An algorithm for a super-stable roommates problem

Tamás Fleiner* Robert W. Irving† David F. Manlove‡

Abstract

In this paper we describe an efficient algorithm that decides if a stable matching exists for a generalized stable roommates problem, where, instead of linear preferences, agents have partial preference orders on potential partners. Furthermore, we may forbid certain partnerships, that is, we are looking for a matching such that none of the matched pairs is forbidden, and yet, no blocking pair (forbidden or not) exists.

To solve the above problem, we generalize the first algorithm for the ordinary stable roommates problem.

Keywords: stable marriages, stable roommates problem, polynomial time algorithm

1 Introduction

The study of stable matching problems was initiated by Gale and Shapley who introduced the *stable marriage problem* in [4]. In this problem, each of n men and n women have a linear preference order on the members of the opposite gender. We ask if there exists a marriage scheme in which no man and woman mutually prefer one another to their eventual partners. The authors prove that the so called deferred acceptance algorithm always finds a stable marriage scheme.

It is natural to ask the same question in a more general, non-bipartite model, in which we have n agents with preference orders on all other agents. This is the so called *stable roommates problem*, and we are looking for a matching (i.e. a pairing of the agents) such that no two agents prefer one another to their assigned partners. Such a matching is called a *stable* one. A significant difference between the stable marriage and the stable roommates models is that for the latter, it might happen that no stable matching exists [4]. A solution for the stable roommates problem was proposed by Irving [5] as an efficient algorithm that either finds a stable matching or concludes that no stable matching exists for the particular problem. Later, Tan [9] used this algorithm to give a good characterization, that is, he proved that for any stable roommates problem, there always exists a so called *stable partition* (that can be regarded as a half integral fractional stable matching [1]) with the property

*Department of Computer Science and Information Theory, Budapest University of Technology and Economics, Magyar Tudósok körútja 2, Budapest, H-1117. fleiner@cs.bme.hu Research was supported by the János Bolyai Research fellowship of the Hungarian Academy of Sciences, by the Royal Society of Edinburgh International Exchange Programme, and by the K69027 and K60802 OTKA grants.

†School of Computing Science University of Glasgow G12 8QQ, UK. Rob.Irving@glasgow.ac.uk Supported by the Engineering and Physical Sciences Research Council grant EP/E011993/1.

‡School of Computing Science University of Glasgow G12 8QQ, UK. David.Manlove@glasgow.ac.uk Supported by the Royal Society of Edinburgh / Scottish Executive Personal Research Fellowship, and Engineering and Physical Sciences Research Council grants GR/R84597/01 and EP/E011993/1.

that either it is a stable matching, or it is a compact proof for the non-existence of a stable matching in the particular model.

In both the stable marriage and the stable roommates problems, strict preferences of the participating agents play a crucial role. However, in many practical situations, we have to deal with indifferences in the preference orders. We model this phenomenon such that preference orders are partial (rather than linear) orders. We can extend the notion of a stable matching to this model in at least three different ways [6]. One possibility is that a matching is *weakly stable* if no pair of agents a, b exists such that they mutually prefer one another to their assigned partner. Ronn proved that deciding the existence of a weakly stable matching is NP-complete for the stable roommates problem [7]. A more restrictive notion is that a matching is *strongly stable* if there are no agents a and b such that a prefers b to his assigned partner and b does not prefer his assigned partner to a. Scott gave an algorithm that finds a strongly stable matching or reports if none exists in $O(m^2)$ time [8]. (Here m stands for the number of edges in the underlying graph.) The most restrictive notion is that of super-stability. A matching is *super-stable* if there exist no two agents a and b such that neither of them prefers his assigned situation in the matching to being a partner of the other. In other words, a matching is super-stable, if it is stable for any linear extensions of the preference orders of the agents. For the case where indifference is transitive (that is, if preferences are determined by scores that can be equal), Irving and Manlove gave an $O(m)$ algorithm to find a super-stable matching, if such a matching exists [6]. Interestingly, the algorithm has two phases, just like Irving's [5], but its second phase is completely different. It is also noted in [6] that the algorithm works without modification for the more general case where preferences are partial (rather than linear) orders.

The motivation of our present work is to give a direct algorithm for an extension of the super-stable matching problem. Our algorithm works similarly to Irving's original algorithm in [5]. This latter algorithm keeps on deleting edges of the underlying graph until it concludes that no stable matching exists or a single stable matching is left. It turns out that deleting an edge is too harsh a transformation: our algorithm uses a finer one as well. For this reason, we extend our model such that we allow so-called forbidden edges and beyond deleting, our algorithm may forbid certain edges. This way we solve a problem that is a common generalization of the stable marriage problem with forbidden pairs (solved by Dias *et al.* with an $O(m)$ algorithm in [2]) and the stable roommates problem with ties in [6]. Note that in our model we also allow that certain edges are "missing" from the underlying graph, that is, some agents might be unacceptable to one another. In particular, a stable matching might contain several agents that have no partners in the matching.

Our present problem, the super-stable matching problem with forbidden edges is known to be polynomial-time solvable. Fleiner *et al.* exhibited a reduction of this problem to 2-SAT [3]. Though the algorithm based on [3] (that applies known algorithms for the reduced problem) works faster than our presented algorithm, an advantage of our direct approach is that there is more hope to find structural results on (super-)stable matchings. Also, by generalizing Irving's algorithm, we can understand what are those crucial properties that allow the algorithm to work. For this reason, our goal here is not a "streamlined" and fast algorithm based on elaborate data structures but a polynomial-time procedure with a compact proof of correctness.

To formalize our problem, we define a *preference model* as a triple $(G, F, \mathcal{O})$, where $G = (V, E)$ is a finite graph, the set F of *forbidden edges* is a subset of the edge set E of G, and $\mathcal{O} = \{\leq_v : v \in V\}$, where $\leq_v$ is a partial order on the star $E(v)$ of v (that is, the set of those edges of G that are incident with vertex v). It is convenient to think that we deal with a market situation: vertices of G are the acting agents and edges of G represent possible partnerships between them. Parallel

edges are allowed in G: the same two agents may form different partnerships, that may yield different profits for them. (For example agents u and v may play tennis or chess and their preferences on these two games might be different.) Partial order $<_v$ is the preference order of agent v on his possible partnerships. Our convention is that partnership e is preferred to partnership f by agent v if $e <_v f$. In this situation, we say that v prefers e to f or e dominates f at v. A subset M of E is a *matching* if edges of M do not share a vertex, that is, each agent participates in at most one partnership. Matching M is *stable* (we omit the 'super' prefix for convenience), if $M \subseteq E \setminus F$ (in other words, no edge of M is forbidden, that is, all edges of M are *free*), and each edge e of E is *dominated by* M, that is, there is an edge $m \in M$ and a vertex $v \in V$ such that $m \leq_v e$. (This means that either $e \in M$ and then because of $m = e$, $m \leq_v e$ holds for any vertex v of e or $e \notin M$ and then there must be an edge m of M that is better: $m <_v e$. In particular, m and e cannot be incomparable in $\leq_v$.) If M is a matching and e is not dominated by M then e is a *blocking edge of* M. The *stable roommates problem with partial orders and forbidden pairs* is the search problem on an input preference model to find a stable matching or to report that no such matching exists. Note that the stable roommates problem with partial orders and forbidden edges can be reduced to the stable roommates problem with partial orders (and no forbidden edges) the following way. For each forbidden edge e, introduce a copy e' parallel to e into the model such that e and e' are equivalent (and incomparable) for vertices of e. Furthermore, in this new model, declare all edges to be free. It is easy to check that the set of stable matchings in the new model is the same as the set of stable matchings (containing no forbidden edges) in the original model.

Note that in the standard terminology, agents have preferences on possible partners, rather than on partnerships. It is easy to see that in our approach, this corresponds to the case where the underlying graph G in the preference model is simple. We also have a slightly different way of defining stability via dominance. Traditionally, we first define the notion of blocking and then we say that a stable matching is a matching that has no blocking edge. Also note that the stable roommates problem [4, 5] is the special case where G is simple, $F = \emptyset$, and each order $<_v$ is linear.

2 The generalized algorithm

Let us fix a preference model $(G_0, F_0, \mathcal{O}_0)$, as the input of our algorithm. We should find a stable matching, if it exists. The algorithm shall handle so-called 1-arcs and 2-arcs that are oriented versions of certain edges of the underlying model. The sets of these arcs after the ith step of the algorithm are denoted by A_i^1 and A_i^2, respectively. In the beginning, $A_0^1 = A_0^2 = \emptyset$. The algorithm works step by step. In the $(i+1)$st step, it changes the actual instance $(G_i, F_i, A_i^1, A_i^2, \mathcal{O}_i)$ into a "simpler" model $(G_{i+1}, F_{i+1}, A_{i+1}^1, A_{i+1}^2, \mathcal{O}_{i+1})$ in such a way that the answer to the latter problem is also a valid answer to the former one. That is,

$$\begin{gathered}\text{any stable matching in } (G_{i+1}, F_{i+1}, \mathcal{O}_{i+1}) \\ \text{is a stable matching in } (G_i, F_i, \mathcal{O}_i) \text{ and}\end{gathered} \tag{1}$$

and

$$\begin{gathered}\text{if there is a stable matching in } (G_i, F_i, \mathcal{O}_i) \text{ then} \\ \text{there has to be a stable matching in } (G_{i+1}, F_{i+1}, \mathcal{O}_{i+1}) \text{ as well.}\end{gathered} \tag{2}$$

We employ four different kinds of transformations to achieve this goal: we find 1-arcs and 2-arcs, we forbid edges and we delete forbidden edges.

To describe these transformations, we need a couple of definitions. We say that edge $e \in E_i(v)$ of G_i (forbidden or not) is a *first choice edge of* v, if there is no edge $f \in E_i(v) \setminus F_i$ with $f <_v e$ (i.e., if no free edge dominates e at vertex v). Note that there can be more than one first choice of v present, moreover, an edge e can be a first choice of both of its vertices and if v is not an isolated vertex then there is at least one first choice of v. If $e = vu$ is a first choice of v then we may change our current instance $(G_i, F_i, A_i^1, A_i^2, \mathcal{O}_i)$ into $(G_{i+1}, F_{i+1}, A_{i+1}^1, A_{i+1}^2, \mathcal{O}_{i+1})$ where $G_{i+1} = G_i, E_{i+1} = E_i, A_{i+1}^1 = A_i^1 \cup \{(vu)\}, A_{i+1}^2 = A_i^2, \mathcal{O}_{i+1} = \mathcal{O}_i$ and we say that (vu) is a *1-arc*. This 1-arc finding transformation clearly satisfies conditions (1) and (2).

An edge $e \in E_i(v)$ is a *second choice of* v if e is not a first choice of v and $e >_v f \notin F_i$ implies that f is a first choice of v. In other words, e is a second choice if any free edge that dominates e at v is a first choice of v and there is at least one such free edge. Note that there can be several second choices of v present in an instance. Moreover, the set of second choices of v is nonempty if and only if there exist two free edges incident to v such that one dominates the other at v. If $e = vu$ is a second choice of v then we may change our current instance $(G_i, F_i, A_i^1, A_i^2, \mathcal{O}_i)$ into $(G_{i+1}, F_{i+1}, A_{i+1}^1, A_{i+1}^2, \mathcal{O}_{i+1})$ where $G_{i+1} = G_i, E_{i+1} = E_i, A_{i+1}^1 = A_i^1, A_{i+1}^2 = A_i^2 \cup \{(uv)\}, \mathcal{O}_{i+1} = \mathcal{O}_i$ and we say that (uv) is a *2-arc*. This 2-arc finding transformation clearly satisfies conditions (1) and (2). Note that the definition of a 2-arc is somewhat counterintuitive: unlike in case of a 1-arc, a 2-arc points to that vertex whose second choice it represents. Later we shall see the reason for this. For each j, we require the following property after the jth step of our algorithm.

$$\text{Each arc of } A_j^1 \text{ is a 1-arc of } (G_j, F_j, \mathcal{O}_j) \text{ and} \tag{3}$$

$$\text{each arc of } A_j^2 \text{ is a 2-arc of } (G_j, F_j, \mathcal{O}_j). \tag{4}$$

Clearly, 1-arc finding and 2-arc finding steps do not violate conditions (3) and (4).

If e is a free edge of G_i, then *forbidding* e means that $G_{i+1} := G_i$, $F_{i+1} := F_i \cup \{e\}$, and $\mathcal{O}_{i+1} := \mathcal{O}_i$. After forbidding, $A_{i+1}^1 = A_i^1$ and $A_{i+1}^2 = A_i^2$, unless we explicitly state otherwise. The algorithm may forbid e if either no stable matching contains e or if e is not contained in all stable matchings of $(G_i, F_i, \mathcal{O}_i)$. (Note that neither of these conditions implies the other.) Such a forbidding transformation clearly satisfies (1) and (2). Forbidding a subset C of E means that we simultaneously forbid all edges of C. We shall do so if properties (1) and (2) hold for $j = i + 1$.

If e is a forbidden edge of G_i then *deleting* e means that we delete e from G_i and F_i to get G_{i+1}: $E_{i+1} := E_i \setminus \{e\}$, $F_{i+1} := F_i \setminus \{e\}$, $A_{i+1}^1 := A_i^1 \setminus \{a \in A_i^1 : a = \vec{e}\}$, $A_{i+1}^2 := A_i^2 \setminus \{a \in A_i^2 : a = \vec{e}\}$ (where $a = \vec{e}$ means that 1-arc or 2-arc a is coming from first or second choice e) and the partial orders in $\mathcal{O}_{i+1}$ are the restrictions of the corresponding partial orders of $\mathcal{O}_i$, to the corresponding stars of G_{i+1}. The algorithm may delete forbidden edge e if there exists no matching in $(G_i, F_i, \mathcal{O}_i)$ that is blocked exclusively by e. This implies that the set of stable matchings in $(G_i, F_i, \mathcal{O}_i)$ and in $(G_{i+1}, F_{i+1}, \mathcal{O}_{i+1})$ is the same, so (1) and (2) clearly hold for $j = i + 1$. As first and second choices do not change after deleting a forbidden edge, properties (3) and (4) are true for $j = i + 1$.

As we mentioned already, our algorithm works in steps and in each step it changes the instance according to some of the above transformations. There is a certain hierarchy between these steps: the current move of the algorithm is always chosen to have the highest priority among the executable steps. Our description of the step types is in the order of this hierarchy.

0th priority (proposal) step If edge $e = vw$ is a first choice of v and does not belong to A_i^1 then find 1-arc vw, that is $A_{i+1}^1 = A_i^1 \cup \{(vw)\}$.

We have seen that conditions (1) and (2) are satisfied after a proposal step and by definition, (3) and (4) also hold for $j = i + 1$. As soon as the algorithm has found all 1-arcs, it looks for a

1st priority (mild rejection) step If $\vec{e} = uv$ is a 1-arc of A_i^1, $A_i^2 = \emptyset$ and $E_i(v) \ni f \not\prec_v e$ (that is, f is not strictly better than e according to v in G_i) then forbid f.

Obviously, if f belongs to some matching M then $e \notin M$, and hence e (being a first choice at u) blocks M. So f does not belong to any stable matching, hence we can safely forbid it. Clearly, any first choice remains a first choice after forbidding edge e, hence (3) remains true for $i+1$. Moreover, after forbidding e a second choice either remains as a second choice or it becomes a first choice. Consequently, for $j = i + 1$, properties (3) and (4) are true with the default choice $A_{i+1}^1 = A_i^1$ and $A_{i+1}^2 = \emptyset$.

Eventually, the algorithm deletes certain forbidden edges in the following way.

2nd priority (firm rejection) step If $e = uv$ is a free 1-arc of A_i^1 and $e <_v f \in E_i(v)$ (e is strictly better than f according to v in G_i) then we delete f.

Note that the above f is already forbidden by a 1st priority mild rejection step. Assume that f blocks matching M, hence, in particular, $e \notin M$. But e, being a first choice of u, also blocks M. So deleting f does not change the set of stable matchings of the preference model.

Note that the so called 1st phase steps in Irving's algorithm [5] for the stable roommates problem are special cases of our proposal and firm rejection steps. It is true for the stable roommates problem that as soon as no more 1st phase steps can be executed, the preference model has the so called first-last property: if some edge $e = uv$ is a first choice of u, then e is the last choice of v. The next lemma shows that generalization of this property holds also in our setting. Assume that the algorithm cannot execute a 0th, 1st or 2nd priority step for $(G_i, F_i, A_i^1, A_i^2, \mathcal{O}_i)$. Let V_i^0 denote the set of those vertices of G_i that are not incident with any free edges, V_i^1 stand for the set of those vertices of G_i that are incident with a bioriented free 1-arc and V_i^2 refer to the set of the remaining vertices of G_i. The following properties are true.

Lemma 2.1. *Assume that no proposal or rejection step is possible in instance $(G_i, F_i, A_i^1, A_i^2, \mathcal{O}_i)$ and let V_i^0, V_i^1 and V_i^2 be defined as above.*

If $v \in V_i^1 \cup V_i^2$ then there is a unique 1-arc entering v and there is a unique 1-arc leaving v and both of these 1-arcs are free. There is no edge of G_i that leaves V_i^0. Bioriented free 1-arcs form a matching M_1 that covers V_i^1 and no more edges are incident with V_i^1 in G_i.

M is a stable matching of $(G_i, F_i, \mathcal{O}_i)$ if and only if the following properties hold: (1) each vertex of V_i^0 is isolated and (2) $M_1 \subseteq M$ and (3) $M \setminus M_1$ is a stable matching of the model restricted to V_i^2.

Proof. Let $v \in V_i^1 \cup V_i^2$. By definition, there is at least one free edge incident with v, hence there is at least one free 1-arc leaving v. On the other hand, no proposal or rejection step (mild or firm) can be made in G_i, hence at most one free 1-arc enters v. By definition, no free 1-arc enters any vertex of V_i^0, and this means that 1-arcs leaving vertices of $V_i^1 \cup V_i^2$ enter this very same vertex set. Consequently, there is a unique free 1-arc leaving and entering each vertex of $V_i^1 \cup V_i^2$. Can there be a forbidden 1-arc e incident with a vertex v of $V_i^1 \cup V_i^2$? The answer is no and we prove it indirectly. Assume that $\vec{e}$ is such a 1-arc. If $\vec{e}$ enters v then v would be able to reject, a contradiction. So $\vec{e} = (vw)$ is a 1-arc of A_i^1 from $V_i^1 \cup V_i^2$ to V_i^0. But w is not incident with any free arcs by definition, thus (vu) is also a 1-arc of A_i^1 that enters vertex u of $V_i^1 \cup V_i^2$, contradiction again. Hence each 1-arc of A_i^1 incident with $V_i^1 \cup V_i^2$ is free.

Let $u \in V_i^0$ and $e = uv$ be an edge of G_i. Clearly $\vec{e} = (uv)$ is a 1-arc and $\vec{e} \in F_i$ by the definition of V_i^0, so $v \in V_i^0$ holds. This means that each edge of G_i incident with a vertex of V_i^0 is completely inside V_i^0.

If v is in V_i^1 then there is a unique 1-arc a that leaves v, so a must be bioriented by the definition of V_i^1. If $e = uv$ is an edge of G_i then either e is the unoriented version of a or e is not a first choice of v, hence $a <_v e$ holds. But in this latter case v should delete e in a firm rejection step as a is a 1-arc entering v. This argument shows that edges of G_i that are incident with V_i^1 are all bioriented and form a matching M_1 covering V_i^1.

Assume now that M is a stable matching of G_i. No edge of G_i incident with a vertex of V_i^0 can block M, hence V_i^0 must consist of isolated vertices. As M is not blocked by an edge of M_1, edges of M_1 all belong to M. As there is no edge of G_i that leaves V_i^2, edges of M in V_i^2 form a stable matching of the restricted model to V_i^2.

Let now M_2 be a stable matching of the model restricted to V_i^2 and assume that V_i^0 consists of isolated vertices. Let $M := M_2 \cup M_1$. Clearly, M is a matching. If some edge e blocks M then e cannot be incident with V_i^0, as these vertices are isolated, and e cannot have a vertex in V_i^1 either, as vertices of V_i^1 are only incident with edges of M_1. Hence e is an edge within V_i^2, contradicting to the fact that M_2 is a stable matching of the model restricted to V_i^2. □

Lemma 2.1 shows that as soon as we have a (forbidden) edge incident with some vertex of V_i^0 for an instance where no proposal or rejection step is possible then there exists no stable matching in our instance, so the algorithm can stop with the conclusion that in the original instance there is no stable matching whatsoever. Another possible conclusion of the algorithm is that eventually no proposal or rejection step can be made and $V_i^2 = \emptyset$ holds. In this case, if V_i^0 consists of isolated vertices then graph G_i is just matching M_1 and this is a stable matching for the instance after the ith step, hence it is also a stable matching for the original instance. So our goal from now on is to get rid off the V^2 part and to achieve this, the algorithm will work only on V_i^2.

Assume that in instance $(G_i, F_i, A_i^1, A_i^2, \mathcal{O}_i)$, the algorithm can execute no 0th, 1st or 2nd priority step. By Lemma 2.1, every vertex v of V_i^2 is incident with at least one free second choice edge: in the "worst case" it is the unique 1-arc pointing to v.

3rd priority (2-arc finding) step If $e = vw \notin A_i^2$ is a second choice of v then find 2-arc wv.

What is the meaning of a 2-arc? Let vv' and uu' be 1-arcs and $u'v$ be a 2-arc. As vv' is the only free edge dominating $u'v$ at v, we get that if uu' is present in a stable matching M then uu' does not dominate uv', hence $vv' \in M$ follows. In other words, 2-arcs represent implications on 1-arcs. This allows us to build an implication structure on the set of 1-arcs.

In this structure, two 1-arcs e and f are called *sm-equivalent*, if there is a directed cycle D formed by 1-arcs and 2-arcs in an alternating manner such that D contains both e and f. (Note that D may use the same vertex more than once.) Sm-equivalence is clearly an equivalence relation and if C is an sm-class and M is a stable matching then either C is disjoint from M or C is contained in M.

Beyond determining sm-equivalence classes, 2-arcs yield further implications between sm-classes: if uu' is a 1-arc of sm-class C and vv' is a 1-arc of sm-class C' and $u'v$ is a 2-arc, then sm-class C "implies" sm-class C' in such a way that if C is not disjoint from stable matching M then M contains both classes C and C'. Assume that sm-class C is on the top of this implication structure, i.e. C is not implied by

any other sm-class (but C may imply certain other classes). Formally, we have that

$$\begin{aligned} &\text{if } vv' \text{ is a 1-arc of } C \text{ and } w'v \text{ is a 2-arc} \\ &\text{then (the unique) 1-arc } ww' \text{ is sm-equivalent to } vv'. \end{aligned} \tag{5}$$

To find a top sm-class C, introduce an auxiliary digraph on the vertices of G_i, such that if uu' is a 1-arc and $u'v$ is a 2-arc, then we introduce an arc uv of the auxiliary graph. It is well known that by depth first search, we can find a source strong component of the auxiliary graph in linear time. If it contains vertices $u_1, u_2, \ldots, u_k$ then it determines a top sm-class $C = \{u_1u'_1, u_2u'_2, \ldots, u_ku'_k\}$ formed by 1-arcs. Note that it is possible here that $u_l = u'_t$ for different l and t. After we have found all 2-arcs (and there are no proposal or rejection steps) in instance $(G_i, F_i, A_i^1, A_i^2, \mathcal{O}_i)$ then the algorithm looks for a

4th priority (2-arc elimination) step If for 1-arcs $u_lu'_l, u_tu'_t \in C$ there are 2-arcs vu_l and vu_t with $vu_l \not\leq_v vu_t$ then forbid vu_l and keep 1-arcs and 2-arcs: $A_{i+1}^1 = A_i^1$ and $A_{i+1}^2 = A_i^2$.

To justify this step, assume that $vu_l \in M$ for some stable matching M of G_i. As vu_l does not dominate vu_t, vu_t has to be dominated at u_t by $u_tu'_t \in M$. As $u_lu'_l$ and $u_tu'_t$ are sm-equivalent, this means that $u_tu'_t$ also belongs to M, a contradiction. So vu_l does not belong to any stable matching and after forbidding it, the set of stable matchings does not change. This proves (1) and (2). As the forbidden edge vu_l is a second choice of u_l and not $\leq_v$-better than vu_t, vu_l is a first choice of neither v nor u_l. Consequently, after forbidding vu_l, first and second choices remain first and second choices, respectively. It follows that a 4th priority step preserves conditions (3) and (4). Note that though a 4th priority step does not change first choices, it may create new second choices hence the algorithm might continue with a 3rd priority step after executing a 4th priority one. Note also that if preferences are linear (rather than partial) orders then no 4th priority step is possible.

If none of the above steps is possible any more then the top sm-equivalence class C can be forbidden. This is the step that corresponds to the 'rotation elimination' step in Irving's algorithm. Note that by the impossibility of a 4th priority step, any top sm-equivalence class $C = \{(u_lu'_l) : 1 \leq l \leq k\}$ has the property that there is exactly one 2-arc entering each u_l, that is, there is a unique second choice of each vertex u_l.

5th priority (top class elimination) step Forbid all edges of C in $(G_i, F_i, \mathcal{O}_i)$ and set $A_{i+1}^2 = \emptyset$.

As we forbid 1-arcs, first and second choices along the vertices of C change after a 5th priority step. In particular, the unique second choices of the u_l vertices of C become first choices. For this reason we change $A_{i+1}^1 = A_i^1 \cup S^{-1}$, where S denotes the set of those 2-arcs that enter some vertex u_l of C and S^{-1} is the set of oppositely oriented arcs of S. After these changes, all arcs in A_{i+1}^1 are clearly first choices of their initial nodes, hence (3) and (4) hold for $j = i + 1$. To justify properties (1) and (2) for the 5th priority step, we distinguish two cases.

Case 1: C is not a matching. This means that $u_l = u'_t$ for some $l \neq t$. As a subset of a matching is a matching, no matching (hence no stable matching) can contain C. So by sm-equivalence, C is disjoint from any stable matching of G_i, and forbidding C does not change the set of stable matchings.

Case 2: C is a matching. Each u_l is adjacent to at least two free edges: the incoming and the outgoing 1-arcs. So each u_l receives at least one free 2-arc. This free 2-arc must come from some u'_t by property (5). Let C' denote the set of free 2-arcs of the form u'_tu_l. As we have seen, each u_l receives at least one arc of C', hence $|C'| \geq k$. As we cannot execute any more 4th priority steps in $(G_i, F_i, \mathcal{O}_i)$, from each u'_t there is at most one arc of C' leaving, implying $|C'| \leq k$. This means that $|C'| = k$ and each u_l receives exactly one arc of C' and each u'_l sends exactly

one arc of C'. As sets $\{u_1, u_2, \ldots, u_k\}$ and $\{u'_1, u'_2, \ldots, u'_k\}$ are disjoint, this means that set C' forms a perfect matching on vertices $u_1, u'_1, u_2, u'_2, \ldots, u_k, u'_k$.

Let M be a stable matching of $(G_i, F_i, \mathcal{O}_i)$. If M is disjoint from C then M is stable in $(G_{i+1}, F_{i+1}, \mathcal{O}_{i+1})$ as well. Otherwise, by sm-equivalence, M contains all edges of C and disjoint from C'. We claim that $M' := M \setminus C \cup C'$ is another stable matching of $(G_i, F_i, \mathcal{O}_i)$ and hence it is a stable matching of $(G_{i+1}, F_{i+1}, \mathcal{O}_{i+1})$, as well.

Indeed: M' is a matching, as C and C' cover the same set of vertices. Each edge $u_l u'_l$ is dominated at u'_l by M' by Lemma 2.1. Each forbidden 2-arc of type $u'_t u_l$ is dominated at u'_t by the 4th priority step. For the remaining edges, if some edge e does not have a vertex u_l then e is dominated the same way in M' as in M. Otherwise, if u_l is a vertex of e then e is neither a first nor a second choice of u_l as we have already checked these edges. This means that the free 2-arc pointing to u_l is dominating e, so C' and thus M' also dominates e at u_l.

The pseudocode below summarizes how our algorithm works. The organization of the steps is justified by the fact that a firm rejection step always deletes a forbidden edge, hence no new first choice is created. Similarly, 2-arc finding and 2-arc elimination steps do not change the set of first choices and preserve properties described in Lemma 2.1.

Input: $(G_0, F_0, \mathcal{O}_0)$ **Output**: Super-stable matching, if exists
$A_0^1 := A_0^2 := \emptyset$, $i := 0$
1 **IF** there is a first choice uv of u that is not a 1-arc
THEN $(G_{i+1}, F_{i+1}, \mathcal{O}_{i+1}, A_{i+1}^1, A_{i+1}^2) := (G_i, F_i, \mathcal{O}_i, A_i^1 \cup \{uv\}, A_i^2)$,
$i := i + 1$, **GO TO 1**
ELSE
2 **IF** mild rejection is possible for some edge uv of G_i
THEN $(G_{i+1}, F_{i+1}, \mathcal{O}_{i+1}, A_{i+1}^1, A_{i+1}^2) := (G_i, F_i \cup \{uv\}, \mathcal{O}_i, A_i^1, A_i^2)$,
$i := i + 1$, **GO TO 1**
ELSE
3 **IF** firm rejection is possible for some edge uv of G_i
THEN $(G_{i+1}, F_{i+1}, \mathcal{O}_{i+1}, A_{i+1}^1, A_{i+1}^2) :=$
$:= (G_i - \{uv\}, F_i \setminus \{uv\}, \mathcal{O}_i|_{G_{i+1}}, A_i^1 \setminus \{uv\}, A_i^2) \setminus \{uv\}$,
$i := i + 1$, **GO TO 3**
ELSE
4 **IF** there is a second choice uv of u that is not a 2-arc
THEN $(G_{i+1}, F_{i+1}, \mathcal{O}_{i+1}, A_{i+1}^1, A_{i+1}^2) := (G_i, F_i, \mathcal{O}_i, A_i^1, A_i^2 \cup \{vu\})$,
$i := i + 1$, **GO TO 4**
ELSE
5 **IF** some 2-arc $uv \in A_i^2$ can be eliminated
THEN $(G_{i+1}, F_{i+1}, \mathcal{O}_{i+1}, A_{i+1}^1, A_{i+1}^2) := (G_i, F_i \cup \{uv\}, \mathcal{O}_i, A_i^1, A_i^2)$,
$i := i + 1$, **GO TO 4**
ELSE
6 **IF** some sm-equivalence class C can be eliminated
THEN $(G_{i+1}, F_{i+1}, \mathcal{O}_{i+1}, A_{i+1}^1, A_{i+1}^2) :=$
$(G_i - C, F_i \setminus C, \mathcal{O}_i, A_i^1 \cup S^{-1}, \emptyset)$, $i := i + 1$, **GO TO 1**
ELSE
7 **IF** each vertex of V_i^0 is isolated
THEN **OUTPUT** super-stable matching E_i
ELSE **OUTPUT** "No super-stable matching exists"
END IF
END IF
END IF
END IF
END IF
END IF
STOP

The following theorem justifies the correctness of our algorithm.

Theorem 2.2. *Assume that the algorithm cannot execute any more step at some instance* $(G_i, F_i, A_i^1, A_i^2, \mathcal{O}_i)$. *Then* $V_i^2 = \emptyset$.

Proof. Assume indirectly that v is a vertex of V_i^2, so by Lemma 2.1, v sends a free 1-arc, and also receives a free 1-arc different from the opposite of the previous one. It follows that there is a 2-arc pointing to v. This implies that a 4th or a 5th priority step can be executed, a contradiction. □

To finish the description of the algorithm, we should recall our earlier remark. By Theorem 2.2, when the algorithm terminates then we have $V_i^2 = \emptyset$, so by Lemma 2.1, if V_i^0 spans some edge then the conclusion is that there is no stable matching, otherwise, if each vertex of V_i^0 is isolated then there is a stable matching of the original instance, and the edge set E_i of G_i forms such a matching. The following theorem estimates the complexity of our algorithm.

Theorem 2.3. *Assume that preference model* $(G, F, \mathcal{O})$ *is such that* G *has* n *vertices and* m *edges we can decide for each edge* $e = uv$ *of* G *whether* e *is a first or second choice of* u *in constant time for any preference model created from* $(G, F, \mathcal{O})$ *after forbidding and deleting edges. The algorithm we described above finds a stable matching or concludes that no stable matching exists in* $O(m \cdot (n + m))$ *time.*

Proof. We have seen that if the algorithm terminates then it has the right answer, so we only need to prove that the running time is $O(m \cdot (n + m))$. As we have seen, in each step, the algorithm changes the current instance by changing the set of 1-arcs or 2-arcs or by forbidding or deleting certain edges. Let us call the latter two transformations *major events*. Clearly, during the course of the algorithm there can be at most $2m$ major events as there are m edges that can be forbidden or eventually deleted. We show that between two consecutive major events the algorithm needs $O(n + m)$ time.

If a major event is a 1st priority (mild rejection) step, then previously we had to find all 1-arcs (in $O(n + m)$ time) and finding the forbidden edge after this can be done in $O(n + m)$ time again. If the major event is a 2nd priority (firm rejection) step then it is preceded by 0th priority proposal steps (taking $O(n+m)$ time again) and checks for 1st priority (mild rejection) steps taking $O(n + m)$ time. We need again $O(n+m)$ time to find the edge to be deleted in the 2nd priority (firm rejection) step.

The next major event type is a 4th priority (2-arc eliminating) step. It is preceded by executing all 0th priority (proposal) steps and checking for 1st and 2nd priority steps that take altogether $O(n+m)$ time. Then we find all 2-arcs in $O(n+m)$ time, find top sm-class C by depth first search in $O(n+m)$ time and find the deleted edge $O(n + m)$ time again.

The remaining major event happens in a 5th priority step. So after the previous major event we had at most $O(n)$ 0th priority proposal steps that take $O(n + m)$ time, checks for 1st and 2nd priority rejection steps taking $O(n + m)$ time, we had to find all 2-arcs in $O(n + m)$ time, we find top sm-class C in $O(n + m)$ time and check for 2-arc elimination in $O(n + m)$ time again.

The above estimates prove that there is $O(n + m)$ time between consecutive major events. We have seen that there are $O(m)$ major events, so our algorithm terminates in $O(m \cdot (n + m))$ time, just as we claimed in the theorem. □

The time complexity in Theorem 2.3 is pretty rough. This is partly due to the fact that in 5th priority (top class elimination) steps we throw away all 2-arcs in spite of the fact that most of them can be reused. Probably by paying more attention to the changes of second choices and by using more appropriate data

structures one can streamline the algorithm to approach the complexity of Irving's original algorithm described in [5]. As we mentioned, our goal was not a competitive algorithm but a description of a polynomial-time method with a compact proof of correctness that gives hope to find further structural results on stable matchings. We think that this goal is achieved.

References

[1] Ron Aharoni, and Tamás Fleiner, On a lemma of Scarf, *J. Combin. Theory Ser. B* (2003) **87**(1) 72–80

[2] Vânia M. F. Dias, Guilherme D. da Fonseca, Celina M. H. de Figueiredo, and Jayme L. Szwarcfiter, The stable marriage problem with restricted pairs, *Theoret. Comput. Sci.* (2003) **306**(1-3) 391–405

[3] Tamás Fleiner, Robert W. Irving, and David F. Manlove, Efficient algorithms for generalised stable marriage and roommates problems, *Theoret. Comput. Sci.* (2007) **381**(1-3) 162–176 and DCS Tech Report, TR-2005-207, `http://www.dcs.gla.ac.uk/publications/` (2005)

[4] D. Gale and L.S. Shapley, College admissions and stability of marriage, *Amer. Math. Monthly* (1962) **69**(1) 9–15

[5] Robert W. Irving, An efficient algorithm for the "stable roommates" problem, *J. Algorithms* (1985) **6**(4) 577-595

[6] Robert W. Irving and David F. Manlove, The stable roommates problem with ties, *J. Algorithms* (2002) **43**(1) 85–105

[7] Eytan Ronn, NP-complete stable matching problems, *J. Algorithms* (1990) **11**(2) 285–304

[8] Sandy Scott, The study of stable marriage problems with ties, 2005, PhD dissertation, University of Glasgow, Department of Computing Science.

[9] Jimmy J. M. Tan, A necessary and sufficient condition for the existence of a complete stable matching, *J. Algorithms* (1991) **12**(1) 154–178