robustness but can be quite robust in the average case:

Example 5. Consider an SM instance $\mathcal{I} = (U, W, \mathcal{P})$ with $U = \{u_1, ..., u_n\}, W = \{w_1, ..., w_n\}$ and $\mathcal{P}$ defined as follows:

$$u_i : w_i \succ w_{i+1} \succ ... \succ w_n \succ w_1 \succ ... \succ w_{i-1}$$

for all $i \in [n]$, and

$$\begin{aligned} w_2 &: u_1 \succ ... \succ u_n \\ w_i &: u_i \succ u_{i+1} \succ ... \succ u_n \succ u_1 \succ ... \succ u_{i-1} \end{aligned}$$

for all $i \in [n] \setminus \{2\}$. Matching $M := \{\{u_i, w_i\} \mid i \in [n]\}$ is stable but swapping w_1 and w_2 in the preference list of m_1 makes M unstable because of the blocking pair $\{m_1, w_2\}$. Thus, the worst-case robustness is 1. However, the average-case robustness is still relatively high as all other pairs need n swaps to become blocking. More accurate, the stable matching probability of M is $1 - \frac{1}{2n(n-1)}$ for $\ell = 1$ and $1 - \frac{2n(n-1)-1}{2n(n-1)(n-2)+\binom{2n}{2}(n-1)^2}$ for $\ell = 2$.

Robustness of Stable Pairs. For an SM instance $\mathcal{I} = (U, W, \mathcal{P})$, a pair $\{u, w\}$ is called *stable* if there is some stable matching M such that $\{m, w\} \in M$. It is *unstable*, if no such stable matching exists. Analogously to stable matchings, we can define robustness for stable pairs. For $|U| = |W| = n$, a stable pair cannot be n-robust because of the same argument as for stable matchings (see Theorem 2.1). In the average-case, the probability of a pair remaining stable after a number of swaps is higher than for matchings, because m or w must be swapped down in order to make $\{m, w\}$ unstable.

Robustness of Stable Agents. For an SM instance $\mathcal{I} = (U, W, \mathcal{P})$, we call an agent $a \in A$ *stable* if there is some stable matching M such that a is assigned. The agent is *unstable* if they are unassigned for all stable matchings. Notice that by the Rural hospitals theorem (McVitie and Wilson [MW70]), a is stable and therefore contained in one stable matching if and only if the agent is contained in all stable matchings. Again, we can define robustness for agents analogously to robustness for matchings and pairs. If $|U| \leq |W|$, then any man is trivially always stable and vice versa. For $|U| \geq |W|$, see Example 6 for a very robust agent.

Example 6. Let $\mathcal{I} = (U, W, \mathcal{P})$ be an SM instance with $U = \{u_1, ..., u_n\}$, $W = \{w_1, ..., w_{n-1}\}$ and preferences

$$u_i : w_1 \succ ... \succ w_{n-1} \qquad\qquad w_j : u_1 \succ ... \succ u_n$$

for all $i \in [n]$ and $j \in [n-1]$. Then, u_1 has a worst-case robustness of $\frac{n(n-1)}{2}$, as swapping down u_1 by i places in the preferences of w_i for every $i \in [n-1]$ results in an instance where u_1 is unstable. Notice that there is no sequence of less swaps to make u_1 unstable as we need to swap every u_i except u_1 in front of u_1 in the preferences of at least one $w \in W$, which requires at least $\sum_{i=2}^{n}(i-1) = \frac{n(n-1)}{2}$ swaps.

Computational problems. The general problem formulation in the worst-case robustness setting will be as follows:

Input: An SM instance $\mathcal{I} = (U, W, \mathcal{P})$, a matching M/a man-woman-pair $\{m^*, w^*\} \in U \times W$/an agent $a \in A$ and a budget $\ell \in \mathbb{N}$.
Question: Is it possible to perform at most ℓ swaps or deletions such that $M/\{m^*, w^*\}/a$ is not stable?
In the average-case setting, since we want to count the number of solutions, the problem formulation will be different:
Input: An SM instance $\mathcal{I} = (U, W, \mathcal{P})$, a matching M/a man-woman-pair $\{m^*, w^*\} \in U \times W$/an agent $a \in A$ and a budget $\ell \in \mathbb{N}$.
Question: How many preference profiles at swap/deletion distance ℓ are there such that $M/\{m^*, w^*\}/a$ is stable?
However, we will slightly adapt these formulations for some specific problems. For example, when considering the worst-case robustness of agents against deletions, it is sensible to not allow the deletion of the designated agent (since then, deleting that agent would make them trivially unstable).

Chapter 3

Worst-Case Robustness

In this chapter, we will analyse the worst-case robustness of SM instances against swaps and deletions, i.e. the number of changes needed to make a matching, pair or agent unstable. These computational problems correspond to the "destructive bribery" setting by Boehmer et al. [Boe+21c]. Whenever the corresponding problem is open, we will also study the "constructive bribery" setting, where we want to make a matching, pair or agent stable with the smallest amount of swaps.
Notice that we defined the worst-case robustness as the largest number d such that the matching, pair or agent cannot be made unstable with d swaps. In this chapter, however, we look for the smallest number d of changes such that the matching, pair or agent can be made unstable. Clearly, when subtracting 1 from the result of our computational problems, we obtain the worst-case robustness as defined initially.

3.1 Robust Matchings

We will first consider the complexity of computing the worst-case robustness of a matching. These problems are computationally straightforward; Therefore, as a generalization of this setting, we will require that the given matching contains a certain number of blocking pairs (instead of at least one). With this additional requirement, the problems become intractable. We show this by reductions from Independent Set and Clique, which are well known to be NP-complete (Karp [Kar]).

3.1.1 Destructive-Matching-Delete

We will start with the change operation of deleting agents. The problem is defined as follows:

Problem 1: Destructive-Matching-Delete
Input: An SM instance $\mathcal{I} = (U, W, \mathcal{P})$, a matching M and a budget $\ell \in \mathbb{N}$.
Question: Is it possible to delete at most ℓ agents such that there is no $M' \subseteq M$ such that M' is stable?

We require that no subset of M is stable, as otherwise one could just delete an arbitrary agent that was assigned in M. However, the problem still remains trivial:

Theorem 3.1. DESTRUCTIVE-MATCHING-DELETE *can be solved in* $\mathcal{O}((n+m)^2)$ *time.*

Proof. There is a trivial solution for $\ell \geq 2$: We delete one woman $w \in W$ and one man $u \in U$ such that $\{u, w\} \notin M$. Any stable matching M' in the resulting instance will contain the edge $\{u, w'\}$ for some $w' \in W \setminus \{w\}$ and therefore $M' \not\subseteq M$. If our budget is smaller than 2, we first check whether M is stable and accept if it is not. If $\ell = 1$ and M is stable, for all $a \in A$, we check whether there exists an agent $a' \in A$ who prefers $M(a)$ to $M(a')$. In that case, $\{a', M(a)\}$ blocks $M' := M \setminus \{\{a, M(a)\}\}$ and we accept. Else, we decline.
To check the stability of M, we need $\mathcal{O}((n+m)^2)$ time. We assume that given an agent it is possible to determine the preferred partner out of two agents of opposite gender in constant time (e.g. by computing the rank of all agents in every preference list in the beginning). Then, searching for an agent a' as described above is possible in $\mathcal{O}(n)$ time for each deleted agent a. Therefore, the total running time is $\mathcal{O}((n+m)^2)$. □

We saw that it is polynomial-time solvable to create one blocking pair given a certain number of deletions. However, one may be interested in deleting agents such that not only one, but a certain number of pairs are blocking. Therefore, we now specify the number of blocking pairs that we want to achieve.

Problem 2: NUM-BLOCKING-DELETE
Input: An SM instance $\mathcal{I} = (U, W, \mathcal{P})$, a matching M, a blocking-pair-threshold $b \in \mathbb{N}$ and a budget $\ell \in \mathbb{N}$.
Question: Is there a set $A' \subseteq A$ with $|A'| \leq \ell$ such that after deleting all agents from A' from the instance, b pairs are blocking for $M \setminus \{\{m, w\} \mid \{m, w\} \cap A' \neq \emptyset\}$?

Theorem 3.2. NUM-BLOCKING-DELETE *is NP-complete.*

Proof. We reduce from INDEPENDENT SET, where we are given a graph $G = (V, E)$ and an integer k and we want to decide whether there is a set $V' \subseteq V$ with $|V'| = k$ such that for all $u, v \in V'$ it holds that $\{u, v\} \notin E$. We construct our NUM-BLOCKING-DELETE instance $\mathcal{I}'$ as follows:
For each $v \in V$, we create a man m_v and a woman w_v. Let $\delta(v)$ denote the degree of v and let $u_1, ..., u_{\delta(v)}$ be the neighbours of v. The preferences of the agents are as follows:

$$m_v : w_v \succ ...$$
$$w_v : m_{u_1} \succ ... \succ m_{u_{\delta(v)}} \succ ...$$

Furthermore, we introduce men m'_i and women w'_i for each $i \in \{1, ..., 2k + |V|\}$. For any i, let $u^i_1, ..., u^i_{k(i)}$ be the vertices with degree $< i$. For all $i \in \{1, ..., |V|\}$ the preferences of m'_i, w'_i are the following:

$$m'_i : w'_i \succ ...$$
$$w'_i : m_{u^i_1} \succ ... \succ m_{u^i_{k(i)}} \succ m'_i \succ ...$$

For the remaining $2k$ agents, i.e. m'_i, w'_i for all $i \in \{|V| + 1, ..., |V| + 2k\}$, all m'_i rank w'_i as their top choice and all w'_i rank all m_v before m'_i before all other agents. Finally, we

set $\ell := k$, $b := k(n + 2k)$ and $M := \{m_v, w_v \mid v \in V\} \cup \{m'_i, w'_i \mid i \in \{1, ..., |V| + 2k\}\}$. Notice that M is stable since every man is matched to his top choice.
($\Rightarrow$) Assume that G contains an independent set V' of size k. We claim that after deleting w_v for all $v \in V'$, b pairs are blocking for M. For each $v \in V'$, after deleting w_v, m_v is unassigned in M. Therefore, each woman w that prefers m_v to $M(w)$ will form a blocking pair with m_v. By construction, this is the case for all w_u with $\{v, u\} \in E$. Notice that since the deleted women correspond to an independent set, no woman w_u is deleted in our solution and therefore all $\delta(v)$ women form a blocking pair. Moreover, $(n - \delta(v)) + 2k$ women w'_i prefer m_v to $M(w'_i)$ and since we do not delete any of these women in our solution, each of them forms a blocking pair with m_v. Altogether, m_v is involved in $\delta(v) + (n - \delta(v)) + 2k = n + 2k$ blocking pairs. As we deleted k women, we have $k(n + 2k)$ blocking pairs in total, fulfilling our required number of b blocking pairs while deleting exactly ℓ agents.
($\Leftarrow$) Assume that in $\mathcal{I}'$, we can delete at most ℓ agents such that b pairs are blocking for M. First, consider the case that some agent $a \in A \setminus \{w_v \mid v \in V\}$ is deleted. No agent a' of opposite gender prefers $M(a)$ to $M(a')$. Thus, the only possibility for $M(a)$ to be contained in a blocking pair is to delete the matched partner $M(b)$ of another agent b such that $\{M(a), b\}$ is blocking. As we can only delete $k - 1$ more agents, the deletion of a can produce at most $k - 1$ blocking pairs. Deleting w_v for any not yet deleted $v \in V$ instead of a will increase the number of blocking pairs, as $\{m'_i, w_v\}$ is blocking for all $i \in \{|V| + 1, ..., 2k + |V|\}$ such that m'_i is not deleted (at least $k + 1$). Therefore, we can assume that only agents w_v for some $v \in V$ are deleted.
Let D be the set of deleted women w_v. Consider a deleted woman $w_v \in D$. $M(w_v) = m_v$ can form a blocking pair with $w'_{\delta(v)}, ..., w'_{n+2k}$ and with all w_u such that $\{u, v\} \in E$, but not with any other woman, since all other women w prefer $M(w)$ to m_v and we assumed that $M(w)$ is not deleted. It follows that each m_v for which w_v was deleted can be part of at most $n + 2k$ blocking pairs. To achieve the total number of $k(n + 2k)$ blocking pairs, each of the k men m_v for which we deleted w_v must be part of exactly $n + 2k$ blocking pairs. Therefore, each m_v must form a blocking pair with every w_u such that $\{u, v\} \in E$. Consequently, no such w_u can be deleted. It follows that $\{u, v\} \notin E$ for every two deleted $w_u, w_v \in D$. By definition, $\{v \mid w_v \in D\}$ is an independent set of size $|D| = k$.
Membership in NP follows from the fact that we can count the number of blocking pairs of a matching in polynomial time. □

3.1.2 Destructive-Matching-Swap

We now consider swaps instead of deletions. Our according computational problem is defined as follows:

Problem 3: Destructive-Matching-Delete
Input: An SM instance $\mathcal{I} = (U, W, \mathcal{P})$, a matching M and a budget $\ell \in \mathbb{N}$.
Question: Is it possible to perform at most ℓ swaps such that M is not stable?

Theorem 3.3. Destructive-Matching-Swap *can be solved in $\mathcal{O}((n + m)^2)$ time.*

Proof. Let $c_a(b, b')$ denote the number of swaps needed to bring b in front of b' in the preferences of a. Our algorithm first computes $c_a(b, M(a))$ for all $a \in A$ and b of opposite gender and then determines the minimum number of swaps to make a matching M unstable by the following expression:

$$\min_{\{u,w\} \in (U \times W) \setminus M} c_u(w, M(u)) + c_w(u, M(w))$$

Over all pairs outside of the matching, we minimise the cost to make one of them blocking. There are $\mathcal{O}(n \cdot m)$ pairs and all $c_a(b, M(a))$ values can also be computed in quadratic time. □

As for the *Delete* setting, we now require that a certain number of pairs (instead of 1) are blocking. The resulting problem is defined as follows:

Problem 4: NUM-BLOCKING-SWAP
Input: An SM instance $\mathcal{I} = (U, W, \mathcal{P})$, a matching M, a blocking-pair-threshold $b \in \mathbb{N}$ and a budget $\ell \in \mathbb{N}$.
Question: Is it possible to perform at most ℓ swaps such that b pairs are blocking for M?

Theorem 3.4. NUM-BLOCKING-SWAP *is NP-complete.*

Proof. We reduce from CLIQUE, where we are given a graph G and an integer $k \in \mathbb{N}$ and we want to decide whether G contains a clique of size k, i.e. whether there is a $V' \subseteq V$ with $|V| = k$ and $\{v, u\} \in E$ for all $v, u \in V$ with $v \neq u$. We construct our NUM-BLOCKING-SWAP instance $\mathcal{I}'$ as follows: For each vertex, we introduce a man m_v and a woman w_v. For each edge, we introduce a man m_e and a woman w_e. Additionally, we set $p := nk + \binom{k}{2}$, $q := k(p+n) + \binom{k}{2}$ and we introduce $r := k(p+n) + (q+2)\binom{k}{2} + 1$ dummy men m_i^d and dummy women w_i^d for each $i \in \{1, ..., r\}$.
Let $v \in V$ be a vertex with incident edges $e_1, ..., e_s$. Let $e \in E$ be an edge with endpoints v_1, v_2. The preferences are as follows:

$$\begin{aligned}
m_v &: w_v \succ w_1^d \succ ... \succ w_p^d \succ w_{e_1} \succ ... \succ w_{e_s} \succ w_{p+1}^d \succ ... \succ w_r^d \succ ... \\
w_v &: m_v \succ m_1^d \succ ... \succ m_r^d \succ ... \\
m_e &: w_e \succ w_1^d \succ ... \succ w_r^d \succ ... \\
w_e &: m_e \succ m_1^d \succ ... \succ m_q^d \succ m_{v_1} \succ m_{v_2} \succ m_{q+1}^d \succ ... \succ m_r^d \succ ...
\end{aligned}$$

For each $i \in \{1, ..., k\}$, the dummy preferences are:

$$\begin{aligned}
m_i^d &: w_i^d \succ w_{i+1}^d \succ ... \succ w_r^d \succ w_1^d \succ ... \succ w_{i-1}^d \succ ... \\
w_i^d &: m_i^d \succ m_{i+1}^d \succ ... \succ m_r^d \succ m_1^d \succ ... \succ m_{i-1}^d \succ ...
\end{aligned}$$

We set the swap budget $\ell := r - 1 = k(p+n) + (q+2)\binom{k}{2}$ and the number of blocking pairs $b := 2\binom{k}{2}$. We set the designated matching to be:

$$M := \{\{m_v, w_v\} \mid v \in V\} \cup \{\{m_e, w_e\} \mid e \in E\} \cup \{\{m_i^d, w_i^d\} \mid i \in \{1, ..., r\}\}$$

Assume that there is a clique $C = (\{v_1, ..., v_k\}, \{e_1, ..., e_{\binom{k}{2}}\})$ in G. For each $v \in V(C)$, we swap w_v down by $p+n$ positions in the preferences of m_v. For each $e \in E(C)$, we swap down m_e by $q+2$ positions in the preferences of w_e. Now, for each of the $2\binom{k}{2}$ pairs $(v, \{v, w\})$ with $v, w \in V(C), e := \{v, w\} \in E(C)$, we have that m_v prefers w_e to w_v and w_e prefers m_v to m_e. It follows that $\{m_v, w_e\}$ is blocking. Notice that we used exactly $k(p+n) + (q+2)\binom{k}{2}$ swaps.

For the other direction, assume that in $\mathcal{I}'$, we can perform at most $k(p+n) + (q+2)\binom{k}{2}$ swaps such that $2\binom{k}{2}$ pairs are blocking. First notice that no blocking pair can contain a dummy agent a: One needs at least r swaps to swap $M(a)$ behind a non-dummy agent in the preferences of a and one needs r swaps to make two dummy agents with different index prefer each other over their matched agents (see Example 3), which exceeds the given swap budget. Moreover, no blocking pair can contain w_v or m_e for any $v \in V$ and $e \in E$, since in their preferences, one cannot swap their matched partners behind a non-dummy agent in less than r swaps and no dummy-agent is contained in a blocking pair. It follows that all blocking pairs must be $\{m_v, w_e\}$ for some $v \in V$ and $e \in E$. Let $e = \{u, u'\}$. Every w_e can only form blocking pairs with m_u and $m_{u'}$, since it cannot form blocking pairs with dummy agents and for any other non-dummy agent m, we would need at least r operations to swap m_e behind m in the preferences of w_e, which exceeds our budget.

Therefore, at least $\binom{k}{2}$ different women w_e must be contained in at least one blocking pair. On the other hand, for each w_e, we need at least $q+1$ swaps in the preferences of w_e to make w_e blocking with some m_v. Assume for contradiction that $\binom{k}{2}+1$ different women are contained in a blocking pair. We need

$$
\begin{aligned}
&(\binom{k}{2} + 1)(q+1) \\
=&\binom{k}{2}(q+1) + q + 1 \\
=&\binom{k}{2}(q+1) + k(p+n) + \binom{k}{2} + 1 \\
=&\binom{k}{2}(q+2) + k(p+n) + 1 \\
=&r
\end{aligned}
$$

swaps, exceeding our budget. Therefore, exactly $\binom{k}{2}$ women w_e are contained in a blocking pair. Thus, we need to spend at least $(q+1)\binom{k}{2}$ swaps in the preferences of women and there are $k(p+n)+(q+2)\binom{k}{2} - (q+1)\binom{k}{2} = k(p+n)+\binom{k}{2}$ swaps left in the preferences of men. For any m_v to be contained in a blocking pair, w_v must be swapped down at least $p+1$ positions in the preferences of m_v (all prior swaps involve dummy women). Suppose for contradiction that at least $k+1$ different men m_v are contained in a blocking pair. In total, at least $(k+1)(p+1)$ swaps are needed. We have:

$$\begin{aligned}
&(k+1)(p+1)\\
=&k(p+1)+nk+\binom{k}{2}\\
=&k(p+n+1)+\binom{k}{2}\\
>&k(p+n)+\binom{k}{2}
\end{aligned}$$

exceeding our remaining budget. Therefore, at most k men m_v can be contained in a blocking pair. However, every woman w_e can only be matched to men corresponding to endpoints of e (remember that we need at least r swaps to make any other pair blocking). Let E^* be the selected edges whose corresponding women are contained in a blocking pair. Since only k men m_v can be contained in a blocking pair, the endpoints of the selected edges are at most k vertices, i.e. $|\bigcup_{e\in E^*} e| \leq k$. By definition, $(\bigcup_{e\in E^*} e, E^*)$ is a clique of size k in G.

Our instance contains $2(n+m+k(nk+\binom{k}{2}+n)(\binom{k}{2}+1)+\binom{k}{2}^2+2\binom{k}{2})$ agents and thus can be computed in polynomial time.

The problem is contained in NP, since we can guess and perform the swaps and then count the blocking pairs in polynomial time. □

3.2 Robust Pairs

We now turn to the problem of computing the worst-case robustness of pairs. Given a stable pair, we want to measure how many operations are necessary so that the pair is not contained in any stable matching. Notably, Boehmer et al. [Boe+21c] used a definition where a pair must be unmatched in at least one matching (in contrast to all stable matchings). However, our definition has the advantage that the counting variants of the constructive and the destructive case are the same.

3.2.1 Destructive-Pair-Swap

Boehmer et al. [Boe+21c] showed that the constructive variant of this problem, where we want to perform ℓ swaps in order to get a stable matching that contains a designated pair, is NP-hard. They also remark that with this result, one can show NP-hardness for their destructive variant of the problem. However, as mentioned above, their destructive variant only requires that there is one stable matching where the designated pair is not contained. In contrast, our problem is defined as follows:

Problem 5: DESTRUCTIVE-PAIR-SWAP

Input: An SM instance $\mathcal{I} = (U, W, \mathcal{P})$, a man-woman-pair $\{m^*, w^*\} \in U \times W$ and a budget $\ell \in \mathbb{N}$.

Question: Is it possible to perform at most ℓ swaps such that no stable matching contains $\{m^*, w^*\}$?

We show that their reduction idea also works for our destructive variant. To this end, we first show that we can model a so-called *Add* operation, where one can add an agent from a predefined set of agents $U_{add} \cup W_{add}$, as a *Swap* operation. Afterwards, we show NP-hardness for the *Add* variant of our problem, implying NP-hardness for the original problem.

Theorem 3.5. *One can reduce* Destructive-Pair-Add *to* Destructive-Pair-Swap *in polynomial time.*

Proof. For every $a \in U_{add} \cup W_{add}$, we add two agents a', a'' where a'' has the same gender as a, but a' is of opposite gender. We set the preferences of the new agents to be $a' : a \succ a'' \succ ...$ and $a'' : a' \succ ...$ and add a' as the top choice of a.
Moreover, we introduce dummy men m_i^d and dummy women w_i^d for $i \in [q := (n-1)(l+1)]$. For any i, their preferences are $m_i^d : w_i^d \succ w^d i+1 \succ ... \succ w_q^d \succ w_1^d \succ ... \succ w_{i-1}^d \succ ...$ and for the dummy women analogous. Boehmer et al. [Boe+21c] showed that if we can perform at most ℓ swaps, all stable matchings must contain $\{\{m_1^d, w_1^d\}, ..., \{m_q^d, w_q^d\}\}$.
In every preference list, between any two adjacent agents, we place $\ell + 1$ dummy agents that have the corresponding dummy agent of opposite gender as their top choice. However, as the only exception, we do not place these dummy agents between a and a'' in the preference list of any a' where $a \in U_{add} \cup W_{add}$. This way, any other swap will not influence the stability of the matching.
Let $\mathcal{I}$ be an instance of Destructive-Pair-Add where there are ℓ agents $a \in U_{add} \cup W_{add}$ such that after their addition to the set of agents, no stable matching contains a designated pair $\{m^{**}, w^*\}$. For an instance $\mathcal{I}'$ of Destructive-Pair-Swap constructed as described above, for every added agent a, we swap a and a'' in the preferences of a'. Then, every stable matching contains $\{a', a''\}$ and a will be matched exactly as in $\mathcal{I}$ in all stable matchings. Every a where this swap is not performed must be matched to a' (mutual top-choice). Notice that all dummy agents can only be matched among themselves and therefore cannot form any blocking pair. It follows that after these swaps, no stable matching contains $\{m^*, w^*\}$ (if the opposite was the case, adding the corresponding agents in $\mathcal{I}$ would create a stable matching containing the designated pair).
For the other direction, notice that it is not possible to swap any two non-dummy agents b and c except for a and a'' in the preferences of a' for some agent $a \in U_{add} \cup W_{add}$, as there are $\ell + 1$ dummy agents between b and c, requiring $\ell + 1$ swaps, but our budget is only ℓ. Assume that a solution contains some swaps involving dummy agents. Then, when omitting these swaps, $\{m^*, w^*\}$ is still not contained in any stable matching (the same non-dummy pairs are blocking for all matchings). Therefore, we can assume that all swaps are between a and a'' in the preferences of some a'. With the same argument as for the other direction, swapping each of these swaps in $\mathcal{I}'$ is equivalent to adding the corresponding agent a in $\mathcal{I}$. □

Theorem 3.6. Destructive-Pair-Swap *is NP-complete.*

Proof. We show that Destructive-Pair-Add is NP-hard and by Theorem 3.5 the result follows. To this end, we reduce from Constructive-Pair-Add, where we are given a man-woman-pair $\{m^*, w^*\}$, a set of initially not added man and women $U_{add} \cup W_{add}$, an SM instance $\mathcal{I} = (U, W, \mathcal{P})$ and a budget $\ell \in \mathbb{N}$ and want to find a set

$X \subseteq U_{add} \cup W_{add}$ with $|X| \leq \ell$ such that after adding all agents from X to our instance $\mathcal{I}$, the pair $\{m^*, w^*\}$ is contained in a stable matching. We can assume that the given instance is constructed as for the reduction from CLIQUE by [Boe+21c]. The general idea of their construction is as follows: Man m^* can only be matched with w^* if every *penalizing woman* is matched to an *edge man*. There are $\binom{k}{2}$ penalizing women and each edge man can only be matched to a penalizing woman if one agent corresponding to the edge and two agents corresponding to the endpoints of the edge are added. Since the budget is $\binom{k}{2} + k$, the used edges must form a clique of size k. Notice that since in their construction $W_{add} = \emptyset$ holds, all agents that can be added are men.

We build the DESTRUCTIVE-PAIR-ADD instance $\mathcal{I}'$ as follows: We add a man m^{**} with preferences $m^{**} : w^* \succ \dots$. We modify the top two preferences of w^* to $m^* \succ m^{**}$. Finally, we set our designated pair to be $\{m^{**}, w^*\}$ and our budget $\ell' := \ell$.

Let $\mathcal{I}$ be an SM instance where one can add ℓ agents $a \in U_{add}$ such that $\{m^*, w^*\}$ is contained in a stable matching M. Let X be the set of these added agents. We claim that after adding the same set X of agents in $\mathcal{I}'$, no stable matching contains $\{m^{**}, w^*\}$. Assume for contradiction that after performing the same additions in $\mathcal{I}'$, there exists a stable matching M' containing $\{m^{**}, w^*\}$. Then, $\{m^*, w^*\}$ must not form a blocking pair in such a matching. As w^* prefers m^* to m^{**}, we need that m^* prefers $M'(m^*)$ to w^*. Therefore, $M'(m^*)$ has to be a penalizing woman $w^\dagger$. Since $\binom{k}{2}$ edge men were selected in $\mathcal{I}$, one edge man m_e which was selected is not matched to a penalizing woman. However, m_e prefers $w^\dagger$ to its partner and vice versa. It follows that the pair $\{w^\dagger, m_e\}$ is blocking, leading to a contradiction.

For the other direction, let $\mathcal{I}'$ be an SM instance where one can add at most ℓ' agents such that $\{m^{**}, w^*\}$ is not contained in any matching. As $\{m^{**}, w^*\}$ is not contained in any matching M', M' must contain $\{m^*, w^*\}$, because if w^* is matched to an agent $m \in U \setminus \{m^*, m^{**}\}$, the pair $\{m^{**}, w^*\}$ is blocking. Thus, every penalizing woman $w^\dagger$ must be matched to an agent she prefers to m^*. Such an agent must be an edge man a. In M, matching every penalizing woman to such an agent and matching m^* to w^*, is then obviously stable.

The problem is contained in NP, since we can guess a preference profile, check the swap distance from the original instance and compute all stable pairs in polynomial time (Gusfield [Gus87]). $\square$

For the worst-case robustness against *Delete* operations, the polynomial-time algorithm introduced by Boehmer et al. [Boe+21c] does not seem to carry over to our definition of worst-case robustness for pairs. The computational complexity of the problem stated below remains open.

Problem 6: DESTRUCTIVE-PAIR-DELETE

Input: An SM instance $\mathcal{I} = (U, W, \mathcal{P})$, a man-woman-pair $\{m^*, w^*\} \in U \times W$ and a budget $\ell \in \mathbb{N}$.

Question: Is it possible to delete at most ℓ agents from $A \setminus \{m^*, w^*\}$ such that no stable matching contains $\{m^*, w^*\}$?

3.3 Robust Agents

Finally, we consider the worst-case robustness of agents, i.e. the number of operations needed to make an agent unassigned. Since the constructive version of these problems, where one wants to find the smallest number of operations such that after performing them, a certain agent is assigned, has not been studied yet (contrary to the matching and pairs case), we also include the constructive version of our problems in our analysis.

3.3.1 Constructive-Agent-Delete

In the first setting, we want to find the smallest set of agents such that after deleting them, a certain agent is assigned in a stable matching:

Problem 7: CONSTRUCTIVE-AGENT-DELETE
Input: An SM instance $\mathcal{I} = (U, W, \mathcal{P})$, an agent $a \in A$ and a budget $\ell \in \mathbb{N}$.
Question: Is it possible to delete at most ℓ agents such that a is contained in a stable matching?

Boehmer et al. [Boe+21c] showed that one can find the minimum number of agent deletions in order to make a designated pair stable (CONSTRUCTIVE-PAIR-DELETE) in $\mathcal{O}((n+m)^2)$ time. Running their algorithm for all possible pairs containing the agent to be assigned a^* then obviously decides whether one can delete at most ℓ agents such that a^* is assigned in a stable matching.

Observation 3.7. CONSTRUCTIVE-AGENT-DELETE *can be solved in* $\mathcal{O}((n+m)^3)$ *time.*

3.3.2 Destructive-Agent-Delete

Problem 8: DESTRUCTIVE-AGENT-DELETE
Input: An SM instance $\mathcal{I} = (U, W, \mathcal{P})$, an agent $a \in A$ and a budget $\ell \in \mathbb{N}$.
Question: Is there a set $A' \subseteq (U \cup W) \setminus \{a\}$ with $|A'| \leq \ell$ such that after deleting the agents in A' from $\mathcal{I}$, a is not assigned in any stable matching?

For the destructive setting, we show that one can model an instance of DESTRUCTIVE-AGENT-DELETE as an instance of CONSTRUCTIVE-PAIR-DELETE. For this, let $\mathcal{I} = (U, W, \mathcal{P})$ be an SM instance with $a \in A := U \cup W$ and let $f(\mathcal{I}, a)$ be an SM instance that contains the agents of $\mathcal{I}$ and one additional agent a' of the opposite gender of a. All other agents rank a' last and a' ranks a first and all other agents in an arbitrary order.

Theorem 3.8. *Let $\mathcal{I} = (U, W, \mathcal{P})$ be an SM instance with $a \in A := U \cup W$. For each $A' \subseteq A \setminus \{a\}$, agent a is unstable in $\mathcal{I}$ after deleting all agents in A' if and only if pair $\{a, a'\}$ is stable in $f(\mathcal{I}, a)$ after deleting all agents in A'.*

Proof. Consider an instance of DESTRUCTIVE-AGENT-DELETE where we want to exclude a from a stable matching.
Assume we can delete agents $a_1, ..., a_i$ in $\mathcal{I}$ such that a is unassigned in a stable matching M. Then, after deleting the same agents in $f(\mathcal{I}, a)$, consider the matching $M' := M \cup \{a, a'\}$. Without loss of generality assume $a \in W$ and $a' \in U$. No pair $\{m, w\}$ with $m \neq a'$ can be blocking for M', as the same pair would also be blocking for M. No

blocking pair for M' can contain a', as a' is matched to its top choice. It follows that M' is stable.
Assume now for the other direction that we can delete agents $a_1, ..., a_i$ in $\mathcal{I}'$ such that $\{a, a'\}$ is contained in a stable matching M'. We claim that after performing the same deletions in $\mathcal{I}$, $M := M' \setminus \{a, a'\}$ is stable. Using the same argument as above, a blocking pair for M must include a. Assume for contradiction that such a blocking pair $\{a, a''\}$ exists for M. Then, a'' prefers a to $M(a) = M'(a)$. However, since a prefers a'' to a', $\{a, a''\}$ would be blocking in M', leading to a contradiction. □

Since f can be computed in linear time and Constructive-Pair-Delete can be computed in $\mathcal{O}((n+m)^2)$ time, we can conclude the following:

Corollary 3.9. Destructive-Agent-Delete *can be solved in $\mathcal{O}((n+m)^2)$ time.*

3.3.3 Constructive-Agent-Swap

For the *Swap* operation, we first consider the constructive setting again:

Problem 9: Constructive-Agent-Swap
Input: An SM instance $\mathcal{I} = (U, W, \mathcal{P})$, an agent $a \in A$ and a budget $\ell \in \mathbb{N}$.
Question: Is it possible to perform at most ℓ swaps in $\mathcal{I}$ such that a is assigned in a stable matching?

As for the Destructive-Pair-Delete problem, we will first briefly argue that one can model *Add* operations as *Swap* operations. Showing that Constructive-Agent-Add is NP-hard then implies NP-hardness for Constructive-Agent-Swap.

Lemma 3.10. *One can reduce* Constructive-Agent-Add *to* Constructive-Agent-Swap *in polynomial time.*

Proof. We model the *Add* operations by *Swap* operations analogously to Theorem 3.6: For each $a \in U_{add} \cup W_{add}$ in the original instance $\mathcal{I}$, we add an agent of opposite gender a' and an agent a'' of the same gender as a, with preference lists $a' : a \succ a'' \succ ...$ and $a'' : a' \succ ...$ and add a' as the top choice of a. We introduce $r := (\ell + 1)(|U| + |W|)$ dummy men m_d^i and the same number of dummy women w_d^i who will prevent undesired swaps. For every adjacent pair $a_u \succ a_v$ in a preference list of the original instance such that u and v are not the top two choices or that the preference list belongs to an agent who is not a' for some $a \in U_{add} \cup W_{add}$, we add ℓ dummy agents of opposite gender $m_{i\ell}, ..., m_{(i+1)\ell}$ (or $w_{i\ell}, ..., w_{(i+1)\ell}$) between u and v, where u is ranked in i-th place in the original preference list. For the top two choices of an agent a' with $a \in U_{add} \cup W_{add}$, we do not add these dummy agents, thus allowing swaps between these two agents to model an *Add* operation. □

Theorem 3.11. Constructive-Agent-Swap *is NP-complete.*

Proof. We show that we can reduce Constructive-Pair-Add to Constructive-Agent-Add and then Lemma 3.10 implies the result. In Constructive-Pair-Add, we are given an SM instance $\mathcal{I}$ and a budget ℓ and the question is whether we can add at most ℓ agents from a predefined set $U_{add} \cup W_{add}$ such that a designated pair $\{m^*, w^*\}$

is contained in a stable matching. It was shown to be NP-complete by Boehmer et al. [Boe+21c]. See Theorem 3.6 for the general idea of their construction. In the new instance $\mathcal{I}'$, we add $|U|$ many women $w'_1, ..., w'_{|U|}$ with arbitrary preferences. We modify the preferences as follows: All men except m^* rank w^* last and rank all w'_i just in front of w^*. The man m^* ranks all w'_i last. All other preferences are not changed. We set the designated agent to be w^*.
Let $d_1, ..., d_p \in U_{add}$ with $p \leq \ell$ be agents such that after adding them to the instance, $\{m^*, w^*\}$ is contained in a stable matching M. In $\mathcal{I}'$, when adding the same agents to the instance, matching M is still stable, since no woman w'_i can form a blocking pair (every man m is assigned since $|U| \leq |W|$ and m prefers $M(m)$ to w'_i). Obviously, w^* is assigned in M.
For the other direction, assume there are at most ℓ agents such that after adding them, w^* is assigned in a stable matching M'. Consider instance $\mathcal{I}$ after adding the same set of agents. Notice that w^* cannot be matched to any man $m \in U \setminus \{m^*\}$, as some w'_i would be unassigned and would form a blocking pair with m. Thus $\{m^*, w^*\} \in M'$. Matching $M := M' \setminus \{\{m, w'_i\} \mid i \in [|U|]\}$ is stable in $\mathcal{I}$: If M' contains any pair $\{m, w'_i\}$, then deleting it will not affect the stability in $\mathcal{I}$, because the unassigned m cannot form any blocking pair. Therefore, $\{m^*, w^*\} \in M$.
We can check whether an agent is assigned in a stable matching by computing an arbitrary stable matching, and thus membership in NP follows. □

3.3.4 Destructive-Agent-Swap

Finally, we consider the problem of deciding whether a stable agent can be made unassigned with a certain number of swaps. The problem is defined as follows:

Problem 10: DESTRUCTIVE-AGENT-SWAP
Input: An SM instance $\mathcal{I} = (U, W, \mathcal{P})$, an agent $a \in A$ and a budget $\ell \in \mathbb{N}$.
Question: Is it possible to perform at most ℓ swaps in $\mathcal{I}$ such that a is not assigned in any stable matching?

We will see that the hardness for this problem can be shown via the hardness of its constructive counterpart.

Theorem 3.12. DESTRUCTIVE-AGENT-SWAP *is NP-complete.*

Proof. We show NP-hardness by reducing from the constructive version CONSTRUCTIVE-AGENT-SWAP. For an instance $\mathcal{I}$ of the constructive version, we add a man m^* with preferences $m^* : w^* \succ ...$ who will be the designated agent and $d := |W| - |U|$ men $\tilde{m}_1, ..., \tilde{m}_d$ which will ensure that m^* cannot be matched to any woman who was unassigned in $\mathcal{I}$. We modify the preferences of any $w \in W \setminus \{w^*\}$ to $w : ... \succ \tilde{m}_1 \succ ... \succ \tilde{m}_d \succ m^*$ where the relation between two men $m, m' \in U$ is not changed. We modify the preferences of w^* to $w^* : ... \succ m^* \succ \tilde{m}_1 \succ \tilde{m}_d$ while again not changing the relation between any two men $m, m' \in U$. Finally, we introduce $(d+1)(\ell+1)$ dummy men and the same number of dummy women which ensure that we cannot change the preference relation of a pair involving a newly added man $m \in U' := \{m^*, \tilde{m}_1, ..., \tilde{m}_d\}$. To this end, between any two neighbouring agents m, m' with $m \in U' \setminus U$ in any preference list, we add $\ell + 1$ dummy men. All dummy women are matched last by all non-dummy men. Dummy

agents prefer each other as described in Theorem 3.6.
Assume that we can perform at most ℓ swaps in $\mathcal{I}$ such that w^* is assigned in a stable matching M. We call the resulting instance $\mathcal{I}_\sigma$. We claim that after performing the same swaps in $\mathcal{I}'$ (resulting in $\mathcal{I}'_\sigma$), m^* will not be assigned in any stable matching.
Assume for contradiction that there is a stable matching M' in $\mathcal{I}'_\sigma$ with $\{m^*, w\} \in M'$ for some $w \in W'$. Since our instance contains at least one more man than women, some man is unassigned. A dummy man is always matched with its partner and therefore, there is some $i \in \{1, ..., d\}$ such that $\tilde{m}_i$ is unassigned (if any other man was unassigned, he would form a blocking pair with the partner of $\tilde{m}_i$). If $w \neq w^*$, $\{\tilde{m}_i, w\}$ is blocking for M'. For the case $w = w^*$, in order for $\{M(w^*), w^*\}$ not to be blocking, he needs to be matched to some woman w he prefers to w^*. The same holds for any man from the original instance $\mathcal{I}$. But then, $M' \setminus \{\{m, w\} \mid m \notin U\}$ would be stable in $\mathcal{I}_\sigma$, which is a contradiction to the Rural Hospitals Theorem (since now, w^* is unassigned). It follows that M' is not stable, a contradiction.
For the other direction, assume that we can perform at most ℓ swaps such that m^* is not assigned in a stable matching M'. As argued in Theorem 3.6, any swap involving dummy agents will not change the preference order of non-dummy agents and can therefore be omitted. Moreover, w^* must be matched to a man she prefers to m^*, thus $\{m, w^*\} \in M'$ for some $m \in U$. We claim that after performing the same swaps in $\mathcal{I}$ as in $\mathcal{I}'$ (which is possible since we omitted all other swaps), $M := M' \setminus \{\{m, w\} \mid m \notin U\}$ is stable in $\mathcal{I}$. Suppose for contradiction that there is a blocking pair $\{m', w'\}$ for M. If w' is assigned in M, the same pair is also blocking M', since $M(m') = M'(m')$, $M(w') = M'(w')$ and the preference relations between $m', M(m'), w'$ and $M(w')$ are not altered. If w' is not assigned, then, in M', it has to be assigned to $\tilde{m}_i$ for some $i \in \{1, ..., d\}$. Again, $\{m', w'\}$ is blocking as w' prefers m' to $\tilde{m}_i$. Thus, we showed that M' is unstable, which is a contradiction.
For the same reason as in the proof for CONSTRUCTIVE-AGENT-SWAP, the problem is contained in NP. □

Chapter 4

Average-Case Robustness

In this chapter, we want to analyse the average-case robustness of matchings, pairs and agents, i.e. the probability that a matching, pair or agent is stable after performing exactly ℓ swaps or deletions. Computing this probability is equivalent to counting the number of solutions at swap/delete distance exactly ℓ, since we can divide this number by the total number of possible instances at swap/delete distance exactly ℓ to obtain the probability. Therefore, in this section, we will analyse the computational complexity of the counting variants of the problems discussed in the previous chapter. The only prerequisite is that we can compute the total number of possible instances in polynomial time. For the *Delete* setting, this is trivial, as there are exactly $\binom{\ell}{N}$ ways to delete ℓ agents from a set of N agents.

Observation 4.1. *Let A be a set of agents and $\ell \in \mathbb{N}$. We can compute the number $\delta_{\mathcal{I}}^{\ell}$ of possible deletions of ℓ agents in A in polynomial time.*

For the *Swap* setting, counting the number of preference profiles at a certain swap distance is not such a trivial task. However, we can show that it is polynomial-time solvable by modelling our SM instance as two elections. The corresponding problem for elections (i.e. computing the number $\sigma_{\mathcal{E}}^{\ell}$ of preference profiles over the same set of candidates that have swap distance exactly ℓ to some central election $\mathcal{E}$) was shown to be polynomial-time solvable via dynamic programming by Boehmer et al. [Boe+20].

Lemma 4.2. *Let $\mathcal{I} = (U, W, \mathcal{P})$ be an SM instance and $\ell \in \mathbb{N}$. We can compute the number $\sigma_{\mathcal{I}}^{\ell}$ of preference profiles at swap distance exactly ℓ in polynomial time.*

Proof. We create two elections $\mathcal{E}, \mathcal{F}$ where election $\mathcal{E}$ has voters U and candidates W and election $\mathcal{F}$ has voters W and candidates U. The preferences of the voters are equal to the preferences of the corresponding agents. We claim that we can compute $\sigma_{\mathcal{I}}^{\ell}$ as follows:

$$\sigma_{\mathcal{I}}^{\ell} = \sum_{i=0}^{\ell} \sigma_{\mathcal{E}}^{i} + \sigma_{\mathcal{F}}^{\ell-i}$$

Each agent $a \in A$ ranks exactly the same agents as its corresponding voter. We can subdivide each preference profile at swap distance ℓ from $\mathcal{I}$ into two preference profiles containing only the preferences of the agents of one gender each. These preference profiles are equivalent to elections $\mathcal{E}, \mathcal{F}$. Clearly, after adding the swap distances of the two preference profiles together, we must obtain ℓ. □

In the counting setting, the constructive and destructive versions of our problems become computationally equivalent (as the total number of solutions where a matching is stable equals the total number of possible solutions minus the total number of solutions where the matching is not stable). Furthermore, the counting version of a problem is always at least as hard as the constructive decision problem (if we know the exact number of solutions, we also know whether a correct solution exists) and at least as hard as the destructive decision problem (if we know the exact number of solutions, we also know whether there is a valid possible solution that is not correct). Therefore, we do not need to analyse the counting complexity of problems where the (constructive or destructive) decision variant is already hard, as we can only encounter hardness. The only counting problems that remain are #Matching-Swap, #Agent-Delete and (possibly) #Pair-Delete. For all three of these problems, we will show #P-hardness in this section. This will imply that we cannot efficiently compute the exact average-case robustness of an SM instance for any of our settings.

4.1 Robust Matchings

We first want to analyse the stable matching probability after performing a number of swaps. To make the problem clearer, we will not allow "redundant" swaps, i.e. we will only allow sequences of ℓ swaps such that the resulting preference profile is at swap distance ℓ from the original instance. For example, swapping a in front of a' in some preference list and then swapping a behind a' again is not allowed.

Problem 11: #Matching-Swap
Input: An SM instance $\mathcal{I} = (U, W, \mathcal{P})$, a matching M and a budget $\ell \in \mathbb{N}$.
Question: In how many preference profiles at swap distance exactly ℓ to $\mathcal{P}$ is M stable?

To show #P-hardness for #Matching-Swap, we will reduce from #Bipartite 2-SAT With No Negations, where we are given a set of variables $V = U \cup Z$ and a set of clauses $C \subseteq U \times Z$. The set of literals is $L := L_U \cup L_Z$ with $L_U := \{u, \bar{u} \mid u \in U\}$ and $L_Z := \{z, \bar{z} \mid z \in Z\}$. We call $A \subseteq L$ a *valid assignment* of our formula if $|\{v, \bar{v}\} \cap A| = 1$ for all $v \in V$. The task is to determine the number of valid assignments $A \subseteq L$ (including negations, even if none of them appear in a clause) such that each clause is fulfilled, i.e. $c \cap A \neq \emptyset$ for each $c \in C$. It was shown to be #P-complete by Provan and Ball [PB83]. An equivalent and more known formulation of this problem is #Bipartite Vertex Cover, which was also proven to be #P-hard by Provan and Ball [PB83]. However, we will choose #Bipartite 2-SAT With No Negations as the problem to reduce from, since we find the notion of variables and their negated form to be more intuitive in this case.
In our reduction, for each literal, we will create a man and a woman that are matched in the designated matching. However, they both prefer agents that correspond to the negated literal or to a literal with which they form a clause to each other. Therefore, in the initial instance, one literal agent forms a blocking pair with the agent corresponding

to the negated literal. To resolve the blocking pair, we must modify the preferences of one of the two literals, which corresponds to setting the variable to true or false. The budget does not allow any other swaps. Furthermore, two agents that correspond to literals that form a clause are blocking in the initial instance. By modifying an agent, this blocking pair is also resolved. Thus, by choosing one of the two literals in the clause, the corresponding blocking pair is resolved.
For the reduction, we need to pay attention to one technical detail: If a literal is not contained in any clause, there are two ways to modify the corresponding agent within the given budget: On the one hand, we can swap upwards the matched partner and on the other hand, we can swap down the blocking agent corresponding to the negated literal. This will produce two distinct preference profiles, which makes it harder to count the total number of preference profiles at the given swap distance. Therefore, we duplicate the complete instance (in the formal reduction, the copy of an agent a is a'). Now, each modification in the preferences of an agent must resolve at least two blocking pairs (the negated literal a and its copy a'). Swapping down both blocking agents is now too expensive and therefore, there is exactly one valid preference profile at the required swap distance for each agent modification.

Theorem 4.3. #MATCHING-SWAP *is #P-complete.*

Proof. For any literal p, we define the number of clauses where p is contained as $c(p) := |\{c \mid c \in C, p \in c\}|$. Let $N = \max_{p \in L} c(p)$.
Let $\mathcal{I} = (U, Z, C)$ be an instance of #BIPARTITE 2-SAT WITH NO NEGATIONS. The agents of our #MATCHING-SWAP instance $\mathcal{I}' = ((U', W', \mathcal{P}), M, \ell)$ are as follows:

$$U' = \{m_v, m'_v, m_{\bar{v}}, m'_{\bar{v}} \mid v \in V\} \cup \{m^d_i \mid i \in N\}$$
$$W' = \{w_v, w'_v, w_{\bar{v}}, w'_{\bar{v}} \mid v \in V\} \cup \{w^d_i \mid i \in N\}$$

For the preferences, consider a variable $u \in U$ and let $p_1, ..., p_{c(u)} \in Z$ be the literals such that there exists some clause $c = \{u, p_i\}$ for each $i \in [c(u)]$. The preferences for the corresponding agents are as follows:

$$m_u : w_{\bar{u}} \succ w'_{\bar{u}} \succ w_{p_1} \succ ... \succ w_{p_{c(u)}} \succ w^d_1 \succ ... \succ w^d_{N-c(u)} \succ w_u \succ ... \qquad w_u : m_u \succ ...$$
$$w_{\bar{u}} : m_u \succ m'_u \succ m^d_1 \succ ... \succ m^d_N \succ m_{\bar{u}} \succ ... \qquad m_{\bar{u}} : w_{\bar{u}} \succ ...$$
$$m'_u : w_{\bar{u}} \succ w'_{\bar{u}} \succ w'_{p_1} \succ ... \succ w'_{p_{c(u)}} \succ w^d_1 \succ ... \succ w^d_{N-c(u)} \succ w'_u \succ ... \qquad w'_u : m'_u \succ ...$$
$$w'_{\bar{u}} : m_u \succ m'_u \succ m^d_1 \succ ... \succ m^d_N \succ m'_{\bar{u}} \succ ... \qquad m'_{\bar{u}} : w'_{\bar{u}} \succ ...$$

Now, let $z \in Z$. The preferences are very similar to agents that correspond to a vertex from U, but the roles of men and women are interchanged. Again, let $p_1, ..., p_{c(z)} \in U$ be all literals such that there exists some clause $c = \{z, p_i\}$ for each $i \in [c(z)]$. The preferences for the corresponding agents are as follows:

$$w_z : m_{\bar{z}} \succ m'_{\bar{z}} \succ m_{p_1} \succ ... \succ m_{p_{c(z)}} \succ m^d_1 \succ ... \succ m^d_{N-c(z)} \succ m_z \succ ... \qquad m_z : w_z \succ ...$$
$$m_{\bar{z}} : w_z \succ w'_z \succ w^d_1 \succ ... \succ w^d_N \succ w_{\bar{z}} \succ ... \qquad w_{\bar{z}} : m_{\bar{z}} \succ ...$$
$$w'_z : m_{\bar{z}} \succ m'_{\bar{z}} \succ m'_{p_1} \succ ... \succ m'_{p_{c(z)}} \succ m^d_1 \succ ... \succ m^d_{N-c(z)} \succ m'_z \succ ... \qquad m'_z : w'_z \succ ...$$
$$m'_{\bar{z}} : w_z \succ w'_z \succ w^d_1 \succ ... \succ w^d_N \succ w'_{\bar{z}} \succ ... \qquad w'_{\bar{z}} : m'_{\bar{z}} \succ ...$$

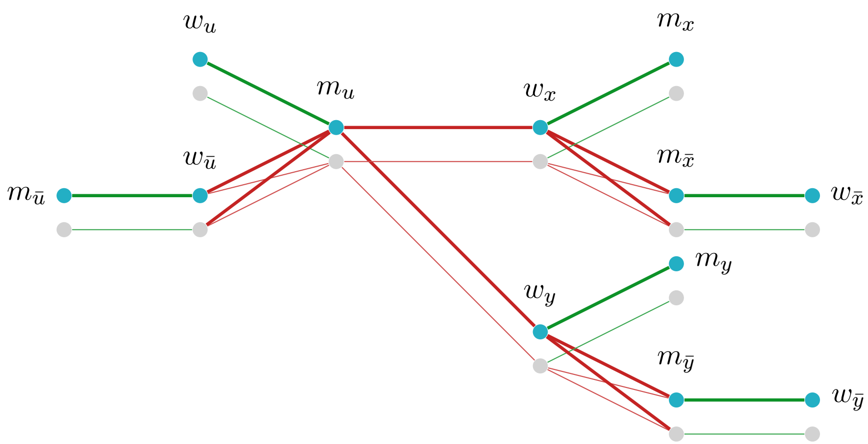


Figure 4.1: Exemplary SM instance constructed from from the formula $(u \lor x) \land (u \lor y)$. The copied part of the instance is coloured gray. Edges in the designated matching are coloured green, while edges that are initially blocking are coloured red. The dummy agents are omitted.

The dummy men have preferences $m_i^d : w_i^d \succ ...$ and the dummy women have preferences $w_i^d : m_i^d \succ ...$ for $i \in [N]$. Our designated matching is

$$M := \{\{m_v, w_v\}, \{m_{\bar{v}}, w_{\bar{v}}\}, \{m'_v, w'_v\}, \{m'_{\bar{v}}, w'_{\bar{v}}\} \mid v \in V\} \cup \{\{m_i^d, w_i^d\} \mid i \in [N]\}$$

Notice that all agents $m_u, w_{\bar{u}}, m'_u, w'_{\bar{u}}$ for $u \in U$ and all agents $w_z, m_{\bar{z}}, w'_z, m'_{\bar{z}}$ for $z \in Z$ rank their matched partner on place $(N+3)$. Moreover, M is not stable in the constructed instance: The set of blocking pairs is

$$\begin{aligned} B :=& \{\{m_p, w_{\bar{p}}\}, \{m_p, w'_{\bar{p}}\}, \{m'_p, w'_{\bar{p}}\}, \{m'_p, w_{\bar{p}}\} \mid p \in L\} \\ &\cup \{\{m_p, w_q\}, \{m'_p, w'_q\} \mid \{p, q\} \in C\} \end{aligned}$$

Note that p can both be a positive and negative literal. One can easily verify that these pairs are indeed blocking. We now argue why there are no other blocking pairs in $\mathcal{I}$: For a blocking pair $\{m, w\}$, it must hold that $w \succ_m M(m)$. Let $m \in U'$. For all $w \in W' \setminus \{w_i^d \mid i \in [N]\}$ that fulfil this condition, we already have that $\{m, w\} \in B$. But since w_i^d is matched to her top choice for all $i \in [N]$, $\{m, w_i^d\}$ is not blocking.
We set our swap budget to $\ell := 2|V|(N + 2)$ and claim that the number of satisfying assignments for the given 2-SAT formula is equal to the number of preference profiles at swap distance exactly ℓ from $\mathcal{I}'$ such that M is stable.
To this end, we define a function f which maps valid assignments of the 2-SAT formula to valid preference profiles in $\mathcal{I}'$. Let $\mathcal{P}$ be the preference profile in $\mathcal{I}'$. For an assignment A, we define $f(A) := \mathcal{P}'$ as follows: Starting from $\mathcal{P}$, for each $p \in L_U$, we swap w_p to the front of the preferences of m_p and we swap w'_p to the front of the preferences of m'_p. For each $p \in L_Z$, we swap m_p to the front of the preferences of w_p and we swap m'_p to the front of the preferences of w'_p.
We now proceed to show that f describes a one-to-one correspondence between satisfying assignments and preference profiles with the desired properties, which implies that their number is equal.

Lemma 4.4. *f is injective.*

Proof. Consider two different valid assignments $A \neq B$. There is at least one literal l with $l \in A$ and $l \notin B$. Without loss of generality, let $l \in L_U$. Now, consider the two constructed instances $f(A)$ and $f(B)$. In $f(A)$, the preferences of m_l are altered, while in $f(B)$ they are not. It follows that $f(A) \neq f(B)$ and thus f is injective. □

Lemma 4.5. *Let A be a satisfying assignment of the 2-SAT formula $\mathcal{I}$. Then, $f(A)$ is a preference profile at swap distance exactly ℓ from $\mathcal{I}'$ where the designated matching M of $\mathcal{I}'$ is stable.*

Proof. First notice that we only improve $M(a)$ in the preferences of any agent a and thus no new blocking pairs can arise. As A is a valid assignment, and thus contains one literal for each variable, for any blocking pair $\{m_p, w_{\bar{p}}\}$ with $p \in L$, we swap $M(m_p)$ to the front of m_p's preferences or $M(w_{\bar{p}})$ to the front of $w_{\bar{p}}$'s preferences, resolving the blocking pair. The same holds for the blocking pairs of the duplicated part. As A is a satisfying assignment, and thus contains one literal for each clause, for any blocking pair $\{m_p, w_q\}$ with $\{p, q\} \in C$, we swap $M(m_p)$ to the front of m_p's preferences or $M(w_q)$ to the front of w_q's preferences, resolving the blocking pair. Again, the same holds for the blocking pairs of the duplicated part. We argued above that no other pairs are blocking in the original instance and thus it follows that M is stable. Notice that to swap $M(a)$ and $M(a')$ to the first place in the preferences of a and a', we need $2(N+2)$ swaps and since any satisfying assignment contains exactly $|V|$ literals, the overall budget is exactly matched. □

Lemma 4.6. *Let $\succ := a : b_1 \succ b_2 \succ ... \succ b_i \succ b^* \succ b_{i+1} \succ ... \succ b_n$ be a preference list. $\succ' := a : b^* \succ b_1 \succ b_2 \succ ... \succ b_n$ is the only preference list at swap distance i to $\succ$ where a prefers b^* to b_1 and to b_2.*

Proof. Since there are $i-1$ agents between b_1 and b^*, one needs at least i swaps to reach a preference list where b^* is preferred to b_1. Clearly, $\succ'$ has swap distance i to $\succ$, as we only improve b^* by i positions. Now consider a preference list $\succ''$ at swap distance i to L where b^* is preferred to b_1 and b_2. With each swap, we have to improve b^* or swap down b_1 or b_2 (otherwise b^* cannot be preferred to b_1). Assume that we swap down b_1 or b_2 in at least one step. If we swap down b_2, the distance between b_1 and b^* is still i, but we only have $i-1$ swaps left, a contradiction. If we swap down b_1, then b_2 is improved with the same swap and the distance between b_2 and b^* is i. Again, since there are only $i-1$ swaps left, this is not possible. It follows that we do not swap down b_1 or b_2 in $\succ''$, and thus $\succ'' = \succ'$. □

Lemma 4.7. *Let $\mathcal{P}'$ be a preference profile at swap distance at most ℓ from $\mathcal{I}'$ such that M is stable. There exists a satisfying assignment A with $f(A) = \mathcal{P}'$.*

Proof. To resolve the blocking pairs $\{m_p, w_{\bar{p}}\}, \{m_p, w'_{\bar{p}}\}, \{m'_p, w'_{\bar{p}}\}$ and $\{m'_p, w_{\bar{p}}\}$ for any $p \in L$, we need to modify m_p and its duplicated version m'_p or we need to modify $w_{\bar{p}}$ and its duplicated version $w'_{\bar{p}}$, that is, swap $M(m_p)$ and $M(m'_p)$ in front of $w_{\bar{p}}$ and $w'_{\bar{p}}$ in the preferences of m_p and m'_p, respectively, or swap $M(w_{\bar{p}})$ and $M(w'_{\bar{p}})$ in front of m_p and m'_p in the preferences of $w_{\bar{p}}$ and $w'_{\bar{p}}$, respectively. Clearly, we need at least $N+2$ swaps

to perform such a modification in one preference list. Since we need to modify $2|V|$ preference lists and our total budget is $2|V|(N+2)$, each modification of a preference list can only take $N+2$ swaps. By Lemma 4.6, the only way to achieve this is to swap the matched agent to the first place. Since we exhausted the complete swap budget, no additional swaps are possible.
As we must modify m_u or $m_{\bar{u}}$ and w_v or $w_{\bar{v}}$ but can only modify at most $2|V|$ agents, the assignment $A := \{p \in L \mid m_p \text{ or } w_p \text{ was modified}\}$ is valid. Clearly, $f(A) = P'$. Since M is stable in P', no pair corresponding to a clause in $\mathcal{I}$ can be blocking. It follows that for each clause $c = \{u, v\} \in C$, either m_u or w_v must have been modified, as otherwise $\{m_u, w_v\}$ is blocking. Therefore, for each clause $c \in C$, there is a literal $p \in A \cap c$ and by definition, A is a satisfying assignment. □

Lemma 4.4, Lemma 4.5 and Lemma 4.7 together show that f describes a one-to-one correspondence between satisfying assignments in $\mathcal{I}$ and preference profiles with swap distance ℓ to $\mathcal{I}'$ where M is stable.
The problem is contained in #P, since we can construct a NTM that nondeterministically chooses a preference profile with swap distance exactly ℓ and checks whether M is stable. Then, the number of accepting branches equals the number of solutions to the problem. □

Together with Lemma 4.2, it follows that computing the probability that a matching is stable at a random preference profile at a certain swap distance is hard.

4.2 Robust Agents

We now turn to the problem of computing the probability that some agent a is stable after a certain number of agents were deleted. We do not consider the analogous problem for pairs in a separate section, since the results for agents implies the result for pairs, as we will see later.

Problem 12: #AGENT-DELETE
Input: An SM instance $\mathcal{I} = (U, W, \mathcal{P})$, an agent $a \in U \cup W$, and a budget $\ell \in \mathbb{N}$.
Question: How many sets of agents $A' \subseteq (U \cup W) \setminus \{a\}$ with $|A'| = \ell$ are there such that after deleting all agents in A' from $\mathcal{I}$, a is assigned in a stable matching?

For an instance $\mathcal{I}$, we will denote by $\#\mathrm{AD}(\mathcal{I})$ the number of solutions for this instance. To show #P-hardness for #AGENT-DELETE, we will give a Turing reduction from #EDGE COVER, where we are given a graph $G = (V, E)$ and an integer $k \in \mathbb{N}$ and want to determine the number of subsets of edges $E' \subseteq E$ with $|E'| = k$ and $\bigcup_{e \in E'} e = V$. The decision version of this problem is polynomial-time solvable (Norman and Rabin [NR59]), but the counting version is #P-hard (Bubley and Dyer [BD97]). We use the notation $n := |V|$ and $m := |E|$.
Let us introduce some notation to better describe the #AGENT-DELETE instances that we will give as input to the oracle. To this end, let $\mathcal{I}^G_{i,\ell}$ be a #AGENT-DELETE instance. We will construct the instance from the given graph G. The agents consist of *edge men* m_e, *vertex men* m^q_v, *extra men* m^*_p and *vertex women* w_v. More formally, the sets of

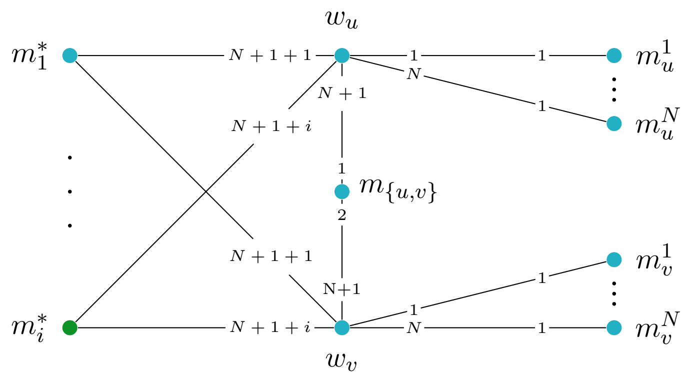


Figure 4.2: One edge and two vertices from an SM instance $\mathcal{I}_{i,\ell}^{G}$. The designated agent m_i^* is coloured green. The edge labels denote the rank of the distant agent of the edge in the preferences of the close agent of the edge.

men and women are:

$$\begin{aligned} U &= \{m_e \mid e \in E\} \cup \{m_v^q \mid v \in V, q \in [N]\} \cup \{m_p^* \mid p \in [i]\} \\ W &= \{w_v \mid v \in V\} \end{aligned}$$

where $N := m + 1$. Let $e = \{u, v\} \in E$. The preferences of the edge men are the following:

$$m_e : w_u \succ w_v \succ ...$$

Let $v \in V$ and $e_1, ..., e_{deg(v)}$ be the edges incident to v. Let $q \in [N]$ and $p \in [i]$. The preferences for the remaining agents are:

$$\begin{aligned} &m_v^q : w_v \succ ... \\ &w_v : m_v^1 \succ ... \succ m_v^N \succ m_{e_1} \succ ... \succ m_{e_{deg(v)}} \succ m_1^* \succ ... \succ m_i^* \succ ... \\ &m_p^* : ... \end{aligned}$$

Our deletion budget is ℓ and will in the most cases be equal to $\ell(i, j) := j + i \cdot N$, where j is the number of edge men that we want to delete. We set the designated agent to be m_i^*.

To give an intuition why the solutions of this instance are connected to edge covers of G, consider the case $i = 1$. Our designated agent is m_1^*. The idea is that we will delete all edge men that do not correspond to an edge of our edge cover. Now, for each vertex woman w_v, there is one edge man that was not deleted and which w_v prefers to m_1^*. Our remaining budget suffices to "select" one vertex $v \in V$, that is, delete the corresponding N vertex men. Intuitively,by selecting a vertex, we check whether our edge cover covers the vertex. In the original instance, every vertex woman w_v is matched to a corresponding vertex man m_v^q for some $q \in [N]$ in any stable matching. However, if all corresponding vertex men were deleted, w_v will be matched with an edge man that corresponds to an edge that is incident to v, or to an extra man, if no such edge exists. Since the remaining edge men correspond to an edge cover, such an edge man always exists, independently of which vertex we chose. Consequently, m_1^* cannot be assigned. On the other hand, if the not-deleted edge men do not form an edge cover, there is a vertex v such that all edge men corresponding to incident edges are deleted. After deleting all vertex men m_v^q for $q \in [N]$, w_v will be matched to our designated agent m_1^*, since all vertex men and edge men that are preferred by w_v were deleted. This means that a yes-instance for Agent-Delete corresponds to a no-instance for Edge Cover. However, “selecting” vertices is not necessarily unique: There could be an edge set E' such that more than one vertex is not covered by E'. In this case, we would count the same non-edge cover multiple times. Therefore, for each non-edge cover E', we need to know how many vertices are not covered. We do this by setting i to the number of vertices that we can select.

Theorem 4.8. #Agent-Delete *is #P-complete.*

Proof. We first define two helpful notions:

Definition 4.9. Consider the #Agent-Delete instance $\mathcal{I}^G_{i,\ell(i,j)}$. We define $D_{i,j}$ as the number of solutions to $\mathcal{I}^G_{i,\ell(i,j)}$ that contain at least one extra man m_p^* for some $p \in [i]$.

Definition 4.10. For a graph $G = (V, E)$, an *i-vertex-isolating set* is an edge set $E' \subseteq E$ such that after deleting all edges in E', there are exactly i vertices that have no incident edges. We denote by $\mathcal{E}_i^j$ the set of all i-vertex-isolating sets of size j.

Notice that for any graph $G = (V, E)$, $C \subseteq E$ is an edge cover if and only if there is no $i \in \{1, ..., n\}$ such that $E \setminus C$ is an i-vertex-isolating set. We will now introduce the most important lemma of the proof, which will establish a connection between vertex-isolating sets and Agent-Delete solutions.

Lemma 4.11. *Let $G = (V, E)$ be a graph and $i \in [n], j \in [m]$. The following equation holds:*

$$\#\mathrm{AD}(\mathcal{I}^G_{i,\ell(i,j)}) = \sum_{j'=0}^{j} \sum_{i'=i}^{n} (\binom{(N+1)(n-i)}{j-j'} \cdot \binom{i'}{i} |\mathcal{E}_{i'}^{j'}|) + D_{i,j}$$

Proof. We first only consider solutions not containing any extra men. Clearly, if the number of such solutions is $\sum_{j'=0}^{j} \sum_{i'=i}^{n} \binom{(N+1)(n-i)}{j-j'} \cdot \binom{i'}{i} |\mathcal{E}_{i'}^{j'}|$, then adding $D_{i,j}$ will give the correct total number of solutions.
As an additional restriction, we first only count solutions that satisfy the following constraints:

1. the edges corresponding to the deleted edge men form a i'-vertex isolating set of size j', where $i' \geq i$ and $j' \leq j$.
2. There are exactly i vertices v such that m_v^q is deleted for all $q \in [N]$. We call such a vertex *selected*. Notice that our budget $\ell = j + i \cdot N$ does not suffice to select more than i vertices, since $j < N$.
3. For each selected $v \in V$, woman w_v is not deleted.

We first show that the number of solutions that fulfil these constraints is exactly the total number of solutions as stated by the theorem minus $D_{i,j}$. Let E' be an i'-vertex-isolating set of size j' in G and $i \leq i' \leq n$, $j' \leq j$. After deleting all edge men m_e with $e \in E'$, the remaining deletion budget is $j - j' + i \cdot N$. Let $V' \subseteq V \setminus \bigcup_{e \in E \setminus E'} e$ with $|V'| = i$ be a subset of i vertices that are isolated in $(V, E \setminus E')$. Clearly, there are $\binom{i'}{i}$ such subsets. After deleting m_v^q for all $v \in V'$ and $q \in [N]$, there are i women who prefer m_p^* for any $p \in [i]$ to all other men, because we deleted all men that corresponded to any $v \in V'$ and to any edge incident to any $v \in V'$. It follows that m_i^* must be assigned. Finally, we can delete $j - j'$ arbitrary vertex agents (except any agent w_v with $v \in V'$) to reach the required deletion budget. There are $N \cdot (|V| - i)$ not yet deleted vertex men and $|V| - i$ vertex women that we can delete. Notice that we cannot delete more edge men since we count all solutions corresponding to a certain vertex-isolating set separately. Thus, for each i'-vertex-isolating set of size j', there exist $\binom{(N+1)(n-i)}{j-j'} \cdot \binom{i'}{i}$ different sets of agents such that after their deletion the designated agent m_i^* is assigned. It follows that there are at least $\sum_{j'=0}^{j} \sum_{i'=i}^{n} \binom{(N+1)(n-i)}{j-j'} \cdot \binom{i'}{i} |\mathcal{E}_{i'}^{j'}|$ many different sets that do not contain extra men and whose deletion ensure that m_i^* is assigned. Clearly, all solutions that we counted until now are different, since we delete different edge men, vertex men or vertex women.

Note that all solutions that we counted until now consisted of j' edge men with $j' \leq j$, whose corresponding edges form a i'-vertex isolating set of size j' with $i' \geq i$, $i \cdot N$ vertex men m_v^q that all belong to the same vertices and $j - j'$ arbitrary vertex agents except w_v for a selected vertex v.

We now argue that there are no solutions that violate the above constraints. To this end, for each violated constraint, we will show that at most $i - 1$ vertex women prefer extra man to any not-deleted vertex man. From this it follows that one of the i extra men is not matched to a vertex woman, and since all vertex women prefer m_j^* to m_i^* for any $j < i$, m_i^* is not assigned.

1. Assume that the deleted edge men do not correspond to an i'-vertex isolating set of size j' in G for some $i' \geq i$, $j' \leq j$. If $j' > j$, the remaining budget does not suffice to delete $i \cdot N$ vertex men and we are in case 2. Thus, we can assume $i' < i$, i.e. less than i vertices are isolated. This means that for each set of i vertex women, at least one of them will prefer an edge man to each extra man. Assume that some extra man m^* is matched to this vertex woman w_v in a stable matching and let e be an edge such that $v \in e$ and m_e is not deleted. In order for $\{m_e, w_v\}$ not to be blocking, m_e must be matched to another vertex woman w_u. It follows that m_u^q was deleted for all $q \in [N]$ (else, w_u would be matched to some m_u^q) and that w_u is not matched to any extra man. Because of our limited budget, we cannot delete all vertex men for more than i vertices. It follows that there are only $i - 1$ vertex women that can be matched to extra men (since u is one of the selected vertices but w_u is not matched to any extra man). We can conclude that m_i^* is not assigned. We can repeat the same argument for the case where w_v is selected but not matched to any extra man.

2. Assume we deleted less than $i \cdot N$ vertex men. For each set of i vertex women, at least one of them will prefer a vertex man to each extra man. Thus, m_i^* cannot be

assigned.

3. Assume that we deleted some w_v such that v was selected, i.e. all m_v^q for $q \in [N]$ were deleted. Because of our limited budget, we cannot select more than i vertices. Since w_v is deleted, there are only $i-1$ not deleted women that were selected and therefore do not prefer a vertex man to any extra man. Again, it follows that m_i^* cannot be assigned.

Thus, there are no other solutions and adding all solutions involving the deletion of extra men, we obtain $\#\mathrm{AD}(\mathcal{I}^G_{i,\ell(i,j)}) = \sum_{j'=0}^{j} \sum_{i'=i}^{n} \binom{(N+1)(n-i)}{j-j'} \cdot \binom{i'}{i} |\mathcal{E}_{i'}^{j'}| + D_{i',j'}$.

□

With the help of Lemma 4.11, we now want to develop a polynomial-time algorithm that, given an oracle for #AGENT-DELETE, computes #EDGE COVER. To this end, we first show that we can compute $D_{i,j}$ in polynomial time via dynamic programming.

Lemma 4.12. *Given an oracle that solves #AGENT-DELETE, we can compute $D_{i,j}$ in polynomial time for any $i \in [n], j \in [m]$.*

Proof. We will compute $D_{i,j}$ via a dynamic programming table $R_{i,j}[k]$, which will denote the number of solutions to #AGENT-DELETE for $\mathcal{I}^G_{i,\ell(i,j)}$ where the first k extra men (and no other extra men) are deleted. We claim that for each $k \in [i]$ we can compute the entries as follows:

$$R_{i,j}[i-1] = \#\mathrm{AD}(\mathcal{I}^G_{1,j,j+iN-(i-1)})$$

$$R_{i,j}[k] = \#\mathrm{AD}(\mathcal{I}^G_{i-k,j,j+iN-k}) - \sum_{r=1}^{i-1-k} \binom{i-1-k}{r} R[k+r] \text{ for all } k < i-1$$

We show by induction over the number of not deleted extra men $i-k$ that $R_{i,j}[k]$ is correctly computed. For $i-k=1$ (and thus $k=i-1$), all extra men except for m_i^* must be deleted (note that by definition, m_i^* cannot be deleted). Therefore, we can modify our instance by deleting all extra men except for one and subtracting $i-1$ from the budget. The resulting instance is exactly $\mathcal{I}^G_{1,j,j+iN-(i-1)}$.
Now, for any k, let $R_{i,j}[k']$ be correctly computed for all $k' \in \{k+1, ..., i-1\}$. From this we can conclude that $R_{i,j}[k]$ is correctly computed as follows: Since we must not count the solutions where more extra men are deleted (as we only are interested in the solutions with exactly k deleted extra men), we subtract these solutions from the result of our oracle call. Consider a solution for $\mathcal{I}^G_{i-k,j,j+iN-k}$ that deletes r extra men. By induction hypothesis, the total number of solutions where the first r extra men are deleted is $R_{i,j}[k+r]$. If a solution with r deleted extra men exists, then exchanging these r extra men with arbitrary different r extra men is still a solution. It follows that for each solution where the first r extra men were deleted, we have $\binom{i-1-k}{r}$ ways to choose the extra men. Consequently, the total number of solutions where r extra men were deleted are $\binom{i-1-k}{r} R_{i,j}[k+r]$. We must subtract this number for each number r of extra men in $\mathcal{I}^G_{i,j,j+iN-k}$ and thus we get $R[k] = \#\mathrm{AD}(\mathcal{I}^G_{i-k,j,j+iN-k}) - \sum_{r=1}^{i-1-k} \binom{i-1-k}{r} R[k+r]$.

Now, we can compute the total number of solutions that contain extra men, which is

$$D_{i,j} = \sum_{k=1}^{i-1} \binom{i-1}{k} R[k]$$

because there are $\binom{i-1}{k}$ ways to choose k out of $i-1$ extra men to delete. □

We now move on to computing the number of i-vertex-isolating sets of size j for any $i \in [n]$, $j \in [m]$. For this, we will fill a dynamic programming table $T[i, j]$ which will store exactly that number. We will fill the table column by column, from $j = 1$ to $j = m$, and for each column we start with $i = n$ and end with $i = 1$. We will calculate $T[i, j]$ for any $i \in [n], j \in [m]$ as follows:

$$T[i,j] = \#\mathrm{AD}(\mathcal{I}^G_{i,\ell(i,j)}) - D_{i,j} - \sum_{j'=0}^{j-1} \sum_{i'=i}^{n} \binom{(N+1)(n-i')}{j-j'} \cdot \binom{i'}{i} \cdot T[i',j'] - \sum_{i'=i+1}^{n} \binom{i'}{i} \cdot T[i',j]$$

The following lemma states that $T[i, j]$ is equal to the number of i-vertex-isolating sets of size j.

Lemma 4.13. *$T[i, j] = |\mathcal{E}_i^j|$ for all $i \in [n], j \in [m]$.*

Proof. We will show the claim by induction over $n(j + 1) - i$. For $n(j + 1) - i = 0$, we have $i = n$ and $j = 0$, and thus, using Lemma 4.11, we obtain

$$\begin{aligned} T[n,0] &= \#\mathrm{AD}(\mathcal{I}^G_{n,\ell(i,0)}) - D_{n,0} \\ &\quad - \sum_{j'=0}^{-1} \sum_{i'=n}^{n} \binom{(N+1)(n-i')}{j-j'} \cdot \binom{i'}{i} \cdot T[i',j'] \\ &\quad - \sum_{i'=n+1}^{n} \binom{i'}{i} \cdot T[i',j] \qquad (4.1) \\ &= \binom{0}{0}\binom{n}{n} |\mathcal{E}_n^0| + D_{n,0} - D_{n,0} = |\mathcal{E}_n^0| \qquad (4.2) \end{aligned}$$

where in (4.1) we use the definition of $T[i, j]$ and in (4.2) we apply Lemma 4.11. Now, assume that the equation is true for all $i \in [n], j \in [m]$ with $n(j + 1) - i < r$. We show that the equation is true for $i \in \{1, ..., n\}, j \in [m]$ such that $n(j + 1) - i = r$. Using Lemma 4.11 and our induction hypothesis that $T[i', j'] = \mathcal{E}_{i'}^{j'}$ for $j' < j$ and $T[i', j] = \mathcal{E}_{i'}^{j}$

for $i' > i$, we obtain

$$\begin{aligned}
T[i,j] =& \#\mathrm{AD}(\mathcal{I}^{G}_{i,\ell(i,j)}) - D_{i,j} \\
&- \sum_{j'=0}^{j-1} \sum_{i'=i}^{n} \binom{(N+1)(n-i')}{j-j'} \cdot \binom{i'}{i} \cdot T[i',j'] \\
&- \sum_{i'=i+1}^{n} \binom{i'}{i} \cdot T[i',j] \qquad (4.3)\\
=& \sum_{j'=0}^{j} \sum_{i'=i}^{n} \binom{(N+1)(n-i')}{j-j'} \cdot \binom{i'}{i} \cdot |\mathcal{E}^{j'}_{i'}| + D_{i,j} - D_{i,j} \\
&- \sum_{j'=0}^{j-1} \sum_{i'=i}^{n} \binom{(N+1)(n-i')}{j-j'} \cdot \binom{i'}{i} \cdot |\mathcal{E}^{j'}_{i'}| - \sum_{i'=i+1}^{n} \binom{i'}{i} \cdot |\mathcal{E}^{j'}_{i'}| \qquad (4.4)\\
=& \binom{(N+1)(n-i)}{j-j} \binom{i}{i} |\mathcal{E}^{j}_{i}| = |\mathcal{E}^{j}_{i}| \qquad (4.5)
\end{aligned}$$

where (4.3) uses the definition of $T[i,j]$ and in (4.4), we apply Lemma 4.11 and also replace $T[i',j']$ by $|\mathcal{E}^{j}_{i}|$ by our induction hypothesis. □

Lemma 4.14. *Given an oracle that solves #*Agent-Delete*, we can compute $T[i,j]$ in polynomial time.*

Proof. To compute the table entry $T[i,j]$, we need to compute at most $i \cdot j$ table entries before. Each table entry can be computed in polynomial time, since, by Lemma 4.12, $D_{i,j}$ can be computed in polynomial time, an oracle query needs constant time and we sum over polynomially many table entries. Therefore, in total, the running time is polynomial. □

Lemma 4.15. *Let $G = (V,E)$ be a graph and let C_k be the number of edge covers of size k in G. Then, it holds that $C_k = \binom{m}{k} - \sum_{i=1}^{n} T[i, m-k]$.*

Proof. By Lemma 4.13, $\sum_{i=1}^{n} T[i, m-k]$ gives the number of vertex-isolating sets of size $m-k$. If $E' \subseteq E$ with $|E'| = m-k$ is not a i'-vertex-isolating set for some $i' \in [n]$, then clearly $E \setminus E'$ is an edge cover of size k. If $E' \subseteq E$ with $|E'| = m-k$ is a i'-vertex-isolating set for some $i' \in [n]$, then clearly $E \setminus E'$ is not an edge cover. It follows that either E' is an edge cover or $E \setminus E'$ is a vertex-isolating set and the result follows. □

Corollary 4.16. *Given an oracle that solves #*Agent-Delete*, for a given graph $G = (V,E)$ and an integer $j \in [m]$, we can compute the number of edge covers of size j in G in polynomial time.*

In the end, we show membership in #P. Consider a NTM that nondeterministically chooses ℓ agents (that are not a) and checks whether after their deletion, a is assigned in a stable matching. The number of accepting branches of this NTM is equal to the number of solutions to our problem. □

Robust Pairs. We can use the hardness of #Agent-Delete to show hardness for the analogous counting problem for stable pairs, i.e. #Pair-Delete. We use the one-to-one correspondence between unstable agents and stable pairs as shown in Theorem 3.8. Notice that the set of possible deletions is the same for both instances ($\mathcal{I}$ and $f(\mathcal{I}, a)$), since we are not allowed to delete a in $\mathcal{I}$ and we are not allowed to delete a or a' in $f(\mathcal{I}, a)$. The one-to-one-correspondence between these two instances directly implies #Agent-Delete($\mathcal{I}$) = $N-$#Pair-Delete, where N is the total number of possible deletions of ℓ agents. Thus, we can clearly compute the number of stable agent solutions of an instance in polynomial time when we are given an oracle for #Pair-Delete. This implies #P-hardness for #Pair-Delete, and since we can nondeterministically choose ℓ agents and check whether a pair is stable after their deletion in polynomial time, we obtain the following corollary:

Corollary 4.17. #Pair-Delete *is #P-complete.*

Together with Observation 4.1, it follows that computing the probability that an agent or pair is stable after deleting a certain number of agents at random is hard.

Chapter 5

Experiments

After having analysed the computational complexity of determining the robustness of a matching, pair or agent, we now want to make statements about the robustness itself. We will focus on swaps as change operation and will not consider *Delete* operations, because we find the swap setting to be more interesting and intuitive. The main questions that we are interested in are the following:

- How many random swaps are required on average to make a typical matching/pair unstable?
- Which cultures to sample SM instances produce more robust stable matchings/pairs?
- What are easy measures to estimate the average-case robustness?
- Are different stable matchings/pairs of the same instance differently robust? What is a method to produce a robust stable matching?

We will present the results of several experiments we conducted about the robustness of stable marriage instances. Since we showed in the last chapters that computing the probability of a matching or of a pair to be stable in case we apply changes uniformly at random is hard, we use a different approach to estimate the average-case robustness of an SM instance: The normalised Mallows model, which is a noise model that is different from the swap model but has favorable sampling properties. We will give the details to it in the next section.

5.1 Setup

Normalised Mallows Model. The Mallows model, given a central preference list $\succ_a$ of agents A with $|A| = n$ and a dispersion parameter ϕ, assigns to each strict and complete order (or preference list) $\succ'_a$ over A a probability $\mathcal{D}^{\succ_a,\phi}_{\text{Mallows}}(\succ'_a) := \frac{1}{Z}\phi^{\kappa(\succ_a,\succ'_a)}$, where $Z = 1 \cdot (1+\phi) \cdot (1+\phi+\phi^2) + ... + (1+\phi+...+\phi^{n-1})$. We can sample a preference list with these probabilities in polynomial time. Note that if $\phi = 0$, $\succ_a$ is assigned probability 1 and all other preference lists are assigned probability 0, while for $\phi = 1$, all preference lists are assigned the same probability.
We use a normalised variant of the dispersion parameter introduced by Boehmer et al.

[Boe+21b]. They use a normalised dispersion parameter norm-ϕ, which is internally converted to ϕ. The advantage of this parameter is that it directly corresponds to the expected number of swaps that is performed in one central preference list to obtained the sampled one. More precisely, the expected swap distance between the central and the sampled preference list is norm-$\phi \cdot \frac{n \cdot (n-1)}{4}$. This allows us to interpret the experimental results using the total number of (expected) swaps, as in the complexity setting.
To measure the robustness of an SM instance $\mathcal{I} = (U, W, \mathcal{P})$ for our experiments, for each $a \in A$, we draw a preference list $\succ'_a$ from the Mallows model with a given norm-ϕ value and the central preference list $\succ_a$. We then check whether a certain matching (or pair) is stable in $\mathcal{I}' := (U, W, \mathcal{P}' := \{\succ'_a \mid a \in A\})$. To estimate the probability that the matching or pair is stable, we repeat this procedure 1000 times in a Monte-Carlo-like fashion. [1]

Complexity of computing stability probabilities in the Mallows model. In the previous chapter, we showed that computing the exact stable matching probability for a specific number of swaps is hard. In fact, the hardness result for #Matching-Swap carries over to the Mallows model. We show this in a very similar fashion to the analogous proof for the election setting by Baumeister and Hogrebe [BH23]. Formally, given a stable marriage instance $\mathcal{I}$, a matching M and a rational number $\phi \in [0, 1]$, the Mallows Stable Matching Probability problem asks for computing the probability that M is stable in an instance generated by the Mallows model with dispersion parameter ϕ from $\mathcal{I}$.

Lemma 5.1. *One can Turing-reduce* #Matching-Swap *to* Mallows Stable Matching Probability *in polynomial time.*

Proof. Assume we are given an SM instance $\mathcal{I} = (U, W, \mathcal{P})$, a matching M and a swap budget r. Let $n = m$, and thus $2n$ be the number of agents in our instance. Our new Mallows Stable Matching Probability instance consists of the exact same SM instance $\mathcal{I}$ and matching M. We set the dispersion parameter $\phi := \frac{1}{(n!)^{2n}}$. Consider an arbitrary preference profile $\mathcal{P}'$ with swap distance j to $\mathcal{P}$. We have

$$\mathcal{D}^{\mathcal{P},\phi}_{\text{Mallows}}(\mathcal{P}') = \frac{1}{Z^{2n}} \frac{1}{(n!)^{2n \cdot j}} = (n!)^{2n} \cdot \frac{1}{Z^{2n}} \frac{1}{(n!)^{2n \cdot (j+1)}} > \sum_{\mathcal{P}'' : \kappa(\mathcal{P}, \mathcal{P}'') > j} \mathcal{D}^{\mathcal{P},\phi}_{\text{Mallows}}(\mathcal{P}'')$$

i.e. the probability assigned to $\mathcal{P}'$ is larger than the summed probabilities assigned to all preference profiles with a larger swap distance. This is because there can be at most $n!$ different preference lists per agent and thus the total number of preference profiles is bounded by $(n!)^{2n}$. Therefore, we can decompose the Mallows Stable Matching Probability answer Φ as follows. Clearly, the probability that M is stable is $N_j \cdot (\frac{1}{Z^{2n}} \phi^j) + y$, where $y < (\frac{1}{z} \phi^j)$ and N_j is the number of preference profiles at swap distance j where M is stable. y resembles the probability that M is stable in some preference profile with higher swap distance. We set $\Phi_0 = \Phi$ and compute for

[1] We also use the normalised Mallows model to generate some of our synthetic instances (see the Norm-Mallows, Mallows-Euclidean, Mallows-Asymmetric and Mallows-Robust cultures in Section 5.1).

$0 \leq j \leq \frac{2n^2 \cdot (n-1)}{2} - 1$:

$$N_j = \lfloor \Phi_j / ((\frac{1}{n!^{2n}})^j / Z^{2n}) \rfloor \text{ and } \Phi_{j+1} = \Phi_j - N_j \cdot ((\frac{1}{n!^{2n}})^j / Z^{2n})$$

Then, N_j gives the number of preference profiles at swap distance j from $\mathcal{I}$ where M is stable. □

Data set. We will now present the data set used for the experiment. It consists of 544 stable marriage instances with $|U| = |W| = 50$. For the rest of the section we assume $n := |U| = |W|$. The data set contains four "special" instances and 540 instances drawn from various statistical cultures, where for each culture, we sample 20 instances. We first present the four special instances:

- **Mutual Agreement (MA)**: In this symmetric instance, two agents rank each other on the same place, i.e. the preferences are built such that $rk_m(w) = rk_w(m)$ for all $m \in U$ and $w \in W$.
- **Mutual Disagreement (MD)**: Contrary to the MA instance, in this instance, if an agent a ranks another agent b very high, b ranks a correspondingly low. Formally, we have that $rk_m(w) = n + 1 - rk_w(m)$ for all $m \in U$ and $w \in W$.
- **Identity (ID)**: In this instance, each man has the same preferences over all women and each woman has the same preferences over all men, i.e. $P_m = P_{m'}$ for all $m, m' \in U$ and $P_w = P_{w'}$ for all $w, w' \in W$.
- **Robust Instance (ROB)**: This instance is Example 3 for $n = 50$.

We now present the statistical cultures. We use the cultures that Boehmer, Heeger, and Szufa [BHS22] already used in the Stable Roommates setting and slightly adapt them for our purposes.

- **Impartial Culture (IC)**: Each preference list $\succ_a$ is sampled uniformly at random from the set of all agents of opposite gender.
- **2 Group-Impartial Culture (2-IC)**: We randomly partition U and W into $U_1 \uplus U_2 = U$ and $W_1 \uplus W_2 = W$ with $\frac{U_1}{U} = \frac{W_1}{W} = p$ for some parameter $p \in [0, 0.5]$. Each man $m \in M_i$ prefers all women $w \in W_i$ over all women of the other group (and vice versa). The preferences within the groups are drawn uniformly at random from the set of all agents of opposite gender for each agent. Our data set contains 20 instances for each $p \in \{0.25, 0.5\}$.
- **1-Dimensional Euclidean (1D)**: For each agent $a \in A$, we sample a point $p_a \in [0, 1]$ uniformly at random. Agent a prefers an agent a' to another agent a'' if $|p_a - p_{a'}| < |p_a - p_{a''}|$.
- **2-Dimensional Euclidean (2D)**: For each agent $a \in A$, we sample a point $p_a \in [0, 1]^2$ uniformly at random. Agent a prefers an agent a' to another agent a'' if the Euclidean distance between a and a' is smaller than the Euclidean distance between a and a''.

- **2-Dimensional Reverse-Euclidean (Rev-Euc)**: In addition to the setting in 2D, we partition U and W into two sets each (U_1, U_2 and W_1, W_2). An agent $a \in U_1 \cup W_1$ ranks the other agents as described in 2D. An agent $b \in U_2 \cup W_2$ ranks the other agents in opposite order, i.e. they prefer agents with a greater Euclidean distance. We are given a parameter $p \in [0, 1]$ that describes the percentage of agents in U_2 and W_2, respectively. Our data set contains 20 instances for each $p \in \{0.05, 0.15, 0.25\}$.

- **Fame-Euclidean (Fame-Euc)**: Given a parameter $f \in [0, 1]$, for each agent, we sample random points $p_a \in [0, 1]^2$ (as for the 2D culture) and $f_a \in [0, f]$. An agent a evaluates another agent of opposite gender b analogously to the 2d culture, but subtracts f_a from this evaluation. The points f_a correspond to the fame of an agent, which influences their rank in the preferences of the other agents. Our data set contains 20 instances for each $f \in \{0.2, 0.4\}$.

- **Expectations-Euclidean (Ex-Euc)**: We first sample a point p_a for each agent $a \in A$ as for the 2D culture. We then generate a second point q_a for each agent using a 2-dimensional Gaussian distribution with mean p_a and standard deviation σ. Each agent a ranks the agents of opposite gender b increasingly by the distance between p_a and q_b. The points p_a correspond to where the agents would like their partner to be located, while q_a is their real location. Our data set contains 20 instances for each $\sigma \in \{0.2, 0.4\}$.

- **Attributes (Attr)** Given a dimension $d \in \mathbb{N}$, we sample two vectors for each agent $p^a, w^a \in [0, 1]^d$ uniformly at random. Agent a ranks other agents b decreasingly by $\sum_{i \in [d]} w_i^a \cdot p_i^b$. Our data set contains 20 instances for each $d \in \{2, 5\}$.

- **Norm-Mallows (N-Mal)** : Starting from a random ordering of men and women, the preference list of each agent is an ordering drawn from the Mallows model with dispersion parameter norm-ϕ from the initial ordering of the agents of opposite gender. Our data set contains 20 instances for each norm-ϕ such that norm-$\phi \in \{0.2, 0.4, 0.6, 0.8\}$.

- **Mallows-Euclidean (Mal-Euc)**: We create the preferences for all agents as described for the 2D-culture. From this instance, we draw a new instance according to the Mallows model with dispersion parameter norm-ϕ. Our data set contains 20 instances for each norm-$\phi \in \{0.2, 0.4\}$.

- **Mallows-Asymmetric (Mal-MD)**: The procedure is analogous to the Mallows-Euclidean culture. Instead of an instance from the 2D culture, we start with the MD instance. Our data set contains 20 instances for each norm-$\phi \in \{0.2, 0.4, 0.6\}$.

- **Mallows-Robust (Mal-ROB)**: The procedure is analogous to the Mallows-Euclidean culture. Instead of an instance from the 2D culture, we start with the Robust instance. We added this culture to our data set in the expectation that its instances will be among the most robust. Our data set contains 20 instances for each norm-$\phi \in \{0.2, 0.4, 0.6, 0.8\}$.

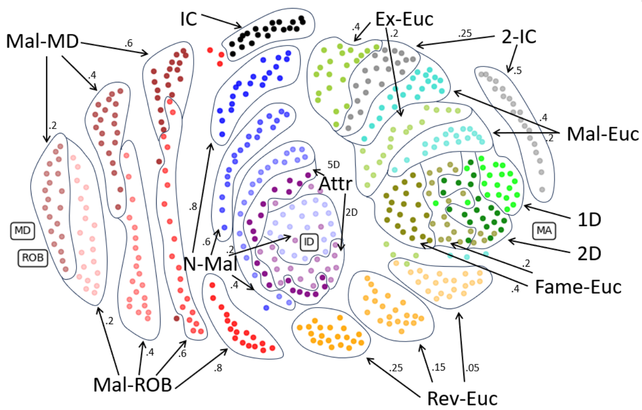


Figure 5.1: The cultures of the data set. Instances of the same culture that have a higher parameter (e.g. norm-ϕ) have the same colour, but are more transparent.

Analyzing the worst-case robustness of the instances from the data set, we observe that all instances except for the MA, ROB and some Mal-ROB 0.2 instances have a worst-case robustness of 1. We will further discuss this in Section 5.2.2. Therefore one can say that it is sensible to analyse the average-case robustness, as the worst-case robustness does not very meaningful.

5.2 Robust Matchings

We will first focus on the robustness of stable matchings. For unstable matchings, the notion of robustness makes little sense. To better quantify the robustness of a stable matching, we will use the notion of the *50%-stability threshold* (or *50%-threshold*), which we define as the smallest norm-ϕ value for which the matching is stable with a probability of less than fifty percent. This is a definition analogous to the *50%-winner threshold* in the voting setting, used by Boehmer et al. [Boe+20]. We will use this value as an experimental measure of the average-case robustness.

5.2.1 Robustness of man-optimal stable matchings

We start with presenting several findings on the average-case robustness of man-optimal (or woman-optimal) stable matchings computed by the Gale-Shapley algorithm. We will consider other matchings in further subsections, but we first focus on those matchings, as it is the oldest and most famous way to compute a stable matching.

Finding 1. For most instances, the robustness of a man-optimal stable matching is low: Performing on average two random swaps in each preference list often makes the matching unstable.

Figure 5.2 depicts the distribution of the 50%-thresholds of all instances. We can see that most instances have a 50%-threshold between 0.002 and 0.003. This corresponds to performing 1.225 to 1.8375 swaps in each preference list (since $0.002 \cdot \frac{50 \cdot 49}{4} = 1.225$,

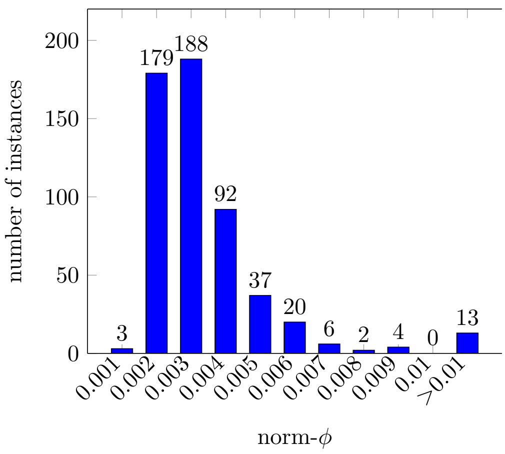


Figure 5.2: Distribution of the 50%-threshold. All values are rounded up to a multiple of 0.001.

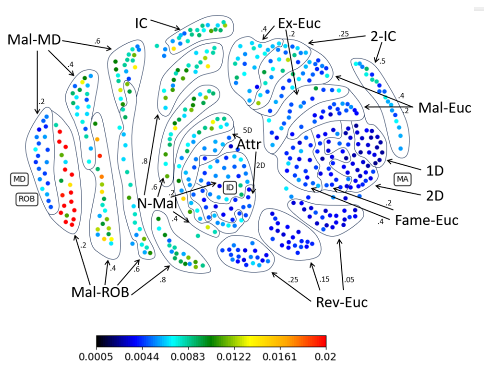
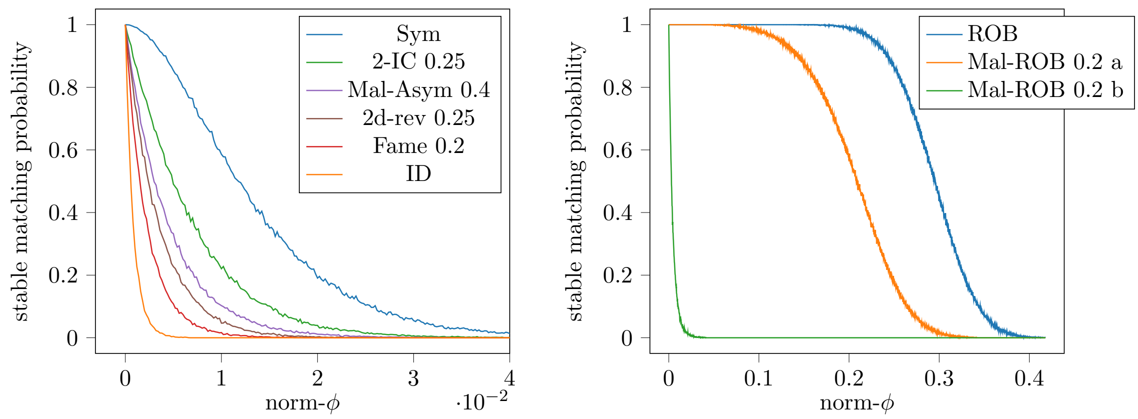


Figure 5.3: 50%-threshold for the man-optimal stable matchings of all instances of the data set. The scale is capped at norm-$\phi = 0.02$.

and analogously for 0.003). Considering that most swaps in an agent's preference list do not involve the matched partner of the agent and thus do not influence the stability of the matching, this number is remarkably low. The intuition is that there are many pairs that only need one swap to become blocking and therefore it is enough if the performed swaps involve the matched agents in only a few preference lists.

Map of stable marriage instances. To better understand which cultures generate robust stable matchings and which instances do not, we examine the map of stable marriage instances introduced by Boehmer, Heeger, and Szufa [BHS22]. Each point corresponds to one instance in our data set. The distance between two points models how different the two instances are according to the so-called *mutual attraction distance*, introduced by Boehmer, Heeger, and Szufa [BHS22]. Each agent a has a vector where the i-th entry is $rk_{p_a(i)}(a)$. We now match agents from the two instances we want to compare such that the ℓ_1 distance between the vectors is minimised. The summed ℓ_1 distances between all matched agents is the mutual-attraction-distance of the two instances. In the map, the labels of the different cultures are connected to the corresponding instances. In Figure 5.1, we see the map for our data set where each culture has a different colour. Using the map, we can visualise different measures for each instance and identify differences between the cultures. For instance, the map that shows the 50%-thresholds of the man-optimal stable matchings of all instances is depicted in Figure 5.3. Examining this map, we can observe that most Euclidean instances have a very low average-case robustness. Instances from the IC culture or close to IC are quite robust. The ROB instance and the instances from the Mal-ROB culture have the highest average-case robustness. The ROB instance has a 50%-threshold close to 0.3, which implies that one needs almost 200 random swaps in each preference list to make the matching unstable. This is 100 times more swaps than for an instance with average robustness, as discussed in Finding 1.



(a) Six instances with low robustness

(b) Three instances with high robustness

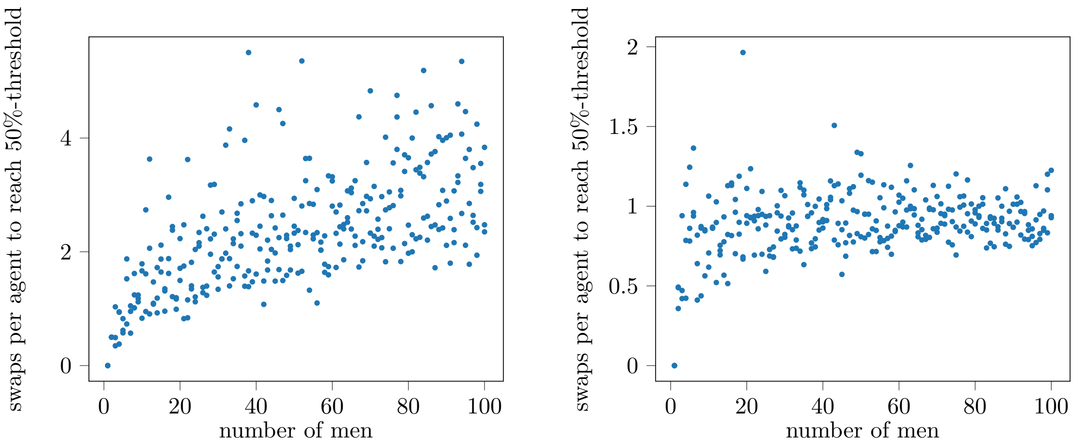
Figure 5.4: Average-case robustness for nine exemplary instances.

One can observe that more structured cultures tend to be less robust than cultures that involve more randomness. For example, most Euclidean instances have a very low 50%-threshold while the uniformly at random generated IC instances are surprisingly robust. Intuitively, in a structured instance, there are natural candidates for blocking pairs that can very easily become blocking because of their mutual liking. For example, consider a 2D instance where the points of one man m and two women w, w' were sampled very close to each other. Suppose that m and w are mutual top-choices and thus $\{m, w\}$ is contained in the initial stable matching. It could be that m ranks w' second and w' ranks m first. In that case, the pair $\{m, w'\}$ has a high probability of becoming blocking (see also Section 5.2.2).

To obtain more information than only the 50%-threshold, we examine how the stable matching probability changes when we increase the norm-ϕ value. To this end, we have chosen nine exemplary instances presented in Figure 5.4. They are chosen to represent instances with average robustness as well as instances with extreme robustness values. Since the range of three of the instances greatly differs from the other six instances, we depict these three more robust instances in Figure 5.4b.

First, we observe that the stable matching probability decreases monotonically with increasing norm-ϕ. For five of the six instances in Figure 5.4a, the stable matching probability decreases very quickly. However, in the MA instance, the stable matching probability does not decrease quickly for norm-ϕ values smaller than 0.005. This can be explained by the higher worst-case robustness of the MA instance: There does not exist one swap that makes the matching unstable and thus the stable matching probability is high when only few swaps are allowed (i.e. for small norm-ϕ). Due to the symmetry of the instance, for a pair to become blocking, we must modify the preferences of both agents of the pair (and thus at least two swaps are needed in any case).

Figure 5.4b shows the stable matching probability for two Mal-ROB instances with norm-ϕ = 0.2 and the ROB instance. The high worst-case robustness of the latter, which we discussed in Example 3, is confirmed by this plot: We can see that for norm-ϕ values smaller 0.2, the stable matching probability is close to 1.



(a) IC culture (b) 2D culture

Figure 5.5: Correlation of instance size and average-case robustness for 300 instances (three instances of size $n = m = i$ for each $i \in [100]$) of two different cultures

For the Mal-ROB instances with norm-$\phi = 0.2$, we observed two different kinds of instances: Very robust instances (marked red in the map) and not very robust instances. In Figure 5.4b, we can see the development of the stable matching probability for the ROB instance and for two Mal-ROB instances with norm-$\phi = 0.2$. Clearly, while the robustness of the orange instance is close to the ROB instance, the green instance is far less robust, despite having the same distance from the ROB instance as the orange instance. The intuitive explanation for this large difference is that, in the orange instance, the man-optimal stable matching is still the same as for the blue ROB instance, while for the green instance, the man-optimal stable matching is different and thus has a far worse worst-case robustness - and thereby also a far worse average-case robustness.

Instance size and Average-Case Robustness. Now, we want to examine the impact of the instance size on the average-case robustness. Recall that in this chapter we assume $n = m$, that is, the number of men and women is equal. Therefore, the instance size only depends on the number of men n. Intuitively, there are two opposing factors that influence how and whether the robustness of matchings changes when increasing the number of agents: On the one hand, when increasing the instance size, we can perform more “useless” swaps, that is, swaps in the preference list $\succ_a$ that do not involve $M(a)$. This would make the instance more robust. On the other hand, by adding more agents, we obtain more potential blocking pairs, making the instance less robust.
We observed that these two effects almost cancle out: Figure 5.5 shows the number of swaps needed per agent to reach the 50%-threshold (i.e. norm-$\phi \cdot \frac{n \cdot (n-1)}{4}$, where norm-$\phi$ is the 50%-threshold) for 300 IC instances and 300 2D instances of different size. The 50%-threshold itself rapidly decreases with increasing instance size, since the total number of possible swaps grows quadratically. In Figure 5.5a, one can observe that larger instances lead to a higher robustness, but the robustness grows very slowly. In Figure 5.5b, far less swaps are required (which also matches the results of Figure 5.3) and

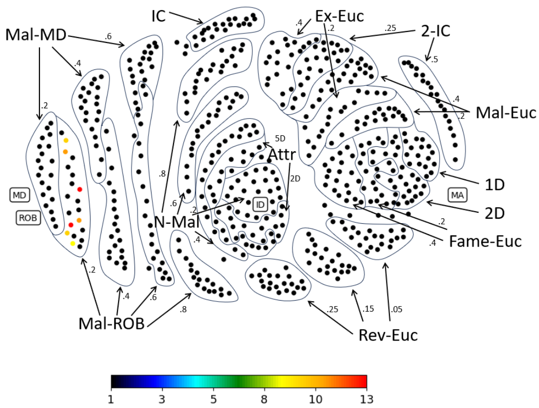


Figure 5.6: Worst-Case Robustness of the instances of the data set.

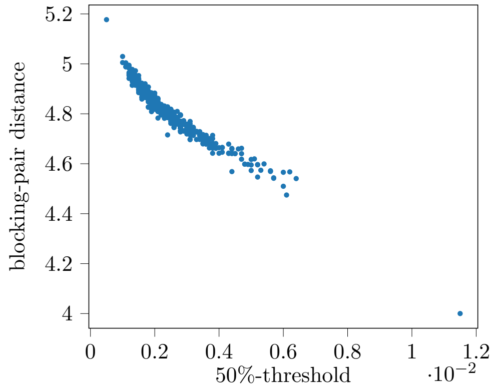


Figure 5.7: Correlation between blocking pair proximity and the 50%-threshold for all instances from the data set except for the instances from the robust cultures (whose values are too extreme).

the robustness does not change significantly when adding more agents.

5.2.2 Easy measures to estimate the robustness

Due to the hardness results in the previous chapter, we use sampling in order to obtain results on the average-case robustness of matchings. However, even the sampling method requires much time if one wants to have reasonably accurate results. Therefore, we want to discuss some simple measures of the average-case robustness and examine their correlation to the 50%-threshold.

Worst-Case Robustness.

Finding 2. The worst-case robustness is a very bad measure for the average-case robustness, as most instances have a worst-case robustness of 1.

The first measure for the average-case robustness that we will investigate is the worst-case robustness as defined in Chapter 2. As can be seen in Figure 5.6, the worst-case robustness is a very bad measure for the average-case robustness. The main reason for this is that in the most cases, the worst-case robustness is 1 (i.e. there is some possible swap that can make the matching unstable if it is performed). This is not surprising considering the fact that in order to have a worst-case robustness greater than 1, for each agent a we would need to have $a \succ_{M(a)} p_a(rk_a(M(a)+1))$, i.e. $M(a)$ must prefer a to the agent following $M(a)$ in the preferences of a for all $a \in A$. The only instances with a worst-case robustness greater than 1 are the MD and ROB instances as well as some instances from the Mal-ROB culture (notice that this culture was designed to have a high worst-case robustness!).

Blocking Pair Proximity. As a second measure, we want to build upon the minimum swap distance for each man-woman pair to being blocking. For an SM instance $\mathcal{I}$, a matching M and a man-woman pair $\{m, w\}$, the *blocking distance* of $\{m, w\}$ to M in $\mathcal{I}$ is

$$\beta_{\mathcal{I}}^{M}(m, w) = \max(0, rk_m(w) - rk_m(M(m))) + \max(0, rk_w(m) - rk_w(M(w)))$$

Clearly, a man-woman pair that has a blocking distance of 1 has a much greater effect on the average-case robustness of this instance than a pair that needs more than one swap to become blocking. In particular, since in each preference list, we can perform around n different swaps, one can argue that the probability that one swap is performed in a preference list is around $\frac{1}{n}$, while performing two specific swaps has a probability of around $\frac{1}{n^2}$. From this intuition, we derive the following robustness measure:

Definition 5.2. Let $\mathcal{I} = (U, W, \mathcal{P})$ be an SM instance, M be a stable matching in $\mathcal{I}$ and $d = n$. Let $\#\beta_{\mathcal{I}}(M, k)$ be the number of man-woman pairs (m, w) with blocking distance k, i.e. $\beta(m, w)_{\mathcal{I}}^{M} = k$. We define the *blocking pair proximity* of M as

$$\pi_{\mathcal{I}}(M) := \log_n \left(\sum_{k=1}^{d} n^{d-k} \cdot \#\beta_{\mathcal{I}}(M, k)\right)$$

where a small blocking pair proximity corresponds to a robust matching. In practice, to avoid very large numbers, we set $d = 5$ and argue that man-woman pairs that have a distance larger 5 to being blocking are "overshadowed" by pairs with smaller blocking distance and thus are practically irrelevant for the average-case robustness. [2] Also notice that two very unrobust matchings could still significantly differ in their blocking pair proximity, if the logarithm was omitted. However, their 50%-threshold is quite similar, since the stable matching probability decreases very rapidly. Thus, we add the logarithm to counteract this effect.
To measure the correlation between this measure and the 50%-threshold, we use the *Pearson Correlation Coefficient*, where 1 means that the two quantities have a perfect positive linear correlation, -1 means that the two quantities have a perfect negative linear correlation and 0 means that the two quantities have no correlation. Using this coefficient, we observe the following:

Finding 3. The blocking pair proximity of a matching is a good indicator of its average-case robustness. The Pearson Correlation Coefficient between the two is -0.965.

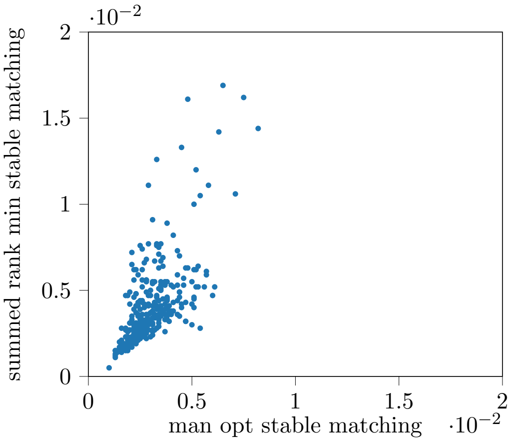
As can be seen in Section 5.2.1, this measure has a very high correlation with the average-case robustness. With a Pearson Correlation Coefficient of -0.965, one can conclude that the blocking pair proximity and the 50%-threshold have a very strong negative correlation [SBS18]. Consequently, in practice, if one wants to measure the average-case robustness of a matching, one could use the blocking pair proximity instead of the more resource-intensive sampling method and still obtain a reasonably good solution.

5.2.3 Robustness of different stable matchings

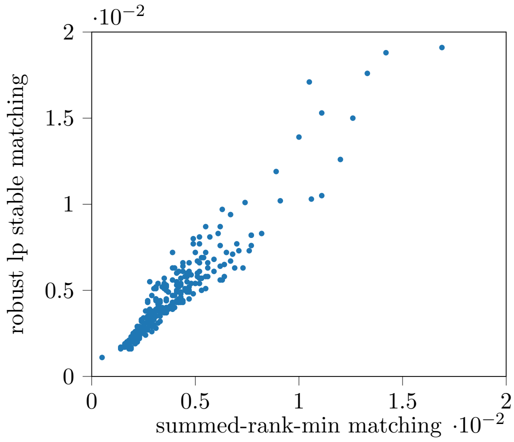
We will consider two alternative methods to find a stable matching. The first is called the *summed-rank minimizing stable matching* (or *summed-rank-min matching*, for short) and

[2] Obviously, for artificial instances with a very high worst-case robustness (e.g. the ROB instance), this simplification cannot be made.

Figure 5.8: Correlation between (a) man-optimal stable matching and summed-rank-minimal stable matching robustness and (b)summed-rank-minimal stable matching and robust stable matching robustness for all instances from the data set (except for the robust and some mallows-robust instances with norm-$\phi = 0.2$)



(a) man-opt and summed-rank-min stable matchings



(b) summed-rank-min and robust stable matchings

was first introduced by Knuth [Knu76]. It minimises the average rank of the matched partner in the preference list of an agent. Thus, one can argue that it is the stable matching that maximises the social welfare. Intuitively, we expect a stable matching where the average rank of the matched partner is small to be more robust, as the average number of swaps required to swap an agent in front of the matched partner is greater. Vate [Vat89] showed that it can be solved in polynomial time via linear programming.

Robust Stable Matching. As a second method, we introduce the *robust stable matching*, which, over all stable matchings of an instance, minimises the expression defined in Definition 5.2. In the following, we give a formulation of computing this matching as an Integer Linear Program (ILP). Using the Gurobi Optimizer, the ILP can be solved reasonably fast for the size of our instances, despite having an exponential worst-case running time. We will have that $U = W = [n]$ and $x[i, j]$ is set to 1 if man i is matched to woman j and to 0 else. Variable $y[i, j]$ is the blocking distance of pair $\{i, j\}$. It is set to some large value (in our case $2n$) if i and j are matched. Finally, $z[i, j, \ell]$ is set to 1 if the blocking distance of the pair $\{i, j\}$ is exactly ℓ. Variable z is needed to count the number of pairs at a certain blocking distance.

$$\sum_{i\in[n]} x[i,j] \leq 1 \qquad \forall j \in [n] \tag{5.1}$$

$$\sum_{j\in[n]} x[i,j] \leq 1 \qquad \forall i \in [n] \tag{5.2}$$

$$\sum_{i,j\in[n]:rk_{i^*}(j)\leq rk_{i^*}(j^*)\vee rk_{j^*}(i)\leq rk_{j^*}(i^*)} x[i,j] \geq 1 \qquad \forall i^*, j^* \in [n] \tag{5.3}$$

$$\begin{aligned}&\sum_{k=1}^{rk_i(j)-1} k\cdot x[i,p_i(rk_i(j)-k)]\\ &+\sum_{k=1}^{rk_j(i)-1} k\cdot x[p_j(rk_j(i)-k)] + 2n\cdot x[i,j] = y[i,j]\end{aligned} \qquad \forall i,j \in [n] \tag{5.4}$$

$$k+1-y[i,j]-k\cdot\sum_{\ell=1}^{k-1} z[i,j,\ell] \leq z[i,j,k] \qquad \forall i,j\in[n], k\in[d] \tag{5.5}$$

$$x[i,j]\in\{0,1\} \qquad \forall i,j\in[n] \tag{5.6}$$

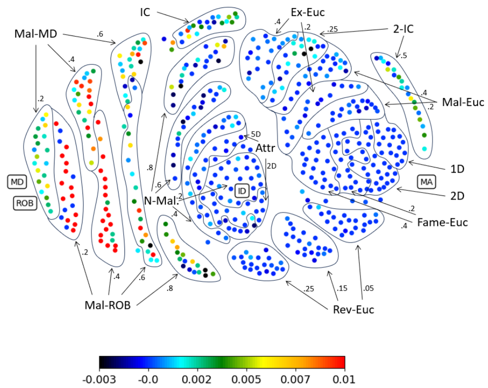
$$\min \sum_{k=1}^{d} n^{d-k}\cdot\sum_{i,j\in[n]} z[i,j,k] \tag{5.7}$$

Conditions (5.1) and (5.2) guarantee that the result is a matching. Inequality (5.3) guarantees that the matching is stable. Inequality (5.4) sets $y[i,j]$ to the distance of the pair $\{i,j\}$ to being blocking. (5.5) ensures that $z[i,j,\ell]$ is set to 1 if the distance of the pair $\{i,j\}$ from being blocking is exactly ℓ. $z[i,j,k]$ will always be assigned the smallest possible value because of the objective function (5.7). We subtract the given sum in (5.5) to ensure that if $z[i,j,k]$ is set to 1, $z[i,j,k+1]$ does not have to be set to 1. Finally, in (5.6), we minimise the blocking pair proximity of the matching. Obviously, we can omit the logarithm and still the same matching minimises the expression, since the logarithm is monotonic. We now compare the robustness of the three matchings. Notice that for some instances, two or all three of the matchings coincide.

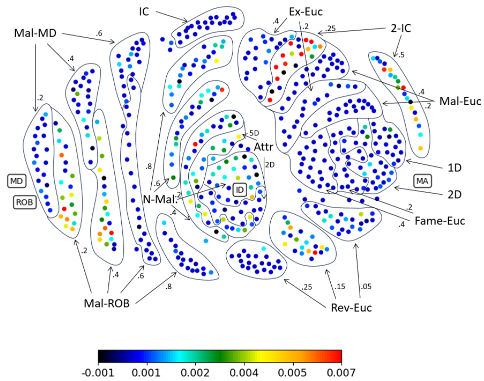
Finding 4. For more than 20% of all instances, the summed-rank minimizing stable matching is significantly more robust than the man-optimal matching. For almost all other instances, the two matchings have a similar robustness. The robust stable matching is the most robust and the summed-rank-min is the second most robust for almost all instances.

In Figure 5.8a, we can observe that for the most instances, the summed-rank minimal stable matching is more robust than the man-optimal stable matching. For 19 instances, the 50%-threshold is even more than three times greater than in the man-optimal matching. The difference between the two matchings is particularly large for the Mal-ROB, Mal-MD, IC and 2-IC instances, as can be seen in Figure 5.9a. As we will see later, these are exactly the cultures that produce instances with many stable pairs (see Figure 5.15). This suggests that in the above cultures, the summed-rank min matching

Figure 5.9: Difference between the 50%-threshold values of (a) the man-optimal and summed-rank-minimal stable matching and (b) the summed-rank-minimal stable matching and the robust stable matching for all instances of the data set.

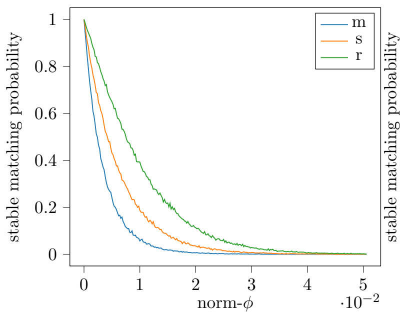


(a) Difference between man-opt and summed-rank min stable matchings. All values ≥ 0.01 are depicted with the same colour.

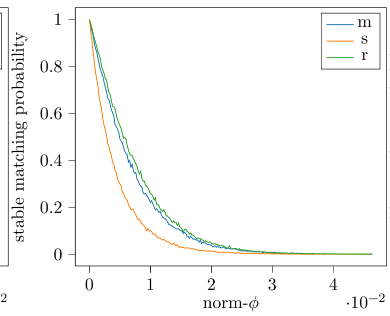


(b) Difference between summed-rank-min and robust stable matchings.

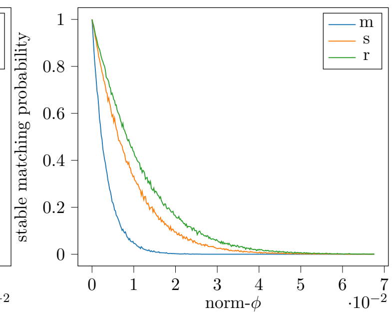
Figure 5.10: Average-case robustness for the man-optimal (m), summed-rank minimizing (s) and robust (r) stable matchings in six exemplary instances.



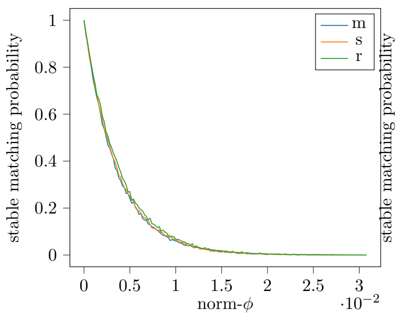
(a) IC



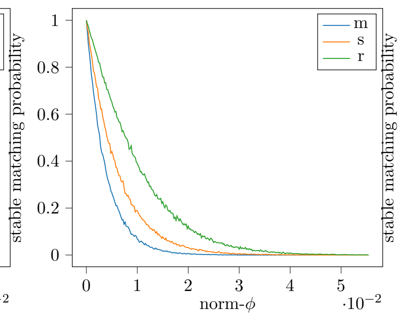
(b) 2g-IC 0.25



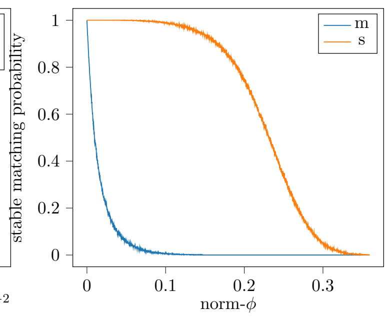
(c) Mal-Asym 0.4 a



(d) Expectation 0.2



(e) Mal-Asym 0.4 b



(f) Mal-ROB 0.2

has the chance to differ more significantly from the man-optimal stable matching and thus also have a significantly higher robustness. In total, we observed that there are 121 instances where the 50%-threshold of the summed-rank-min matching is at least 0.001 greater than the man-optimal stable matching. For 222 instances, the two matchings are completely identical. There are only very few (7) instances where the man-optimal matching performs better than the summed-rank minimal matching. An example can be seen in Figure 5.10b. On the other hand, the summed-rank-min matching can also be extremely more robust, as seen in Figure 5.10f. In this Mal-ROB instance, the summed-rank-min matching corresponds to the stable matching of the ROB instance, while the man-optimal matching differs.
We now turn to the robust stable matching, which, as we will see, never performs worse than the man-optimal matching and also outperforms the summed-rank minimal matching in many instances. In Figure 5.8b, we see the correlation between the 50%-thresholds of the summed-rank min and robust stable matchings. While one can observe that the robust stable matching is always at least as robust as the summed-rank-min matching, the difference between the two thresholds is not as large as for the man-optimal and summed-rank min stable matching, i.e. there is no instance where the robust matching is by far more robust than the summed-rank-min matching. Nevertheless, Figures 5.10a and 5.10e show that the robust stable matching can still be clearly more robust than the other matchings. As seen in Figure 5.9, the difference between summed-rank-min and robust stable matchings is particularly large for instances from the Mal-ROB, 2-IC, Rev-Euc, N-Mal and Attr cultures. We observed that there are 77 instances where the 50%-threshold of the robust stable matching is at least 0.001 greater than the 50%-threshold of the summed-rank-min stable matching (so one could argue that for around 14% of the instances, the robust stable matching is significantly more robust than the summed-rank-min matching), while for 284 instances, the two matchings are exactly identical. For all remaining instances, the two matchings have a very similar robustness.
As mentioned above, for some instances, two or all three matchings coincide. The instances we chose to present in Figure 5.10 have distinct man-optimal, summed-rank-min and robust matchings (except for Figure 5.10f, where the summed-rank-min and robust matching coincide) but we aim to represent all instances regarding their robustness (that is, for each instance in the data set, there is some exemplary instance that has a similar robustness). For some, mostly Euclidean, instances, the matchings are the same or they differ but their robustness is very similar. An example is shown in Figure 5.10d. But, generally, for instances where the robustness differs, the robust and summed-rank-min matchings are mostly closer to each other than to the man-optimal matching (for example in Figure 5.10c).

5.2.4 Unstable matchings

We want to shortly examine the stable matching probability of matchings that are initially not stable. As there is a very large number of unstable matchings, we restrict ourselves to one specific (possibly unstable) matching, which is defined analogously to the summed-rank min stable matching, but the stability criterion is omitted, i.e. we consider the matching maximizing social welfare, regardless of whether it is stable or not. More formally, we define this matching as follows:

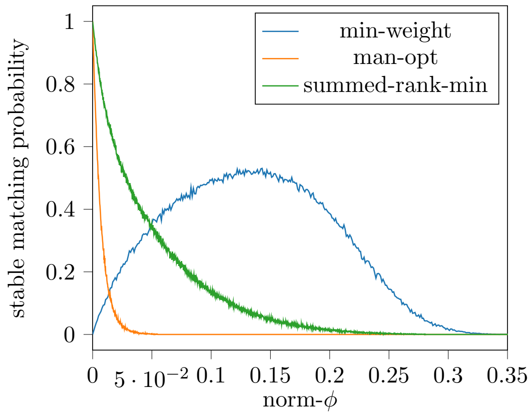


Figure 5.11: Average-Case Robustness for the man-optimal and summed-rank minimal stable matchings as well as for the min-weight matching for Mal-ROB 0.25.

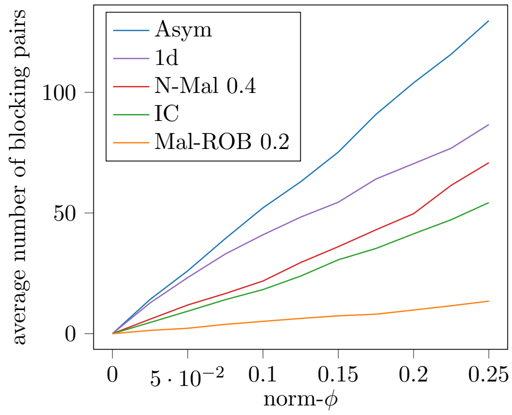


Figure 5.12: Average number of blocking pairs for the man-optimal matchings of five exemplary instances, for norm-ϕ-values from 0 to 0.25.

Definition 5.3. Let $\mathcal{I} = (U, W, \mathcal{P})$ be an SM instance. The *minimum weight matching* is the (not necessarily stable) perfect matching M that minimises $\sum_{a \in A} rk_a(M(a))$.

Examining the robustness of the minimum weight matching for all instances of the data set, we observed the following:

Finding 5. Most matchings that are not stable in the initial preferences have a very low stable matching probability for any norm-ϕ value.

For any instance except for the Mal-ROB culture and the MA and ROB instances, it contained at least 4 blocking pairs which resulted in having a stable matching probability of 0% for every norm-ϕ value. However, in Figure 5.11, we present one special Mal-ROB instance with norm-$\phi = 0.2$ where the min-weight matching initially contains only one blocking pair. In this case, the matching becomes even more robust than the initially stable matchings. Notice that this is an outlier and for usual instances that are not closely related to the ROB instance, the minimum-weight matching is very unrobust.

5.2.5 Number of blocking pairs

In this section, instead of only considering stable and unstable matchings, we count the number of blocking pairs for a matching when adding random noise. The justification is that while an instance may have a low stable matching probability, the average number of blocking pairs in the perturbed instance may still be relatively low. In Theorem 3.4, we showed that this problem is already hard for the worst-case robustness and thus we cannot efficiently compute the average number of blocking pairs for a specific number of swaps. Therefore, we again use the Mallows model to determine the average number of blocking pairs.
Figure 5.12 shows the development of the average number of blocking pairs for increasing norm-ϕ values for five exemplary instances of the data set. The instances not depicted

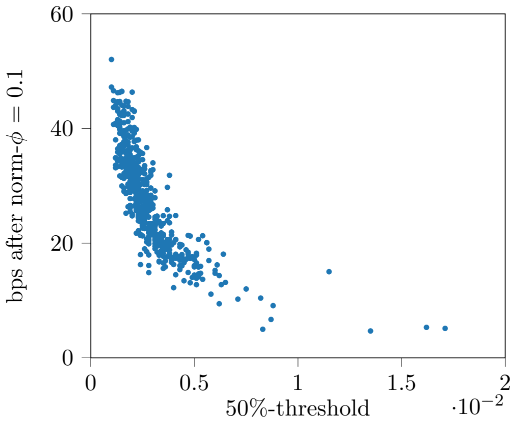


Figure 5.13: Correlation of the 50%-threshold and the average number of blocking pairs in a perturbed instance with norm-$\phi = 0.1$.

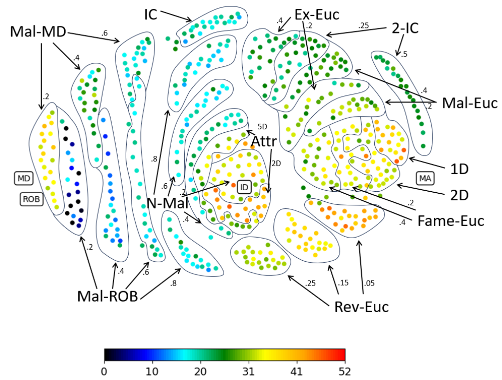


Figure 5.14: Average number of blocking pairs for norm-$\phi = 0.1$ for each instance of our data set.

in the plot behave quite similarly to these five selected instances. Contrary to the stable matching probability, and as can be seen in Figure 5.12, the number of blocking pairs increases almost linearly with increasing norm-ϕ (at least for small norm-ϕ values). Therefore, we use the average number of blocking pairs for norm-$\phi = 0.1$ as our measure instead of looking at a specific probability (e.g. 50%, as we did for the stability setting).

Finding 6. The number of blocking pairs and the 50% stability threshold have a quite strong correlation. However, for very (similarly) unrobust instances, the number of blocking pairs after some random noise can significantly differ.

In Figure 5.13, we can see that, at least for the man-optimal matchings, two instances that have a very similar 50%-stability-threshold can have a very different average number of blocking pairs. Therefore, it is sensible to examine which instances have a particularly low number of average blocking pairs. Figure 5.14 shows the map of our instances coloured according to their average number of blocking pairs. Notice that a small value now corresponds to a robust instance. While the robust instances remain similar to the stability setting, we can now better differentiate between non-robust instances. We observe that Euclidean cultures that do not involve perturbing single preference lists (like 1D, 2D, Attr and Rev-Euc) have the most blocking pairs. The instances have less blocking pairs when they are closer to the Mal-ROB and IC instances.

5.3 Robust Pairs

In this section, we analyse the robustness of stable pairs as defined in Chapter 2. Analogously to the matching setting, for each SM instance $\mathcal{I}$ and stable pair $\{m, w\}$, we define the 50%-stability threshold as the smallest norm-ϕ-value such that the probability of

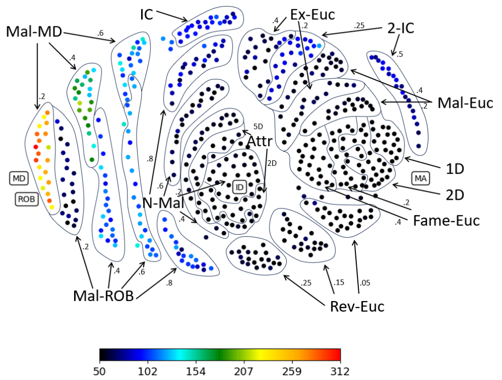


Figure 5.15: Number of stable pairs in the initial instance.

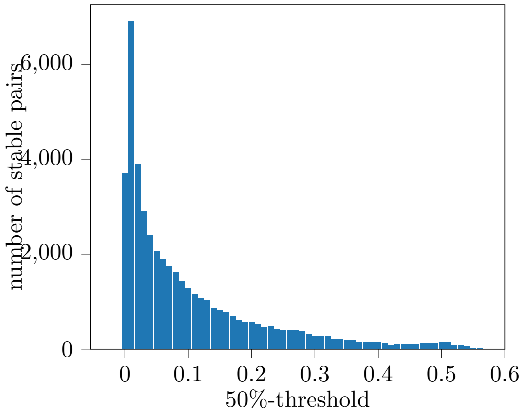


Figure 5.16: Distribution of the 50%-thresholds of all stable pairs in our data set.

$\{m, w\}$ being stable in an instance drawn from the Mallows model with this dispersion parameter from $\mathcal{I}$ is below 50%. To compute all stable pairs of an instance, we use the algorithm by Gusfield [Gus87]. Clearly, each instance contains at least n stable pairs. In Figure 5.15, one can see the number of stable pairs for each instance in our data set. Especially the MD and Mal-MD have many stable pairs. The distribution of the 50%-threshold of the stable pairs of all instances can be seen in Figure 5.16. Notably, stable pairs with a 50%-threshold above 0.1 are by no means an exception. In fact, as we will see later with the average stable pair robustness, an average stable pair often achieves a robustness greater than 0.1 and the low robustness values seen in Figure 5.16 are mostly due to the many and unrobust stable pairs of the MD and Mal-MD instances.

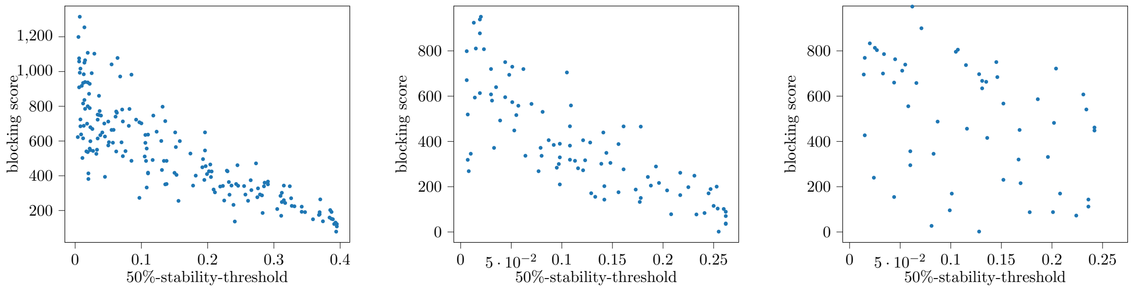
5.3.1 Easy robustness measures

As for the matching setting, we want to find a measure that has a high correlation with the 50%-stability-threshold. Again, we cannot hope for an exact measure due to our hardness result in Corollary 4.17. Furthermore, the stable pair robustness does not only depend on the distance of some specific pairs to being blocking (as in the matching case), but it also depends on the stability of other pairs, making it more unlikely to find an easy measure with a strong correlation.

For each woman w' that m prefers to w, we will add a penalty depending on the rank of m in w' (namely $n - rk_{w'}(m)$). If w' ranks m high, $\{m, w'\}$ has a good chance to be blocking in a matching containing $\{m, w\}$.

Definition 5.4. Let $\mathcal{I} = (U, W, \mathcal{P})$ be an SM instance with $|U| = |W| = n$ and let $\{m, w\}$ be a pair with $m \in U$ and $w \in W$. The *blocking score* of $\{m, w\}$ is

$$\mathcal{B}_{\mathcal{I}}(m, w) = \sum_{i=1}^{rk_m(w)-1} n - rk_{p_m(i)}(m) + \sum_{i=1}^{rk_w(m)-1} n - rk_{p_w(i)}(w)$$



(a) Mal-Asym 0.4 (b) 2g-IC 0.5 (c) Norm-Mallows 0.4

Figure 5.17: Correlation between the 50%-threshold and the blocking score for all stable pairs of three instances.

Note that if m and w are mutual top-choices, the blocking score is 0. As for the blocking pair proximity of a matching, small values correspond to a high robustness. The correlation with the 50%-threshold showed the following:

Finding 7. The blocking score does not achieve a correlation as strong as the blocking pair proximity in the matching setting, the Pearson Correlation Coefficient being around -0.54.

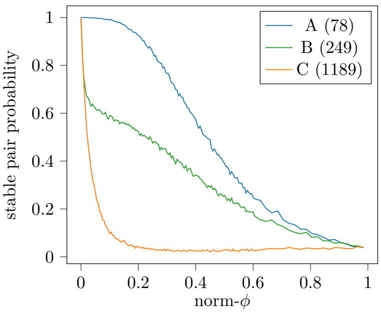
However, it can still be a good way to compare the pair robustness of stable pairs of the same instance. Figure 5.17 shows the correlation for the stable pairs of three exemplary instances. For two of them, we can observe a somewhat strong correlation, while for the Norm-Mal instance, the blocking score is not a good indicator of the 50%-threshold.

To quantify the robustness of the stable pairs of an instance, based on the 50%-threshold of stable pairs, for each instance, we introduce the following measures:

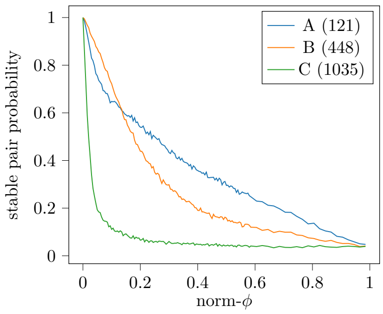
- the **most robust stable pair** is the stable pair with the highest 50%-stability threshold.
- the **least robust stable pair** is the stable pair with the lowest 50%-stability threshold.
- the **average stable pair robustness** is the average over the 50%-stability thresholds of all stable pairs.
- the **robustness variance** is the variance over the 50%-stability thresholds of all stable pairs.

In the following subsections, we will closer examine these indicators for the instances of our data set. In Figure 5.18, we chose six instances that are representative for the data set regarding the robustness of their stable pairs. For each instance, we depict the most and least robust stable pair as well as some interesting stable pairs in between. We will use these plots to draw attention to some interesting observations regarding our pair robustness indicators.

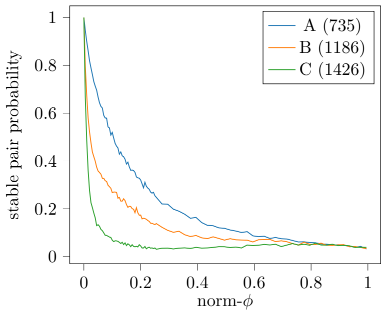
Figure 5.18: Average-case robustness of some exemplary stable pairs for six instances. For each pair, its blocking score is written in brackets.



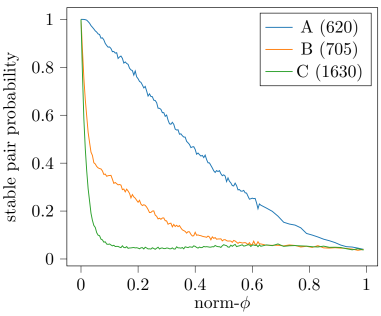
(a) Mal-Asym 0.6



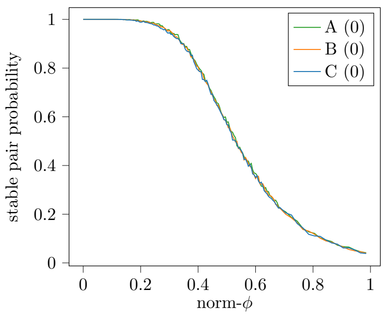
(b) Fame-Euc 0.4



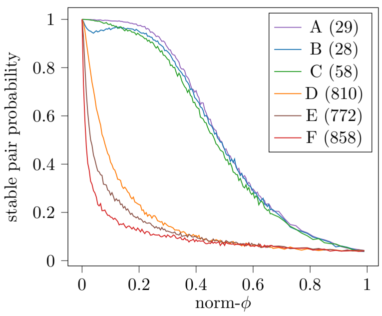
(c) Rev-Euc 0.05 a



(d) Rev-Euc 0.05 b



(e) ROB



(f) Mal-ROB 0.2

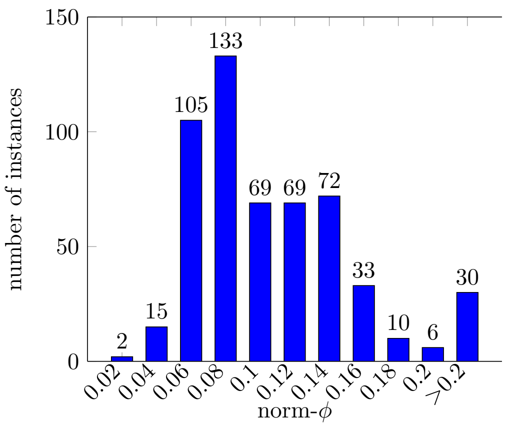


Figure 5.19: Distribution of the average stable pair robustness. All values are rounded up to a multiple of 0.02.

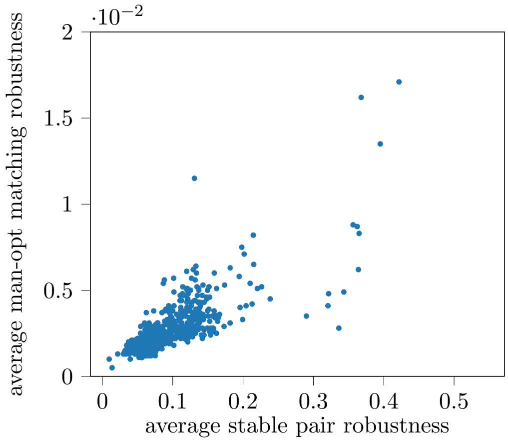


Figure 5.20: Correlation between the 50%-stability threshold and the average stable pair robustness for the instances of our data set. Instances with an extremely high robustness are left out.

5.3.2 Average stable pair robustness

We plotted the distribution of the average stable pair robustness for all instances of the data set in Figure 5.19. We can directly observe the following:

Finding 8. The average stable pair robustness is much higher than the matching robustness of an instance. On average, around 50 random swaps per preference list are needed to make a pair unstable.

This finding matches our intuition, since, while it is enough to create one blocking pair to make a matching unstable, a certain pair in this matching can still be stable in some other stable matching. Thus, it is much harder for a matching to be stable than for a single pair.

As in the matching setting, there are some instances from the Mal-ROB culture that are particularly robust. However, **??** shows that the difference to the "normal" instances is smaller in the pairs setting. For instance, consider the stable pair robustness of the Mal-Asym instance depicted in Figure 5.18a. The figure shows the most robust and least robust stable pair as well as a pair with an average robustness. The average stable pair robustness for this instance is 0.15. However, the stable pair probability decreases slowly for most stable pairs: For a norm-ϕ value of 0.4, the blue and green pairs have still a quite high stable pair probability.
Still, the 50%-stability-threshold is not a bad indicator of the average stable pair robustness, as can be seen in Figure 5.20. The Pearson Correlation Coefficient is 0.65.

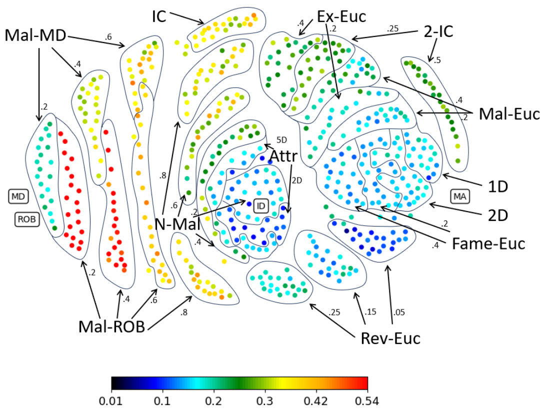


Figure 5.21: Average stable pair robustness for all instances of the data set.

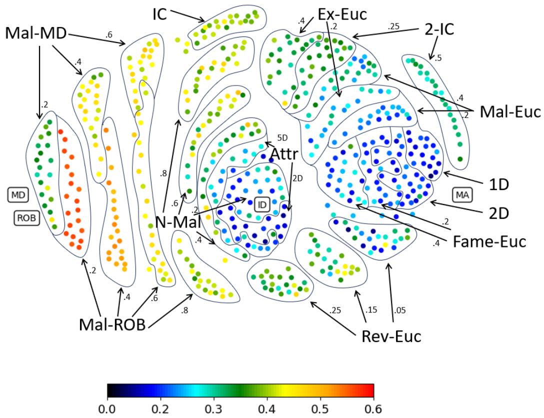


Figure 5.22: 50%-threshold of the most robust pair for each instance of the data set.

5.3.3 Most robust stable pair

First, one should note that it is not quite clear how to intuitively define the most robust stable pair. While we chose to look for the stable pair with the highest 50%-threshold, Figure 5.18b shows that there could be other legitimate definitions that would yield other results. For our definition, the blue pair is the most robust, but if one is interested in smaller changes/norm-ϕ values, one could perfectly argue that the orange pair is the most robust. Generally, stable pairs do not behave as monotonously as stable matchings. In fact, the blue stable pair in Figure 5.18f, after having a stable pair probability of 0.94 at norm-$\phi = 0.04$, increases to 0.97 and only drops below 0.94 again at a norm-ϕ value greater 0.2.
We now turn to Figure 5.22, which depicts the most robust stable pair for all instances of the data set. One can observe that the most robust stable pair greatly differs from instance to instance. Unsurprisingly, pairs in the Mal-ROB culture tend to be more robust. Generally, the right part of the map has small values for the most robust stable pair (see Figure 5.18c for an example), while the most robust stable pair for the instances on the left of the map has a high 50%-threshold. An instance has usually a worse most robust stable pair if it is close to the ID or MA instance, as can be seen in Figure 5.22. The correlation to the matching-50%-threshold is fairly low (the Pearson Correlation Coefficient being 0.3), among other things because of (extreme) outliers (see Figure 5.23). Ergo, finding a robust stable pair in an instance does not imply that the instance admits robust stable matchings.

5.3.4 Least robust stable pair

Most instances have stable pairs that are very unrobust. Thus, the least robust stable pair is not a very meaningful measure. For the same reason, we do not examine the difference between the most and least robust stable pair, as it mostly almost corresponds to the value of the most robust pair. Only four non-special instances have a least robust stable pair whose threshold is significantly larger than zero, and all of them are from the

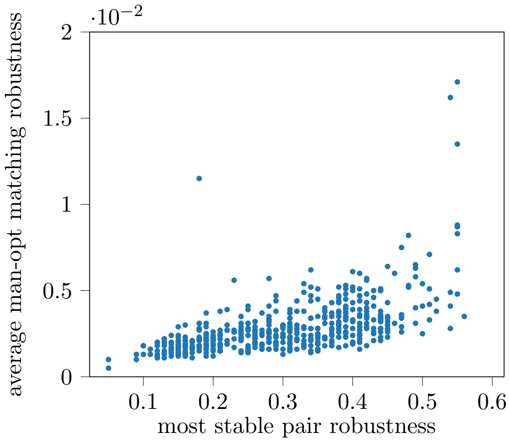


Figure 5.23: Correlation between the 50%-stability threshold and the most robust stable pair robustness for the instances of our data set. Instances with an extremely high robustness are left out.

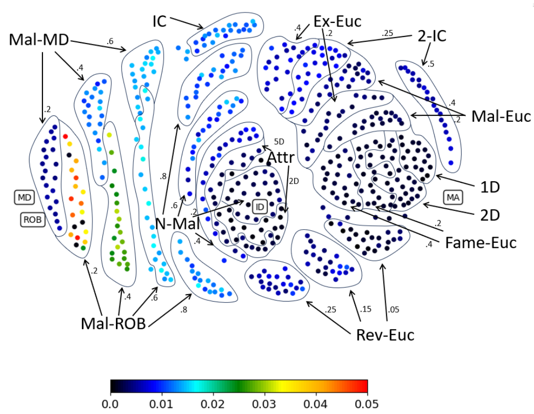


Figure 5.24: Robustness variance for each instance of the data set.

Mal-ROB culture.

5.3.5 Robustness variance

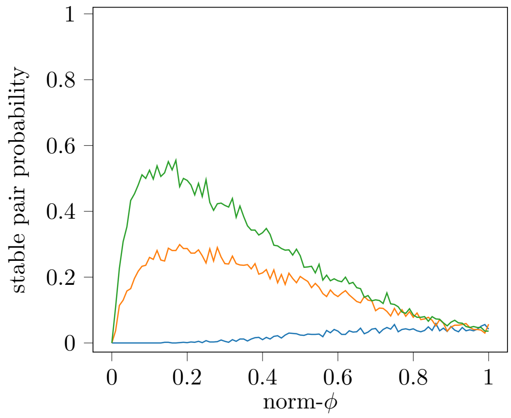
With this measure, we try to quantify by how much the robustness of two different stable pairs differ. As already suggested when regarding the robustness of the different stable pairs in Figure 5.18, we observed the following:

Finding 9. Two different stable pairs can vary significantly in their robustness, the average robustness variance of an instance being 0.0085 norm-ϕ.

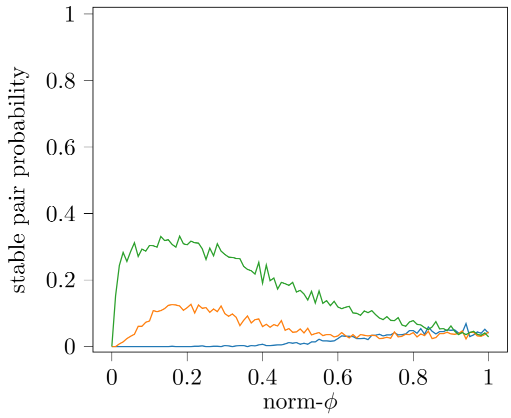
As can be seen in Figure 5.24, the left side of the map again has clearly greater values. Even though the ROB instance itself has a very low robustness variance (see Figure 5.18e), most Mal-ROB instances have a very high robustness variance. More precisely, the stable pairs of the Mal-ROB instances that are also stable in the ROB instance are very robst, while all other stable pairs are not robust. This is illustrated in Figure 5.18f. For instances of other cultures, we often encounter single stable pairs that are much more robust than any other stable pair of the instance, as in Figure 5.18d. This results in a greater robustness variance and also shows that it is sensible to regard the average and most robust stable pairs separately.
One should also notice that the map of the robustness variance is very similar to the map of the number of stable pairs in the initial instance, suggesting that if an instance has only few stable pairs, they are similarly robust.

5.3.6 Unstable pairs

At last, we want to examine the development of the stable pair probability of unstable pairs. While clearly any unstable pair has a stable pair probabilty of 0 for norm-$\phi = 0$,



(a) IC



(b) 2d-rev 0.05

Figure 5.25: Average-case robustness of some exemplary unstable pairs for two different instances. Each line corresponds to one stable pair.

we expect some unstable pairs to have a much higher stable pair probability (for some norm-ϕ values) than other stable pairs (e.g. because of a higher mutual agreement of the agents). Contrary to unstable matchings, we observe that such unstable pairs with high stable pair probability exist and are not uncommon:

Finding 10. Pairs that are initially unstable can have a quite high stable pair probability for small norm-ϕ values.

Section 5.3.6 shows some exemplary unstable pairs for two instances. We show the unstable pairs with the highest, lowest and an average stable pair probability. For most instances, there is some unstable pair that has a stable pair probability around 0.5 for norm-ϕ values close to 0.2. However, the stable pair probabilty of most unstable pairs does not increase significantly and behaves similar to the blue pairs in Figures 5.25a and 5.25b.

Chapter 6

Conclusion

In this work, we analysed the complexity of computing the worst- and average-case robustness of stable matchings, agents and pairs in Stable Marriage instances and conducted experiments on the robustness of matchings and pairs. In the decision setting, among others, we showed that regarding stable agents, *Swap* operations lead to easier computational problems than *Delete* operations. On the other hand, while the worst-case robustness of matchings can be easily computed, requiring a certain number of blocking pairs makes the problems intractable. In the counting setting, we have shown that #Matching-Swap, #Agent-Delete and #Agent-Pair are #P-hard, while all other counting variants of the considered problems are already intractable in the decision setting, implying that it is hard to compute the exact average-case robustness of a stable matching, pair or agent against swaps or deletions. In the course of this, we encountered interesting connections between robustness in stable marriage and graph problems such as Edge Cover and Bipartite Vertex Cover.

Despite these computational hardness results, we saw that a sampling approach can be used to measure the robustness of stable matchings and pairs in practice. Conducting these experiments, we observed several interesting facts regarding the robustness in stable marriage instances. First of all, stable matchings are very unrobust in general. Therefore, in dynamic settings where the preferences change, one should be careful assuming that an initially stable matching remains stable over time, especially when we are close to a Euclidean setting, i.e. when agents prefer other agents that are similar to them. On the other hand, matching market designers have some means to make their stable matchings more robust. For instance, while men- or women-optimal stable matchings are very unrobust, using the summed-rank minimising stable matching or the robust stable matching method usually leads to significantly more robust stable matchings. The fact that the summed-rank minimising stable matchings are more robust than the stable matchings produced by the Gale-Shapley algorithm allows the pleasant interpretation that 'unfair' stable matchings where one side is preferred over the other are less robust to changes in the real world than stable matchings that are considered fairer. Furthermore, we saw that while there is some correlation between the worst- and average-case robustness, the worst-case robustness cannot distinguish between most instances; this fact further motivates the study of the average-case robustness of stable matchings.

Analysing the robustness relative to the number of blocking pairs and the robustness of stable pairs allows to further distinguish between instances in our data set. We observed that stable pairs are much more robust than stable matchings, but stable pairs of one instance can vary greatly in their robustness.

In the experimental section, the Mal-ROB instances turned out to have very extreme robustness values. While they can serve as an example of how extreme instances can behave, one should probably reduce the number of these instances for similar work in the future to obtain a clearer picture with less outliers.

For future work, settling the complexity of Destructive-Pair-Delete is the most pressing open question. On the one hand, such a reduction must allow a rotation/cycle involving the designated pair, since otherwise the construction would also hold for the polynomial-time solvable case of Boehmer et al. [Boe+21c]. On the other hand, most such reduction attempts that we tried have failed so far because one could always delete a small number of agents to destroy the rotation/cycle. The question arises whether one can always "destroy" a rotation and possible follow-up rotations by deleting a small number of agents, which would probably imply that the problem is polynomial-time solvable.

Beyond that, one may investigate the parametrised complexity of our problems. While some of our reductions imply parametrised hardness (for instance, Destructive-Pair-Swap is W[1]-hard with respect to the budget), our reductions in the counting section do not settle the average-case complexity of these problems parametrised by the budget. Furthermore, the correlation between the 50%-stability threshold and our blocking pair proximity is so strong that one could analyse whether the blocking pair proximity admits any theoretical lower and upper bounds to the exact average-case robustness. Moreover, while the summed-rank minimizing matching of an instance can be found in polynomial time via linear programming, we only give an ILP formulation for the robust stable matching. It would be interesting to analyse whether one can obtain a polynomial-time algorithm by relaxing this formulation or whether finding such a matching is NP-hard.

Additionally, one can conduct further experiments, analysing the robustness of agents instead of matchings and pairs, as well as the robustness against deletions instead of swaps. While we would expect that agents are even more robust than pairs, a deletion intuitively seems to have a greater impact on stability than a swap, leading to a lower robustness. Finally, it would be interesting to use real-world data in order to examine which cultures are particularly close to real-world applications and to verify or refute observations made with our synthetic data.

Literature

[Abd+05] Atila Abdulkadiroğlu, Parag A Pathak, Alvin E Roth, and Tayfun Sönmez. *The Boston public school match.* In: *American Economic Review* 95.2 (2005), pp. 368–371 (cit. on p. 9).

[ALG20] Mohammad Akbarpour, Shengwu Li, and Shayan Oveis Gharan. *Thickness and Information in Dynamic Matching Markets.* In: *J. Political Econ.* 128.3 (2020), pp. 783–815 (cit. on p. 10).

[Azi+20] Haris Aziz, Péter Biró, Serge Gaspers, Ronald de Haan, Nicholas Mattei, and Baharak Rastegari. *Stable Matching with Uncertain Linear Preferences.* In: *Algorithmica* 82.5 (2020), pp. 1410–1433 (cit. on p. 10).

[Azi+22] Haris Aziz, Péter Biró, Tamás Fleiner, Serge Gaspers, Ronald de Haan, Nicholas Mattei, and Baharak Rastegari. *Stable matching with uncertain pairwise preferences.* In: *Theor. Comput. Sci.* 909 (2022), pp. 1–11 (cit. on p. 10).

[BCK22] Kristóf Bérczi, Gergely Csáji, and Tamás Király. *Manipulating the outcome of stable matching and roommates problems.* In: *CoRR* abs/2204.13485 (2022). arXiv: `2204.13485` (cit. on p. 10).

[BD97] Russ Bubley and Martin Dyer. *Graph orientations with no sink and an approximation for a hard case of# SAT.* In: *Proceedings of the Eighth Annual ACM-SIAM Symposium on Discrete Algorithms.* 1997, pp. 248–257 (cit. on p. 36).

[BH23] Dorothea Baumeister and Tobias Hogrebe. *On the complexity of predicting election outcomes and estimating their robustness.* In: *SN Computer Science* 4.4 (2023), p. 362 (cit. on pp. 10, 46).

[Bha+15] Sayan Bhattacharya, Martin Hoefer, Chien-Chung Huang, Telikepalli Kavitha, and Lisa Wagner. *Maintaining Near-Popular Matchings.* In: *Proceedings of the 42nd International Colloquium on Automata, Languages, and Programming (ICALP '15).* Springer, 2015, pp. 504–515 (cit. on p. 10).

[BHN22a] Niclas Boehmer, Klaus Heeger, and Rolf Niedermeier. *Deepening the (Parameterized) Complexity Analysis of Incremental Stable Matching Problems.* In: *Proceedings of the 47th International Symposium on Mathematical Foundations of Computer Science (MFCS '22).* Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2022, 21:1–21:15 (cit. on p. 10).

[BHN22b] Niclas Boehmer, Klaus Heeger, and Rolf Niedermeier. *Theory of and Experiments on Minimally Invasive Stability Preservation in Changing Two-Sided Matching Markets.* In: *Proceedings of the Thirty-Sixth AAAI Conference on Artificial Intelligence (AAAI '22).* AAAI Press, 2022, pp. 4851–4858 (cit. on p. 10).

[BHS22] Niclas Boehmer, Klaus Heeger, and Stanisław Szufa. *A Map of Diverse Synthetic Stable Roommates Instances.* In: *arXiv preprint arXiv:2208.04041* (2022) (cit. on pp. 11, 47, 50).

[BLY20] Mariagiovanna Baccara, SangMok Lee, and Leeat Yariv. *Optimal dynamic matching.* In: *Theor. Econ.* 15.3 (2020), pp. 1221–1278 (cit. on p. 10).

[Boe+20] Niclas Boehmer, Robert Bredereck, Piotr Faliszewski, and Rolf Niedermeier. *On the robustness of winners: counting briberies in elections.* In: *arXiv preprint arXiv:2010.09678* (2020) (cit. on pp. 31, 49).

[Boe+21a] Niclas Boehmer, Robert Bredereck, Piotr Faliszewski, and Rolf Niedermeier. *Winner Robustness via Swap- and Shift-Bribery: Parameterized Counting Complexity and Experiments.* In: *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI 2021, Virtual Event / Montreal, Canada, 19-27 August 2021.* Ed. by Zhi-Hua Zhou. ijcai.org, 2021, pp. 52–58 (cit. on p. 10).

[Boe+21b] Niclas Boehmer, Robert Bredereck, Piotr Faliszewski, Rolf Niedermeier, and Stanisław Szufa. *Putting a compass on the map of elections.* In: *arXiv preprint arXiv:2105.07815* (2021) (cit. on p. 45).

[Boe+21c] Niclas Boehmer, Robert Bredereck, Klaus Heeger, and Rolf Niedermeier. *Bribery and control in stable marriage.* In: *Journal of Artificial Intelligence Research* 71 (2021), pp. 993–1048 (cit. on pp. 10–12, 19, 24–27, 29, 70).

[Boe+22] Niclas Boehmer, Robert Bredereck, Piotr Faliszewski, and Rolf Niedermeier. *A Quantitative and Qualitative Analysis of the Robustness of (Real-World) Election Winners.* In: *Equity and Access in Algorithms, Mechanisms, and Optimization, EAAMO 2022, Arlington, VA, USA, October 6-9, 2022.* ACM, 2022, 7:1–7:10 (cit. on p. 10).

[Boe+23] Niclas Boehmer, Robert Bredereck, Dusan Knop, and Junjie Luo. *Fine-grained view on bribery for group identification.* In: *Auton. Agents Multi Agent Syst.* 37.1 (2023), p. 21 (cit. on p. 10).

[Bre+20] Robert Bredereck, Jiehua Chen, Dušan Knop, Junjie Luo, and Rolf Niedermeier. *Adapting Stable Matchings to Evolving Preferences.* In: *Proceedings of the Thirty-Fourth AAAI Conference on Artificial Intelligence (AAAI '20).* AAAI Press, 2020, pp. 1830–1837 (cit. on p. 10).

[BSS22] Markus Brill, Ulrike Schmidt-Kraepelin, and Warut Suksompong. *Margin of victory for tournament solutions.* In: *Artif. Intell.* 302 (2022), p. 103600 (cit. on p. 10).

[CSS21] Jiehua Chen, Piotr Skowron, and Manuel Sorge. *Matchings under Preferences: Strength of Stability and Tradeoffs.* In: *ACM Trans. Economics and Comput.* 9.4 (2021), 20:1–20:55 (cit. on p. 9).

[DL05] Ettore Damiano and Ricky Lam. *Stability in dynamic matching markets.* In: *Games Econ. Behav.* 52.1 (2005), pp. 34–53 (cit. on p. 10).

[Dov22] Laura Doval. *Dynamically stable matching.* In: *Theor. Econ.* 17.2 (2022), pp. 687–724 (cit. on p. 10).

[DP23] Michelle Döring and Jannik Peters. *Margin of Victory for Weighted Tournament Solutions*. In: *Proceedings of the 2023 International Conference on Autonomous Agents and Multiagent Systems, AAMAS 2023, London, United Kingdom, 29 May 2023 - 2 June 2023.* Ed. by Noa Agmon, Bo An, Alessandro Ricci, and William Yeoh. ACM, 2023, pp. 1716–1724 (cit. on p. 10).

[Eib+23] Eduard Eiben, Gregory Z. Gutin, Philip R. Neary, Clément Rambaud, Magnus Wahlström, and Anders Yeo. *Preference swaps for the stable matching problem*. In: *Theor. Comput. Sci.* 940.Part (2023), pp. 222–230 (cit. on p. 10).

[Fei+20] Itai Feigenbaum, Yash Kanoria, Irene Lo, and Jay Sethuraman. *Dynamic Matching in School Choice: Efficient Seat Reallocation After Late Cancellations.* In: *Manag. Sci.* 66.11 (2020), pp. 5341–5361 (cit. on p. 10).

[FHH09] Piotr Faliszewski, Edith Hemaspaandra, and Lane A. Hemaspaandra. *How Hard Is Bribery in Elections?* In: *J. Artif. Intell. Res.* 35 (2009), pp. 485–532 (cit. on p. 10).

[FR16] Piotr Faliszewski and Jörg Rothe. *Control and Bribery in Voting.* In: *Handbook of Computational Social Choice.* Ed. by Felix Brandt, Vincent Conitzer, Ulle Endriss, Jérôme Lang, and Ariel D. Procaccia. Cambridge University Press, 2016, pp. 146–168 (cit. on p. 10).

[Gaj+20] Karthik Gajulapalli, James A. Liu, Tung Mai, and Vijay V. Vazirani. *Stability-Preserving, Time-Efficient Mechanisms for School Choice in Two Rounds.* In: *Proceedings of the 40th IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS '20).* Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020, 21:1–21:15 (cit. on pp. 9, 10).

[Gan+22] Rohith Reddy Gangam, Tung Mai, Nitya Raju, and Vijay V. Vazirani. *A Structural and Algorithmic Study of Stable Matching Lattices of "Nearby" Instances, with Applications*. In: *42nd IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science, FSTTCS 2022, December 18-20, 2022, IIT Madras, Chennai, India.* Ed. by Anuj Dawar and Venkatesan Guruswami. Vol. 250. LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2022, 19:1–19:20 (cit. on p. 9).

[Gan+23] Rohith Reddy Gangam, Tung Mai, Nitya Raju, and Vijay V. Vazirani. *The Stable Matching Lattice under Changed Preferences, and Associated Algorithms*. In: *CoRR* abs/2304.02590 (2023). arXiv: `2304.02590` (cit. on p. 9).

[Gen+17a] Begum Genc, Mohamed Siala, Barry O'Sullivan, and Gilles Simonin. *Finding Robust Solutions to Stable Marriage.* In: *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence (IJCAI '17).* ijcai.org, 2017, pp. 631–637 (cit. on p. 9).

[Gen+17b] Begum Genc, Mohamed Siala, Barry O'Sullivan, and Gilles Simonin. *Robust Stable Marriage.* In: *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence (AAAI '17).* AAAI Press, 2017, pp. 4925–4926 (cit. on p. 9).

[Gen+19] Begum Genc, Mohamed Siala, Gilles Simonin, and Barry O'Sullivan. *Complexity Study for the Robust Stable Marriage Problem.* In: *Theor. Comput. Sci.* 775 (2019), pp. 76–92 (cit. on p. 9).

[GS62] David Gale and Lloyd S Shapley. *College admissions and the stability of marriage.* In: *The American Mathematical Monthly* 69.1 (1962), pp. 9–15 (cit. on p. 9).

[Gus87] Dan Gusfield. *Three fast algorithms for four problems in stable marriage.* In: *SIAM Journal on Computing* 16.1 (1987), pp. 111–128 (cit. on pp. 26, 61).

[HHA10] Günter J Hitsch, Al Hortaçsu, and Dan Ariely. *Matching and sorting in online dating.* In: *American Economic Review* 100.1 (2010), pp. 130–163 (cit. on p. 9).

[Kar] Richard M Karp. *Reducibility among combinatorial problems* (cit. on p. 19).

[KLM16] Varun Kanade, Nikos Leonardos, and Frédéric Magniez. *Stable Matching with Evolving Preferences.* In: *Proceedings of Approximation, Randomization, and Combinatorial Optimization (APPROX/RANDOM 2016).* Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2016, 36:1–36:13 (cit. on p. 10).

[KMT+11] John Kennes, Daniel Monte, Norovsambuu Tumennasan, et al. *The daycare assignment problem.* Department of Economics and Business Economics, Aarhus BSS, 2011 (cit. on p. 9).

[Knu76] Donald Ervin Knuth. *Marriages stables.* In: *Technical report* (1976) (cit. on p. 55).

[Liu21] Ce Liu. *Stability in Repeated Matching Markets.* In: *CoRR* abs/2007.03794v2 (2021) (cit. on p. 10).

[Mal57] C. Mallows. *Non-null ranking models.* In: *Biometrica* 44 (1957), pp. 114–130 (cit. on p. 11).

[Man13] David F. Manlove. *Algorithmics of Matching Under Preferences.* Vol. 2. Series on Theoretical Computer Science. WorldScientific, 2013 (cit. on p. 9).

[MV18] Tung Mai and Vijay V. Vazirani. *Finding Stable Matchings That Are Robust to Errors in the Input.* In: *Proceedings of the 26th Annual European Symposium on Algorithms (ESA '18).* Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2018, 60:1–60:11 (cit. on p. 9).

[MW70] David G McVitie and Leslie B Wilson. *Stable marriage assignment for unequal sets*. In: *BIT Numerical Mathematics* 10.3 (1970), pp. 295–309 (cit. on p. 18).

[NR59] Robert Z Norman and Michael O Rabin. *An algorithm for a minimum cover of a graph*. In: *Proceedings of the American Mathematical Society* 10.2 (1959), pp. 315–319 (cit. on p. 36).

[PB83] J Scott Provan and Michael O Ball. *The complexity of counting cuts and of computing the probability that a graph is connected*. In: *SIAM Journal on Computing* 12.4 (1983), pp. 777–788 (cit. on p. 32).

[SBS18] Patrick Schober, Christa Boer, and Lothar A Schwarte. *Correlation coefficients: appropriate use and interpretation*. In: *Anesthesia & analgesia* 126.5 (2018), pp. 1763–1768 (cit. on p. 54).

[SYE13] Dmitry Shiryaev, Lan Yu, and Edith Elkind. *On elections with robust winners*. In: *International conference on Autonomous Agents and Multi-Agent Systems, AAMAS '13, Saint Paul, MN, USA, May 6-10, 2013*. Ed. by Maria L. Gini, Onn Shehory, Takayuki Ito, and Catholijn M. Jonker. IFAAMAS, 2013, pp. 415–422 (cit. on p. 10).

[Vat89] John H Vande Vate. *Linear programming brings marital bliss*. In: *Operations Research Letters* 8.3 (1989), pp. 147–153 (cit. on p. 55).