Stable Marriage with Groups of Similar Agents

Kitty Meeks[1] and Baharak Rastegari[2]

[1] School of Computing Science, University of Glasgow, Glasgow, UK
kitty.meeks@glasgow.ac.uk
[2] Electronics and Computer Science, University of Southampton, Southampton, UK
b.rastegari@soton.ac.uk

Abstract. Many important stable matching problems are known to be NP-hard, even when strong restrictions are placed on the input. In this paper we seek to identify structural properties of instances of stable matching problems which will allow us to design efficient algorithms using elementary techniques. We focus on the setting in which all agents involved in some matching problem can be partitioned into k different *types*, where the type of an agent determines his or her preferences, and agents have preferences over types (which may be refined by more detailed preferences within a single type). This situation would arise in practice if agents form preferences solely based on some small collection of agents' attributes. We also consider a generalisation in which each agent may consider some small collection of other agents to be exceptional, and rank these in a way that is not consistent with their types; this could happen in practice if agents have prior contact with a small number of candidates. We show that (for the case without exceptions), the well-known NP-hard matching problem Max SMTI (that of finding the maximum cardinality stable matching in an instance of stable marriage with ties and incomplete lists) belongs to the parameterised complexity class FPT when parameterised by the number of different types of agents needed to describe the instance. This tractability result can be extended to the setting in which each agent promotes at most one "exceptional" candidate to the top of his/her list (when preferences within types are not refined), but the problem remains NP-hard if preference lists can contain two or more exceptions and the exceptional candidates can be placed anywhere in the preference lists.

1 Introduction

Matching problems occur in various applications and scenarios such as the assignment of children to schools, college students to dorm rooms, junior doctors to hospitals, and so on. In all the aforementioned, and similar, problems, it is understood that the participants (which we will refer to as agents) have preferences over other agents, or subsets of agents. The majority of the literature assumes that these preferences are ordinal, and that is the assumption we make in this work as well. Moreover, it is widely accepted that a "good" and "reasonable" solution to a matching problem must be *stable*, where stability is defined

according to the context of the problem at hand. Intuitively speaking, a stable solution guarantees that no subset of agents find it in their best interest to leave the prescribed solution and seek an assignment amongst themselves. Unfortunately, many interesting and important stable matching problems are known to be NP-hard even for highly restricted cases.

In this paper we focus on the *Stable Marriage problem (SM)*, which is perhaps the most widely studied matching problem. In an instance of SM we have two disjoint sets of agents, men and women, each having a strict preference ordering over the individuals of the opposite sex (candidates). A solution to this problem is a *matching*, that is a mapping from men to women where each man is matched to at most one woman and vice versa. Each agent prefers being matched to remaining unmatched. A matching is *stable* if there are no two agents a and b who prefer each other to their assigned partners. If such a pair exists, we say that (a, b) is a *blocking pair*. *Stable Marriage with Incomplete lists (SMI)* is a generalisation of SM where agents are permitted to declare some candidates unacceptable. In their seminal work, Gale and Shapley [11] showed that every instance of SMI admits a stable matching that can be found in polynomial time by their proposed algorithm (GS). A simple extension of GS can be used to identify stable matchings in domains where agents are additionally allowed to express indifference between two or more candidates (*Stable Marriage with Ties and Incomplete lists (SMTI)*). However, it is known that (in contrast with SMI) an instance of SMTI might admit stable matchings of different sizes, and GS does not necessarily find the largest. In many practical applications, it is important to match as many agents as possible, but finding a matching which achieves this is much more computationally challenging: Max SMTI, the problem of determining a maximum cardinality stable matching (i.e., a stable matching with the largest size amongst all stable matchings) in an instance of SMTI, is known to be NP-hard [3,14,16,20], even when the input is heavily restricted.

Most hardness results in the study of stable matching problems are based on the premise that agents may have arbitrary preference lists. In practice, however, agents' preferences are likely to be more structured and correlated. In this work, we consider a setting where agents can be grouped into k different "types", where the type of an agent determines (most of) the agent's preferences, and also how s/he is compared against other agents. If we allow each agent to have a different type, this setup does not place any restrictions on the instance. However, we are interested in the setting where the number of types required to describe an instance is much smaller than the total number of agents: such a situation would arise in practice if agents derive their preferences by considering some small collection of attributes of other agents (where each of these attributes has a small number of possible values). As an example, consider the hospitals-residents job market in which junior doctors or residents are to be assigned to hospital posts. It is highly plausible that agents in this market base their preferences on small collection of candidates' attributes. E.g. hospitals might rank applicants based on their exam grade, interview score, etc, and junior doctors might rank the hospitals based on the programs they offer, their reputation, their geographic location,

etc. Similar observations have been made in the literature (see [2,4]) regarding stable marriage market and stable roommate market respectively, where agents form preferences based on candidates' attributes such as attractiveness, intelligence, wealth, etc. In this setting, we obtain our set of types by first partitioning agents by their profile of attributes, then further partitioning each set by the preference list over other profiles of attributes. Note that the number of possible preference lists depends only on the number of possible attribute profiles.

The notion of types is also useful if we are interested in a relaxation of stability, where agents are only willing to form a private arrangement with a partner who is distinctly superior to their current partner with respect to an important characteristic. It is reasonable to assume that in practice a certain amount of effort is required by both agents in a blocking pair to make a private arrangement outside the matching, and so agents are unlikely to make this effort for a very small improvement in their utility. Suppose that an agent is only willing to make the effort to form a private arrangement if it results in a significantly better partner, specifically one which has a significantly better value for the most important attribute. In this case we only need to consider attributes which are the most important for at least one agent, and moreover we might reasonably consider only a small number of categories of values for these attributes.

The simplest model is to assume that the agents of the same type are completely indistinguishable. That is, they have the same preference lists, and every other agent that finds their type acceptable is indifferent between them. Equivalently, we can say that each type has a preference ordering over types of the candidates, which need not be complete or strict. We also consider two generalisations of this basic model. In the first generalisation, agents no longer have to be indifferent between agents of the same type: they can refine their preference lists arbitrarily (so that agents of the same type still occur consecutively), so long as the preference lists for agents of the same type are identical. In the second generalisation, we instead enrich the basic model by allowing each agent to consider some small number of other agents "exceptional": such agents can appear anywhere in the preference list, regardless of their type. This situation with exceptions might arise in practice if, for example, an agent knows some of the candidates directly or through a third-party connection and, based on this additional information, ranks them disregarding their type, e.g. at the top or bottom of his/her preference list.

Our contribution. We consider the parameterised complexity of Max SMTI in all three settings. In the basic model and the extension which allows consistently refined preference lists, we show that the problem is in FPT parameterised by the number of types. In both settings the problem further becomes polynomial-time solvable if all preferences over types are strict. When exceptions are allowed in the preference lists, we demonstrate that the problem is once again in FPT, parameterised by the number of types, if each agent considers at most one agent exceptional, whom s/he promotes to the top of his/her preference list. On the other hand, if two arbitrarily placed exceptions are allowed, Max SMTI remains NP-hard, even if the number of types is bounded by a constant.

Due to shortage of space we have omitted or shortened the proofs. We refer the reader to the full version of the paper [18].

1.1 Definitions

Let N denote a set of n agents, which is composed of two disjoint sets. We use the term *candidates* to refer to the agents on the opposite side of the market to that of an agent under consideration. Each agent finds a subset of candidates acceptable and ranks them in order of preference. Preference orderings need not be strict, so it is possible for an agent to be indifferent between two or more candidates. We write $b \succ_a c$ to denote that agent a prefers candidate b to candidate c, and $b \simeq_a c$ to denote that a is indifferent between b and c. We write $b \succeq_a c$ to denote that a either prefers b to c or is indifferent between them. The indifference relation $\simeq_a$ implies an equivalence relation on acceptable candidates for a; each equivalence class under $\simeq_a$ is referred to as a tie.

In an instance of SMTI, a *matching* M is a pairing of men and women such that no one is paired with an unacceptable partner, each man is paired with at most one woman, and each woman is paired with at most one man. We write $(a, b) \in M$ to say that a and b are matched in M. We use $M(a)$ to denote the agent matched to a in M. We write $M(a) = \varnothing$ if agent a is unmatched in M. We assume that every agent prefers being matched to an acceptable candidate to remaining unmatched. Given an instance of SMTI, a matching M is *(weakly) stable* if there is no pair $(a, b) \notin M$ where a prefers b to his current partner in M, i.e., $b \succ_a M(a)$, and vice versa. For further background and terminology on stable matchings we refer the reader to [15].

We are concerned with the *parameterised complexity* of computational problems that are intractable in the classical sense. Parameterised complexity provides a multivariate framework for the analysis of hard problems: if a problem is known to be NP-hard, so that we expect the running-time of any algorithm to depend exponentially on some aspect of the input, we can seek to restrict this combinatorial explosion to one or more *parameters* of the problem rather than the total input size. This has the potential to provide an efficient solution to the problem if the parameter(s) in question are much smaller than the total input size. A parameterised problem with total input size n and parameter k is considered to be tractable if it can be solved by a so-called *FPT algorithm*, an algorithm whose running time is bounded by $f(k) \cdot n^{\mathcal{O}(1)}$, where f can be any computable function. Such problems are said to be *fixed parameter tractable*, and belong to the complexity class FPT. For further background on the theory of parameterised complexity, we refer the reader to [7,8,10].

1.2 Related Work

The NP-hardness of Max SMTI has been shown for a variety of restricted settings, for example: (1) even if each man's list is strictly ordered, and each woman's list is either strictly ordered or is a tie of length 2 [16], (2) even if each mans preference list is derived from a strictly-ordered master list of

women, and each woman's preference list is derived from a master list of men that contains only one tie [14], and (3) even if the SMTI instance has symmetric preferences; that is, for any acceptable (man, woman) pair (m_i, w_j), $rank(m_i, w_j) = rank(w_j, m_i)$ [20], where $rank(a, b)$ is defined to be one plus the number of candidates that a prefers to b.

There are a limited number of works addressing fixed-parameter tractability in stable matching problems. Marx and Schlotter [17] gave the first parameterised complexity results on Max SMTI. They showed that the problem is in FPT when parameterised by the total length of the ties, but is W[1]-hard when parameterised by the number of ties in the instance, even if all the men have strictly ordered preference lists. Very recently, three different works have studied hard stable matching problems from the perspective of parameterised complexity. Mnich and Schlotter [19] obtained results on the parameterised complexity of finding a stable matching which matches a given set of distinguished agents and has as few blocking pairs as possible. Gupta et al. [13] showed that several hard stable matching problems, including Max SMTI, are W[1]-hard when parameterised by the treewidth of the graph obtained by adding an edge between each pair of agents that find each other mutually acceptable. Gupta et al. [12] studied above guarantee parameterisations of the problem of finding a stable matching that balances between the dissatisfaction of men and women, with parameters that capture the degree of dissatisfaction.

Settings in which agents are partitioned into different types, or derive their preferences based on a set of attributes assigned to each candidate, have been considered for the problems of sampling and counting stable matchings in instances of SM or SR (Stable Roommate problem); see, e.g., [2,4,5]. Echenique et al. [9] studied the problem of characterising matchings that are rationalisable as stable matchings when agents' preferences are unobserved. They focused on a restricted setting that translates into assigning each agent a type based on several attributes, and assuming that agents of the same type are identical and have identical preferences. They remarked that empirical studies on marriage typically make such an assumption [6]. Bounded agent types have been considered by Aziz and de Keijzer [1] and Shrot et al. [22] to derive polynomial-time results for the coalition structure generation problem, an important issue in cooperative games when the goal is to partition the participants into exhaustive and disjoint coalitions in order to maximise the social welfare.

2 Our basic model: agents of the same type are indistinguishable

In this section we begin with a formal definition of the simplest model we consider, in which agents' preferences can be derived directly from the preferences of types over types of candidates. We then identify a necessary and sufficient condition, in terms of the type of the least desirable partner assigned to any agent of each type, for a matching to be stable in this model. We use this to show that, if there are k types, we can solve Max SMTI by solving $k^{\mathcal{O}(k)} \cdot \log n$

instances of Max Flow on directed networks with $\mathcal{O}(k)$ vertices and maximum edge capacity $\mathcal{O}(n)$. This implies that Max SMTI, parameterised by k, belongs to FPT.

2.1 Definition of typed instances

Assume that there are k types available for agents. Let $[k]$ denote the set $\{1, 2, \ldots, k\}$. Let N_i denote the set of agents that are of type i. Thus we have that the set of agents $N = \bigcup_{i \in [k]} N_i$. Each type i has a preference ordering over types of the candidates, which need not be complete or strict. We assume, without loss of generality, that $|N_i| > 0$ for all $i \in [k]$, and that each type finds at least one other type acceptable. We write $j \succ_i \ell$ if agents of type i strictly prefer agents of type j to agents of type ℓ. We write $j \simeq_i \ell$ to denote that agents of type i are indifferent between agents of types j and ℓ, and $j \succeq_i \ell$ if agents of type i prefer agents of type j to those of type ℓ or are indifferent between the two. We assume that given every two agents x and y of the same type:

1. x and y have identical preference lists , and
2. all other agents are indifferent between x and y.

These requirements imply that any agent either finds all agents of a given type acceptable (and is indifferent between them) or finds none of them acceptable. We say that an instance of a stable matching problem satisfying these requirements is *typed*, and refer to the standard problems with input of this form as Typed Max SMTI etc. Note that Typed Max SMTI remains NP-hard when k is considered to be part of the input: we can always create a typed instance by assigning each agent its own type.

A typed instance I of SMTI is given as input by specifying the number of types k and, for each type i, the set N_i of agents of type i as well as the preference ordering $\succ_i$ over types of the candidates. Observe that, if we are only given the preference list for each agent as input, it is straightforward to compute, in polynomial time, the coarsest partition of the agents into types that satisfies the definition of a typed instance (see [18]). Having found such a partition, the preference lists over types can also be constructed efficiently.

Example 1. Assume we have 4 types for the agents, all men are of type 1 and types 2, 3 and 4 correspond to women. Let the preference ordering of type 1 over types of women be as follows, where the preference list is ordered from left to right in decreasing order of preference, and the types in round brackets are tied: (2 3) 4. Assume that there are 7 women and w_1 and w_2 are of type 2, w_3 and w_4 are of type 3, and w_5, w_6 and w_7 are of type 4. Therefore, the preference lists of all men under the typed model are as follows: (w_1 w_2 w_3 w_4) (w_5 w_6 w_7).

2.2 An FPT algorithm for Typed Max SMTI

Let I be a typed instance of SMTI, and let M be a matching in I. We may assume without loss of generality that every agent is matched, by creating sufficiently

many dummy agents of type $k+1$ which are inserted at the end of each man's and woman's (possibly incomplete) preference list. We define $\text{worst}_M(i)$ to be the type of the least desirable agent with which any agent of type i is matched in M, breaking ties arbitrarily (e.g. lexicographically). Note that $\text{worst}_M(i)$ would be a dummy type if an agent of type i is unmatched (i.e. matched to a dummy agent) in M. Let $\text{type}(a)$ denote the type of a given agent a.

The key observation is that, in order to determine whether or not M is stable, it suffices to examine the values of $\text{worst}_M(i)$ for each $i \in [k]$.

Lemma 1. *Let I be a typed instance of SMTI. Then a matching M in I is stable if and only if there is no pair $(i,j) \in [k]^{(2)}$ such that $j \succ_i \text{worst}_M(i)$ and $i \succ_j \text{worst}_M(j)$.*

We say that a matching M *realises* a given function $\text{worst} : [k] \to [k+1]$ if, for each $i \in [k]$, the least desirable partner any agent of type i has in M is of type no worse than $\text{worst}(i)$. We say that a function $\text{worst} : [k] \to [k+1]$ is *I-stable* for an instance I of SMTI if there is no pair $(i,j) \in [k]^{(2)}$ such that $j \succ_i \text{worst}(i)$ and $i \succ_j \text{worst}(j)$. Given any I-stable function worst, we write $\max(\text{worst})$ for the maximum cardinality of any matching in I that realises worst. Using Lemma 1, it is straightforward to check that, given a typed instance I of SMTI, the cardinality of a solution to Max SMTI can be found by taking the largest value of $\max(\text{worst})$ over all I-stable functions worst.

Corollary 1. *Let I be a typed instance of SMTI. Then the cardinality of the largest stable matching in I is equal to* $\max\{\max(\text{worst}) : \text{worst is } I\text{-stable}\}$.

We next show that, given an arbitrary I-stable function worst, we can compute $\max(\text{worst})$ in time polynomial in k and $\log n$. We do this by solving $\mathcal{O}(\log n)$ instances of Max Flow on a directed network.

Lemma 2. *Let I be a typed instance of SMTI, and fix an I-stable function* worst*. We can compute* $\max(\text{worst})$ *in time $\mathcal{O}(k^3 \log^2 n)$.*

Proof (Proof sketch.). The proof is structured as follows. Suppose that in total there are n_1 women and n_2 men, so we have that $n = n_1 + n_2$. Note that $\max(\text{worst})$ is at most $\min\{n_1, n_2\}$, which in turn is at most $\lfloor n/2 \rfloor$. Therefore, using a binary search strategy, we can determine the maximum size of a matching realising worst by solving $\mathcal{O}(\log n)$ instances of the decision problem "Is $\max(\text{worst})$ at least c?", where $c \in \{1, \ldots, \min\{n_1, n_2\}\}$. We show that we can determine whether $\max(\text{worst}) \geq c$ by solving Max Flow on a directed network D with $\mathcal{O}(k)$ vertices, in which the maximum capacity of any edge is $\mathcal{O}(n)$ (see Figure 1); we can construct D from I in time $\mathcal{O}(k^2 \log n)$. Max Flow can be solved on D in time $\mathcal{O}(k^3 \log n)$, using an algorithm due to Orlin [21], where the $\log n$ factor is required to carry out arithmetic operations on integers of size $\mathcal{O}(n)$. Therefore, we conclude that we can compute $\max(\text{worst})$ in time $\mathcal{O}(k^3 \log^2 n)$. □

It then follows that Typed Max SMTI is in FPT parameterised by the number k of different types in the instance.

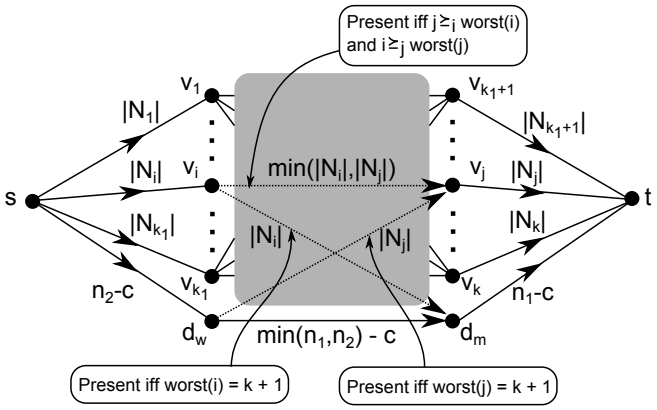

Fig. 1. Network D, constructed from an instance of SMTI in the proof of Lemma 2. Types $1, \ldots, k_1$ are types of women and types $k_1 + 1, \ldots, k$ are types of man. Each vertex v_i corresponds to type i and both vertices d_w and d_m correspond to the dummy type. In total there are n_1 women and n_2 men. We have $\max(\text{worst}) \geq c$ if and only if the maximum flow in D is equal to $n_1 + n_2 - c$.

Corollary 2. Typed Max SMTI *can be solved in time* $k^{\mathcal{O}(k)} \cdot \log^2 n + \mathcal{O}(n)$. *If we are only interested in computing the size of the maximum cardinality matching, and not the matching itself, this can be done in time* $k^{\mathcal{O}(k)} \cdot \log^2 n$.

3 Agents of the same type refine their preferences in the same way

In this section, we generalise the model from Section 2 by allowing agents to refine their preferences over candidates within a particular type, so long as agents of the same type still have identical preference lists. Our key result is that refining preferences in this way can never change the size of the largest stable matching, compared with the corresponding typed instance. We also use the tools we develop to deal with this generalisation to show that Max SMTI becomes polynomially solvable if preferences over types are strict, both in this setting and under the basic model.

3.1 Definition of consistently-refined-typed instances

Consider a generalisation of typed instances in which agents are no longer necessarily indifferent between two agents of the same type, however agents of the same type occur consecutively in preference lists. This means that for any two agents x and y of the same type i:

1. x and y have identical preference lists,
2. no agent of a different type appears between x and y in any preference list, and

3. if a tie in a preference list contains agents of two or more types, then that tie is in fact a union of types.

The third criterion allows us to define in a consistent way what it means for agents of type i to strictly prefer type j to type ℓ or to be indifferent between them. We will say that agents of type i prefer type j to type ℓ if and only if given every pair of agents x of type j and y of type ℓ all agents in N_i prefer x to y. On the other hand, if type i is indifferent between types j and ℓ it means that, in the preference list for each agent x of type i, all agents in $N_j \cup N_\ell$ belong to a single tie.

If an instance of a stable matching problem satisfies these slightly weaker requirements, we say that the instance is *consistently-refined-typed*, and refer to the standard problems with input of this form as Consistently-Refined-Typed Max SMTI etc.

A consistently-refined-typed instance I of SMTI is given as an input by specifying the number of types k and, for each type i, the set N_i of agents of type i as well as the preference ordering $\succ_i$ over agents. Note that for typed instances $\succ_i$ specified preferences over types, whereas here the preferences are over agents. However, we can compute preferences over types from preferences over agents in time $\mathcal{O}(kn)$. Note that if we are only given the preference list for each agent as input (i.e., no information about types is given), it is straightforward to compute, in polynomial time, the coarsest partition of the agents into types that satisfies the definition of consistently-refined-typed instance (see [18]).

Example 2. Assume we have 4 types for the agents, all men are of type 1 and types 2, 3, and 4 correspond to women. Assume also that we have 3 men m_1, m_2 and m_3, and 7 women where w_1 and w_2 are of type 2, w_3 and w_4 are of type 3, and w_5, w_6 and w_7 are of type 4. Let all men have the preference ordering $(w_1\ w_2\ w_3\ w_4)\ w_6\ (w_5\ w_7)$, women of types 2 and 3 have the preference ordering $(m_1\ m_2\ m_3)$, and women of type 4 have the preference ordering $m_2\ m_1$. This setting constitutes a consistently-refined-typed instance. It is easy to compute the preferences of type 1 agents over the types of women, which is (2 3) 4, similar to that of Example 1. Allowing men to have the preference ordering $(w_1\ w_2)\ w_3\ w_4\ (w_5\ w_6\ w_7)$, while keeping everything else unchanged, also gives us a consistently-refined-typed instance. In this new instance agents of type 1 have the strict preference ordering 2 3 4 over the types of women.

3.2 An FPT algorithm for Consistently-Refined-Typed Max SMTI

To extend the result for Typed Max SMTI to Consistently-Refined-Typed Max SMTI, we need the following result.

Lemma 3. *Let I be a consistently-refined-typed instance of SMTI and suppose that M is a matching in I such that there is no pair $(i,j) \in [k]^{(2)}$ where $j \succ_i \text{worst}_M(i)$ and $i \succ_j \text{worst}_M(j)$. Then there is a stable matching M' such that, for every $(i,j) \in [k]^{(2)}$, both M and M' contain the same number of pairs that consist of one agent of type i and another of type j. Moreover, given M, we can compute M' in time $\mathcal{O}(kn)$.*

Let I be a consistently-refined-typed instance of SMTI and let I' be a typed instance of SMTI that is obtained from I by ignoring the refined preferences within each type (i.e. every agent is indifferent between the candidates of the same type). It follows from the definition of stability that every matching that is stable in I is also stable in I'. Lemma 3 implies that for any stable matching M in I', there exists a stable matching M' in I of the same cardinality as M. Thus, in order to find a maximum cardinality matching in a consistently-refined-typed instance I of SMTI, it suffices to (1) solve the typed problem (i.e. ignore the refined preferences within each type) and then (2) use the algorithm provided in the proof of Lemma 3 (see [18]) to convert the solution to a matching of the same cardinality that is stable in the instance I. In fact, in (1) it is enough to only compute the maximum flow f (and not the matching M); the flow f, that can be computed in time $k^{\mathcal{O}(k)} \cdot \log^2 n$, provides sufficient information for the algorithm described in the proof of Lemma 3 to construct M' in time $\mathcal{O}(kn)$. Deriving a typed instance from a consistently-refined-typed instance can be done easily in time $\mathcal{O}(kn)$. It thus follows that Consistently-Refined-Typed Max SMTI is in FPT parameterised by the number k of different types in the instance.

Theorem 1. Consistently-Refined-Typed Max SMTI *can be solved in time* $k^{\mathcal{O}(k)} \cdot \log^2 n + \mathcal{O}(kn)$.

3.3 Strict preferences over types

Elsewhere in the paper, we assume that agents can be indifferent between agents of two or more types. It turns out that Max SMTI becomes easier if we restrict the set of possible instances by assuming that agents have strict preferences over types. We prove this by breaking ties arbitrarily in a consistent way for each type, to obtain an instance I' of the polynomially-solvable problem SMI, and then using Lemma 3 to argue that the cardinality of the largest stable matching in I' is the same as that in our original instance. This argument is based on a private communication with David Manlove.

Theorem 2. *When preferences over types are strict,* Typed Max SMTI *and* Consistently-Refined-Typed Max SMTI *are polynomial-time solvable. Furthermore, all stable matchings are of the same size.*

4 Exceptions in preference lists

We have argued for the existence of typed instances, where $k \ll n$, based on the premise that agents' preferences are formed based on a small collection of candidates' attributes. In practice, it seems likely that an agent might have access to additional information about some small subset of the candidates, either through personal acquaintance or some third-party connection; we say that an agent considers such candidates to be *exceptional*. This additional information may alter the agent's opinion of candidates relative to that derived from the

attributes alone, and so affect where these candidates are placed in his/her preference ordering. In this section we consider a generalisation of typed instances in which each agent may find some small collection of other agents to be exceptional and ranks them without regard to their types. Note that if only a small number of the agents in our instance consider one or more candidates to be exceptional, we can capture this information in a typed instance: each agent with exceptions in their preference list can be assigned their own type.

We say that an instance I of a stable matching problem is a *(c,Any)-exception-typed* instance, for a given constant c, if I is a typed instance in which each agent finds at most c number of the candidates exceptional and may rank them anywhere in his/her preference list. Two special cases are *(c,Top)-exception-typed* and *(c,Bottom)-exception-typed* instances where the exceptions are promoted to the top, or demoted to the bottom, of the preference lists, respectively. We refer to the standard problems with input of this form as (c,Any)-Exception Typed Max SMTI etc.

In this section we show that (1,Top)-Exception Typed Max SMTI belongs to FPT, but that (2,Any)-Exception Typed Max SMTI remains NP-hard even when there are only a constant number of types. The computational complexity of (1,Any)-Exception Typed Max SMTI and (2,Top)-Exception Typed Max SMTI remain open.

We begin with the case of (1,Top)-Exception Typed Max SMTI. For each agent a let $\mathrm{ex}(a)$ denote the exceptional candidate from a's point of view; $\mathrm{ex}(a) = \varnothing$ if a does not find any candidate exceptional. Formally, we say that I is a *(1,Top)-exception-typed* instance of SMTI if, given every two agents x and y of the same type:

1. x and y have identical preference lists when restricted to $N \setminus \{\mathrm{ex}(x), \mathrm{ex}(y)\}$, and
2. all other agents who do not find either x or y exceptional are indifferent between x and y.

Without loss of generality we can assume that there is no pair of agents who each consider the other to be exceptional in a (1,Top)-exception-typed instance of SMTI. If there are such pairs, they must be assigned to each other in any stable matching; so we can remove all such pairs to reduce to an instance that satisfies this assumption. A (1,Top)-exception-typed instance of SMTI is given as input by, in addition to the specifications needed for a typed instance (see Section 2.1), providing for each agent his or her exceptional candidate (if s/he has one).

Let I be a (1,Top)-exception-typed instance of SMTI, and let M be a matching in I. As in Section 2.2, we may assume without loss of generality that every agent is matched, by creating sufficiently many dummy agents of type $k+1$ which are inserted at the end of each man's and woman's (possibly incomplete) preference list. In order to obtain an analogue of the stability criterion given in Lemma 1 in this setting, we need some more notation.

Recall that we write $j \simeq_i \ell$ if agents of type i are indifferent between types j and ℓ. It is straightforward to see that $\simeq_i$ defines an equivalence relation on $[k]$

for each i. Given $j \in [k]$, we write $\text{class}_i(j)$ for the equivalence class under $\simeq_i$ which contains j. For each equivalence class J under $\simeq_i$, we say that the agent x of type i has *subtype* $i[J]$ if:

1. some agent y, with $\text{type}(y) \in J$, considers x exceptional, and
2. there is no agent z, such that $\text{type}(z) \succ_i j$ for $j \in J$, who considers x exceptional.

Thus $\text{subtype}(x) = i[J]$ if the most desirable agents who consider x exceptional have types from J. If an agent x of type i is not considered exceptional by any agent, we say that x has subtype $i[\{k+1\}]$. We also introduce a second dummy type 0, which is inserted at the head of each type's preference list and corresponds to exceptional candidates. We write $N_{i[J]}$ for the set of agents of subtype $i[J]$. Observe that the sets $N_{i[J]}$ can be computed in time $\mathcal{O}(n)$: for each agent x, $\text{subtype}(x)$ can be computed in time $\mathcal{O}(n)$ with suitable data structures.

We will need a variation on the function worst_M, which we call $\text{worst}^{\text{ex}}{}_M$. For any non-empty set $N_{i[J]}$, $\text{worst}^{\text{ex}}{}_M(i[J])$ is the type of the least desirable partner received by an agent of subtype $i[J]$ who is not matched with an agent they find exceptional; if the least desirable partners assigned to agents of subtype $i[J]$ belong to two or more different types between which agents of type i are indifferent, we define $\text{worst}^{\text{ex}}{}_M(i[J])$ to be the lexicographically first such type. If every agent of subtype $i[J]$ is matched with a partner they find exceptional, we set $\text{worst}^{\text{ex}}{}_M(i[J]) = 0$. Therefore $\text{worst}_M(i)$, as defined in Section 2.2, is the least desirable type out of $\{\text{worst}^{\text{ex}}{}_M(i[J]) : N_{i[J]} \neq \emptyset\}$.

We say that a matching M in an instance I of (1,Top)-exception-typed SMTI realises the function worst^{ex}, mapping nonempty subtypes $i[J]$ to values in $\{0, 1, \ldots, k+1\}$, if $\text{worst}^{\text{ex}}{}_M(i[J]) \succeq_i \text{worst}^{\text{ex}}(i[J])$ whenever $N_{i[J]} \neq \emptyset$. We can now characterise stability in a (1,Top)-exception-typed instance.

Lemma 4. *Let I be a (1,Top)-exception-typed instance of SMTI. Then a matching M in I is stable if and only if there is no pair $(i,j) \in [k]^{(2)}$ such that*

1. *$j \succ_i \text{worst}_M(i)$ and $i \succ_j \text{worst}_M(j)$, or*
2. *$i \succ_j \text{worst}^{\text{ex}}{}_M(j[\text{class}_j(i)])$.*

We will say that a function worst^{ex} is *I-exception-stable* for a (1,Top)-exception-typed instance I of SMTI if there is no pair $(i,j) \in [k]^{(2)}$ such that either $j \succ_i \text{worst}(i)$ and $i \succ_j \text{worst}(j)$ or $i \succ_j \text{worst}^{\text{ex}}(j[\text{class}_j(i)])$. Given any I-exception-stable function worst^{ex}, we write $\max(\text{worst}^{\text{ex}})$ for the maximum cardinality of any matching in I that realises worst^{ex}. We have an analogous result to Corollary 1 in this setting.

Lemma 5. *Let I be a (1,Top)-exception-typed instance of SMTI. Then the cardinality of the largest stable matching in I is equal to*

$$\max\{\max(\text{worst}^{\text{ex}}) : \text{worst}^{\text{ex}} \text{ is } I\text{-exception-stable}\}.$$

By solving a collection of instances of Maximum Matching in suitable undirected graphs, we are able to compute $\max(\text{worst}^{\text{ex}})$ (and generate a stable matching of this size) in time $\mathcal{O}(n^{5/2} \log n)$, for any I-exception-stable function worst^{ex}. Our FPT result now follows.

Theorem 3. (1,Top)-Exception Typed Max SMTI *can be solved in time* $\mathcal{O}\left(k^{k^2}(k^3 + n^{5/2}\log n)\right)$.

In contrast with this positive result, we show that only a small relaxation of the requirements on exceptions results in a problem that is NP-hard, even if the number of types is bounded by a constant. We show that, if we allow each agent to declare two candidates exceptional, and these two candidates can appear anywhere in the agent's preference list, then Max SMTI remains NP-hard under sever restrictions. In fact, we give a reduction from the NP-complete problem Clique to the special case Com SMTI, which involves deciding whether a given instance of SMTI admits a complete stable matching (i.e., a matching that matches all agents).

Theorem 4. (2,Any)-Exception Typed COM SMTI *is* NP*-complete, even if only men have exceptions in their preference lists, preferences over types are strict, and there are three types each of men and women.*

5 Discussion and Future Work

We believe that the same techniques used in this paper can be extended to prove analogous results in all three settings for the Hospitals-Residents problem (a many-one generalisation of SMTI) and the Stable Roommates problem (a non-bipartite generalisation of SMTI). For typed and consistently-refined-typed instances, a standard cloning argument gives the corresponding results for the Hospitals-Residents problem immediately.

We note that our FPT results can also be derived using Integer Linear Programming (ILP) techniques: for each function worst (or worst^{ex}), the corresponding optimisation problem can be encoded as an ILP instance. For the problems studied here, the ILP approach results in worse running times than the algorithms we have described, but this alternative approach might be helpful in tackling other stable matching problems involving types.

It would be interesting to investigate what further generalisations of our model yield FPT algorithms for NP-hard stable matching problems. In particular, the complexity of (1,Bottom)-Exception-Typed Max SMTI, (1,Any)-Exception-Typed Max SMTI, and (2,Top)-Exception-Typed Max SMTI remain open. Moreover, we could consider further restrictions with two or more exceptions, for example if an exceptional candidate can only be moved to the top or bottom of its type. Another intriguing question would be to understand how the complexity of MAX SMTI and other stable matching problems changes when agents on only one side of the market are associated with types.

Acknowledgements. The first author is supported by a Personal Research Fellowship from the Royal Society of Edinburgh (funded by the Scottish Government). Both authors are extremely grateful to David Manlove for his insightful comments on a preliminary version of this manuscript.

References

1. Aziz, H., de Keijzer, B.: Complexity of coalition structure generation. In: Proceedings of the 10th International Conference on Autonomous Agents and Multiagent Systems, AAMAS'11. pp. 191–198 (2011)
2. Bhatnagar, N., Greenberg, S., Randall, D.: Sampling stable marriages: why spouse-swapping won't work. In: Proceedings of the 19th ACM/SIAM Symposium on Discrete Algorithms, SODA '08. pp. 1223–1232. ACM-SIAM (2008)
3. Biró, P., Manlove, D., Mittal, S.: Size versus stability in the marriage problem. Theoretical Computer Science **411**, 1828–1841 (2010)
4. Chebolu, P., Goldberg, L.A., Martin, R.: The complexity of approximately counting stable matchings. Theoretical Computer Science **437**, 35–68 (2012)
5. Chebolu, P., Goldberg, L.A., Martin, R.: The complexity of approximately counting stable roommate assignments. Journal of Computer and System Sciences **78**(5), 1579–1605 (2012)
6. Choo, E., Siow, A.: Who marries whom and why. Journal of Political Economy **114**(1), 175–201 (2006)
7. Cygan, M., Fomin, F., Kowalik, Ł., Lokshtanov, D., Marx, D., Pilipczuk, M., Pilipczuk, M., Saurabh, S.: Parameterized Algorithms. Springer International Publishing (2015)
8. Downey, R.G., Fellows, M.R.: Fundamentals of Parameterized Complexity. Springer London (2013)
9. Echenique, F., Lee, S., Shum, M., Yenmez, M.B.: The revealed preference theory of stable and extremal stable matchings. Econometrica **81**(1), 153–171 (2013)
10. Flum, J., Grohe, M.: Parameterized Complexity Theory. Springer (2006)
11. Gale, D., Shapley, L.: College admissions and the stability of marriage. American Mathematical Monthly **69**, 9–15 (1962)
12. Gupta, S., Roy, S., Saurabh, S., Zehavi, M.: Balanced stable marriage: How close is close enough. Tech. Rep. 1707.09545, CoRR, Cornell University Library (2017)
13. Gupta, S., Saurabh, S., Zehavi, M.: On treewidth and stable marriage. Tech. Rep. 1707.05404, CoRR, Cornell University Library (2017)
14. Irving, R., Manlove, D., Scott, S.: The stable marriage problem with master preference lists. Discrete Applied Mathematics **156**(15), 2959–2977 (2008)
15. Manlove, D.: Algorithmics of Matching Under Preferences. World Scientific (2013)
16. Manlove, D., Irving, R., Iwama, K., Miyazaki, S., Morita, Y.: Hard variants of stable marriage. Theoretical Computer Science **276**(1-2), 261–279 (2002)
17. Marx, D., Schlotter, I.: Parameterized complexity and local search approaches for the stable marriage problem with ties. Algorithmica **58**(1), 170–187 (2010)
18. Meeks, K., Rastegari, B.: Solving hard stable matching problems involving groups of similar agents. Tech. Rep. 1708.04109, CoRR, Cornell University Library (2018)
19. Mnich, M., Schlotter, I.: Stable marriage with covering constraints-a complete computational trichotomy. In: Proceedings of the 10th International Symposium on Algorithmic Game Theory, SAGT'17. pp. 320–332 (2017)
20. O'Malley, G.: Algorithmic Aspects of Stable Matching Problems. Ph.D. thesis, University of Glasgow, Department of Computing Science (2007)
21. Orlin, J.B.: Max flows in $\mathcal{O}(nm)$ time, or better. In: Proceedings of the 45th Annual ACM Symposium on Theory of Computing, STOC '13. pp. 765–774. ACM (2013)
22. Shrot, T., Aumann, Y., Kraus, S.: On agent types in coalition formation problems. In: Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems, AAMAS'10. pp. 757–764 (2010)