The Hospitals/Residents Problem with Ties

Robert W. Irving[1], David F. Manlove[1]* and Sandy Scott[2]

[1] Dept. of Computing Science, University of Glasgow, Glasgow G12 8QQ, Scotland
Email: {rwi,davidm}@dcs.gla.ac.uk
[2] Dept. of Mathematics, University of Glasgow, Glasgow G12 8QQ, Scotland
Email: ssc@maths.gla.ac.uk

Abstract. The hospitals/residents problem is an extensively-studied many-one stable matching problem. Here, we consider the hospitals/residents problem where ties are allowed in the preference lists. In this extended setting, a number of natural definitions for a stable matching arise. We present the first linear-time algorithm for the problem under the strongest of these criteria, so-called *super-stability.* Our new results have applications to large-scale matching schemes, such as the National Resident Matching Program in the US, and similar schemes elsewhere.

1 Introduction

The Hospitals/Residents problem (HR) [4, 14] is a many-one stable matching problem which is so-named because of its application to large-scale matching schemes, such as the National Resident Matching Program in the US [12], the Canadian Resident Matching Service [1], and the Scottish Pre-registration house officer Allocations (SPA) matching scheme [6]. Each of these centralised schemes administers the annual match of graduating medical students to hospital appointments in its respective country.

An instance of HR involves a set $\mathcal{R}$ of *residents* and a set $\mathcal{H}$ of *hospitals*, each resident $r \in \mathcal{R}$ seeking a post at one hospital, and each hospital $h \in \mathcal{H}$ having $q(h) \geq 1$ posts. Each resident in $\mathcal{R}$ ranks a subset of $\mathcal{H}$ in strict order, and each hospital $h \in \mathcal{H}$ ranks its applicants in strict order. An agent $p \in \mathcal{R} \cup \mathcal{H}$ finds an agent $q \in \mathcal{R} \cup \mathcal{H}$ *acceptable* if q appears on p's preference list; p finds q *unacceptable* otherwise. A *matching* M is a subset of $\mathcal{R} \times \mathcal{H}$, where $(r, h) \in M$ implies that (i) r, h find each other acceptable, (ii) r is assigned to at most one hospital in M, and (iii) at most $q(h)$ residents are assigned to h in M. A matching M for an instance of HR is *stable* if M admits no *blocking pair.* A blocking pair (r, h) for M is a resident r and hospital h such that (i) r, h find each other acceptable, (ii) r either is unassigned or prefers h to his assigned hospital in M, and (iii) h either is undersubscribed or prefers r to the worst resident assigned to it in M. If (r, h) form a blocking pair with respect to a matching M, then (r, h) is said to *block* M. Also, if $(r, h) \in M$ for some stable matching M, then we say

* Supported by Engineering and Physical Sciences Research Council grant number GR/M13329.

that (r, h) is a *stable pair*, and r is a *stable partner* of h (and vice versa). Note that, in view of the definitions of a matching and a blocking pair, we assume throughout this paper, without loss of generality, that an agent p finds an agent q acceptable if and only if q finds p acceptable. We say that the preference list of a resident $r \in \mathcal{R}$ (resp. hospital $h \in \mathcal{H}$) is *complete* if r (resp. h) finds all hospitals in $\mathcal{H}$ (resp. residents in $\mathcal{R}$) acceptable.

The classical Stable Marriage problem (SM) [4, 14, 8] is a restriction of HR in which each hospital has exactly one post, the number of hospitals equals the number of residents, and all preference lists are complete. For a given instance I of HR, the Gale/Shapley algorithm for SM [2] may be extended in order to find a stable matching for I (such a matching in I always exists) in $O(mn)$ time, where $n = |\mathcal{R}|$ and $m = |\mathcal{H}|$ [4, Section 1.6.3]. The Gale/Shapley algorithm incorporates a sequence of proposals from one set of agents to the other; if the residents propose to the hospitals (the *resident-oriented algorithm*), then we obtain a stable matching M which is uniquely favourable to the residents: every resident assigned in M is assigned to his best stable partner, and every resident unassigned in M is unassigned in any stable matching [4, Section 1.6.3]. Analogously, if the hospitals propose to the residents (the *hospital-oriented algorithm*), then we obtain a stable matching M which is uniquely favourable to the hospitals: every hospital $h \in \mathcal{H}$ is assigned either its $q(h)$ best stable partners, or a set of fewer than $q(h)$ residents; in the latter case, no other resident is assigned to h in any stable matching [4, Section 1.6.2].

Although an instance of HR may admit more than one stable matching, every stable matching has the same size, matches exactly the same set of residents, and fills exactly the same number of posts at each hospital; indeed any hospital that is undersubscribed in one stable matching is assigned exactly the same set of residents in all stable matchings. (These results are collectively known as the 'Rural Hospitals Theorem' [12, 3, 13].)

Ties in the preference lists. A natural generalisation of HR occurs when each agent's preference list need not be strictly ordered, but may include ties – we refer to this extension as the Hospitals/Residents problem with Ties (HRT). When ties are permitted, more than one definition of stability is possible [5].

According to the weakest of these stability notions, a matching M is *weakly stable* [5] if M admits no blocking pair[3], where a blocking pair (r, h) for M is a resident r and hospital h such that (i) r, h find each other acceptable, (ii) r either is unassigned or strictly prefers h to his assigned hospital in M, and (iii) h either is undersubscribed or strictly prefers r to the worst resident assigned to it in M. Given an instance I of HRT, the existence of a weakly stable matching is guaranteed: by breaking the ties in I arbitrarily, we obtain an instance I' of HR, and clearly a stable matching in I' is weakly stable in I. Indeed, a converse of sorts holds, giving the following proposition, whose proof is straightforward and is omitted.

[3] Note that throughout this paper, the form of stability to which the term *blocking pair* refers should be clear from the context.

Proposition 1. *Let I be an instance of HRT, and let M be a matching in I. Then M is weakly stable in I if and only if M is stable in some instance I' of HR obtained from I by breaking the ties in I in some way.*

However, the weakly stable matchings in I may be of different cardinality, and each of the problems of finding the maximum or minimum size of weakly stable matching in an HRT instance is NP-hard, though approximable within a factor of 2 [7, 10].

A stronger form of stability may be defined as follows: a matching M is *super-stable* [5] if M admits no blocking pair, where a blocking pair (r, h) for M is a resident r and hospital h such that (i) $(r, h) \notin M$, (ii) r, h find each other acceptable, (iii) r either is unassigned or strictly prefers h to his assigned hospital in M or is indifferent between them, and (iv) h either is undersubscribed or strictly prefers r to the worst resident assigned to it in M or is indifferent between them. Clearly a super-stable matching is weakly stable. Additionally, the super-stability definition gives rise to the following analogue of Proposition 1 (again, the proof is straightforward and is omitted):

Proposition 2. *Let I be an instance of HRT, and let M be a matching in I. Then M is super-stable in I if and only if M is stable in every instance I' of HR obtained from I by breaking the ties in I in some way.*

It should be clear that an instance I of HRT may not admit a super-stable matching: as a simple example, suppose that each hospital has just one post, and every agent's list is a single tie of length 2. It is the purpose of this paper to present optimal $O(mn)$ algorithms – linear in the size of the problem instance – for determining whether a given instance of HRT admits a super-stable matching, and if it does, to construct such a matching. The first algorithm, presented in Section 2, is resident-oriented in that it involves a sequence of proposals from the residents to the hospitals, and has similar optimality implications for the residents to those of the resident-oriented algorithm for HR. Also in Section 2, we prove an analogue of the Rural Hospitals Theorem for HRT. The second algorithm, presented in Section 3, is the hospital-oriented version, incorporating proposals from the hospitals to the residents, with analogous optimality implications for the hospitals to those of the hospital-oriented algorithm for HR.

For space reasons, the majority of our attention is focused on the resident-oriented algorithm for HRT. It is this algorithm that is likely to be of more significance to implementors of large-scale matching schemes, since recent pressure from student bodies has ensured that all three matching schemes mentioned above essentially employ the resident-oriented algorithm for HR.

Applications. Note that permitting ties in the preference lists has important practical applications. In the context of centralised matching schemes, some participating hospitals with many applicants have found the task of producing a strictly ordered preference list difficult, and they have expressed a desire to include ties in their lists. In such a setting, choosing the weak stability definition leads to two problems: (i) finding a weakly stable matching that matches as many residents as possible, and (ii) the possibility of, say, a resident r persuading, by some means, a hospital h to accept r at the expense of some allocated resident r', if h is indifferent between r and r'. The super-stability definition clearly avoids problem (ii), and additionally guards against problem (i), as is demonstrated by the following proposition, which is a consequence of Propositions 1 and 2, and the Rural Hospitals Theorem for HR.

Proposition 3. *Let I be an instance of HRT, and suppose that I admits a super-stable matching M. Then the Rural Hospitals Theorem holds for the set of weakly stable matchings in I.*

Thus Proposition 3 tells us that if a super-stable matching exists, then all weakly stable matchings are of the same size, and match exactly the same set of residents. Of course, as observed earlier, a super-stable matching need not exist. Nonetheless, it is arguable that a super-stable matching should be preferred by a practical matching scheme in cases when one does exist. In Section 4, we address the issue of the existence of super-stable matchings in an HRT instance.

Previous work. As mentioned above, optimal algorithms for constructing stable matchings in an instance of HR are known. For the case where ties are permitted, there is an optimal $O(n^2)$ algorithm, due to Irving [5], for determining whether a given (one-one) instance of Stable Marriage in which preference lists are complete but may incorporate ties (henceforth SMT) admits a super-stable matching, and for constructing one if it does, where n is the number of men and women. However, the problem of formulating such an algorithm for the (many-one) HRT case has remained open until now.

2 Resident-oriented algorithm for HRT

For a given instance of HRT, Algorithm HRT-Super-Res, shown in Figure 1, determines whether a super-stable matching exists, and if so will find such a matching. We shall describe informally the execution of Algorithm HRT-Super-Res. Before doing so, we make a number of definitions.

For a given instance I of HRT, suppose that $(r, h) \in M$ for some super-stable matching M. Then (r, h) is a *super-stable pair*, and r is a *super-stable partner* of h (and vice versa). The term *delete the pair* (r, h), implies that r, h are to be deleted from each other's preference lists. By the *head* of a resident's preference list, we mean the set of one or more hospitals, tied in his current list (i.e. his preference list after any deletions have been carried out), which he strictly prefers to all other hospitals in his list. Similarly, the *tail* of a hospital's list refers to the set of one or more residents, tied in its current list, to whom it strictly prefers all other residents in its list. By the term *reduced lists*, we mean the current lists at the termination of Algorithm HRT-Super-Res.

Algorithm HRT-Super-Res involves a sequence of proposals from the residents to the hospitals, in the spirit of the resident-oriented Gale/Shapley algorithm for HR. A resident proposes simultaneously to *all* hospitals at the head

```
assign each resident to be free;
assign each hospital to be totally unsubscribed;
for each hospital $h$ loop
    $full(h)$ := false;
end loop;
while some resident $r$ is free and has a nonempty list loop
    for each hospital $h$ at the head of $r$'s list loop
        provisionally assign $r$ to $h$;   {$r$ "proposes" to $h$}
        if $h$ is oversubscribed then                                        (†)
            for each resident $s'$ at the tail of $h$'s list loop
                if $s'$ is provisionally assigned to $h$ then
                    break the assignment;
                end if;
                delete the pair $(s', h)$;
            end loop;
        end if;
        if $h$ is full then                                                  (‡)
            $full(h)$ := true;
            $s$ := worst resident provisionally assigned to $h$;  {any one, if $> 1$}
            for each strict successor $s'$ of $s$ on $h$'s list loop
                delete the pair $(s', h)$;
            end loop;
        end if;
    end loop;
end loop;
if some resident is multiply assigned or
(some hospital $h$ is undersubscribed and $full(h)$) then
    no super-stable matching exists;
else
    the assignment relation is a super-stable matching;
end if;
```

Fig. 1. Algorithm HRT-Super-Res.

of his list, and all proposals are provisionally accepted. If a hospital h becomes oversubscribed, it turns out that none of h's worst-placed assignees (there must be more than one), nor any residents tied with these assignees in h's list, can be a super-stable partner of h – such pairs (r, h) are deleted. If a hospital h is full, then no resident strictly inferior than h's worst-placed assignee(s) can be a super-stable partner of h – again such pairs (r, h) are deleted. The proposal sequence terminates once every resident either is assigned to a hospital or has an empty list. At this point, it turns out that if a resident is assigned to more than one hospital, or some hospital is undersubscribed but was previously full, then no-super-stable matching exists. Otherwise, the assignment relation is a super-stable matching.

In order to establish the correctness of Algorithm HRT-Super-Res, a number of lemmas follow. The first three of these deal with the case that the assignment relation is claimed to be a super-stable matching. In what follows, I is an instance of HRT, in which $\mathcal{R}$ is the set of residents and $\mathcal{H}$ is the set of hospitals.

Lemma 1. *If, at the termination of the while loop of Algorithm HRT-Super-Res, the algorithm reports that the assignment relation M is a super-stable matching, then M is indeed a matching.*

Proof. Clearly, no hospital is oversubscribed in M. Also, no resident is multiply assigned in M, for otherwise the algorithm would have reported that no super-stable matching exists, a contradiction. □

Lemma 2. *If the pair (r, h) is deleted during an execution of Algorithm HRT-Super-Res, then that pair cannot block any matching generated by Algorithm HRT-Super-Res, comprising pairs that are never deleted.*

Proof. Let M be a matching generated by Algorithm HRT-Super-Res, comprising pairs that are never deleted, and suppose that (r, h) is deleted during execution of the algorithm. If h is full in M, then h strictly prefers its worst-placed assignee in M to r, since r is a strict successor of any undeleted entries in the reduced list of h. Hence (r, h) does not block M in this case. Now suppose that h is undersubscribed in M. As the pair (r, h) is deleted by the algorithm, then during some iteration of the while loop, h must have been full. Hence the algorithm would have reported that no super-stable matching exists rather than generating M, a contradiction. □

Lemma 3. *If, at the termination of the while loop of Algorithm HRT-Super-Res, the algorithm reports that the assignment relation M is a super-stable matching, then M is indeed a super-stable matching.*

Proof. By Lemma 1, the assignment relation M is a matching. Now suppose that M is blocked by some pair (r, h). Then r and h are acceptable to each other, so that each is on the original preference list of the other. By Lemma 2, the pair (r, h) has not been deleted. Hence each is on the reduced list of the other.

As the reduced list of r is nonempty, r is assigned to some hospital h' in M. Now $h' \neq h$, as (r, h) blocks M. If r strictly prefers h to h', then the pair (r, h) has been deleted, since h' is at the head of the reduced list of r, a contradiction. Thus r is indifferent between h and h', so that r proposed to h during the execution of the algorithm. Hence r is assigned to h in M, for otherwise the pair (r, h) would have been deleted, a contradiction. Thus (r, h) does not block M, a contradiction. □

The next lemma shows that Algorithm HRT-Super-Res will never delete a pair that could belong to some super-stable matching.

Lemma 4. *No super-stable pair is ever deleted during an execution of Algorithm HRT-Super-Res.*

Proof. Suppose, for a contradiction, that (r, h) is the first super-stable pair to be deleted during an execution of Algorithm HRT-Super-Res. Let M be a super-stable matching in I such that $(r, h) \in M$.

Case (i). Suppose that (r, h) is deleted as a result of h being oversubscribed. Consider the assignment relation G at point (†) in the same iteration of the while loop. At this point, some resident s is provisionally assigned to h in G, where $(s, h) \notin M$ and h strictly prefers s to r or is indifferent between them, since $(r, h) \in M$ and h cannot be oversubscribed in M. There is no super-stable matching in which s is assigned to a hospital h' which he strictly prefers to h. For otherwise, the super-stable pair (s, h') would have been deleted before (r, h), in order for s to propose to h, a contradiction. Thus either s is unassigned in M, or s is assigned to h' in M, where s strictly prefers h to h' or is indifferent between them. In any of these cases, (s, h) blocks M, a contradiction.

Case (ii). Suppose that (r, h) is deleted as a result of h being full. Consider the assignment relation G at point (‡) in the same iteration of the while loop. At this point, some resident s is provisionally assigned to h in G, where $(s, h) \notin M$ and h strictly prefers s to r, since $(r, h) \in M$ and r is not assigned to h in G. As in part (i), there is no super-stable matching in which s is assigned a hospital which he strictly prefers to h. Thus again, (s, h) blocks M, a contradiction. □

The next two lemmas deal with the case that Algorithm HRT-Super-Res claims the non-existence of a super-stable matching.

Lemma 5. *If, at the termination of the while loop of Algorithm HRT-Super-Res, some resident is multiply assigned, then I admits no super-stable matching.*

Proof. Let G be the assignment relation at the termination of the while loop. Suppose, for a contradiction, that there exists a super-stable matching M in I.

Firstly, we claim that some hospital must have fewer assignees in M than it has provisional assignees in G. For, suppose not. Let $p_G(h)$ denote the provisional assignees of hospital h in G, and let $p_M(h)$ denote the assignees of hospital h in M, for any $h \in \mathcal{H}$. Then by hypothesis,

$$\sum_{h \in \mathcal{H}} |p_M(h)| \geq \sum_{h \in \mathcal{H}} |p_G(h)|. \tag{1}$$

Now if some resident r is not provisionally assigned to a hospital in G, then the reduced list of r is empty, so that by Lemma 4, r is unassigned in any super-stable matching. Thus, letting R_1 denote the residents who are provisionally assigned to at least one hospital in G, and letting R_2 denote the residents who are assigned to a hospital in M, we have $|R_2| \leq |R_1|$. Hence

$$\sum_{h \in \mathcal{H}} |p_M(h)| = |R_2| \leq |R_1| < \sum_{h \in \mathcal{H}} |p_G(h)|$$

as some resident is multiply assigned in G, which contradicts Inequality 1. Thus the claim is established, so that some hospital h has fewer assignees in M than it has provisional assignees in G. Hence h is undersubscribed in M, since no hospital is oversubscribed in G. In particular, some resident r is assigned to h in G but not in M. Thus by Lemma 4, r cannot be assigned to a hospital in M which he strictly prefers to h. Hence (r, h) blocks M, a contradiction. □

Lemma 6. *If some hospital h became full during the while loop of Algorithm HRT-Super-Res, and h subsequently ends up undersubscribed at the termination of the while loop, then I admits no super-stable matching.*

Proof. Let G be the assignment relation at the termination of the while loop. Suppose, for a contradiction, that there exists a super-stable matching M in I. By Lemma 5, no resident is multiply assigned in G. Let h' be a hospital which became full during the while loop and subsequently ends up undersubscribed in G. Then there is some resident r' who was provisionally assigned to h' at some point during the while loop, but is not assigned to h' in G. Thus the pair (r', h') was deleted during some iteration of the while loop, so that $(r', h') \notin M$ by Lemma 4.

Now let $p_G(h), p_M(h), R_1, R_2$ be defined as in the proof of Lemma 5. Firstly, we claim that if any hospital h is undersubscribed in M, then every resident provisionally assigned to h in G is also assigned to h in M. For, if some resident r is assigned to h in G but not in M, then (r, h) blocks M, since h is undersubscribed in M, and by Lemma 4, r cannot be assigned to a hospital in M which he strictly prefers to h.

Secondly, we claim that each hospital has the same number of provisional assignees in G as it has assignees in M. For, by the first claim, any hospital that is full in G is also full in M, and any hospital that is undersubscribed in G fills as many places in M as it does in G. Hence $|p_M(h)| \geq |p_G(h)|$ for each $h \in \mathcal{H}$. As in the proof of Lemma 5, we also have

$$\sum_{h\in\mathcal{H}} |p_M(h)| = |R_2| \leq |R_1| = \sum_{h\in\mathcal{H}} |p_G(h)|$$

since no resident is multiply assigned in G. Hence $|p_M(h)| = |p_G(h)|$ for each $h \in \mathcal{H}$.

Thus (r', h') blocks M, since h' is undersubscribed in M by the second claim, and by Lemma 4, r' cannot be assigned to a hospital in M which he strictly prefers to h'. □

Together, Lemmas 1-6 establish the correctness of Algorithm HRT-Super-Res. In addition, Lemma 4 implies that there is an optimality property for the partner of a given assigned resident in any super-stable matching output by the algorithm. In particular, we have proved:

Theorem 1. *For a given instance of HRT, Algorithm HRT-Super-Res determines whether or not a super-stable matching exists. If such a matching does exist, all possible executions of the algorithm find one in which every assigned resident has as good a partner as in any super-stable matching, and every unassigned resident is unassigned in all super-stable matchings.*

By a suitable choice of data structures, Algorithm HRT-Super-Res can be implemented to run in $O(mn)$ time and space, where $m = |\mathcal{H}|$ and $n = |\mathcal{R}|$. The time bound follows by noting that the number of iterations of the while loop is bounded by the number of deletions from the preference lists. Note that the complexity of Algorithm HRT-Super-Res can also be expressed in terms of L, the total length of all preference lists in the HRT instance: clearly the running time is then $O(L)$. Since SM is a special case of HRT, the $\Omega(L)$ lower bound of Ng and Hirschberg [11] for SM implies that Algorithm HRT-Super-Res for HRT is optimal.

We now present the Rural Hospitals Theorem for HRT under super-stability.

Theorem 2. *Let I be a given instance of HRT. Then:*

1. *Each hospital is assigned the same number of residents in all super-stable matchings.*
2. *Exactly the same residents are unassigned in all super-stable matchings.*
3. *Any hospital that is undersubscribed in one super-stable matching is matched with exactly the same set of residents in all super-stable matchings.*

Proof. Let M, M' be two super-stable matchings in I. Let I' be an instance of HR obtained from I by resolving the ties in I arbitrarily. Then by Proposition 2, each of M, M' is stable in I'. By the Rural Hospitals Theorem for stable matchings in an instance of HR [4, Theorem 1.6.3], each hospital is assigned the same number of residents in M and M', exactly the same residents are unassigned in M and M', and any hospital that is undersubscribed in M is matched with exactly the same set of residents in M'. $\square$

3 Hospital-oriented algorithm for HRT

In this section, we consider the hospital-oriented analogue of Algorithm HRT-Super-Res, namely Algorithm HRT-Super-Hosp, shown in Figure 2. We begin by describing the execution of Algorithm HRT-Super-Hosp informally.

Algorithm HRT-Super-Hosp involves a sequence of proposals from the hospitals to the residents, in the spirit of the hospital-oriented Gale/Shapley algorithm for HR. A hospital h proposes simultaneously to the most preferred resident r on h's list not already provisionally assigned to h, and to all other residents tied with r in h's list. These proposals are provisionally accepted. If a resident r becomes multiply assigned and is indifferent between his provisional assignees, it turns out that neither of r's provisional assignees, nor any hospitals tied with them in r's list, can be a super-stable partner of r – such pairs (r, h') are deleted. If a resident r receives a proposal from a hospital h, then no hospital h' to whom r strictly prefers h can be a super-stable partner of r – again such pairs (r, h') are deleted. The proposal sequence terminates once every hospital is either full or provisionally assigned to everyone on its current list. At this point, it turns out that if a hospital is oversubscribed, or some resident is unassigned but was previously provisionally assigned, then no super-stable matching exists. Otherwise, the assignment relation is a super-stable matching.

```
assign each resident to be free;
assign each hospital to be totally unsubscribed;
**for each** resident $r$ **loop**
    $assigned(r)$ := false;
**end loop**;
**while** some hospital $h$ is undersubscribed **and**
$h$'s list contains a resident $r'$ not provisionally assigned to $h$ **loop**
    $r'$ := most preferred such resident in $h$'s list;  {any one, if $> 1$}
    **for each** resident $r$ tied with $r'$ in $h$'s list **loop**   {including $r'$}
        provisionally assign $r$ to $h$;   { $h$ "proposes" to $r$}
        $assigned(r)$ := true;
        **if** $r$ is multiply assigned **and**
        $r$ is indifferent between his provisional assignees **then**
            **for each** hospital $h'$ at the tail of $r$'s list **loop**
                **if** $r$ is provisionally assigned to $h'$ **then**
                    break the assignment;
                **end if**;
                delete the pair $(r, h')$;
            **end loop**;
        **else**
            **for each** strict successor $h'$ of $h$ on $r$'s list **loop**
                **if** $r$ is provisionally assigned to $h'$ **then**
                    break the assignment;
                **end if**;
                delete the pair $(r, h')$;
            **end loop**;
        **end if**;
    **end loop**;
**end loop**;
**if** (some resident $r$ is not assigned **and** $assigned(r)$) **or**
some hospital is oversubscribed **then**
    no super-stable matching exists;
**else**
    the assignment relation is a super-stable matching;
**end if**;
```

Fig. 2. Algorithm HRT-Super-Hosp.

In order to establish the correctness of Algorithm HRT-Super-Hosp, a number of lemmas follow. We omit the proofs, which use similar techniques to those of Section 2. We begin by stating the analogues of Lemmas 3 and 4 for Algorithm HRT-Super-Hosp. In what follows, I is an instance of HRT, in which $\mathcal{R}$ is the set of residents and $\mathcal{H}$ is the set of hospitals.

Lemma 7. *If, at the termination of the while loop of Algorithm HRT-Super-Hosp, the algorithm reports that the assignment relation M is a super-stable matching, then M is indeed a super-stable matching.*

Lemma 8. *No super-stable pair is ever deleted during an execution of Algorithm HRT-Super-Hosp.*

The next two lemmas deal with the case that Algorithm HRT-Super-Hosp claims the non-existence of a super-stable matching.

Lemma 9. *If, at the termination of the while loop of Algorithm HRT-Super-Hosp, some hospital is oversubscribed, then I admits no super-stable matching.*

Lemma 10. *If some resident r became assigned during the while loop of Algorithm HRT-Super-Hosp, and r subsequently ends up unassigned at the termination of the while loop, then I admits no super-stable matching.*

Together, Lemmas 7-10 establish the correctness of Algorithm HRT-Super-Hosp. In addition, Lemma 8 implies that there is an optimality property for the assignees of a given fully-subscribed hospital in any super-stable matching output by the algorithm. In particular, we have proved:

Theorem 3. *For a given instance of HRT, Algorithm HRT-Super-Hosp determines whether or not a super-stable matching exists. If such a matching does exist, all possible executions of the algorithm find one in which every hospital $h \in \mathcal{H}$ is assigned either its $q(h)$ best super-stable partners, or a set of fewer than $q(h)$ residents; in the latter case, no other resident is assigned to h in any super-stable matching.*

As is the case for Algorithm HRT-Super-Res, by considering suitable data structures, Algorithm HRT-Super-Hosp can be implemented to run in $O(mn)$ time and space, where $m = |\mathcal{H}|$ and $n = |\mathcal{R}|$. Again, the time bound follows by noting that the number of iterations of the while loop is bounded by the number of deletions from the preference lists. Note that the complexity of Algorithm HRT-Super-Hosp can also be expressed in terms of L, the total length of all preference lists in the HRT instance: clearly the running time is then $O(L)$. As is the case for Algorithm HRT-Super-Res, this time bound is optimal.

4 Existence of super-stable matchings

Algorithm HRT-Super-Res has been implemented and some preliminary experiments have been carried out, in order to give an indication of the likelihood of a super-stable matching existing in a given HRT instance. There are clearly several parameters that can be varied in these tests, such as the numbers of residents and hospitals, the capacities of the hospitals, the lengths of the preference lists, and the number, position and sizes of the ties. A range of vectors of values for the aforementioned parameters were considered, and for each vector, a set of random instances was created, each satisfying the particular constraints on the instance. Finally, the percentage of instances in each set admitting a super-stable matching was computed.

Perhaps not surprisingly, the empirical results suggest that the probability of a super-stable matching existing decreases as the size of the instance increases, and also decreases as the number and length of the ties increase. However, it was found that the probability of a super-stable matching existing is likely to be much higher if the ties occur on one side only, for example in the hospitals' lists and not in the residents' lists (further details may be found in [15]). This is a situation that is likely to occur naturally in practice: for example, in the context of resident/hospital matching schemes, residents are typically asked to rank a relatively small number of hospitals, and might find it easier to produce a strictly ordered preference list than would a large hospital with many applicants.

Due to the large number of different parameters that can be varied in empirical tests, clearly such experiments cannot hope to provide a comprehensive analysis of the likelihood of a super-stable matching existing in an arbitrary HRT instance. It remains open to establish theoretical bounds on the probability of a super-stable matching existing in a given random instance of HRT.

5 Concluding remarks

In this paper we have highlighted the importance of the super-stability criterion in HRT, with reference to large-scale matching schemes. Current practice in the SPA scheme, for example, is that hospitals are permitted to express ties in their preference lists. However, any ties are broken arbitrarily so as to give an instance with strictly ordered lists. Hence by Proposition 1, the SPA scheme will produce matchings that can only guarantee to be weakly stable in the original instance. We suggest that such centralised matching schemes should first search for a super-stable matching using Algorithm HRT-Super-Res, and only if none exists should they settle for a weakly stable matching.

We finish with an open problem. A third stability criterion, so-called *strong stability*, can be applied to an HRT instance [5]. In the strong stability case, the definition of a blocking pair is similar to that of the super-stability case, except that at most one agent in the pair is permitted to express indifference between the other agent and its (possibly worst) partner(s) in the matching. Clearly a super-stable matching is strongly stable, and a strongly stable matching is weakly stable. Additionally, the strong stability and super-stability definitions coincide if the ties belong to the preference lists of one set of agents only. As is the case for super-stability, a given instance of HRT may not admit a strongly stable matching (see [5] for further details). However, there is an $O(n^4)$ algorithm, due to Irving [5], for determining whether a given instance of SMT admits a strongly stable matching, and for constructing one if it does, where n is the number of men and women. An extended version of this algorithm, also of $O(n^4)$ complexity, has been formulated by Manlove for SMTI (the variant of SMT in which preference lists may be incomplete) [9]. We leave open the problem of constructing a polynomial-time algorithm, or establishing NP-completeness, for HRT under strong stability.

References

1. Canadian Resident Matching Service. How the matching algorithm works. Web document available at `http://www.carms.ca/algorith.htm`.
2. D. Gale and L.S. Shapley. College admissions and the stability of marriage. *American Mathematical Monthly*, 69:9–15, 1962.
3. D. Gale and M. Sotomayor. Some remarks on the stable matching problem. *Discrete Applied Mathematics*, 11:223–232, 1985.
4. D. Gusfield and R.W. Irving. *The Stable Marriage Problem: Structure and Algorithms*. MIT Press, 1989.
5. R.W. Irving. Stable marriage and indifference. *Discrete Applied Mathematics*, 48:261–272, 1994.
6. R.W. Irving. Matching medical students to pairs of hospitals: a new variation on an old theme. In *Proceedings of ESA '98: the Sixth European Symposium on Algorithms*, volume 1461 of *Lecture Notes in Computer Science*, pages 381–392. Springer-Verlag, 1998.
7. K. Iwama, D. Manlove, S. Miyazaki, and Y. Morita. Stable marriage with incomplete lists and ties. In *Proceedings of ICALP '99: the 26th International Colloquium on Automata, Languages, and Programming*, volume 1644 of *Lecture Notes in Computer Science*, pages 443–452. Springer-Verlag, 1999.
8. D.E. Knuth. *Stable Marriage and its Relation to Other Combinatorial Problems*, volume 10 of *CRM Proceedings and Lecture Notes*. American Mathematical Society, 1997. English translation of *Mariages Stables*, Les Presses de L'Université de Montréal, 1976.
9. D.F. Manlove. Stable marriage with ties and unacceptable partners. Technical Report TR-1999-29, University of Glasgow, Department of Computing Science, January 1999.
10. D.F. Manlove, R.W. Irving, K. Iwama, S. Miyazaki, and Y. Morita. Hard variants of stable marriage. Technical Report TR-1999-43, University of Glasgow, Department of Computing Science, September 1999. Submitted for publication.
11. C. Ng and D.S. Hirschberg. Lower bounds for the stable marriage problem and its variants. *SIAM Journal on Computing*, 19:71–77, 1990.
12. A.E. Roth. The evolution of the labor market for medical interns and residents: a case study in game theory. *Journal of Political Economy*, 92(6):991–1016, 1984.
13. A.E. Roth. On the allocation of residents to rural hospitals: a general property of two-sided matching markets. *Econometrica*, 54:425–427, 1986.
14. A.E. Roth and M.A.O. Sotomayor. *Two-sided matching: a study in game-theoretic modeling and analysis*, volume 18 of *Econometric Society Monographs*. Cambridge University Press, 1990.
15. S. Scott. Implementation of matching algorithms. Master's thesis, University of Glasgow, Department of Computing Science, 1999.