



University
of Glasgow

Meeks, K. and Rastegari, B. (2020) Solving hard stable matching problems involving groups of similar agents. *Theoretical Computer Science*, 844, pp. 171-194.

(doi: [10.1016/j.tcs.2020.08.017](https://doi.org/10.1016/j.tcs.2020.08.017))

This is the Author Accepted Manuscript.

There may be differences between this version and the published version. You are advised to consult the publisher's version if you wish to cite from it.

<https://eprints.gla.ac.uk/222769/>

Deposited on: 1 September 2020

Enlighten – Research publications by members of the University of Glasgow
<http://eprints.gla.ac.uk>

Solving Hard Stable Matching Problems Involving Groups of Similar Agents *

Kitty Meeks

School of Computing Science, University of Glasgow, Glasgow, UK

Baharak Rastegari*

Department of Electronics and Computer Science, University of Southampton, Southampton, 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), several well-studied NP-hard stable matching problems including MAX SMTI (that of finding the maximum cardinality stable matching in an instance of stable marriage with ties and incomplete lists) belong to the parameterised complexity class FPT when parameterised by the number of different types of agents needed to describe the instance. For MAX SMTI 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, even if the number of types is bounded by a constant.

Keywords: Stable matchings, Stable marriage problem, Stable roommate problem, NP-hard problems, Parameterized complexity, Fixed-parameter tractable algorithms

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.

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

*A preliminary version of this paper which includes some of the results (without proofs) has been published in the proceedings of the 14th Conference on Web and Internet Economics (WINE'18). WINE'18 paper contains only the results on MAX SMTI.

*Corresponding author

Email addresses: kitty.meeks@glasgow.ac.uk (Kitty Meeks), b.rastegari@soton.ac.uk (Baharak Rastegari)

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 a 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 [1, 2]) regarding stable marriage market and stable roommates 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 (discussed in Section 4) 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 (discussed in Section 5), 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 (discussed in Section 6), 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.

We show that we can solve some of the most important hard stable matching problems efficiently from the point of view of parameterised complexity when the number of types is taken as the parameter. We obtain our results by reducing a given hard stable matching problem to the problem of solving a number of instances of a tractable problem, where this number is a function of the number of types. Some of our results rely on the fixed-parameter tractability of Integer Linear Programming (ILP). We also demonstrate that, by imposing further restrictions, some of these hard stable matching problems become polynomial-time solvable.

We would like to remark that we do not perceive our work as a fully realistic model but an initial exploration of the potential to exploit parameters that describe the level of similarity between agent's preference lists. We believe that our work is a first step towards what might be a good model of reality. The fact that we have obtained several positive results for this simple model motivates looking at more complicated extensions that are hopefully more realistic for most applications.

2. Preliminaries

In this section we introduce the main concepts we use in the paper; we begin with some definitions, then provide a brief introduction to parameterised complexity, before describing existing results on Integer Programming. We conclude this section by providing an overview of related work.

2.1. Definitions

In this section we provide the key definitions for the stable matching settings we study; for further background and terminology we refer the reader to [3].

Perhaps the most widely studied matching problem is the *Stable Marriage problem (SM)*, which is defined as follows. 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). *Stable Marriage with Incomplete lists (SMI)*, *Stable Marriage with Ties (SMT)* and *Stable Marriage with Ties and Incomplete lists (SMTI)* are generalisations of SM where agents are permitted to declare some candidates unacceptable, are allowed to express indifference between two or more candidates, or both, respectively. The *Stable Roommates problem (SR)* is a non-bipartite generalisation of SM. Extensions allowing for incomplete lists and indifference in preference lists are defined the same way as for SM. *Hospitals/Residents problem (HR)* is a famous extension of SMI that models many practical applications, including the assignment of junior doctors to hospitals, by allowing agents on one side of the market to be assigned to multiple agents on the other side of the market. *Hospitals/Residents problem with Ties (HRT)* is an extension of HR that allows for indifference in preference lists (note that in the standard terminology, both HR and HRT allow for incomplete lists; see [3]).

Let N denote a set of n agents, which in a bipartite matching setting (i.e. SMTI or HRT) is composed of two disjoint sets. Each hospital h in an instance of HRT is associated with a capacity $q(h)$ that denotes the number of posts it offers. When in a bipartite matching setting, we use the term *candidates* to refer to the agents on the opposite side of the market to that of an agent under consideration. In non-bipartite settings, candidates refer to all the other agents except the one under consideration.

Each agent finds a subset of candidates acceptable and ranks them in order of preference. Preference orderings need not to be strict, so it is possible for an agent to be indifferent between two or more candidates. We write $b \succ_a c$, or equivalently $c \prec_a b$, 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, and say that a *weakly prefers* b to c .

Definition 1 (Matching). *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. In an SRTI instance, a matching is a pairing of agents such that each agent is matched with at most one other agent whom s/he additionally finds acceptable. In an instance of HRT, a matching is a pairing of hospitals and residents such that no agent is paired with an unacceptable candidate, each resident is matched with at most one hospital, and each hospital h is matched with at most $q(h)$ residents.*

We write $(a, b) \in M$ to say that a and b are matched in M . We use $M(a)$ to denote the agent (or the set of agents in the case of hospitals in HRT) matched to a in M . We write $M(a) = \emptyset$ if agent a is unmatched in M . We assume that every agent prefers being matched to an acceptable candidate to remaining unmatched.

Definition 2 (Blocking pair). *Given an instance of SMTI or SRTI and a matching M , a pair of agents $(a, b) \notin M$ constitute a blocking pair if a prefers b to his current partner in M , i.e., $b \succ_a M(a)$, and vice versa. Given an instance of HRT and a matching M , an acceptable (resident, hospital) pair (r, h) is a blocking pair if (i) r prefers h to $M(r)$, and (ii) either $|M(h)| < q(h)$ or h prefers r to its worst assigned resident in M .*

Definition 3 (Stable matching). *A matching M is a stable matching if it admits no blocking pair.*

In their seminal work, Gale and Shapley [4] 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 instances of SMTI and mechanisms very similar to GS have been used to compute a stable assignment of residents to hospitals in instances of HRT. An instance of SR need not admit a stable matching [4]. Irving [5] provided a polynomial time algorithm that finds a stable matching in an instance of SR, or reports that none exists. Gusfield and Irving [6] showed that it is straightforward to generalise this algorithm to instances of SRI.

2.2. Hard Stable Matching Problems

As explained above, the problem of identifying a stable matching, or showing that none exists, can be solved efficiently for instances of SMTI, HRT and SRI. In contrast to the case for stable marriage, allowing indifference in the stable roommates problem makes the quest for a stable matching a difficult task. Ronn [7] showed that the following problem is NP-complete.

WEAK SRT*Input:* An instance I of SRT.*Output:* A stable matching in I if any exists.

The importance of stable solutions has been argued and stressed by economists (see e.g.[8, 9, 10]) and finding a stable matching is at the core of many practical applications such as the assignment of residents to hospitals in the National Resident Matching Program (NRMP) in the United States. In many practical applications, however, it is also important to match as many agents as possible, and thus finding a maximum cardinality stable matching (i.e., a stable matching with the largest size amongst all stable matchings) is a crucial issue.

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; this gives rise to a very natural problem.

MAX SMTI*Input:* An instance I of SMTI.*Output:* A maximum cardinality stable matching in I .

This problem is known to be NP-hard [11, 12, 13, 14, 15], even when the input is heavily restricted.

The concern of computing a maximum size stable matching extends to instances of HRT. As SMTI can be seen as a special case of HRT in which every hospital has capacity exactly one, the NP-hardness of MAX SMTI implies that **MAX HRT**, the problem of determining the maximum cardinality stable matching in an instance of HRT, is also NP-hard. Conversely, we can express any instance of MAX HRT as an instance of MAX SMTI using a standard cloning argument [3, 16]: for each hospital h in the HRT instance, we create $q(h)$ identical agents in the SMTI instance, each with capacity one.

NP-hardness of WEAK SRT implies that **MAX SRT**, the problem of identifying a maximum cardinality stable matching in an instance of SRT or reporting that none exists, is also NP-hard. Note that this problem is hard even when all agents find all other agents acceptable.

Depending on the application, one might be willing to tolerate a small degree of instability if that leads to larger matchings. Two different measurements for the degree of instability have been introduced in the literature: the number of blocking pairs and the number of blocking agents (i.e. agents who belong to at least one blocking pair). This gives rise to two further optimisation problems:

MAX SIZE MIN BP SMI*Input:* An instance I of SMI.*Output:* A matching of maximum cardinality in I which has the minimum number of blocking pairs out of all such matchings.**MAX SIZE MIN BA SMI***Input:* An instance I of SMI.*Output:* A matching of maximum cardinality in I which has the minimum number of blocking agents out of all such matchings.

Both these problems are NP-hard and hard to approximate [11]. Note that these two problems are hard even when there are no ties in preference lists.

Since an instance of SR may not admit a stable matching, it is of interest to find a matching with minimum number of blocking pairs or blocking agents. Abraham et al. [17] showed that the following problem is NP-hard and hard to approximate.

MIN BP SR*Input:* An instance I of SR.*Output:* A matching in I with the minimum number of blocking pairs.

Note that this problem remains hard even if all agents rank all the other agents in strict order of preference. Very recently, Chen et al. [18] showed that the following related problem is also NP-hard.

MIN BA SRI*Input:* An instance I of SRI.*Output:* A matching in I with the minimum number of blocking agents.

2.3. Parameterised Complexity

In this paper 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. It should be emphasised that, for a problem to be in FPT, the exponent of the polynomial must be independent of the parameter value; problems which satisfy the weaker condition that the running time is polynomial for any constant value of the parameter(s) (so that the degree of the polynomial may depend on the parameters) are said to belong to the class XP.

For further background on the theory of parameterised complexity, we refer the reader to [19, 20].

2.4. The complexity of Integer Programming

Many of the algorithms we present in this paper make use of an algorithm for INTEGER LINEAR PROGRAMMING in some way. This problem is formally stated as follows: given an $m \times k$ matrix A and two m -dimensional vectors \mathbf{b} and \mathbf{c} (all with coefficients in \mathbb{Z}), find a m -dimensional vector $\mathbf{x} \in \mathbb{Z}^k$ which minimises the scalar product $\mathbf{c}^T \cdot \mathbf{x}$, subject to the m linear constraints given by $A\mathbf{x} \leq \mathbf{b}$, or else report that no vector satisfying the constraints exists. Note that we can easily translate problems in which we wish to maximise rather than minimise the objective function into this form, and also we can express constraints based on linear equalities as a combination of linear inequalities; for simplicity of presentation we will use both of these generalisations when expressing problems as instances of INTEGER LINEAR PROGRAMMING.

While INTEGER LINEAR PROGRAMMING is NP-hard in general, one of the most celebrated results in parameterised complexity is that this problem belongs to FPT when parameterised by the number of variables.

Theorem 1 (Theorem 6.5 on page 136 in [21], based on [22, 23, 24]). *An INTEGER LINEAR PROGRAMMING instance of size L with k variables can be solved using*

$$\mathcal{O}(k^{2.5k+o(k)} \cdot (L + \log M_x) \log(M_x M_c))$$

arithmetic operations and space polynomial in $L + \log M_x$, where M_x is an upper bound on the absolute value a variable can take in a solution, and M_c is the largest absolute value of a coefficient in the vector \mathbf{c} .

In Section 4.4 we also need to solve instances of INTEGER QUADRATIC PROGRAMMING, a variant of INTEGER LINEAR PROGRAMMING in which the objective function is quadratic. Formally, given a $k \times k$ integer matrix Q , an $m \times k$ integer matrix A and an m -dimensional integer vector \mathbf{b} , our goal is to find a vector $\mathbf{x} \in \mathbb{Z}^k$ which minimises $\mathbf{x}^T Q \mathbf{x}$, subject to the m linear constraints $A\mathbf{x} \leq \mathbf{b}$, or else report that no vector satisfying the constraints exists. As before, we note that we can easily generalise this definition to deal with maximisation problems and constraints in the form of linear equalities. Lokshantov recently gave an FPT algorithm for this problem.

Theorem 2 ([25]). *INTEGER QUADRATIC PROGRAMMING is in FPT parameterised by $k + \alpha$, where α is the maximum absolute value of any entry in the matrices A and Q .*

2.5. Related Work

NP-hard matching problems.. 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 [14], (2) even if each man’s 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 [13], and (3) even if the SMTI instance has symmetric preferences; that is, for any acceptable (man, woman) pair (m_i, w_j) , $rank(m_i, w_j) = rank(w_j, m_i)$ [15], where $rank(a, b)$ is defined to be one plus the number of candidates that a prefers to b . As SMTI is a special case of HRT, the NP-hardness of finding a maximum stable matching in the latter follows directly from the NP-hardness of this problem in the former.

The NP-hardness of WEAK SRT, and hence MAX SRT, holds even if each preference list is either strictly ordered or contains a tie of length 2 at the head.

MAX SIZE MIN BP SMI and MAX SIZE MIN BA SMTI are NP-hard and hard to approximate [26] even if each agent’s preference list is of length at most 3 [11, 27], but polynomial-time solvable if agents on one side of the market have preference lists of length at most 2 [11].

Parameterised complexity of matching problems.. There are a limited number of works addressing fixed-parameter tractability in stable matching problems. Marx and Schlotter [28] gave the first parameterised complexity results on MAX SMTI. They show 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. Abraham et al. [17] showed that MIN BP SR is in XP when parameterised by the number of blocking pairs. Very recently, five different works have studied hard stable matching problems from the perspective of parameterised complexity. In [29], the authors 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. In [30] it is shown 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. In [31], the authors study above guarantee parameterisations of the problem of finding a stable matching that balances the dissatisfaction of men and women, with parameters that capture the degree of dissatisfaction. Chen et al. [18] study the parameterised complexity of several NP-hard problem in the stable roommate setting, including MIN BP SRI and MIN BA SRI. They show that MIN BP SRI (respectively MIN BA SRI) is W[1]-hard when parameterised by the number of blocking pairs (respectively by the number of blocking agents), even if the length of each preference list is at most five. Adil et al. [32] study the parameterised complexity of MAX SMTI and MAX SRTI, as well as the NP-hard problem of finding a stable matching of minimum cardinality given an instance of SMTI: MIN SMTI. They show that both MAX SMTI and MIN SMTI are fixed-parameter tractable with respect to the solution size, and that MAX SRTI is fixed-parameter tractable with respect to the size of the maximum cardinality matching.

Attributes and types.. 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 (see, e.g., [1, 2, 33]). In [34], the authors study the problem of characterising matchings that are rationalisable as stable matchings when agents’ preferences are unobserved. They focus 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 remark that empirical studies on marriage typically make such an assumption [35]. Bounded agent types have been considered in [36, 37] 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.

3. Summary of results

We study the NP-hard problems defined in Section 2.2 in several settings where agents are associated with types.

1. In the simplest model where agents of the same type are completely indistinguishable (*typed* instances, defined in Section 4), we show that the following problems are in FPT parameterised by the number k of different types: MAX SRTI (Corollary 7), MAX SMTI (Corollary 10), MAX HRT (Corollary 11), MAX SIZE MIN BP SMTI (Theorem 14), MAX SIZE MIN BA SMTI (Theorem 16), MAX SIZE MIN BA SRTI (Theorem 17), and MIN BA SRTI (Corollary 18).

2. In our first generalisation of the basic model where agents are allowed to *consistently* refine their preferences over agents of the same type (*consistently-refined-typed* instances, defined in Section 5), we show that the following problems are in FPT parameterised by the number k of different types: MAX SRTI (Theorem 20), MAX SMTI and MAX HRT (Corollary 21), MAX SIZE MIN BP SMTI and MAX SIZE MIN BA SMTI (Corollary 24), MAX SIZE MIN BA SRTI and MIN BP SRTI and MIN BA SRTI (Corollary 25).
3. In our second generalisation of the basic model where we allow each agent to consider small number of other agents *exceptional*, who can appear anywhere in his or her preference list, regardless of their type (defined in Section 6), we obtain the following results. We show that when each agent finds exactly one other agent exceptional and ranks that agent at the top of the preference list, then MAX SMTI is in FPT parameterised by the number k of different types (Corollary 30). If each agent finds at least two other agent exceptional and can rank them anywhere in his or her preference list, then the following problems are NP-hard even when $k = 6$ (hence they are para-NP-hard): MAX SMTI (Corollary 32), MAX HRT and MAX SRTI and MAX SIZE MIN BP SMTI and MAX SIZE MIN BA SMTI and MAX SIZE MIN BP SRTI and MAX SIZE MIN BA SRTI (Corollary 33).

Moreover, we show that MAX SRTI, and hence MAX SMTI and MAX HRT, are polynomial-time solvable given a typed or consistently-refined-typed instance, if agents' preferences over types are strict (Theorem 22 and Corollary 23).

4. 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. This implies that 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.

We then identify a necessary and sufficient condition, in terms of the types of the least and the second least desirable partners 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 SRTI by solving $k^{\mathcal{O}(k)}$ instances of INTEGER LINEAR PROGRAMMING. This implies that MAX SRTI and hence MAX SMTI, parameterised by k , belong to FPT. We then show that we can solve MAX SMTI more efficiently 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 alternative approach reduces the time complexity of MAX SMTI from $k^{\mathcal{O}(k^2)} \log^3 n + \mathcal{O}(n)$ to $k^{\mathcal{O}(k)} \cdot \log^2 n + \mathcal{O}(n)$. The $\mathcal{O}(n)$ part of the time complexities accounts for the time required to construct the maximum cardinality matching.

Lastly, we extend the methods for solving MAX SRTI to provide FPT algorithms for MAX SIZE MIN BP SMTI, MAX SIZE MIN BA SMTI, MAX SIZE MIN BP SRTI, MAX SIZE MIN BA SRTI, MIN BP SRTI, and MIN BA SRTI.

4.1. Definition of typed instances

Assume that there are k types available for agents. Let $[k]$ denote the set $\{1, 2, \dots, k\}$. Given a set X , let X^c and $X^{(c)}$ denote the sets of ordered and unordered c -element subsets of X respectively. 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 $j \succ_i \ell$, or equivalently $\ell \prec_i j$, 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 when restricted to $N \setminus \{x, y\}$, 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

all the NP-hard problems we have discussed in Section 2.2 remain NP-hard when k is considered to be part of the input: we can always turn a standard instance to a typed one by assigning each agent its own type.

We can translate the definition of a blocking pair (Definition 2) into the setting of typed instances.

Proposition 3. *Let x and y be agents of type i and type j respectively, and suppose that $M(x)$ is of type j' and $M(y)$ is of type i' . Then (x, y) is a blocking pair in M if and only if $j \succ_i j'$ and $i \succ_j i'$.*

A typed instance I of SRTI 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. 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 in a stable marriage setting and that 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: $(w_1 w_2 w_3 w_4) (w_5 w_6 w_7)$.*

4.2. An FPT algorithm for TYPED MAX SRTI

Let I be a typed instance of SRTI, 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 agent's (possibly incomplete) preference list; we therefore have that $j \succ_i k+1$ if and only if type i considers type j to be acceptable. For each pair of types $(i, j) \in [k+1]^2$, we write $n_{i,j}^M$ for the number of pairs in M consisting of one agent of type i and another of type j . We define $\text{worst}_M(i)$ and $\text{second_worst}_M(i)$ to be the types of the least desirable agent and the second least desirable agent with which any agent of type i is matched in M , respectively, breaking ties arbitrarily (e.g. lexicographically). Note that $\text{worst}_M(i)$ would be the dummy type if an agent of type i is unmatched (i.e. matched to a dummy agent) in M . If there is only one agent of type i , then $\text{second_worst}_M(i)$ is undefined, in which case we let $\text{second_worst}_M(i) = \emptyset$. Note that it is possible to have $\text{second_worst}_M(i) = \text{worst}_M(i)$. 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 $\text{worst}_M(i)$ and $\text{second_worst}_M(i)$ for each $i \in [k]$.

Lemma 4. *Let I be a typed instance of SRTI. Then a matching M in I is stable if and only if (1) there is no pair of types $(i, j) \in [k]^2$, $i \neq j$, such that $j \succ_i \text{worst}_M(i)$ and $i \succ_j \text{worst}_M(j)$, and (2) there is no pair of types (i, i) , $i \in [k]$, such that there are at least two agents of type i and $i \succ_i \text{second_worst}_M(i)$.*

Proof. Suppose first that M is not stable. In this case, by definition, there exists some pair of agents (a, b) such that a and b are not matched together but each prefers the other over their current partner. First suppose that a and b are of different types and without loss of generality assume that a is of type i and b is of type j , $i \neq j$. Then we know that $j \succ_i \text{type}(M(a)) \succeq_i \text{worst}_M(i)$, and similarly $i \succ_j \text{type}(M(b)) \succeq_j \text{worst}_M(j)$. Now suppose that a and b are both of the same type i . Then we know that type i likes type $\text{type}(M(a))$ or type $\text{type}(M(b))$ at least as well as type $\text{second_worst}_M(i)$. Without loss of generality assume that $\text{type}(M(a)) \succeq_i \text{second_worst}_M(i)$. Since (a, b) is a blocking pair, $i = \text{type}(b) \succ_i \text{type}(M(a))$, and therefore $i \succ_i \text{second_worst}_M(i)$.

Conversely, suppose that M is stable. Suppose for a contradiction that at least one of the two conditions in the statement of the theorem does not hold. First suppose that the first condition does not hold, i.e. there is a pair of types (i, j) , $i \neq j$, such that $j \succ_i \text{worst}_M(i)$ and $i \succ_j \text{worst}_M(j)$. Then there is some agent a of type i which is matched with an agent of type $\text{worst}_M(i)$, so in particular a is matched with a partner less desirable than any agent of type j . Similarly, there is some agent b of type j which is matched with an agent of type $\text{worst}_M(j)$ and hence is matched with a partner less desirable than any agent of type i . Thus a and b both prefer each other to their current partner, and so form a blocking pair. This contradicts the assumption that M is stable. Now suppose that the second condition does not hold, i.e. there exists a type i corresponding to at least two agents where $i \succ_i \text{second_worst}_M(i)$. Then there are two agents a and b of type i who are matched to an agent of type $\text{worst}_M(i)$ and an agent of type $\text{second_worst}_M(i)$, respectively. Thus a and b both prefer each other to their current partner, and so form a blocking pair. This contradicts the assumption that M is stable. \square

Definition 4 (Realising). We say that a matching M in an instance of SRTI realises given functions $\text{worst} : [k] \rightarrow [k+1]$ and $\text{second_worst} : [k] \rightarrow [k+1] \cup \emptyset$ if, for each $i \in [k]$, the least desirable and the second least desirable partners any agent of type i has in M are of types no worse than $\text{worst}(i)$ and $\text{second_worst}(i)$ respectively.

We say that a pair of functions $(\text{worst}, \text{second_worst})$ is feasible if for each type i , (1) $\text{worst}(i)$ is either a type acceptable to type i or the dummy type, (2) $\text{second_worst}(i)$ is either a type acceptable to type i or the dummy type, or \emptyset (only if there is only one agent of type i), and (3) if $\text{worst}(i) = \text{second_worst}(i)$ then there exist at least two agents of type $\text{worst}(i)$.

Definition 5 (I-stability). We say that a pair of functions $(\text{worst}, \text{second_worst})$ is I -stable for an instance I of SRTI if it is feasible, there is no pair $(i, j) \in [k]^2$ such that $j \succ_i \text{worst}(i)$ and $i \succ_j \text{worst}(j)$, and there is no pair (i, i) , $i \in [k]$, such that there are at least two agents of type i and $i \succ_i \text{second_worst}(i)$.

Given a pair of I -stable functions $(\text{worst}, \text{second_worst})$, we write $\max(\text{worst}, \text{second_worst})$ for the maximum cardinality of any matching in I that realises worst and second_worst . Using Lemma 4, it is straightforward to check that, given a typed instance I of SRTI, the cardinality of a solution to MAX SRTI can be found by taking the largest value of $\max(\text{worst}, \text{second_worst})$ over all pairs of I -stable functions $(\text{worst}, \text{second_worst})$.

Corollary 5. Let I be a typed instance of SRTI. Then the cardinality of the largest stable matching in I is equal to

$$\max\{\max(\text{worst}, \text{second_worst}) : (\text{worst}, \text{second_worst}) \text{ is } I\text{-stable}\}.$$

We next show that, given a pair of I -stable functions $(\text{worst}, \text{second_worst})$, there is an FPT algorithm (with parameter k) to compute $\max(\text{worst}, \text{second_worst})$.

Lemma 6. Let I be a typed instance of SRTI, and fix a pair of I -stable function $(\text{worst}, \text{second_worst})$. We can compute $\max(\text{worst}, \text{second_worst})$ in time $k^{\mathcal{O}(k^2)} \log^3 n$.

Proof. Suppose that in total there are n agents. We compute $\max(\text{worst}, \text{second_worst})$ by solving a suitable instance of an Integer Linear Programming. For each pair of values $(i, j) \in [k+1]^2$, we have a variable $n_{i,j}$ which represents the number of pairs we will include in our matching which consist of one agent of type i and another of type j . Recall that N_i is the set of agents of type i . We then have the following integer linear program:

$$\begin{aligned} & \text{maximize} && \sum_{1 \leq i \leq j \leq k} n_{i,j} \\ & \text{subject to} && \sum_{j \in [k+1]} n_{i,j} = |N_i|, && \forall i \in [k] \\ & && n_{i,j} \geq 0, && \forall i, j \in [k+1] \\ & && \sum_{j \succeq_i \text{worst}(i)} n_{i,j} = |N_i|, && \forall i \in [k] \\ & \text{and} && \sum_{j \succeq_i \text{second_worst}(i)} n_{i,j} \geq |N_i| - 1, && \forall i \in [k]. \end{aligned}$$

The first two constraints ensure that every agent is involved in exactly one pair, perhaps with a dummy agent. The third constraint ensures that no agent of type i is assigned a partner of type worse than $\text{worst}(i)$, and the last constraint ensures that no more than one agent of type i is assigned a partner of type worse than $\text{second_worst}(i)$. The objective function seeks to maximise the total number of pairs that do not involve dummy agents.

The above integer linear program has $3k + (k+1)^2$ constraints and $(k+1)^2$ variables. The upper bound on the absolute value a variable can take is n . Therefore, by Theorem 1, this maximisation problem for any pair of candidate functions $(\text{worst}, \text{second_worst})$ can be solved in time $k^{\mathcal{O}(k^2)} \log^3 n$. \square

It then follows that TYPED MAX SRTI is in FPT parameterised by the number k of different types in the instance. A high-level description of our algorithm for computing the size of a maximum cardinality stable matching in an instance of TYPED MAX SRTI is presented as Algorithm 1.

ALGORITHM 1: Algorithm for TYPED MAX SRTI

Input: Typed instance I of SRTI with n agents and k types

Output: Size of a maximum cardinality matching in I

$max_size := 0$

```
for each function worst :  $[k] \rightarrow [k + 1]$  do
  for each function second_worst :  $[k] \rightarrow [k + 2]$  do
    if (worst, second_worst) is  $I$ -stable then
      if  $\max(\text{worst}, \text{second\_worst}) > max\_size$  then
        |  $max\_size := \max(\text{worst}, \text{second\_worst})$ 
      end
    end
  end
end
end
return  $max\_size$ 
```

Corollary 7. TYPED MAX SRTI can be solved in time $k^{\mathcal{O}(k^2)} \cdot \log^3 n + \mathcal{O}(n)$. 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 $k^{\mathcal{O}(k^2)} \cdot \log^3 n$.

Proof. Let I be a typed instance of SRTI. We consider each possible pair of functions $worst : [k] \rightarrow [k + 1]$ and $second_worst : [k] \rightarrow [k + 2]$ in turn; there are at most $(k + 1)^k (k + 2)^k$ such pairs of functions. We can determine in time $\mathcal{O}(k^2)$ whether $(worst, second_worst)$ is I -stable. If none of the candidate pairs of functions is I -stable, we conclude that there is no stable matching and terminate. For each I -stable pair of functions $(worst, second_worst)$, we compute $\max(worst, second_worst)$ in time $k^{\mathcal{O}(k^2)} \log^3 n$, by Lemma 6. We then take the maximum value of $\max(worst, second_worst)$ over all I -stable pairs of functions $(worst, second_worst)$ which, by Corollary 5, is equal to the cardinality of the largest stable matching in I . Given the number of pairs of agents of types i and j which are to be matched together for all $1 \leq i \leq j \leq k + 1$ (as calculated by the ILP), and using suitable data structures, a maximum cardinality matching can easily be constructed in time $\mathcal{O}(n)$. \square

4.3. An FPT algorithm for TYPED MAX SMTI

SMTI is a bipartite restriction of SRTI and therefore, by Corollary 7, we see that TYPED MAX SMTI and TYPED MAX HRT belong to FPT when parameterised by the number of types in the instance. In this section we show that the running time can be improved in the bipartite setting of SMTI and HRT.

In this setting, we no longer need to consider the function $second_worst$ in order to determine stability or otherwise of the matching: it is enough to consider only the type of the least desirable partner assigned to any agent of each type. The following characterisation of stability in terms of the function $worst$ can easily be deduced from the proof of Lemma 4.

Lemma 8. Let I be a typed instance of SMTI. Then a matching M in I is stable if and only if there is no pair $(i, j) \in [k]^2$, $i \neq j$, such that $j \succ_i worst_M(i)$ and $i \succ_j worst_M(j)$.

We also extend the terminology of the previous section to describe a function $worst$ as I -stable if there is no pair $(i, j) \in [k]^2$, $i \neq j$, such that $j \succ_i worst(i)$ and $i \succ_j worst(j)$. Similarly, we say that a matching M realises the function $worst$ if the least desirable partner of any agent of type i is no worse than $worst(i)$, and write $\max(worst)$ for the cardinality of the largest matching which realises $worst$. With this terminology, it is clear that (as in Corollary 5) the cardinality of the largest stable matching in a typed instance I of SMTI is equal to $\max\{\max(worst) : worst \text{ is } I\text{-stable}\}$.

These observations alone would allow us to simplify the ILP formulation by omitting the final constraint, and would reduce the number of instances of ILP that must be solved to $\mathcal{O}(k^k)$. However, we can make further improvements by using a network flow method to compute the value of $\max(worst)$ for each relevant function $worst$ in polynomial time. A high-level description of this algorithm is presented as Algorithm 2.

Lemma 9. Let I be a typed instance of SMTI, and fix an I -stable function $worst$. We can compute $\max(worst)$ in time $\mathcal{O}(k^3 \log^2 n)$.

ALGORITHM 2: Algorithm for computing $\max(\text{worst})$ in an instance of SMTI

Input: Typed instance I of SMTI with n_1 women and n_2 men, and an I -stable function worst **Output:** $\max(\text{worst})$ $min := 0$ $max := \min\{n_1, n_2\}$ **while** $max > min$ **do** $c := \lceil \frac{max+min}{2} \rceil$ Construct directed network D from I **if** $\text{MaxFlow}(D) = n_1 + n_2 - c$ **then** $min := c$ **end** **else** $max := c - 1$ **end****end****return** min

Proof. 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 $c \in \{1, \dots, \min\{n_1, n_2\}\}$. We will 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 $\mathcal{O}(k^2 \log n)$. In particular, we show that the maximum flow in D is equal to $n_1 + n_2 - c$ if and only if $\max(\text{worst}) \geq c$. MAX FLOW can be solved on D in time $\mathcal{O}(k^3 \log n)$, using an algorithm due to Orlin [38], 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 $\mathcal{O}(k^3 \log^2 n)$.

We now show how to construct the network D . Assume, without loss of generality, that types $1, \dots, k_1$ are types of women and types $k_1 + 1, \dots, k$ are types of man. We construct a directed network $D = (V, E)$ with vertex set $V = \{s, t, v_1, \dots, v_k, d_w, d_m\}$, where vertex v_i corresponds to type i , and d_w and d_m both correspond to the dummy type. The edge set E consists of the following directed edges with associated capacities:

- $\overrightarrow{sv_i}$ for $1 \leq i \leq k_1$, with capacity $|N_i|$,
- $\overrightarrow{sd_w}$, with capacity $n_2 - c$,
- $\overrightarrow{v_i v_j}$ for all pairs (i, j) with $1 \leq i \leq k_1$ and $k_1 + 1 \leq j \leq k$ such that $j \succeq_i \text{worst}(i)$ and $i \succeq_j \text{worst}(j)$, with capacity $\min\{|N_i|, |N_j|\}$,
- $\overrightarrow{v_i d_m}$ for all $1 \leq i \leq k_1$ such that $\text{worst}(i) = k + 1$, with capacity $|N_i|$,
- $\overrightarrow{d_w v_j}$ for all $k_1 + 1 \leq j \leq k$ such that $\text{worst}(j) = k + 1$, with capacity $|N_j|$,
- $\overrightarrow{d_w d_m}$, with capacity $\min\{n_1, n_2\} - c$,
- $\overrightarrow{v_j t}$ for $k_1 + 1 \leq j \leq k$, with capacity $|N_j|$,
- $\overrightarrow{d_m t}$, with capacity $n_1 - c$.

By construction, there are $\mathcal{O}(k)$ vertices in the network and the maximum capacity of any edge is $\mathcal{O}(n)$. Therefore, it is clear that we can construct D from I in time $\mathcal{O}(k^2 \log n)$ where the $\log n$ factor allows for the time required to read integers of size $\mathcal{O}(n)$ in the input. So it only remains to show that we can decide whether $\max(\text{worst}) \geq c$ by solving MAX FLOW on D .

We claim that the maximum flow from s to t in D is equal to $n_1 + n_2 - c$ if and only if there is a matching of cardinality at least c which realises worst (i.e., if and only if $\max(\text{worst}) \geq c$). Note that the flow clearly cannot exceed this value, summing the capacities of the outgoing edges of s .

We first prove that if there exists a matching M of cardinality at least c that realises worst , then the maximum flow from s to t in D is equal to $n_1 + n_2 - c$. We define a flow f in D of value $n_1 + n_2 - c$ as follows:

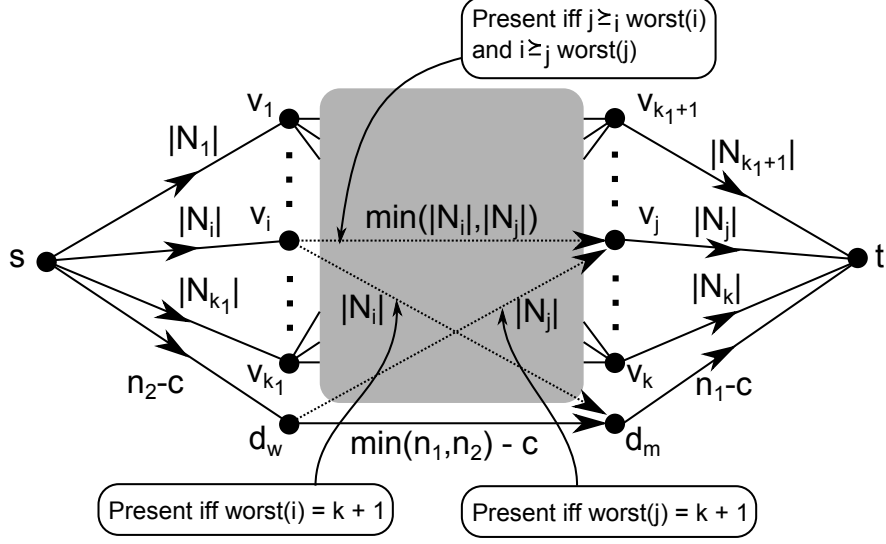


Figure 1: Network D , constructed from an instance of SMTI in the proof of Lemma 9. Types $1, \dots, k_1$ are types of women and types $k_1 + 1, \dots, k$ are types of man. Vertices d_w and d_m both correspond to the dummy type.

- for all edges incident with s or t , we assign a flow equal to the capacity;
- for each edge $\overrightarrow{v_i v_j}$, we assign a flow equal to the number of pairs in M consisting of one agent of type i and one agent of type j ;
- for each edge $\overrightarrow{v_i d_m}$, we assign a flow equal to the number of unassigned agents of type i in M ;
- for each edge $\overrightarrow{d_w v_j}$, we assign a flow equal to the number of unassigned agents of type j in M ; and
- to $\overrightarrow{d_w d_m}$, we assign a flow equal to $|M| - c$.

We need to show that f is feasible and that value of f is $n_1 + n_2 - c$. It is easy to verify that no edge is assigned a flow greater than its capacity, so to prove that f is feasible it suffices to demonstrate that the flow is conserved at each vertex other than s and t . For any vertex v_i with $1 \leq i \leq k_1$, the flow into v_i is equal to $|N_i|$, and the total flow out of v_i is the sum of the numbers of agents of type i matched with agents of type j (for all $j \geq k_1 + 1$) plus the number of agents of type i that are unmatched, which must be equal to $|N_i|$. A symmetric argument holds for each vertex v_j with $k_1 + 1 \leq j \leq k$. The flow into d_w is $n_2 - c$, and the total flow out of d_w is equal to the total number of men that are unmatched in M , plus $|M| - c$: since each pair in M contains exactly one distinct man, the number of unmatched men in M is equal to $n_2 - |M|$, so the total flow out of d_w is $n_2 - |M| + (|M| - c) = n_2 - c$. A symmetric argument for d_m completes our claim that we have defined a feasible flow. It remains to show that f does indeed have value $n_1 + n_2 - c$: note that every outgoing edge from s is saturated, and the sum of the capacities of these edges is equal to $n_1 + n_2 - c$.

Conversely, we prove that if the maximum flow from s to t in D is equal to $n_1 + n_2 - c$, then there exists a matching M of cardinality at least c that realises worst. We define a matching M_f in I by greedily including $f(\overrightarrow{v_i v_j})$ pairs consisting of one agent of type i and one agent of type j , for every edge $\overrightarrow{v_i v_j}$ in D . We need to show that M_f : (1) is a matching, (2) has cardinality at least c , and (3) realises worst. To check that M_f is a matching, we need to verify that we have not assigned more than $|N_i|$ agents of any type i . If i is a type of women, then the total number of pairs involving an agent of type i is the total flow out from v_i , which cannot be more than the capacity of the single incoming edge to v_i , which is equal to $|N_i|$. Similarly, if i is a type of men, then the total number of pairs involving an agent of type i is the total flow into v_i , which cannot be more than the capacity of the single outgoing edge from v_i , which is equal to $|N_i|$. The cardinality of M_f is equal to the sum of flows on edges $\overrightarrow{v_i v_j}$. As the set of such edges, together with all outgoing edges of d_w and all incoming edges of d_m , forms an s - t cut in D (in which all edges are directed the same way), we can conclude that the cardinality of M_f is equal to the value of f minus the sum of flows on outgoing edges of d_w and incoming edges of d_m . Note that the sum of flows on outgoing edges of d_w cannot be

more than the capacity of the single incoming edge of d_w , which is $n_2 - c$, and similarly the sum of flows on incoming edges of d_m cannot be more than $n_1 - c$. Thus we see that $|M_f|$ is at least $n_1 + n_2 - c - (n_2 - c) - (n_1 - c) = c$. It is clear by construction of D and M_f that M_f realises worst. This completes our proof. \square

The next corollary then follows along the same lines as Corollary 7.

Corollary 10. 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 $k^{\mathcal{O}(k)} \cdot \log^2 n$.

To solve TYPED MAX HRT, we need only to use a standard cloning argument [3, 16] and for each hospital h in the instance create $q(h)$ identical agents each with capacity one. Let $m =$ number of residents + sum of hospitals' capacities.

Corollary 11. TYPED MAX HRT can be solved in time $k^{\mathcal{O}(k)} \cdot \log^2 m + \mathcal{O}(m)$.

4.4. FPT algorithms for TYPED MAX SIZE MIN BP SMTI, TYPED MAX SIZE MIN BA SMTI and TYPED MAX SIZE MIN BA SRTI

As stated in Section 1, in some settings the size of the matching takes priority over the stability criterion. That is, the mechanism designers are willing to tolerate a small degree of instability if that leads to a matching of larger size. We can extend the methods from previous sections to deal with this situation in both the bipartite and non-bipartite setting; for simplicity of presentation we begin with the case of SMTI and then discuss how to extend the method to SRTI.

We begin by considering the problem of minimising the total number of blocking pairs. For the rest of this section, we assume that types 1 to k' are types of women, and types $k' + 1$ to k are types of men. We further assume that all preference lists are extended to include the dummy type as their least desirable acceptable type. If an agent a is unmatched in M , we say that $M(a)$ is of type $k + 1$, the dummy type.

Using Proposition 3, we can obtain an expression for the total number of blocking pairs in M that are comprised of an agent of type i and an agent of type j .

Lemma 12. The number of blocking pairs (x, y) in M such that x is a woman of type i and y is a man of type j is given by

$$\left(\sum_{i' \prec_j i} n_{i',j}^M \right) \left(\sum_{j' \prec_i j} n_{i,j'}^M \right) = \sum_{\substack{i' \in \{1, \dots, k', k+1\} \\ j' \in \{k'+1, \dots, k, k+1\}}} n_{i',j}^M n_{i,j'}^M (\mathbb{1}_{i' \prec_j i} \mathbb{1}_{j' \prec_i j}),$$

where $\mathbb{1}_{a \prec_b c}$ is an indicator function that returns one if $a \prec_b c$ and zero otherwise.

Proof. It is easy to verify that the left-hand side and the right-hand side of the equation are equal. For the remainder of the proof, we focus on the left-hand side of the equation. By Proposition 3, we know that the set of blocking pairs consisting of one agent of type i and another of type j is precisely

$$\{(x, y) : \text{type}(x) = i \text{ and } \text{type}(y) = j \text{ and } \text{type}(M(x)) \prec_i j \text{ and } \text{type}(M(y)) \prec_j i\}.$$

The cardinality of this set is thus equal to the number of agents of type i that are matched to an agent of type inferior to type j (i.e. $\sum_{j' \prec_i j} n_{i,j'}^M$) multiplied by the number of agents of type j that are matched to an agent of type inferior to type i (i.e. $\sum_{i' \prec_j i} n_{i',j}^M$). The result follows immediately. \square

Summing over all possibilities for i and j gives the following result.

Lemma 13. The total number of blocking pairs in M is given by

$$\sum_{\substack{1 \leq i \leq k' \\ k'+1 \leq j \leq k}} \sum_{\substack{i' \in \{1, \dots, k', k+1\} \\ j' \in \{k'+1, \dots, k, k+1\}}} n_{i',j}^M n_{i,j'}^M (\mathbb{1}_{i' \prec_j i} \mathbb{1}_{j' \prec_i j}).$$

We can now prove our main result concerning TYPED MAX SIZE MIN BP SMTI.

Theorem 14. TYPED MAX SIZE MIN BP SMTI *belongs to FPT when parameterised by the number k of different types in the given instance.*

Proof. We begin by computing, in polynomial time, the cardinality C_{\max} of a maximum matching in our instance. Our strategy then is to formulate MAX SIZE MIN BP SMTI as an instance of INTEGER QUADRATIC PROGRAMMING. We have a variable $n_{i,j}$ for each $i \in \{1, \dots, k'\} \cup \{k+1\}$ and $j \in \{k'+1, \dots, k\} \cup \{k+1\}$, which corresponds to the number of pairs we will match together consisting of one agent of type i and another of type j . Our goal is to minimise the following objective function

$$\sum_{\substack{1 \leq i \leq k' \\ k'+1 \leq j \leq k}} \sum_{\substack{i' \in \{1, \dots, k', k+1\} \\ j' \in \{k'+1, \dots, k, k+1\}}} n_{i',j} n_{i,j'} (\mathbb{1}_{i' \prec_j i} \mathbb{1}_{j' \prec_i j}).$$

Note that this can be written in the form $x^T Q x$ where x is the vector with entries $n_{i,j}$ for $i \in \{1, \dots, k'\} \cup \{k+1\}$ and $j \in \{k'+1, \dots, k\} \cup \{k+1\}$, and the entry of Q in the column corresponding to $n_{i',j}$ and row corresponding to $n_{i,j'}$ is equal to $\mathbb{1}_{i' \prec_j i} \mathbb{1}_{j' \prec_i j}$. We have two linear constraints which enforce that every agent is involved in exactly one pair (perhaps with a dummy agent), and that the number of pairs that do not involve dummy agents is equal to the maximum possible cardinality of a matching:

$$\sum_{j \in [k+1]} n_{i,j} = |N_i| \quad \forall i \in [k]$$

and
$$\sum_{1 \leq i < j \leq k} n_{i,j} = C_{\max}.$$

It is therefore clear, using Lemma 13, that the minimum possible value of the objective function, subject to the constraints, is equal to the minimum possible number of blocking pairs in a maximum cardinality matching.

Thus, by Theorem 2, we have an FPT algorithm to solve our instance of MAX SIZE MIN BP SMTI. \square

We now consider the problem of minimising the number of agents which are involved in at least one blocking pair. We start by characterising the conditions under which an agent of a particular type can belong to one or more blocking pairs; this characterisation follows immediately from the definition of a blocking pair.

Lemma 15. *Let x be an agent of type i and assume that $M(x)$ is of type j (which would be a dummy type if x is unmatched). Then x belongs to a blocking pair if and only if there is some agent y of type j' who is paired with an agent of type i' (which would be a dummy type if y is unmatched) such that $i \succ_{j'} i'$ and $j' \succ_i j$.*

Using this characterisation, we can now prove our main result concerning TYPED MAX SIZE MIN BA SMTI.

Theorem 16. TYPED MAX SIZE MIN BA SMTI *belongs to FPT when parameterised by the number k of different types in the given instance.*

Proof. For any matching M , we can define a collection of at most $k(k+1)/2$ boolean variables $v_{i,j}$ (for $1 \leq i < j \leq k+1$, where type $k+1$ is a dummy type), so that $v_{i,j}$ is true if and only if the matching contains at least one pair involving an agent of type i and an agent of type j (and unmatched agents are considered to be matched with agents of the dummy type $k+1$). For a given matching M , this collection of variables defines a vector \mathbf{v}_M in $\{0, 1\}^{k(k+1)/2}$, which we call the *type-signature* of the matching M .

Note that there are at most $2^{k(k+1)/2} = \mathcal{O}(2^{k^2})$ possible type-signatures for a matching; we will consider in turn each possible type-signature \mathbf{v} such that $v_{i,j} = 0$ for all pairs of types such that i and j do not find each other mutually acceptable, and determine the minimum number of agents which can be involved in blocking pairs in a maximum matching which has type-signature \mathbf{v} (if a maximum matching with this type-signature exists). Minimising over this set of optimal solutions will give the desired answer.

We now describe how to compute the minimum number of agents involved in blocking pairs in a maximum matching with type-signature \mathbf{v} or else to report that no such maximum matching exists. Our strategy is to encode the problem as an instance of INTEGER LINEAR PROGRAMMING.

Once again, we have variables $n_{i,j}$ (for $1 \leq i < j \leq k+1$) which correspond to the number of pairs in the matching consisting of one agent of type i and another of type j . We now define the constraints. As usual, we need to ensure that every agent is involved in exactly one pair (potentially involving a dummy agent), and as in the proof of

Theorem 14 we need to enforce that the number of pairs that do not involve dummy agents is equal to the maximum cardinality of any matching in our instance. Moreover, we need to make sure that our matching does indeed have type-signature equal to \mathbf{v} . This gives rise to the following linear constraints.

$$\begin{aligned}
& \sum_{1 \leq i < j \leq k} n_{i,j} = C_{\max} \\
& \sum_{1 \leq j \leq k} n_{i,j} = |N_i| && \forall i \in [k] \\
& n_{i,j} > 0 && \forall 1 \leq i < j \leq k + 1 \text{ with } v_{i,j} = 1 \\
\text{and } & n_{i,j} = 0 && \forall 1 \leq i < j \leq k + 1 \text{ with } v_{i,j} = 0.
\end{aligned}$$

Finally, we define our objective function, which captures the number of agents which are involved in at least one blocking pair. By Lemma 15, we know that an agent of type i matched with an agent of type j belongs to a blocking pair if and only if there exist $i' \succ_{j'} i'$ and $j' \succ_i j$ such that $v_{i',j'} = 1$. Thus, for a given type-signature \mathbf{v} , we can compute for each $1 \leq i, j \leq k + 1$ the indicator variable $b_{i,j}$ which takes the value 1 if an agent of type i matched with an agent of type j in a matching with type-signature \mathbf{v} will belong to a blocking pair, and takes the value 0 otherwise. It is now clear that the total number of agents that are involved in at least one blocking pair in the matching is

$$\sum_{1 \leq i < j \leq k + 1} n_{i,j} (b_{i,j} + b_{j,i}).$$

This is our linear objective function. □

We now generalise Theorem 16 to the non-bipartite setting; the proof strategy is very similar, but we include full details here for completeness.

Theorem 17. TYPED MAX SIZE MIN BA SRTI *belongs to FPT when parameterised by the number k of different types in the given instance.*

Proof. Similar to the proof of Theorem 16, for any matching M we define a collection of at most $k(k+3)/2$ boolean variables $v_{i,j}$ (for $1 \leq i \leq j \leq k+1$, where type $k+1$ is a dummy type), so that $v_{i,j}$ is true if and only if the matching contains at least one pair involving an agent of type i and an agent of type j (and unmatched agents are considered to be matched with agents of the dummy type $k+1$). Note that in this setting we additionally have variables $v_{i,i}$ for each $1 \leq i \leq k$, so that $v_{i,i}$ is true if and only if at least one pair in M consists of two agents of type i . Once again, we call the vector \mathbf{v}_M in $\{0, 1\}^{k(k+3)/2}$ the *type-signature* of the matching M .

As before, we observe that there are at most $2^{k(k+3)/2} = \mathcal{O}(2^{k^2})$ possible type-signatures for a matching, and so we can consider in turn each possible type-signature \mathbf{v} (such that $v_{i,j} = 0$ whenever types i and j do not find each other mutually acceptable) and determine the minimum number of agents which can be involved in blocking pairs in a maximum matching which has type-signature \mathbf{v} (if a maximum matching with this type-signature exists). Minimising over this set of optimal solutions will give the desired answer.

Once again, we do this by encoding the problem as an instance of INTEGER LINEAR PROGRAMMING. As usual, we have variables $n_{i,j}$ for $1 \leq i \leq j \leq k+1$ corresponding to the number of pairs consisting of one agent of type i and another of type j . The constraints are very similar to those used in the proof of Theorem 16, the only change being that we now need to allow for the possibility of agents being matched with others of the same type.

$$\begin{aligned}
& \sum_{1 \leq i \leq j \leq k} n_{i,j} = C_{\max} \\
& \sum_{1 \leq j \leq k} n_{i,j} = |N_i| && \forall i \in [k] \\
& n_{\{i,j\}} > 0 && \forall 1 \leq i \leq j \leq k + 1 \text{ with } v_{i,j} = 1 \\
\text{and } & n_{i,j} = 0 && \forall 1 \leq i \leq j \leq k + 1 \text{ with } v_{i,j} = 0.
\end{aligned}$$

The only change we need to the objective function from the proof of Theorem 16 is that we allow for the case $i = j$ in the summation: our goal here is to minimise

$$\sum_{1 \leq i \leq j \leq k+1} n_{i,j}(b_{i,j} + b_{j,i}).$$

To see that this correctly counts blocking agents involved in same-type pairs, recall that $n_{i,i}$ is the number of *pairs* consisting of two agents of type i , so the number of *agents* involved in such pairs is $2n_{i,i}$. If an agent of type i matched to another agent of the same type forms a blocking pair, it follows that $b_{i,i} = 1$ and that all agents of type i matched to other agents of type i are blocking agents; the number of such agents is $2n_{i,i} = n_{i,i}(b_{i,i} + b_{i,i})$, as required. \square

It is straightforward to modify the IQP in the proof of Theorem 17 to solve MIN BA SRTI. We only need to remove the constraint that enforces the matching to be of size C_{\max} .

Corollary 18. TYPED MIN BA SRTI *belongs to FPT when parameterised by the total number of types.*

5. Agents of the same type refine their preferences in the same way

In this section, we generalise the model from Section 4 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 SRTI, and hence also MAX SMTI and MAX HRT, become polynomially solvable if preferences over types are strict, both in this setting and under the basic model. Lastly, we extend the results in Section 4.4 to provide FPT algorithms for MAX SIZE MIN BP SMTI, MAX SIZE MIN BA SMTI, MAX SIZE MIN BP SRTI, MAX SIZE MIN BA SRTI, MIN BP SRTI, and MIN BA SRTI in this refined setting.

5.1. 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 when restricted to $N \setminus \{x, y\}$,
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 SRTI 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 specifies preferences over types, whereas in the refined setting 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.

We illustrate this definition with two short examples.

Example 2. Assume we have 4 types for the agents in a stable marriage setting and that 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.

Example 3. Assume that we are in a stable roommates setting with six agents a, b, \dots, f of type 1. assume that the agents' preference orderings are as follows:

$$\begin{array}{lll} a : (b c) (d e f) & b : (a c) (d e f) & c : (a b) (d e f) \\ d : (a b c) (e f) & e : (a b c) (d f) & f : (a b c) (d e) \end{array}$$

This setting constitutes a consistently-refined-typed instance and it is easy to compute the refined preferences within type 1 which is $(a b c) (d e f)$.

5.2. An FPT algorithm for CONSISTENTLY-REFINED-TYPED MAX SRTI

To extend the result for TYPED MAX SRTI to CONSISTENTLY-REFINED-TYPED MAX SRTI, we need the following result.

Lemma 19. Let I be a consistently-refined-typed instance of SRTI and suppose that M is a matching in I such that (1) there is no pair $(i, j) \in [k]^2$, $i \neq j$, where $j \succ_i \text{worst}_M(i)$ and $i \succ_j \text{worst}_M(j)$, and (2) there is no pair (i, i) , $i \in [k]$, such that there is at least two agents of type i and $i \succ_i \text{second_worst}(i)$. 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 $\mathcal{O}(kn)$.

Proof. We construct a stable matching M' such that $n_{i,j}^{M'} = n_{i,j}^M$. Let $\bigcup_{i,j \in [k], i \leq j} M'_{i,j} = M'$ be a decomposition of M' where $M'_{i,j}$ is the projection of M' onto agent types i and j . The polynomial-time construction of M' takes place in two steps. High-level descriptions of these steps are presented as Algorithms 3 and 4.

ALGORITHM 3: Step 1 in the proof of Lemma 19

Input: Typed instance I of SRTI with n agents and k types, and matching M in I that satisfies conditions of Lemma 19

Output: The number of agents of each type i that are to be matched to agents of each type j

```

for  $i = 1$  to  $k$  do
   $\langle i_1, i_2, \dots, i_{k_i} \rangle \leftarrow$  preference ordering of type  $i$  over accepted types, ties broken arbitrarily
  All agents of type  $i$  are set to be available
  for  $j = i_1$  to  $i_{k_i}$  do
    if  $j \neq i$  then
       $A_i^j \leftarrow$  topmost  $n_{i,j}^M$  available agents of type  $i$ , from the perspective of type  $j$ 
    end
    else
       $A_i^j \leftarrow$  topmost  $2 \cdot n_{i,j}^M$  available agents of type  $i$ , from the perspective of type  $j$ 
    end
    All agents in  $A_i^j$  become unavailable
  end
end
return  $(A_i^j)_{1 \leq i, j \leq k}$ 

```

Step 1: To start with, all agents are available. For each type i , take the candidate types in type i 's decreasing order of preference with ties broken arbitrarily, $\langle i_1, i_2, \dots, i_{k_i} \rangle$, where i_s denotes the type that is ranked s 'th by type i . Starting with $j = i_1$, if $j \neq i$, take the topmost (from the perspective of an agent of type j) $n_{i,j}^M$ available agents

of type i , and put them in A_i^j ; these agents become unavailable from now on. If $j = i$ then take the topmost $2 \cdot n_{i,i}^M$ available agents of type i , and put them in A_i^i ; these agents become unavailable from now on. A_i^j includes agents of type i that are to be matched to agents of type j . Note that as the preference lists of agents of type j over agents of type i may include ties, it may not be possible to determine exactly who are the topmost available $n_{i,j}^M$ agents in type i . To be more precise, when going down the preference list of agents of type j over available agents of type i , we may reach a tie τ including $z > 1$ available agent where we need to pick $x < z$ number of them. If this happens, arbitrarily pick x agents from τ . This step can be done in time $\mathcal{O}(kn)$.

ALGORITHM 4: Step 2 in the proof of Lemma 19

Input: Typed instance I of SRTI with n agents and k types, and for each pair of types $(i, j) \in [k]^2$ the values A_i^j and A_j^i computed in Step 1

Output: Matching M' in I

$M' \leftarrow \emptyset$

$\ell := |A_i^j|$

$\langle x_1, x_2, \dots, x_\ell \rangle \leftarrow$ agents in A_i^j in non-decreasing order of preference with respect to type j

$\langle y_1, y_2, \dots, y_\ell \rangle \leftarrow$ agents in A_j^i in non-decreasing order of preference with respect to type i

for $z = 1$ **to** ℓ **do**

 | Add (x_z, y_z) to M'

end

return M'

Step 2: We now show how to generate $M'_{i,j}$, for each pair of types (i, j) , given A_i^j and A_j^i computed in Step 1. We do so by computing a perfect stable matching amongst the agents in $A_i^j \cup A_j^i$. To do this, we list the agents in A_i^j in non-increasing order of preference with respect to j , and similarly for A_j^i , then pair up each agent with the agent having the same position on the other list. The total time required to do this for all pairs (i, j) is $\mathcal{O}(n)$.

It follows immediately from this construction that, for each i and j , $n_{i,j}^{M'} = n_{i,j}^M$. So it only remains to prove that M' is stable. Assume for a contradiction that M' admits a blocking pair (a, b) where $a, b, M'(a)$ and $M'(b)$ are of types i, j, j' and i' respectively. Five “kinds” of blocking pairs are possible, depending on how a and b compare each others’ types against the types of their partners. We examine each of them and show that M' can admit none.

- *Suppose that $i \neq j$, a prefers type j to type j' , and b prefers type i to type i' .* In the assumption of the lemma we have that in the given M there is no pair $(i, j) \in [k]^2$ where $j \succ_i \text{worst}_M(i)$ and $i \succ_j \text{worst}_M(j)$. By construction of M' , $\text{worst}_M(i)$ remains unchanged under M' for all types i . It thus directly follows that M' cannot admit such a blocking pair.
- *Suppose that $j \simeq_i j'$ ($j \neq j'$) or $i \simeq_j i'$ ($i \neq i'$).* We may assume that $j \simeq_i j'$, $j \neq j'$; a symmetric argument holds when $i \simeq_j i'$, $i \neq i'$. Then, by our assumption that the instance is consistently-refined-typed, any agent of type i , and hence a , is indifferent between all agents who are of type j or type j' . Therefore, $M(a) \simeq_a b$ and (a, b) cannot be a blocking pair.
- *Suppose that $i = i'$ and $j = j'$.* The existence of such a blocking pair implies that $M'_{i,j}$ constructed in Step 2 is not stable with respect to the preferences of agents in $A_{i,j} \cup A_{j,i}$, a contradiction.
- *Suppose that either $i = i'$ or $j = j'$, but not both.* Without loss of generality assume that $j = j'$ and b prefers type i to type i' . Since a prefers b to $M'(a)$, and $M'(a) \in A_{j,i}$, it follows from the construction in Step 1 that, since b is not in $A_{j,i}$ and is in $A_{j,i'}$, any agent of type j (including b) either prefers type i' to type i or is indifferent between them, a contradiction.
- *Suppose that $i = j$ so that a and b are of the same type.* In the assumption of the lemma we have that given M there is no pair $(i, i) \in [k]^2$ where $i \succ_i \text{second_worst}_M(i)$. By the construction of M' , $\text{second_worst}_M(i)$ remains unchanged under M' for all types i . It thus directly follows that M' cannot admit such a blocking pair.

□

Let I be a consistently-refined-typed instance of SRTI and let I' be a typed instance of SRTI 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 19 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 SRTI, 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 19 to convert the solution to a matching of the same cardinality that is stable in the instance I . 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 SRTI is in FPT parameterised by the number k of different types in the instance.

Theorem 20. CONSISTENTLY-REFINED-TYPED MAX SRTI can be solved in time $k^{\mathcal{O}(k^2)} \log^3 n + \mathcal{O}(kn)$.

We also have the following immediate corollary.

Corollary 21. CONSISTENTLY-REFINED-TYPED MAX SMTI and CONSISTENTLY-REFINED-TYPED MAX HRT are in FPT parameterised by the number k of different types in the instance.

5.3. Strict preferences over types

Elsewhere in the paper, we assume that agents can be indifferent between agents of two or more types. We now show that MAX SRTI becomes easier if we restrict the set of possible instances by assuming that agents have strict preferences over types. This argument is based on a private communication with David Manlove. Note that ties are still allowed but they need to be within the same type. Let a *cross-type tie* denote a tie between agents of different types.

Theorem 22. TYPED MAX SRTI and CONSISTENTLY-REFINED-TYPED MAX SRTI are polynomial-time solvable when there are no cross-type ties. Furthermore, all stable matchings (if any exists) are of the same size.

Proof. Since TYPED MAX SRTI is a special case of CONSISTENTLY-REFINED-TYPED MAX SRTI, it suffices to demonstrate that the result holds for the latter problem. Let I be an instance of CONSISTENTLY-REFINED-TYPED MAX SRTI with no cross-type ties, and let I' be another instance of CONSISTENTLY-REFINED-TYPED MAX SRTI obtained from I by breaking any remaining ties in the preference lists arbitrarily and consistently (so that agents of the same type still have identical preference lists). Note that I' is in fact an instance of SRI. All stable matchings in an instance of SRI (if it admits any) have the same cardinality [6] and so we can find a maximum cardinality stable matching in I' , or report that none exists, in polynomial time [6]. We will argue that if I' admits no stable matching then neither does I , otherwise, a maximum cardinality stable matching M in I' is in fact a maximum cardinality stable matching in I . Therefore CONSISTENTLY-REFINED-TYPED MAX SRTI is polynomial-time solvable when there are no cross-type ties.

Consider I_0 , the typed instance of SRTI obtained from I by ignoring preferences within types (i.e. placing all agents of the same type in a single tie). Note that (as preferences over types are strict in I) if we apply the same process to I' , we also obtain the same typed instance I_0 . By Lemma 19 we know that, given any stable matching M_0 in I_0 , there is a stable matching M' in I' of the same cardinality as M_0 . Therefore, if I' admits no stable matching neither does I_0 . Furthermore, if I_0 admits no stable matching then it follows from the definition of stability that neither does I .

Now assume that I' admits a stable matching. It is clear that M must be stable in I , since (as each agent's preference list in I' is obtained from that in I by breaking ties) any blocking pair with respect to M in the instance I would also be a blocking pair in I' . It remains to argue that there cannot be any stable matching of cardinality greater than $|M|$ in I . Since all stable matchings in I' have the same cardinality, we can conclude that (following Lemma 19) every stable matching in I_0 has the same cardinality as M . Moreover, as any matching that is stable in I must also be stable in I_0 , it follows that all stable matchings in I have the same cardinality as M . Thus M is a maximum cardinality stable matching in I , as required. \square

The above result, combined with Corollary 21, gives us the following result.

Corollary 23. (CONSISTENTLY-REFINED-)TYPED MAX SMTI and (CONSISTENTLY-REFINED-)TYPED MAX HRT are polynomial-time solvable when there are no cross-type ties. Furthermore, all stable matchings are of the same size.

5.4. FPT algorithms for CONSISTENTLY-REFINED-TYPED MAX SIZE MIN BP SMTI and CONSISTENTLY-REFINED-TYPED MAX SIZE MIN BA SMTI

In this section we show how we can apply the ideas from the Section 5.2 to solve MAX SIZE MIN BP/BA SMTI for consistently-refined-typed instances.

Corollary 24. CONSISTENTLY-REFINED-TYPED MAX SIZE MIN BP SMTI and CONSISTENTLY-REFINED-TYPED MAX SIZE MIN BA SMTI belong to FPT when parameterised by the number of types.

Proof. We use the same strategy for both problems. First, we solve the problem for the corresponding typed instance (ignoring the more refined preferences within types). Note that the number of blocking pairs or blocking agents achieved in this simplification of the problem clearly gives a lower bound on the minimum number that can be achieved if we take into account all information; we will argue that in fact we can always obtain a matching which does not increase either quantity when we take into account the full preference lists.

To do this, we follow the method described in Lemma 19. Given the number of pairs $n_{i,j}$ consisting of agents of types i and j for each $1 \leq i < j \leq k$, this method allows us to construct a matching M where, for each $1 \leq i < j \leq k$ we have exactly $n_{i,j}$ pairs involving one agent of type i and one of type j , and there is no blocking pair (x, y) such that x is currently matched to an agent of the same type as y . Thus the only blocking pairs in M are those of the form (x, y) where x is of type i and y of type j , and $j \succ_i \text{type}(M(x))$ and $i \succ_j \text{type}(M(y))$. But these are precisely the blocking pairs that occur in the relaxation to a typed instance.

Thus we can indeed obtain a solution to MAX SIZE MIN BP SMTI or MAX SIZE MIN BA SMTI by applying the appropriate algorithm to find the number of pairs of each type under the relaxation to a typed instance, and then use the method of Lemma 19 to extend this to a matching which does not introduce any additional blocking pairs when the full preference lists are taken into consideration. \square

Combining this result with the techniques of Section 4.4, we obtain the following corollary.

Corollary 25. CONSISTENTLY-REFINED-TYPED MAX SIZE MIN BA SRTI and CONSISTENTLY-REFINED-TYPED MIN BA SRTI belong to FPT when parameterised by the total number of types.

6. 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 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.

We show $(1, \text{TOP})$ -EXCEPTION TYPED MAX SMTI, in which each agent finds at most one candidate exceptional, whom s/he promotes to the top of his/her list, belongs to FPT. In contrast, if we allow for two (or more) exceptions, and exceptional candidates can appear anywhere in the preference lists, then the problem remains NP-hard even when there are only a constant number of types (i.e. it is para-NP-hard). It follows that $(2, \text{ANY})$ -EXCEPTION TYPED MAX SMTI cannot be in XP (and hence in FPT) unless $P=NP$.

6.1. An FPT algorithm for (1,Top)-EXCEPTION TYPED MAX SMTI

We consider a generalisation of typed instances in which (i) every agent considers at most one other agent to be exceptional, and (ii) every agent would strictly prefer to be assigned to the agent they consider to be exceptional than to any other agent. For each agent a let $\text{ex}(a)$ denote the exceptional candidate from a 's point of view; $\text{ex}(a) = \emptyset$ if a does not find any candidate exceptional. Then, I is a $(1, \text{Top})$ -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 $N \setminus \{\text{ex}(x), \text{ex}(y)\}$, 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 $(1, \text{Top})$ -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 $(1, \text{Top})$ -exception-typed instance of SMTI is given as input by, in addition to the specifications needed for a typed instance (see Section 4.1), providing for each agent his or her exceptional candidate (if s/he has one).

Let I be a $(1, \text{Top})$ -exception-typed instance of SMTI, and let M be a matching in I . As in Section 4.2, 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 4 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 4.3, 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 $(1, \text{Top})$ -exception-typed SMTI realises the function worst^{ex} , mapping nonempty subtypes $i[J]$ to values in $\{0, 1, \dots, k+1\}$, 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 $(1, \text{Top})$ -exception-typed instance.

Lemma 26. *Let I be a $(1, \text{Top})$ -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. $i \succ_j \text{worst}^{\text{ex}}_M(j[\text{class}_j(i)])$.

Proof. Suppose first that M is not stable. In this case there is a blocking pair (x, y) where x is of type i and y is of type j . We show that at least one of the two aforementioned conditions must hold. If neither x nor y finds the other exceptional, then it must be that they prefer each other's type to the type of their assigned partner. This implies that $j \succ_i \text{type}(M(x)) \succeq_i \text{worst}_M(i)$ and $i \succ_j \text{type}(M(y)) \succeq_j \text{worst}_M(j)$ so condition (1) holds. If one of x and y finds the other exceptional, we may assume without loss of generality that x finds y exceptional. We may

further assume, without loss of generality, that x is the (equally) most desirable agent (from the point of view of y) who prefers y to his/her current partner. This means there is no agent of type ℓ , with $\ell \succ_j \text{type}(x) = i$, who considers y to be exceptional, implying that $y \in N_{j[\text{class}_j(i)]}$. Hence, by the definition of a blocking pair, we see that $i = \text{type}(x) \succ_j \text{type}(M(y)) \succeq_j \text{worst}^{\text{ex}}_M(j[\text{class}_j(i)])$, and condition (2) holds.

Conversely, it is straightforward to see that if we have any pair (i, j) satisfying condition (1) or (2) then there will be a blocking pair and hence M will not be stable. If condition (1) holds, then $\text{worst}_M(i)$ and $\text{worst}_M(j)$ must both be greater than 0, and the condition implies that there is a pair (x, y) of type (i, j) who prefer each other to their partners. If condition (2) holds, then there exists an agent $y \in N_{j[\text{class}_j(i)]}$ who is considered exceptional by some agent x whose type is in $\text{class}_j(i)$, and y prefers any agent whose type belongs to $\text{class}_j(i)$ to his/her current partner, so (x, y) forms a blocking pair. \square

We will say that a function worst^{ex} is *I-exception-stable* for a $(1, \text{Top})$ -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 $i \succ_j \text{worst}^{\text{ex}}(j[\text{class}_j(i)])$. Observe that Lemma 26 gives us the following straightforward corollary.

Corollary 27. *Let I be a $(1, \text{Top})$ -exception-typed instance of SMTI. Suppose that M is a matching in I which realises the I-exception-stable function worst^{ex} , and that M contains a pair (x, y) where x considers y to be exceptional and y is of subtype $i[J]$. Then $\text{type}(x) \in J$ and $\text{worst}^{\text{ex}}(i[J]) \in J$.*

Proof. Since $\text{subtype}(y) = i[J]$, there exists $j \in J$ such that some agent z of type j considers y to be exceptional. By definition of subtype, the fact that x considers y to be exceptional means that $j \succeq_i \text{type}(x)$. Thus, as M realises worst^{ex} , we see that

$$j \succeq_i \text{type}(x) = \text{type}(M(y)) \succeq_i \text{worst}^{\text{ex}}(i[J]).$$

If any of the preferences in this equation is in fact strict, we would violate condition (2) of Lemma 26, contradicting our assumption that worst^{ex} is *I-exception stable*. Therefore we must have $j \simeq_i \text{type}(x) \simeq_i \text{worst}^{\text{ex}}(i[J])$, implying that both $\text{type}(x)$ and $\text{worst}^{\text{ex}}(i[J])$ are in $\text{class}_i(j) = J$, as required. \square

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 5 in this setting.

Lemma 28. *Let I be a $(1, \text{Top})$ -exception-typed instance of SMTI. Then the cardinality of the largest stable matching in I is equal to*

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

We now argue that we can compute $\max(\text{worst}^{\text{ex}})$ for any *I-exception-stable* function worst^{ex} in polynomial time.

Lemma 29. *Let I be a $(1, \text{Top})$ -exception-typed instance of SMTI, and fix an I-exception-stable function worst^{ex} . We can compute $\max(\text{worst}^{\text{ex}})$, and generate a stable matching of size $\max(\text{worst}^{\text{ex}})$, in time $\mathcal{O}(n^{5/2} \log n)$.*

Proof. Suppose that there are n agents in total, with n_1 women and n_2 men. Using a binary search strategy, we can determine the maximum size of a matching realising worst^{ex} by solving $\mathcal{O}(\log n)$ instances of the decision problem “Is $\max(\text{worst}^{\text{ex}})$ at least c ?”, where $c \in \{1, \dots, \min(n_1, n_2)\}$. We will show that to determine whether $\max(\text{worst}^{\text{ex}}) \geq c$ it is enough to decide whether an undirected graph $G = (V, E)$ with $\mathcal{O}(n)$ vertices, constructed from I in time $\mathcal{O}(n^2)$, admits a perfect matching. Solving the latter can be done in time $\mathcal{O}(\sqrt{|V||E|})$, which coupled with the fact that $|V| = \mathcal{O}(n)$ gives us $\mathcal{O}(n^{5/2})$. Our proof is constructive; that is, if G admits a perfect matching E' , E' corresponds to a stable matching in I of size at least c . Therefore, we conclude that we can compute $\max(\text{worst}^{\text{ex}})$, and generate a stable matching of size $\max(\text{worst}^{\text{ex}})$, in time $\mathcal{O}(n^{5/2} \log n)$.

We now show how to construct $G = (V, E)$. The vertex-set V contains one vertex corresponding to each agent in I , together with $n - 2c$ dummy vertices. The edge-set E contains an edge between the agents x , of type $a[C]$, and y , of type $b[D]$, if and only if x and y find each other mutually acceptable and either

1. neither x nor y considers the other to be exceptional, $b \succeq_a \text{worst}^{\text{ex}}(a[C])$, and $a \succeq_b \text{worst}^{\text{ex}}(b[D])$, or
2. $a \in D$, x considers y to be exceptional, and $\text{worst}^{\text{ex}}(b[D]) \in D$, or
3. $b \in C$, y considers x to be exceptional, and $\text{worst}^{\text{ex}}(a[C]) \in C$.

We additionally have an edge between each pair of dummy vertices, and between each dummy vertex and every agent x such that $\text{worst}^{\text{ex}}(i[J]) = k + 1$ where $i[J] = \text{subtype}(x)$; that is, if worst^{ex} allows for x to be unmatched.

We note that the number of vertices in G is $2(n - c)$, and so is even. It is clear that we can construct G from I and worst^{ex} in time $\mathcal{O}(n^2)$. We claim that G contains a perfect matching if and only if $\max(\text{worst}^{\text{ex}}) \geq c$.

We first prove that if $\max(\text{worst}^{\text{ex}}) \geq c$, then G admits a perfect matching. Let M be a matching of maximum cardinality which realises worst^{ex} .

We start by arguing that we can find a set of $|M|$ independent edges in G , none of which is incident with a dummy vertex; these edges will be in one-to-one correspondence with the pairs in M . Let (x, y) be a pair of agents which is matched in M , where x is of subtype $a[C]$ and y is of subtype $b[D]$: we want to show that $xy \in E$. As we know that x and y do not both consider the other to be exceptional, we may assume without loss of generality that y does not consider x to be exceptional. There are then two cases to consider.

1. **Neither x nor y considers the other to be exceptional.** Since M is a matching that realises worst^{ex} and $(x, y) \in M$, it follows that $b \succeq_a \text{worst}^{\text{ex}}(a[C])$ and $a \succeq_b \text{worst}^{\text{ex}}(b[D])$. Therefore, by the construction of G , $xy \in E$.
2. **x considers y to be exceptional.** In this case, by the construction of G , we have that $xy \in E$ if and only if $a \in D$ and $\text{worst}^{\text{ex}}(b[D]) \in D$. Both conditions follow immediately from Corollary 27 as M realises worst^{ex} and worst^{ex} is I -exception stable.

Thus we can indeed find a set E_1 of $|M|$ independent edges in G , which are in one-to-one correspondence with the pairs of agents matched in M . We now extend E_1 to cover the vertices corresponding to agents not matched in M . Suppose that x is unmatched in M . Since M realises worst^{ex} , we must have $\text{worst}^{\text{ex}}(i[J]) = k + 1$ where $i[J] = \text{subtype}(x)$ and hence, by construction of G , there is an edge from x to every dummy vertex. Recall that the number of dummy vertices in G is $n - 2c$ and $|M| \geq c$. Therefore the number of unmatched agents in M is $n - 2|M| \leq n - 2c$, implying that there are enough dummy vertices to cover unmatched agents: we can greedily choose a set E_2 of $n - 2|M|$ independent edges which cover the remaining vertices in G that correspond to agents, so that no two edges in $E_1 \cup E_2$ are incident (each edge in E_2 connects a vertex corresponding to an unmatched agent in M to a dummy vertex). Finally, if there remain any uncovered dummy vertices (in the case that $|M| > c$) we can extend our matching $E_1 \cup E_2$ to cover all such vertices by selecting further independent edges greedily, as all dummy vertices are pairwise adjacent. Hence G contains a perfect matching.

Conversely, we prove that if G contains a perfect matching E' , then $\max(\text{worst}^{\text{ex}}) \geq c$. Let M be the matching in I obtained by matching agents x and y together if and only if $xy \in E'$. The number of pairs in M is equal to the number of edges in E' that are not incident with any dummy agent; this is at least $\frac{n - (n - 2c)}{2} = c$. It therefore remains to show that M is a matching in I that realises worst^{ex} . Let x be any agent, and suppose that $\text{subtype}(x) = i[J]$. It follows immediately from the construction of G that x and $M(x)$ find each other mutually acceptable, and so M is a matching in I . To prove that M realises worst^{ex} we only need to show that $M(x)$ is at least as desirable, from the point of view of x , as any agent of type $\text{worst}^{\text{ex}}(i[J])$. There are four cases to consider:

1. **x is unmatched in M .** In this case we must have an edge xz in E' , where z is a dummy vertex; such an edge is only present in G if $\text{worst}^{\text{ex}}(i[J]) = k + 1$, in which case we are done.
2. **$M(x) = y$ and neither x nor y considers the other exceptional.** Suppose that y is of type ℓ . In this case, the fact that $xy \in E$ implies that $\ell \succeq_i \text{worst}^{\text{ex}}(i[J])$ (by construction of G), so we have $\text{type}(M(x)) \succeq_i \text{worst}^{\text{ex}}(i[J])$ as required.
3. **$M(x) = y$ and x considers y to be exceptional.** In this case x is matched to his/her most desired partner, so the condition must be satisfied.
4. **$M(x) = y$ and y considers x to be exceptional.** Suppose that y is of type ℓ . In this case, the fact that $xy \in E$ implies that $\ell \in J$ and $\text{worst}^{\text{ex}}(i[J]) \in J$. As x is indifferent between all types belonging to J , we have that $\text{type}(M(x)) = \ell \succeq_i \text{worst}^{\text{ex}}(i[J])$, as required.

Hence M is indeed a matching in I which realises worst^{ex} and $|M| \geq c$. This completes the proof. \square

Corollary 30. (1, TOP)-EXCEPTION TYPED MAX SMTI can be solved in time $\mathcal{O}\left(k^{k^2}(k^3 + n^{5/2} \log n)\right)$.

Proof. Suppose that I is the input to our instance of (1, TOP)-EXCEPTION TYPED MAX SMTI. We consider each possible function $\text{worst}^{\text{ex}} : [k] \times [k] \rightarrow \{0, 1, \dots, k+1\}$ in turn; there are at most $(k+2)^{k^2}$ such functions. We can determine in time $\mathcal{O}(k^3)$ whether worst^{ex} is I -exception-stable. For each I -exception-stable function worst^{ex} , we compute $\max(\text{worst}^{\text{ex}})$ in time $\mathcal{O}(n^{5/2} \log n)$ by Lemma 29. We then take the maximum value of $\max(\text{worst}^{\text{ex}})$ over all I -exception-stable functions worst^{ex} which, by Lemma 28, is equal to the cardinality of the largest stable matching in I . The proof of Lemma 29 is constructive, that is, it also generates a stable matching of size $\max(\text{worst}^{\text{ex}})$. \square

6.2. (2, ANY)-EXCEPTION TYPED MAX SMTI is NP-hard for a constant number of types

In this section we investigate the effects of allowing each agent to consider more than one candidate to be exceptional. 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 is NP-hard even under severe restrictions. In fact, we show that the special case COM SMTI, which involves deciding whether a given instance of SMTI admits a perfect stable matching (i.e., a matching that matches all agents), is NP-complete even if the number of types is bounded by a constant. Our proof is by reduction from the NP-complete problem CLIQUE, defined as follows: given an undirected graph $G = (V, E)$ and $r \in \mathbb{N}$, does G contain a clique on at least r vertices?

Theorem 31. (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.

Proof. The problem obviously is in NP as we can easily check in quadratic time whether a given matching is perfect and stable. To prove NP-hardness, we reduce from CLIQUE. Let $(G = (V, E), r)$ be the input to an instance of CLIQUE, and suppose that $V = \{v_1, \dots, v_n\}$ and $E = \{e_1, \dots, e_m\}$. We will construct an instance I of (2-ANY)-EXCEPTION TYPED MAX SMTI so that G contains a clique on r vertices if and only if there is a perfect stable matching in I .

Our instance I has six types in total; types 1, 2 and 3 are types of men and types 4, 5 and 6 are types of women. Type 1 contains one man corresponding to each edge in E ; abusing notation, we shall write e_i for the man corresponding to the edge e_i . Type 4 contains one woman corresponding to each vertex in V , and again we abuse notation to write v_i for the woman corresponding to the vertex v_i . Types 2 and 3 consist of r and $n-r$ men respectively, and types 5 and 6 consist of $\binom{r}{2}$ and $m - \binom{r}{2}$ women respectively.¹ The preferences over types for each type is as follows:

Type 1:	6 5 4	Type 4:	2 1 3
Type 2:	4	Type 5:	1
Type 3:	4	Type 6:	1

We now describe the exceptions. Only men of type 1 have any exceptions in their preference lists. Each man e_i of type 1 considers two women of type 4 to be exceptional, and he ranks these two women in a tie between types 6 and 5. If $e_i = (v_j, v_\ell)$ then e_i considers v_j and v_ℓ exceptional. This completes the definition of I . Clearly I can be constructed from (G, r) in polynomial time.

We claim that G contains a clique on r vertices if and only if I admits a perfect stable matching.

First, we argue that if (G, r) is a yes-instance, then I admits a perfect stable matching. Let $U \subset V$ be a set of r vertices in G which induces a clique. We construct matching M in I as follows. Each woman in U (who are all of type 4) is matched to a man of type 2. Let F denote the set of $\binom{r}{2}$ edges with both endpoints in U . Each man in F (who is of type 1) is matched to a woman of type 5. The remaining women of type 4 are matched to men of type 3, and the remaining men of type 1 are matched to women of type 6. M clearly matches all agents. Note that the matched pairs are all of the following four pairs of types: (1,5), (1,6), (2,4), and (3,4). It remains to show that M is stable.

Note that no agent of type 2, 3, 5 or 6 can be involved in a blocking pair: all these agents are matched in M , and are indifferent between all candidates they find acceptable. Hence any blocking pair is between a man of type 1 and a woman of type 4. Moreover, no blocking pair can involve a man m of type 1 who is matched with a woman of type 6, or a woman w of type 4 who is matched with a man of type 2: such agents (m or w) are already matched with their first (equal) choice of partner. Therefore, if there is a blocking pair, it is of the form (e, v) where e is a man of type 1, v is a woman of type 4, $M(e)$ is of type 5 and $M(v)$ is of type 3. By construction of M , as $M(e)$ is of type 5, we know that $e \in F$. Thus, the two women that e finds exceptional both belong to U and so are matched to men of type

¹If there is a clique on r vertices then there must be at least $\binom{r}{2}$ number of edges, hence $m - \binom{r}{2} \geq 0$.

2. It follows that e does not find v exceptional. Thus e does not prefer v (of type 4) to his current partner of type 5, and so (v, e) cannot form a blocking pair. We can therefore conclude that M is indeed stable.

Conversely, we prove that if there is a perfect stable matching M in I , then (G, r) is a yes-instance. Since M matches every agent, all men of types 2 and 3 must be matched to women of type 4 (i.e. women corresponding to vertices), and all women of types 5 and 6 must be matched to men of type 1 (i.e. men corresponding to edge). The numbers of agents in each type mean that in a perfect matching no man corresponding to an edge can be matched to a woman corresponding to a vertex. Let W be the set of r women who are matched to men of type 2. We will argue that W must induce a clique in G .

Suppose, for a contradiction, that W does not induce a clique. This means that there are at most $\binom{r}{2} - 1$ edges of G with both endpoints in W . In particular, there is a woman of type 5 who is matched to a man of type 1, corresponding to some edge e whose endpoints are not both in W . Suppose, without loss of generality, that v is an endpoint of e that is not in W . Since $v \notin W$, we know that $M(v)$ is of type 3. This means that v strictly prefers e (of type 1) to her current partner. Moreover, since v is an endpoint of e , she is considered exceptional by e and so e prefers v to any woman of type 5, including $M(e)$. Thus, (v, e) is a blocking pair, contradicting the assumption that M is a stable matching. We therefore conclude that W must induce a clique and so (G, r) is a yes-instance. \square

The next result then follows immediately.

Corollary 32. *(2,ANY)-EXCEPTION TYPED MAX SMTI is NP-hard, 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.*

We note that Theorem 31 in fact implies hardness in this setting for several other problems we have discussed in the paper: SMTI is a special case of both SRTI and HRT, and in order to determine the existence or otherwise of a perfect stable matching it is certainly enough to determine the minimum possible number of blocking pairs or blocking agents in a maximum size stable matching.

Corollary 33. *The following problems are all NP-hard, even if there are only six types, preferences over types are strict, and only agents of one type have exceptions in their preference lists:*

- MAX HRT,
- MAX SRTI,
- MAX SIZE MIN BP SMTI,
- MAX SIZE MIN BA SMTI,
- MAX SIZE MIN BP SRTI, and
- MAX SIZE MIN BA SRTI.

7. Summary and Future Work

We studied settings in which agents are partitioned into k different types, and the type of an agent determines (most of) his or her preferences, as well as how s/he is compared against other agents. We considered a basic setting and two generalisations. In the basic model, referred to as typed, agents have preferences over types and agents of the same types have identical preferences. In the first generalisation, referred to as consistently-refined-typed, agents' preferences may be refined by more detailed preferences within a single type, subject to the requirement that agents of the same type still have identical preference lists. In the second generalisation, referred to as exception-typed, each agent may regard some small collection of other agents to be exceptional and rank them without regard to their types.

If k is considered to be part of the input, then our models do not place any restrictions on preference lists; we can create a type for each agent and hence let $k = n$. Thus we can deduce that MAX SMTI, MAX HRT, MAX SRT, MAX SIZE MIN BP/BA SMI, MAX SIZE MIN BP/BA SRI, MIN BP SR, and MIN BA SRI are all NP-complete, in either of our three settings, when k is part of the input.

We are interested in, and have argued for the existence of, scenarios where k is not part of the input and is much smaller than n . Under both typed and consistently-refined-typed settings, we showed that MAX SMTI, MAX HRT, MAX SRTI, MAX SIZE MIN BP/BA SMTI, MAX SIZE MIN BA SRTI, and MIN BA SRTI belong to the

parameterised complexity class FPT when parameterised by the number of different types of agents, and so admit efficient algorithms when this number of types is small. We were further able to prove that MAX SMTI, MAX HRT and MAX SRTI are polynomial-time solvable when agents have strict preferences over types.

Under the exception-typed setting, we showed that if each agent finds one candidate exceptional and promotes that candidate to the top of his/her preference list, then MAX SMTI belongs to FPT parameterised by k . In contrast, if we allow for each agent to find two or more candidates exceptional who can appear anywhere in the preference lists, then MAX SMTI remains NP-hard, even when the number of types is bounded by a constant.

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, as well as the complexity of MAX SRTI, MAX SIZE MIN BP/BA SMTI, MAX SIZE MIN BP/BA SRTI, and MIN BP/BA SRTI under the exception-typed setting. 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 MAX SIZE MIN BP/BA SMTI 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] N. Bhatnagar, S. Greenberg, D. Randall, Sampling stable marriages: why spouse-swapping won't work, in: Proceedings of the 19th ACM/SIAM Symposium on Discrete Algorithms, SODA '08, ACM-SIAM, 2008, pp. 1223–1232.
- [2] P. Chebolu, L. A. Goldberg, R. Martin, The complexity of approximately counting stable matchings, Theoretical Computer Science 437 (2012) 35–68.
- [3] D. Manlove, Algorithmics of Matching Under Preferences, World Scientific, 2013.
- [4] D. Gale, L. Shapley, College admissions and the stability of marriage, American Mathematical Monthly 69 (1962) 9–15.
- [5] R. Irving, On the stable room-mates problem, Tech. Rep. CSC/86/R5, University of Glasgow, Department of Computing Science (1986).
- [6] D. Gusfield, R. Irving, The Stable Marriage Problem: Structure and Algorithms, MIT Press, 1989.
- [7] E. Ronn, NP-complete stable matching problems, Journal of Algorithms 11 (1990) 285–304.
- [8] A. Roth, The evolution of the labor market for medical interns and residents: a case study in game theory, Journal of Political Economy 92 (6) (1984) 991–1016.
- [9] A. Roth, A natural experiment in the organization of entry level labor markets: Regional markets for new physicians and surgeons in the U.K, American Economic Review 81 (1991) 415–440.
- [10] A. Roth, X. Xing, Jumping the gun: imperfections and institutions related to the timing of market transactions, American Economic Review 84 (4) (1994) 992–1044.
- [11] P. Biró, D. Manlove, S. Mittal, Size versus stability in the marriage problem, Theoretical Computer Science 411 (2010) 1828–1841.
- [12] K. Iwama, D. Manlove, S. Miyazaki, Y. Morita, Stable marriage with incomplete lists and ties, in: Proceedings of ICALP '99: the 26th International Colloquium on Automata, Languages, and Programming, Vol. 1644 of Lecture Notes in Computer Science, Springer, 1999, pp. 443–452.
- [13] R. Irving, D. Manlove, S. Scott, The stable marriage problem with master preference lists, Discrete Applied Mathematics 156 (15) (2008) 2959–2977.

- [14] D. Manlove, R. Irving, K. Iwama, S. Miyazaki, Y. Morita, Hard variants of stable marriage, *Theoretical Computer Science* 276 (1-2) (2002) 261–279.
- [15] G. O’Malley, Algorithmic aspects of stable matching problems, Ph.D. thesis, University of Glasgow, Department of Computing Science (2007).
- [16] A. Roth, M. Sotomayor, *Two-Sided Matching: a Study in Game-Theoretic Modeling and Analysis*, Vol. 18 of *Econometric Society Monographs*, Cambridge University Press, 1990.
- [17] D. Abraham, P. Biró, D. Manlove, “Almost stable” matchings in the Roommates problem, in: *Proceedings of the 3rd Workshop on Approximation and Online Algorithms, WAOA ’05*, Vol. 3879 of *Lecture Notes in Computer Science*, Springer, 2006, pp. 1–14.
- [18] J. Chen, D. Hermelin, M. Sorge, H. Yedidsion, How hard is it to satisfy (almost) all roommates?, in: *45th International Colloquium on Automata, Languages, and Programming, ICALP 2018*, July 9-13, 2018, Prague, Czech Republic, 2018, pp. 35:1–35:15.
- [19] R. G. Downey, M. R. Fellows, *Fundamentals of Parameterized Complexity*, Springer London, 2013.
- [20] J. Flum, M. Grohe, *Parameterized Complexity Theory*, Springer, 2006.
- [21] M. Cygan, F. Fomin, Ł. Kowalik, D. Lokshtanov, D. Marx, M. Pilipczuk, M. Pilipczuk, S. Saurabh, *Parameterized Algorithms*, Springer International Publishing, 2015.
- [22] A. Frank, É. Tardos, An application of simultaneous Diophantine approximation in combinatorial optimization, *Combinatorica* 7 (1) (1987) 49–65.
- [23] R. Kannan, Minkowski’s convex body theorem and integer programming, *Mathematics of Operations Research* 12 (3) (1987) 415–440.
- [24] H. Lenstra Jr, Integer programming with a fixed number of variables, *Mathematics of Operations Research* 8 (4) (1983) 538–548.
- [25] D. Lokshtanov, Parameterized integer quadratic programming: Variables and coefficients, arXiv:1511.00310 [cs.DS] (2015).
- [26] P. Biró, D. Manlove, S. Mittal, Size versus stability in the marriage problem, Tech. Rep. TR-2008-283, University of Glasgow, Department of Computing Science (2008).
- [27] K. Hamada, K. Iwama, S. Miyazaki, An improved approximation lower bound for finding almost stable maximum matchings, *Information Processing Letters* 109 (18) (2009) 1036–1040.
- [28] D. Marx, I. Schlotter, Parameterized complexity and local search approaches for the stable marriage problem with ties, *Algorithmica* 58 (1) (2010) 170–187.
- [29] M. Mnich, I. Schlotter, Stable marriage with covering constraints—a complete computational trichotomy, in: *Proceedings of the 10th International Symposium on Algorithmic Game Theory, SAGT’17*, 2017, pp. 320–332.
- [30] S. Gupta, S. Saurabh, M. Zehavi, On treewidth and stable marriage, Tech. Rep. 1707.05404, CoRR, Cornell University Library (2017).
- [31] S. Gupta, S. Roy, S. Saurabh, M. Zehavi, Balanced stable marriage: How close is close enough, Tech. Rep. 1707.09545, CoRR, Cornell University Library (2017).
- [32] D. Adil, S. Gupta, S. Roy, S. Saurabh, M. Zehavi, Parameterized algorithms for stable matching with ties and incomplete lists, *Theoretical Computer Science* 723 (2018) 1 – 10.
- [33] P. Chebolu, L. A. Goldberg, R. Martin, The complexity of approximately counting stable roommate assignments, *Journal of Computer and System Sciences* 78 (5) (2012) 1579–1605.
- [34] F. Echenique, S. Lee, M. Shum, M. B. Yenmez, The revealed preference theory of stable and extremal stable matchings, *Econometrica* 81 (1) (2013) 153–171.

- [35] E. Choo, A. Siow, Who marries whom and why, *Journal of Political Economy* 114 (1) (2006) 175–201.
- [36] H. Aziz, B. de Keijzer, Complexity of coalition structure generation, in: *Proceedings of the 10th International Conference on Autonomous Agents and Multiagent Systems, AAMAS'11*, 2011, pp. 191–198.
- [37] T. Shrot, Y. Aumann, S. Kraus, On agent types in coalition formation problems, in: *Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems, AAMAS'10*, 2010, pp. 757–764.
- [38] J. B. Orlin, Max flows in $\mathcal{O}(nm)$ time, or better, in: *Proceedings of the 45th Annual ACM Symposium on Theory of Computing, STOC '13*, ACM, 2013, pp. 765–774.