Strong Stability in the Hospitals / Residents Problem

Robert W. Irving, David F. Manlove, and Sandy Scott

UNIVERSITY of GLASGOW

Department of Computing Science
University of Glasgow
Glasgow G12 8QQ
UK

Technical Report
TR-2002-123
December 2002
Revised May 2005

Strong Stability in the Hospitals/Residents Problem

Robert W. Irving, David F. Manlove*, and Sandy Scott

Department of Computing Science, University of Glasgow, Glasgow G12 8QQ, UK.
Email: {rwi,davidm,sas}@dcs.gla.ac.uk.

Abstract

We study a version of the well-known Hospitals/Residents problem in which participants' preferences may involve ties or other forms of indifference. In this context, we investigate the concept of *strong stability*, arguing that this may be the most appropriate and desirable form of stability in many practical situations. When the indifference is in the form of ties, we describe an $O(a^2)$ algorithm to find a strongly stable matching, if one exists, where a is the number of mutually acceptable (resident,hospital) pairs. We also give a lower bound in this case in terms of the complexity of determining whether a bipartite graph contains a perfect matching. By way of contrast, we prove that it becomes NP-complete to determine whether a strongly stable matching exists if the preferences are allowed to be arbitrary partial orders.

Keywords: stable matching problem; strong stability; hospitals/residents problem; polynomial-time algorithm; lower bound; NP-completeness.

1 Introduction

The Hospitals/Residents problem [6] is a many-to-one extension of the classical Stable Marriage problem (SM), so-called because of its widespread application to matching schemes that assign graduating medical students (residents) to hospital posts. In particular the National Resident Matching Program (NRMP) in the USA [19], the Canadian Resident Matching Service [1], and the Scottish PRHO Allocations (SPA) matching scheme [9] all make use of algorithms that solve variants of this problem.

An instance of the classical Hospitals/Residents problem (HR) involves two sets, namely a set R of *residents* and a set H of *hospitals*. Each hospital $h \in H$ has a specified number p_h of posts, referred to as its *quota*. Each resident $r \in R$ ranks a

*Supported by award NUF-NAL-02 from the Nuffield Foundation, grant GR/R84597/01 from the Engineering and Physical Sciences Research Council, and Royal Society of Edinburgh / Scottish Executive Personal Research Fellowship.

subset of H in strict order of preference, and each hospital $h \in H$ ranks, again in strict order, those residents who have ranked h. These are the *preference lists* for the instance. Note that preference lists are *consistent* in the sense that a resident r appears on a hospital h's list if and only if h appears on r's list. Consistency of preference lists will be assumed throughout. A (resident,hospital) pair (r, h) is *acceptable* if each of r and h are on the other's preference list, and we denote by A (a subset of $R \times H$) the set of acceptable pairs, with $|A| = a$.

An *assignment* M is a subset of A. If $(r, h) \in M$ then we say that r is *assigned to* h, and h is *assigned* r. We denote by $M(r)$ the hospital to which r is assigned in M (this is null if r is unassigned), and by $M(h)$ the set of residents assigned to h in M (the set of *assignees* of h in M). A hospital $h \in H$ is *under-subscribed*, *fully-subscribed* (or simply *full*) or *over-subscribed* according as $|M(h)|$ is less than, equal to, or greater than p_h respectively. Hospital h is *empty* in M if $M(h) = \emptyset$.

A *matching* M is an assignment such that $|\{h \in H : (r, h) \in M\}| \leq 1$ for all $r \in R$, and $|\{r \in R : (r, h) \in M\}| \leq p_h$ for all $h \in H$. Given a pair $(r, h) \in A$, we define $h \prec_r M(r)$ to mean that either r is unassigned in M, or r prefers h to $M(r)$. Likewise, we define $r \prec_h M(h)$ to mean that either h is under-subscribed in M, or h prefers r to at least one member of $M(h)$.

A matching M in an instance of HR is *stable* if there is no pair $(r, h) \in A \setminus M$, such that $h \prec_r M(r)$ and $r \prec_h M(h)$. If such a pair (r, h) exists it is said to be a *blocking pair* for the matching M, or to *block* M. The existence of a blocking pair potentially undermines the matching, since both members of the pair could improve their situation by becoming assigned to each other.

A special case of HR arises when $p_h = 1$ for all $h \in H$ – this is the Stable Marriage problem with Incomplete lists (SMI). A further special case arises when, additionally, $|R| = |H|$ and $A = R \times H$ – this is SM. In both SMI and SM, the residents and hospitals are referred to as men and women respectively. Every instance of SM admits at least one stable matching, and such a matching may be found in linear time using the Gale/Shapley (GS) algorithm [4]. An extended version of the GS algorithm finds a stable matching for a given HR instance in $O(a)$ time [4, 6], so that in particular, every instance of HR admits at least one stable matching.

Although an instance of HR may admit more than one stable matching, it is known that every stable matching has the same size, matches the same set of residents and fills the same number of posts at each hospital. Further, any hospital that is under-subscribed in some stable matching is assigned the same set of residents in every stable matching. Collectively these results are known as the Rural Hospitals Theorem [19, 5, 20].

Recent pressure from student bodies associated with the NRMP has ensured that the extended version of the GS algorithm that is employed by the scheme is now *resident-oriented*, meaning that it produces the *resident-optimal* stable matching for a given instance of HR [18]. This is the unique stable matching M_0 in which every resident assigned in M_0 is assigned to the best hospital that he/she could obtain in any stable matching, and any resident unassigned in M_0 is unassigned in any stable matching.

In this paper we consider generalisations of HR in which preferences involve some form of indifference. This is highly relevant for practical matching schemes — for example, a popular hospital may be unable or unwilling to produce a strict ranking over all of its many applicants.

The most natural form of indifference involves ties. A set R' of k residents forms a *tie* of length k in the preference list of hospital h if h does not prefer r_i to r_j for any $r_i, r_j \in R'$ (i.e. h is *indifferent* between r_i and r_j), and for any other resident r who is acceptable to h, either h prefers r to each resident in R', or h prefers each resident in R' to r. A tie in a resident's list is defined similarly. For convenience in what follows, we consider an untied entry in a preference list as a tie of length 1. We denote by HRT the variant of HR in which preference lists can include arbitrary ties.

In certain practical applications, a variety of external factors could contribute to a given preference structure, yielding a more complex form of indifference that cannot in general be represented using only ties, but may be represented by an arbitrary partial order [2]. We denote by HRP the variant of HR in which each preference 'list' is a partial order.

Given an instance I of HRT or HRP, a *derived* instance of HR is any instance of HR obtained from I by resolving the indifference (breaking all of the ties or extending each partial order to a total order).

These extensions of the original problem force a re-evaluation of the concept of a blocking pair. We could view a pair (r, h) to be a blocking pair if, by coming together (a) both parties would be better off, or (b) neither party would be worse off, or (c) one party would be better off and the other no worse off. These three possibilities give rise to the notions of *weak stability*, *super-stability*, and *strong stability*, respectively, first considered by Irving [8] in the context of SMT (the variant of SM in which preference lists may include ties). We now formally define these three forms of stability.

A matching M in an instance of HRT or HRP is *weakly stable* if there is no pair $(r, h) \in A \setminus M$, such that $h \prec_r M(r)$ and $r \prec_h M(h)$. A weakly stable matching exists for every instance of HRP, and can be found by forming a derived instance of HR, and applying the extended GS algorithm [10]. It turns out that, in contrast to HR, weakly stable matchings for an instance of HRT may have different sizes, and it is notable that the problem of finding the largest weakly stable matching, and various other problems involving weak stability, are NP-hard [12, 17].

To define super-stability and strong stability we need to extend our notation. For a given matching M and pair $(r, h) \in A \backslash M$, we define $h \lhd_r M(r)$ to mean that r is unassigned in M, or that r prefers h to $M(r)$, or is indifferent between them. Likewise, $r \lhd_h M(h)$ means that $|M(h)| < p_h$, or h prefers r to at least one member of $M(h)$, or is indifferent between r and at least one member of $M(h)$.

A matching M is *super-stable* if there is no pair $(r, h) \in A \setminus M$, such that $h \lhd_r M(r)$ and $r \lhd_h M(h)$. By contrast with weak stability, it is straightforward to show that there are instances of HRT and HRP for which no super-stable matching exists [8]. However, there is an $O(a)$ algorithm to determine whether an instance of HRT admits a super-stable matching, and to find one if it does [10]. With some straightforward modifications, this algorithm is also applicable in the more general context of HRP. Further, the analogue of the Rural Hospitals Theorem also holds for HRP under super-stability [10].

A matching M is *strongly stable* if there is no pair $(r, h) \in A \setminus M$, such that either (i) $h \prec_r M(r)$ and $r \lhd_h M(h)$; or (ii) $h \lhd_r M(r)$ and $r \prec_h M(h)$. If $(r, h) \in M$

for some strongly stable matching M, we say that (r, h) is a *strongly stable pair*, and that r is a *strongly stable partner* of h and vice versa. Again, it is easy to construct an instance of HRT that does not admit a strongly stable matching [8]. Clearly, as is implied by the terminology, a super-stable matching is strongly stable, and a strongly stable matching is weakly stable.

There is a sense in which strong stability can be viewed as the most appropriate criterion for a practical matching scheme when there is indifference in the preference lists, and that in cases where a strongly stable matching exists, it should be chosen instead of a matching that is merely weakly stable. Consider a weakly stable matching M for an instance of HRT or HRP, and suppose that $h \prec_r M(r)$, while h is indifferent between r and its worst assignee r', and $|M(h)| = p_h$. Such a pair (r, h) would not constitute a blocking pair for weak stability. However, r might have such an overriding preference for h over $M(r)$ that he is prepared to engage in persuasion, even bribery, in the hope that h will reject r' and accept r instead. Hospital h, being indifferent between r and r' may yield to such persuasion, and, of course, a similar situation could arise with the roles reversed. However, the matching cannot be potentially undermined in this way if it is strongly stable. On the other hand, insisting on super-stability seems unnecessarily restrictive, for if (r, h) is a blocking pair for super-stability but not for strong stability, then neither r nor h has any real incentive to seek a change. Furthermore, the super-stability property is less likely to be attainable in practice.

As strong stability is sufficient to avoid the possibility of a matching being undermined by persuasion or bribery, it is therefore a desirable property in cases where it can be achieved.

In this paper we present an $O(a^2)$ algorithm for finding a strongly stable matching, if one exists, given an instance of HRT, thus solving an open problem described in [10]. Our algorithm is resident-oriented in that it finds a strongly stable matching with similar optimality properties to those of the resident-optimal stable matching in HR, as mentioned above. This algorithm is a non-trivial extension of the strong stability algorithms for SMT and SMTI (the variant of SMI in which preference lists may include ties) due to Irving [8] and Manlove [15]. We also show that the analogue of the Rural Hospitals Theorem for HR holds for HRT under strong stability. These results are presented in Section 2; in Section 3 we establish the complexity of the algorithm to be $O(a^2)$. Further, in Section 4 we prove that the complexity of any algorithm for HRT under strong stability has the same lower bound as applies to the problem of determining if a bipartite graph has a perfect matching. By contrast, we show in Section 5 that the problem of deciding whether a given instance of HRP admits a strongly stable matching is NP-complete. Finally, Section 6 presents some concluding remarks and a discussion of related work in the literature that appeared subsequently to the original publication of the results presented here [11].

2 An algorithm for strong stability in HRT

In this section we describe our algorithm, called Algorithm HRT-strong, for finding a strongly stable matching, if one exists, given an instance of HRT, and prove its correctness. Before doing so, we present some definitions relating to the algorithm.

During the execution of the algorithm, residents become *provisionally assigned* to hospitals, and it is possible for a hospital to be provisionally assigned a number of residents that exceeds its quota. We describe a hospital as *replete* if at any time during the execution of the algorithm it has been over-subscribed or fully-subscribed.

The algorithm proceeds by deleting from the preference lists certain pairs that cannot be strongly stable. By the *deletion* of a pair (r, h), we mean the removal of r and h from each other's lists, and, if r is provisionally assigned to h, the breaking of this provisional assignment. By the *head* and *tail* of a preference list at a given point we mean the first and last ties respectively on that list (recalling that a tie can be of length 1). We say that a resident r is *dominated* in a hospital h's list if h prefers to r at least p_h residents who are provisionally assigned to it.

A resident r who is provisionally assigned to a hospital h is said to be *bound* to h if h is not over-subscribed or r is not in h's tail (or both). The *provisional assignment graph* is a bipartite graph G containing a vertex for each resident and each hospital, with (r, h) forming an edge if resident r is provisionally assigned to hospital h. A *feasible matching* in G is a matching M such that, if r is bound to one or more hospitals, then r is assigned to one of these hospitals in M, and subject to this restriction, M has maximum possible cardinality.

A *reduced assignment graph* G_R is formed from a provisional assignment graph as follows. For each resident r, and for each hospital h such that r is bound to h, we delete the edge (r, h) from the graph, and we reduce the quota of h by one; furthermore, we remove *all* other edges incident to r. Each isolated resident vertex is then removed from the graph. Finally, if the quota of any hospital h is reduced to 0, or h becomes an isolated vertex, then h is removed from the graph. For each surviving h we denote by p'_h the revised quota of h.

Given a set Z of residents in G_R, define $\mathcal{N}(Z)$, the *neighbourhood* of Z, to be the set of hospital vertices adjacent in G_R to a resident vertex in Z. The *deficiency* of Z is defined by $\delta(Z) = |Z| - \sum_{h \in \mathcal{N}(Z)} p'_h$. It is not hard to show that, if Z_1 and Z_2 are maximally deficient, then so also is $Z_1 \cap Z_2$, so there is a unique minimal set with maximum deficiency. This is the *critical set*.

Algorithm HRT-strong, displayed in Figure 2, begins by assigning each resident to be free (i.e., not assigned to any hospital). The iterative stage of the algorithm involves each free resident in turn being provisionally assigned to the hospital(s) at the head of his list. If, by gaining a new provisional assignee, a hospital h becomes fully- or over-subscribed then each pair (r, h), such that r is dominated on h's list, is deleted. This continues until every resident is provisionally assigned to one or more hospitals or has an empty list. We then find the reduced assignment graph G_R (note that G_R is formed afresh from the current provisional assignment graph in each loop iteration) and the critical set Z of residents. As we will see later, no hospital in $\mathcal{N}(Z)$ can be assigned a resident from among those in its tail in any strongly stable matching, so all such pairs are deleted. The iterative step is then reactivated, and this entire process continues until Z is empty, which must happen eventually, since if Z is found to be non-empty, then at least one pair is subsequently deleted from the preference lists.

Let M be any feasible matching in the final provisional assignment graph G. As we will show, if M itself is not strongly stable then no strongly stable matching exists, otherwise the algorithm outputs M.

assign each resident to be free;
assign each hospital to be empty;
repeat {
  while (some resident r is free **and** r has a non-empty list)
    for (each hospital h at the head of r's list) {
      provisionally assign r to h;
      if (h is fully-subscribed **or** h is over-subscribed)
        for (each resident r' dominated on h's list)
          delete the pair (r', h); }
  form the reduced assignment graph;
  find the critical set Z of residents;
  for (each hospital $h \in \mathcal{N}(Z)$)
    for (each resident r in the tail of h's list)
      delete the pair (r, h);
} **until** $Z == \emptyset$;
let G be the final provisional assignment graph;
let M be a feasible matching in G;
if (M is not strongly stable)
  no strongly stable matching exists;
else
  output the strongly stable matching specified by M;

Figure 1: Algorithm HRT-strong.

The correctness of Algorithm HRT-strong, and an optimality property of any strongly stable matching that it finds, are established below. The first lemma shows that the algorithm never deletes a strongly stable pair.

Lemma 2.1. *No strongly stable pair is ever deleted during an execution of Algorithm* HRT-*strong.*

Proof. Suppose that the pair (r, h) is the first strongly stable pair deleted during some execution of the algorithm, and let M' be a strongly stable matching in which r is assigned to h. There are two cases to consider.

Case 1: Suppose (r, h) is deleted as a result of some other resident, r' say, becoming provisionally assigned to h, so that r is dominated on h's list. Call the set of residents provisionally assigned to h at this point R'. None of the residents in R' can be assigned to a hospital they prefer to h in any strongly stable matching, for otherwise some strongly stable pair must have been deleted before (r, h), as h must be in the head of each of the lists of the residents in R'. In M', at least one of the residents in R', r'' say, cannot be assigned to h, so r'' is either unassigned in M', or prefers h to $p_{M'}(r'')$, or is indifferent between h and $p_{M'}(r'')$. It follows that (r'', h) blocks M', a contradiction.

Case 2: Suppose that (r, h) is deleted because h is provisionally assigned a resident in the critical set Z at some point, and at that point r is in h's tail. We refer to the set of lists at that point as the *current lists*. Let Z' be the set of residents in Z who are assigned in M' to a hospital from the head of their current list, and let H' be the set of hospitals in $\mathcal{N}(Z)$ who are assigned in M' at least one resident from the tail of their current list. Then $h \in H'$, so $H' \neq \emptyset$. Consider $r^* \in Z$. Now

r^* cannot be assigned in M' to a hospital that he prefers to any member of the head of his current list, for otherwise some strongly stable pair must have been deleted before (r, h). Hence, any resident r^* in Z who is provisionally assigned to h must be in Z', otherwise (r^*, h) would block M'. Thus $Z' \neq \emptyset$.

We now claim that $\mathcal{N}(Z \setminus Z')$ is not contained in $\mathcal{N}(Z) \setminus H'$. For, suppose that the containment does hold. Then

$$\begin{aligned} |Z \setminus Z'| - \textstyle\sum_{h \in \mathcal{N}(Z \setminus Z')} p_h &\geq |Z \setminus Z'| - \textstyle\sum_{h \in \mathcal{N}(Z) \setminus H'} p_h \\ &= |Z| - \textstyle\sum_{h \in \mathcal{N}(Z)} p_h - (|Z'| - \textstyle\sum_{h \in H'} p_h). \end{aligned}$$

But $|Z'| - \sum_{h \in H'} p_h \leq 0$, because every resident in Z' is assigned in M' to a hospital in H'. Hence $Z \setminus Z'$ has deficiency greater than or equal to that of Z, contradicting the fact that Z is the critical set. Thus the claim is established.

Hence there must be a resident $r_1 \in Z \setminus Z'$ and a hospital $h_1 \in H'$ such that r_1 is provisionally assigned to h_1. Since r_1 is either unassigned in M' or prefers h_1 to $p_{M'}(r_1)$ and h_1 is indifferent between r_1 and at least one member of $p_{M'}(h_1)$, the pair (r_1, h_1) blocks M', a contradiction. □

We continue with three auxiliary lemmas.

Lemma 2.2. *Every resident who is assigned to a hospital in the final provisional assignment graph G must be assigned in any feasible matching M.*

Proof. The result is true by definition for any bound resident. Consider the other residents assigned in G. Any x of them must be collectively adjacent in G_R to hospitals with at least x posts, otherwise one of them is in the critical set Z, and hence $Z \neq \emptyset$. But, by a simple extension of Philip Hall's Theorem, this means that they are all assigned in any maximum cardinality matching in G_R, and hence they must be assigned in any feasible matching M. □

Lemma 2.3. *Let M be a feasible matching in the final provisional assignment graph G. If (a) some non-replete hospital h has fewer assignees in M than provisional assignees in G, or (b) some replete hospital h is not full in M, then no strongly stable matching exists.*

Proof. Suppose that M' is a strongly stable matching for the instance. Every resident provisionally assigned to a hospital in the final assignment graph G must be assigned to a hospital in M (by Lemma 2.2), and any resident not provisionally assigned in G must have an empty list and hence no strongly stable partners (by Lemma 2.1). It follows that $|M'| \leq |M|$.

Suppose that condition (a) is satisfied. Then some non-replete hospital h' satisfies $|p_M(h')| < d_G(h')$, where $d_G(h')$ is the degree of vertex h' in G, i.e., the number of residents provisionally assigned to h'. As h' is non-replete, it follows that $d_G(h') < p_{h'}$. Now $|p_M(h)| \leq \min(d_G(h), p_h)$ for all $h \in H$. Hence

$$|M| = \sum_{h \in H} |p_M(h)| < \sum_{h \in H} \min(d_G(h), p_h). \tag{1}$$

Now suppose that $|p_{M'}(h)| \geq \min(d_G(h), p_h)$ for all $h \in H$. Then $|M'| > |M|$ by 1, a contradiction. Hence $|p_M(h'')| < \min(d_G(h''), p_h'')$ for some $h'' \in H$. Hence h''

is under-subscribed in M', and some resident r' is provisionally assigned to h'' in G but not assigned to h'' in M'. By Lemma 2.1, r' is not assigned to a hospital in M' that he prefers to h''. Hence (r', h'') blocks M', a contradiction.

Now suppose that condition (b) is satisfied. Let H_1 and H_2 be the set of replete and non-replete hospitals respectively. Then some $h' \in H_1$ satisfies $|p_M(h')| < p_{h'}$. Condition (a) cannot be satisfied, for otherwise the first part of the proof shows that M' does not exist. Hence $|p_M(h)| = d_G(h) < p_h$ for all $h \in H_2$. Now $p_M(h) \leq p_h$ for all $h \in H_1$. Hence

$$|M| = \sum_{h \in H_1} |p_M(h)| + \sum_{h \in H_2} |p_M(h)| < \sum_{h \in H_1} p_h + \sum_{h \in H_2} d_G(h). \tag{2}$$

Now suppose that $|p_{M'}(h) \geq p_h$ for all $h \in H_1$ and $|p_{M'}(h) \geq d_G(h)$ for all $h \in H_2$. Then $|M'| > |M|$ by 2, a contradiction. Hence either (i) $|p_{M'}(h'') < p_h''$ for some $h'' \in H_1$ or (ii) $|p_{M'}(h'')| < d_G(h'')$ for some $h'' \in H_2$. In Case (ii) we reach a similar contradiction to that arrived at for condition (a). In Case (i), h'' is under-subscribed in M'. As h'' is replete, there exists some resident r' who was provisionally assigned to h'' during the execution of the algorithm, but is not assigned to h'' in M'. By Lemma 2.1, r' is not assigned to a hospital in M' that he prefers to h''. Hence (r', h'') blocks M', a contradiction. □

Lemma 2.4. *Suppose that, in the final assignment graph G, a resident is bound to two different hospitals. Then no strongly stable matching exists.*

Proof. Suppose that a strongly stable matching exists for the instance. Let M be a feasible matching in the final provisional assignment graph G. Denote by H_1 the set of over-subscribed hospitals in G, let $H_2 = H \setminus H_1$ and let $d_G(h)$ denote the degree of the vertex h in G - i.e., the number of residents provisionally assigned to h. Denote by R_1 the set of residents bound to one or more hospitals in G, and by R_2 the other residents assigned to one or more hospitals in G. Note that for each $h \in H_2$, any resident assigned to h in G is bound to h, and hence is in R_1.

By Lemma 2.2, we have

$$|M| = |R_1| + |R_2|. \tag{3}$$

Also, by Lemma 2.3, we have

$$|M| = \sum_{h \in H_1} p_h + \sum_{h \in H_2} d_G(h). \tag{4}$$

If some resident is bound to more than one hospital then, by considering how quotas are reduced when the residents of R_1 are removed in deriving G_R from G, it follows that

$$\sum_{h \in H_1} (p_h - p'_h) + \sum_{h \in H_2} d_G(h) > |R_1|. \tag{5}$$

Combining 3, 4 and 5 gives

$$\sum_{h \in H_1} p'_h < |R_2|.$$

Since no member of H_2 belongs to G_R, the residents in R_2 are collectively adjacent only to hospitals in H_1, and so the preceding inequality suffices to establish that the critical set is non-empty, a contradiction. □

The next lemma shows that a feasible matching M may be used to determine the existence or otherwise of a strongly stable matching for the given instance.

Lemma 2.5. *Let M be a feasible matching in the final provisional assignment graph. If M is not strongly stable then there is no strongly stable matching for the instance.*

Proof. Suppose M is not strongly stable, and let (r, h) be a blocking pair for M. Suppose r prefers h to $p_M(r)$, or r is unassigned in M. Then (r, h) has been deleted, which can only happen if h is replete. To see this, suppose that h is not replete, but (r, h) was deleted because h was a neighbour of some resident $r' \in Z$ at a point when r was in h's tail. Suppose that the residents in $Z' \subseteq Z$ are provisionally assigned to h' in G_R. Then $0 < |Z'| \leq p'_{h'}$. Let $Z^* = Z \setminus Z'$. Then $\mathcal{N}(Z^*) \subseteq \mathcal{N}(Z) \setminus \{h'\}$ so that

$$\sum_{h \in \mathcal{N}(Z^*)} p'_h \leq \sum_{h \in \mathcal{N}(Z)} (p'_h - p'_{h'}).$$

Hence

$$\begin{array}{rcl} \delta(Z^*) & = & |Z^*| - \sum_{h \in \mathcal{N}(Z^*)} p'_h \\ & = & |Z| - |Z'| - \sum_{h \in \mathcal{N}(Z^*)} p'_h \\ & \geq & |Z| - |Z'| - (\sum_{h \in \mathcal{N}(Z)} (p'_h - p'_{h'})) \\ & = & |Z| + p'_{h'} - |Z'| - \sum_{h \in \mathcal{N}(Z)} p'_h \\ & \geq & |Z| - \sum_{h \in \mathcal{N}(Z)} p'_h \\ & = & \delta(Z). \end{array}$$

If $\delta(Z^*) > \delta(Z)$ then Z^* contradicts the fact that Z is maximally deficient. Hence $\delta(Z^*) = \delta(Z)$. But $Z^* \subset Z$, contradicting the minimality of Z. Thus h is replete. If h is full in M then h prefers all its assignees to r, since r is a strict successor of any undeleted entries in h's list, contradicting the fact that (r, h) is a blocking pair for M. If h is not full in M then h is a replete hospital which is not full in M, so by Lemma 2.3, no strongly stable matching exists for the instance, and we are done. We now consider the case where r is indifferent between h and $p_M(r)$.

Suppose h is not full in M. If h is replete then, by Lemma 2.3, no strongly stable matching exists for the instance and we are done. If h is not replete then r must be bound to h, and since r is not assigned to h in M, by the definition of a feasible matching r must be bound to $p_M(r)$. But then r is bound to two hospitals, so by Lemma 2.4 no strongly stable matching exists for the instance.

Now suppose h is full in M. For (r, h) to block M, h must prefer r to at least one of its assignees in M. But then r is bound to h, and since r is not assigned to h in M, r must be bound to $p_M(r)$. But then again r is bound to two hospitals, so by Lemma 2.4 no strongly stable matching exists for the instance. □

Lemma 2.5 proves the correctness of Algorithm HRT-strong. Further, Lemma 2.1 shows that there is an optimality property for each assigned resident in any strongly stable matching output by the algorithm. To be precise, we have proved:

Theorem 2.6. *For a given instance of* HRT*, Algorithm* HRT*-strong determines whether or not a strongly stable matching exists. If such a matching does exist, all possible executions of the algorithm find one in which every assigned resident is assigned as favourable a hospital as in any strongly stable matching, and any unassigned resident is unassigned in every strongly stable matching.*

For obvious reasons, we call any matching found by the above algorithm *resident-optimal.*

Now we show that the Rural Hospitals Theorem holds for HRT under strong stability. For the following lemma and theorem we assume that we have an HRT instance that admits a strongly stable matching.

Lemma 2.7. *For a given* HRT *instance, let M be the matching obtained by Algorithm* HRT*-strong and let M' be any strongly stable matching. If a hospital h is not full in M' then every resident assigned to h in M is also assigned to h in M'.*

Proof. Suppose r is assigned to h in M, but not in M'. Then (r, h) blocks M' since h is under-subscribed in M' and r cannot prefer any of his strongly stable partners to h. □

Theorem 2.8. *For a given* HRT *instance I,*

1. *each hospital is assigned the same number of residents in every strongly stable matching;*
2. *the same residents are assigned in every strongly stable matching;*
3. *any hospital that is under-subscribed in some strongly stable matching is assigned the same set of residents in every strongly stable matching.*

Proof. Let M be the strongly stable matching obtained by Algorithm HRT-strong, and let M' be any strongly stable matching such that $M' \neq M$.

1. We first observe that any resident r who is unassigned in M cannot be assigned in M', since r must have an empty list (hence (r, h) has been deleted for every hospital h that r finds acceptable, and by Lemma 2.1 no strongly stable pair is deleted during the execution of Algorithm HRT-strong). It follows that $|M'| \leq |M|$. By Lemma 2.7, any hospital that is full in M is also full in M', while any hospital that is not full in M fills at least as many posts in M' as in M. It follows that $|M'| \geq |M|$, and so, combining this with the earlier inequality, $|M| = |M'|$. This equality and the conclusions drawn earlier from Lemma 2.7 imply that every hospital is assigned the same number of residents in M and M'.
2. As has already been observed, $|M| = |M'|$, and no resident who is unassigned in M can be assigned in M', so the same set of residents are assigned in M and M'.
3. As has already been observed, $|M'| \leq |M|$, and the result follows for M' by Lemma 2.7 and the first part of the proof.

Since M' is an arbitrary strongly stable matching, these results follow for every strongly stable matching. □

Example 2.9. An example instance is displayed in Figure 2.9. The residents are labeled r_i $(1 \leq i \leq 6)$ and the hospitals are labeled h_i $(1 \leq i \leq 3)$. The entry for hospital h_i takes the form h_i : (p_{h_i}) P_{h_i}, where P_{h_i} is h_i's preference list. The entry for a resident is similar, but without the quota element. Ties in the preference lists are represented by parentheses. We assume that the residents become assigned to the hospitals at the head of their lists in subscript order. The while loop of Algorithm HRT-strong terminates with every resident except r_6 provisionally assigned to every hospital in the first tie on their preference list. Resident r_6 has an empty list, because (r_6, h_3) was deleted as a result of r_1 becoming provisionally assigned to h_3, causing r_6 to be dominated on the list of h_3. Only one edge is removed from the provisional assignment graph to form the reduced assignment graph, as only resident r_2 is bound to a hospital, namely h_2. The isolated vertices are then removed from the graph, leaving residents r_1, r_3, r_4 and r_5, and hospitals h_2 and h_3. It can then be shown that every resident in the reduced assignment graph is in the critical set, and the neighbourhood of the critical set is $\{h_2, h_3\}$. The lists after the relevant deletions have been made are displayed in Figure 2.9. By following the same process, it can be shown that a second iteration of the main loop of Algorithm HRT-strong terminates with an empty critical set, and there are two feasible matchings, $(r_1, h_1), (r_2, h_2), (r_3, h_2), (r_4, h_3), (r_5, h_1)$ and $(r_1, h_1), (r_2, h_2), (r_3, h_2), (r_4, h_1), (r_5, h_3)$, one of which is output by the algorithm, and both are strongly stable.

	Initial preference lists			Lists after first loop iteration	
r_1 :		$(h_2\ h_3)\ h_1\ h_4$	r_1 :		$h_1\ h_4$
r_2 :		$h_2\ h_1$	r_2 :		$h_2\ h_1$
r_3 :		$h_3\ h_2\ h_1$	r_3 :		$h_2\ h_1$
r_4 :		$h_2\ (h_1\ h_3)$	r_4 :		$(h_1\ h_3)$
r_5 :		$h_2\ (h_1\ h_3)$	r_5 :		$(h_1\ h_3)$
r_6 :		h_3	r_6 :		
h_1 :	(2)	$r_2\ (r_1\ r_3)\ (r_4\ r_5)$	h_1 :	(2)	$r_2\ (r_1\ r_3)\ (r_4\ r_5)$
h_2 :	(2)	$r_3\ r_2\ (r_1\ r_4\ r_5)$	h_2 :	(2)	$r_3\ r_2$
h_3 :	(1)	$(r_4\ r_5)\ (r_1\ r_3)\ r_6$	h_3 :	(1)	$(r_4\ r_5)$
h_4 :	(1)	r_1	h_4 :	(1)	r_1

Figure 2: The preference lists for an example HRT instance.

3 Implementation and analysis of Algorithm HRT-strong

For the implementation and analysis of Algorithm HRT-strong, we require to describe the efficient construction of maximum cardinality matchings and critical sets in a context somewhat more general than that of simple bipartite graphs.

Consider a *capacitated* bipartite graph $G = (V, E)$, with bipartition $V = R \cup H$, in which each vertex $h \in H$ has a positive integer *capacity* c_h. In this context, a *matching* is a subset M of E such that $|\{h : \{r, h\} \in M\}| \leq 1$ for all $r \in R$, and $|\{r : \{r, h\} \in M\}| \leq c_h$ for all $h \in H$. For any vertex x, a vertex joined to x by an edge of M is called a *mate* of x. A vertex $r \in R$ with no mate, or a vertex $h \in H$ with fewer than c_h mates, is said to be *exposed*. An *alternating path* in G relative to M is any simple path in which edges are alternately in, and not in, M. An *augmenting* path is an alternating path of odd length both of whose endpoints are exposed. It is immediate that an augmenting path has one endpoint in R and the other in H.

The following lemmas may be established by straightforward extension of the corresponding results for one-to-one bipartite matching.

Lemma 3.1. *Let P be the set of edges on an augmenting path relative to a matching M in a capacitated bipartite graph G. Then $M' = M \oplus P$ is a matching of cardinality $|M| + 1$ in G.*

Lemma 3.2. *A matching M in a capacitated bipartite graph has maximum cardinality if and only if there is no augmenting path relative to M in G.*

The process of replacing M by $M' = M \oplus P$ is called *augmenting M* along the path P.

With these lemmas, we can extend to the context of capacitated bipartite graphs the classical augmenting path algorithm for a maximum cardinality matching. The algorithm starts with an arbitrary matching – say the empty matching – and repeatedly augments the matching until there is no augmenting path. The search for an augmenting path relative to M is organised as a restricted breadth-first search in which only edges of M are followed from vertices in H and only edges not in M are followed from vertices in R, to ensure alternation. The number of iterations is $O(\min(|R|, \sum c_h))$, and each search can be completed in $O(|E|)$ time, since there are no isolated vertices. During the breadth-first search, we record the parent in the BFS spanning tree of each vertex. This enables us to accomplish the augmentation in $O(|E|)$ time, observing that, for each vertex $h \in H$, the set of mates can be updated in constant time by representing the set as, say, a doubly linked list, and storing a pointer into this list from any child node in the BFS spanning tree. Hence, overall, the augmenting path algorithm in a capacitated bipartite graph can be implemented to run in $O((\min(|R|, \sum c_h))|E|)$ time.

Now that we have ascertained that we can efficiently find a maximum cardinality matching in the reduced assignment graph, the following lemma points the way to finding the critical set.

Lemma 3.3. *Given a maximum cardinality matching M in the capacitated bipartite graph G_R, the critical set Z consists of the set U of unassigned residents together with the set U' of residents reachable from a vertex in U via an alternating path.*

Proof. Let $C = U \cup U'$. It is immediate that $\delta(C) = \delta(G)(= |U|)$, for if $\mathcal{N}(C)$ were such that

$$\sum_{h \in \mathcal{N}(C)} p'_h > |U'|$$

then there would be an augmenting path relative to M, contradicting the maximality of M.

Further, the critical set Z must contain every resident who is unassigned in some maximum cardinality matching in G. For if M' is an arbitrary such matching (of size $|R| - \delta(G)$), and if $r \in R \setminus Z$ is not assigned in M', then Z must contain at least $|Z| - \delta(G) + 1$ assigned residents. To see this consider that there must be $\delta(G)$ unassigned residents, with at most $\delta(G) - 1$ of these residents contained in Z. Hence Z contains at most $|Z| - \delta(G) + 1$ residents. It follows that

$$\sum_{h \in \mathcal{N}(Z)} p'_h \geq |Z| - \delta(G) + 1$$

or

$$|Z| - \sum_{h \in \mathcal{N}(Z)} p'_h \leq \delta(G) - 1$$

contradicting the required deficiency of Z.

But, for every $r \in U'$, there is a maximum cardinality matching in which r is unassigned, obtainable from M via an alternating path from a resident in U to r. Hence, $C \subseteq Z$, and since $\delta(C) = \delta(Z)$, the proof is complete. □

During each iteration of the repeat-until loop of Algorithm HRT-strong we need to form the reduced assignment graph, which takes $O(a)$ time, then search for a maximum cardinality matching in the bipartite graph G_R. This allows us to use Lemma 3.3 to find the critical set. The key to the analysis of Algorithm HRT-strong, as with Algorithm STRONG in [8], is bounding the total amount of work done in finding the maximum cardinality matchings.

It is clear that work done other than in finding the maximum cardinality matchings and critical sets is bounded by a constant times the number of deleted pairs, and so is $O(a)$.

Suppose that Algorithm HRT-strong finds a maximum cardinality matching M_i in the reduced assignment graph G_R at the ith iteration. Suppose also that, during the ith iteration, x_i pairs are deleted because they involve residents in the critical set Z, or residents tied with them in the list of a hospital in $\mathcal{N}(Z)$. Suppose further that in the $(i+1)$th iteration, y_i pairs are deleted before the reduced assignment graph is formed. Note that any edge in G_R at the ith iteration which is not one of these $x_i + y_i$ deleted pairs must be in G_R at the $(i+1)$th iteration, since a resident can only become bound to a hospital when he becomes provisionally assigned to it. In particular at least $|M_i| - x_i - y_i$ pairs of M_i remain in G_R at the $(i+1)$th iteration. Hence, in that iteration, we can start from these pairs and find a maximum cardinality matching in $O(\min(na, (x_i + y_i + z_i)a))$ time, where n is the number of residents and z_i is the number of edges in G_R at the $(i+1)$th iteration which were not in G_R at the ith iteration.

Let s denote the number of iterations carried out, let $S = \{1, 2, \ldots, s\}$, and let $S' = S \backslash \{s\}$. Let $T \subseteq S'$ denote those indices i such that $\min(na, (x_i + y_i + z_i)a) = na$, and let $t = |T|$. Then the algorithm has time complexity $O(\min(n,p)a + tna + a\sum_{i \in S' \backslash T}(x_i + y_i + z_i))$, where p is the total number of posts, and the first term is for the first iteration. But $\sum_{i \in S'}(x_i + y_i) \leq a$ and $\sum_{i \in S'} z_i \leq a$ (since these summations are bounded by the total number of deletions and provisional assignments, respectively), and since $x_i + y_i + z_i \geq n$ for each $i \in T$, it follows that

$$tn + \sum_{i \in S' \backslash T}(x_i + y_i + z_i) \leq \sum_{i \in S'}(x_i + y_i + z_i) \leq 2a.$$

Thus

$$\sum_{i \in S' \setminus T} (x_i + y_i + z_i) \leq 2a - tn.$$

After the end of the final iteration a feasible matching is constructed by taking the final maximum cardinality matching and combining it with the bound (resident,hospital) pairs. This operation is clearly bounded by the number of bound pairs, hence is $O(a)$. It follows that the overall complexity of Algorithm HRT-strong is $O(\min(n,p)a + tna + a(2a - tn)) = O(a^2)$.

Note that while there is at least one algorithm for finding maximum cardinality matchings in the context of capacitated bipartite graphs with better time complexity than the one we use here (see e.g. [3]), it is not clear how we can use this algorithm to give an improvement in the running time of Algorithm HRT-strong.

4 A lower bound for finding a strongly stable matching

To establish the lower bound of this section, we let STRONGLY STABLE MATCHING IN HRT be the problem of deciding whether a given instance of HRT admits a strongly stable matching.

Let n denote the number of participants (i.e. residents and hospitals) in a given instance of HRT. We show that, for any function f on n, where $f(n) = \Omega(n^2)$, the existence of an $O(f(n))$ algorithm for STRONGLY STABLE MATCHING IN HRT would imply the existence of an $O(f(n))$ algorithm for PERFECT MATCHING IN BIPARTITE GRAPHS (the problem of deciding whether a given bipartite graph admits a perfect matching).

The result is established by the following simple reduction from PERFECT MATCHING IN BIPARTITE GRAPHS to STRONGLY STABLE MATCHING IN HRT.

Let $G = (V, E)$ be a bipartite graph with bipartition $V = R \cup H$. Let $R = \{r_1, \ldots, r_n\}$ and $H = \{h_1, \ldots, h_n\}$, and, without loss of generality, assume that G contains no isolated vertices. Also, for each i ($1 \leq i \leq n$), let P_i denote the set of vertices in H adjacent to r_i.

We form an instance I of HRT as follows. Let $p_i = 1$ for all i. Form a preference list for each participant in I as follows:

$$r_i : \ (P_i) \ \ (H \setminus P_i) \qquad\qquad h_i : \ (R) \qquad\qquad (1 \leq i \leq n).$$

In a given participant's preference list (S) denotes all members of the set S listed as a tie in the position where the symbol occurs.

It is straightfoward to verify that G admits a perfect matching if and only if I admits a strongly stable matching. Clearly the reduction may be carried out in $O(n^2)$ time. Hence, for any function f on n, where $f(n) = \Omega(n^2)$, an $O(f(n))$ algorithm for STRONGLY STABLE MATCHING IN HRT would solve PERFECT MATCHING IN BIPARTITE GRAPHS in $O(f(n))$ time. The current best algorithm for PERFECT MATCHING IN BIPARTITE GRAPHS has complexity $O(\sqrt{n}m)$ [7], where m is the number of edges in G. Finally we note that, since each hospital in I has quota 1, the lower bound established in this section also applies to STRONG STABLE MATCHING IN SMT (the restriction of STRONGLY STABLE MATCHING IN HRT to SMT).

5 NP-completeness of strong stability in HRP

In this section we establish the NP-completeness of STRONGLY STABLE MATCHING IN SMP, which is the problem of deciding whether a given instance of SMP admits a strongly stable matching. Here, SMP denotes the variant of SM in which each person's preferences over the members of the opposite sex are represented as an arbitrary partial order (henceforth this preference structure is referred to as a *preference poset*). Clearly SMP is a special case of HRP in which $|R| = |H|$, $A = R \times H$ and each hospital has quota 1. It therefore follows immediately that the problem of deciding whether a given instance of HRP admits a strongly stable matching is also NP-complete.

To prove our result we give a reduction from the following problem:

> *Name:* RESTRICTED SAT.
> *Instance:* Boolean formula B in CNF, where each variable v occurs in exactly two clauses of B as literal v, and in exactly two clauses of B as literal $\overline{v}$.
> *Question:* Is B satisfiable?

We firstly establish the NP-completeness of RESTRICTED SAT.

Lemma 5.1. RESTRICTED SAT *is NP-complete.*

Proof. Clearly RESTRICTED SAT belongs to NP. To show NP-hardness, we give a reduction from the more general version of RESTRICTED SAT, in which each variable v occurs in at most two clauses as literal v, and in at most two clauses as literal $\overline{v}$. This problem is NP-complete [6, p.210]. Let B be an instance of this problem, and suppose that some variable v occurs in zero or one clauses of B as literal v (the argument is similar for $\overline{v}$). Construct a Boolean formula B' by introducing new variables x,y,z, and by adding the following clauses to B:

$$(v \vee y) \wedge (x \vee y \vee \overline{z}) \wedge (\overline{x} \vee \overline{y} \vee z) \wedge (x \vee \overline{y} \vee \overline{z}) \wedge (\overline{x} \vee z).$$

Clearly B is satisfiable if and only if B' is satisfiable. In addition, each of the new variables w introduced occurs in exactly two clauses of B' as literal w, and in exactly two clauses of B' as literal $\overline{w}$. Thus successive applications of this reduction will yield an instance of RESTRICTED SAT; clearly the overall transformation is polynomial. □

We now state and prove the main result of this section.

Theorem 5.2. STRONGLY STABLE MATCHING IN SMP *is NP-complete.*

Proof. Clearly STRONGLY STABLE MATCHING IN SMP is in NP. To show NP-hardness, we give a polynomial reduction from RESTRICTED SAT, which is NP-complete as mentioned above. Let B be a Boolean formula in CNF, given as an instance of this, in which $X = \{x_1, x_2, \ldots, x_n\}$ is the set of variables and $C = \{c_1, c_2, \ldots, c_m\}$ is the set of clauses. For each i $(1 \leq i \leq n)$ and for each r $(1 \leq r \leq 2)$, let $c(x_i^r)$ (respectively $c(\overline{x}_i^r)$) denote the clause corresponding to the rth occurrence of literal x_i (respectively $\overline{x}_i$).

We now construct an instance I of SMP, as follows. Let $U = X^1 \cup X^2 \cup \overline{X}^1 \cup \overline{X}^2 \cup Z$ be the set of men in I, and let $W = Y^1 \cup Y^2 \cup \overline{Y}^1 \cup \overline{Y}^2 \cup C$ be the set of women in I, where

$$\begin{array}{llll} X^r = \{x_i^r : 1 \le i \le n\} & (1 \le r \le 2), & Y^r = \{y_i^r : 1 \le i \le n\} & (1 \le r \le 2), \\ \overline{X}^r = \{\overline{x}_i^r : 1 \le i \le n\} & (1 \le r \le 2), & \overline{Y}^r = \{\overline{y}_i^r : 1 \le i \le n\} & (1 \le r \le 2), \\ Z = \{z_i : 1 \le i \le m\}, & & C = \{c_i : 1 \le i \le m\}. & \end{array}$$

Clearly $|U| = |W| = 4n + m$. Now, for each person p in I, we formulate $\prec_p^*$, the preference poset of p. In order to define $\prec_p^*$, we will construct a relation $\prec_p$, where $q \prec_p r$ implies that p prefers q to r. We then obtain the partial order $\prec_p^*$ by taking the transitive closure of $\prec_p$. Note that p is indifferent between q and r if and only if q, r are incomparable in $\prec_p^*$ (i.e. neither $q \prec_p^* r$ nor $r \prec_p^* q$ holds). For each person q we will also define a subset $P(q)$ of members of the opposite sex; if $r \in P(q)$ we say that r is *proper* for q.

- *Preference poset of* x_i^r *(*$1 \le i \le n,\ 1 \le r \le 2$*)*: $\overline{y}_i^1 \prec_{x_i^r} c(x_i^r)$, $\overline{y}_i^2 \prec_{x_i^r} c(x_i^r)$, $y_i^r \prec_{x_i^r} p$, for every $p \in W \backslash P(x_i^r)$, where $P(x_i^r) = \{c(x_i^r), y_i^r, \overline{y}_i^1, \overline{y}_i^2\}$.
- *Preference poset of* $\overline{x}_i^r$ *(*$1 \le i \le n,\ 1 \le r \le 2$*)*: $y_i^1 \prec_{\overline{x}_i^r} c(\overline{x}_i^r)$, $y_i^2 \prec_{\overline{x}_i^r} c(\overline{x}_i^r)$, $\overline{y}_i^r \prec_{\overline{x}_i^r} p$, for every $p \in W \backslash P(\overline{x}_i^r)$, where $P(\overline{x}_i^r) = \{c(\overline{x}_i^r), \overline{y}_i^r, y_i^1, y_i^2\}$.
- *Preference poset of* z_i *(*$1 \le i \le m$*)*: $y \prec_{z_i} p$, for every $y \in P(z_i)$ and for every $p \in W \backslash P(z_i)$, where $P(z_i) = Y^1 \cup Y^2 \cup \overline{Y}^1 \cup \overline{Y}^2$.
- *Preference poset of* y_i^r *(*$1 \le i \le n,\ 1 \le r \le 2$*)*: $x_i^r \prec_{y_i^r} \overline{x}_i^1$, $x_i^r \prec_{y_i^r} \overline{x}_i^2$. Let $P(y_i^r) = \{x_i^r, \overline{x}_i^1, \overline{x}_i^2\} \cup Z$.
- *Preference poset of* $\overline{y}_i^r$ *(*$1 \le i \le n,\ 1 \le r \le 2$*)*: $\overline{x}_i^r \prec_{\overline{y}_i^r} x_i^1$, $\overline{x}_i^r \prec_{\overline{y}_i^r} x_i^2$. Let $P(\overline{y}_i^r) = \{\overline{x}_i^r, x_i^1, x_i^2\} \cup Z$.
- *Preference poset of* c_i *(*$1 \le i \le m$*)*: $\prec_{c_i} = \emptyset$. Let $P(c_i)$ contain those members of $X^1 \cup X^2 \cup \overline{X}^1 \cup \overline{X}^2$ corresponding to the literal-occurrences in clause c_i.

It is easy to verify that, for any two people q, r of the opposite sex, r is proper for q if and only if q is proper for r.

Now suppose that B admits a satisfying truth assignment f. We form a matching M in I as follows. For each clause c_i in B ($1 \le i \le m$), pick any literal-occurrence $x \in X^1 \cup X^2 \cup \overline{X}^1 \cup \overline{X}^2$ corresponding to a true literal in c_i, and add (x, c_i) to M. For any x_i^r left unmatched ($1 \le i \le n$, $1 \le r \le 2$), add (x_i^r, y_i^r) to M. Similarly, for any $\overline{x}_i^r$ left unmatched ($1 \le i \le n$, $1 \le r \le 2$), add $(\overline{x}_i^r, \overline{y}_i^r)$ to M. Finally, there remain m members of $Y^1 \cup Y^2 \cup \overline{Y}^1 \cup \overline{Y}^2$ that are as yet unmatched. Add to M a perfect matching between these women and the men in Z. It is straightforward to verify that M is strongly stable in I.

Conversely suppose that I admits a strongly stable matching M. Then it is not difficult to see that $(m, w) \in M$ implies that w is proper for m and vice versa. Also, for each i ($1 \le i \le n$), c_i is matched in M to some man $x \in X^1 \cup X^2 \cup \overline{X}^1 \cup \overline{X}^2$ corresponding to an occurrence of a literal in clause c_i of B. Suppose that $x = x_i^r$

for some i $(1 \leq i \leq n)$ and r $(1 \leq r \leq 2)$ (the argument is similar if $x = \overline{x}_i^r$). Then by the strong stability of M, $(\overline{x}_i^1, \overline{y}_i^1) \in M$ and $(\overline{x}_i^2, \overline{y}_i^2) \in M$. Thus we may form a truth assignment f for B as follows: if $x = x_i^r$ then set variable x_i to have value T, otherwise if $x = \overline{x}_i^r$ then set variable x_i to have value F. Any remaining variable whose truth value has not yet been assigned can be set to T. Clearly f is a satisfying truth assignment for B. □

6 Concluding remarks and subsequent work

In this paper we have described a polynomial-time algorithm for the problem of finding a strongly stable matching, if one exists, given an instance of HRT. By constrast we have shown that the corresponding existence question becomes NP-complete for HRP.

Algorithm HRT-strong as presented in Section 2 is resident-oriented in that, given an HRT instance that admits a strongly stable matching, the algorithm constructs a resident-optimal strongly stable matching (the optimality properties of this matching are described by Theorem 2.6). A hospital-oriented counterpart to Algorithm HRT-strong appears in [21]. Given an HRT instance that admits a strongly stable matching, the algorithm outputs a hospital-optimal strongly stable matching, in which each hospital has at least as favourable a set of assignees as it can have in any strongly stable matching. This algorithm has $O(a^2)$ complexity, as is the case for Algorithm HRT-strong.

Subsequent to this work, Kavitha et al. [13] presented an algorithm that finds a strongly stable matching, or reports that none exists, in $O(ka)$ time, given an instance I of SMTI, where k is the total number of men and women in I. They also extended this to a resident-oriented algorithm for HRT, with time complexity $O(a(|R| + \sum_{h \in H} p_h))$. Additionally Malhotra [14] gave an $O(a^2)$ algorithm, an extension of Algorithm HRT-strong, that finds a strongly stable matching, or reports that none exists, given an instance I of the many-to-many Stable Marriage problem with ties (a generalisation of HRT). Using the techniques of [13], it is likely that this algorithm can be modified to run in $O(a \sum_{q \in Q} p_q)$ time, where Q is the set of participants in I. Malhotra's algorithm can also be used as a hospital-oriented algorithm for HRT under strong stability.

Given an instance I of HRT that admits a strongly stable matching M, Theorem 2.8 implies that all strongly stable matchings in I have size $|M|$. However it is possible that I could admit weakly stable matchings of sizes smaller than $|M|$ and larger than $|M|$. To show this, consider the HRT instance given in Figure 6, in which each hospital has quota 1. This instance admits two strongly stable matchings of size 5, namely $\{(r_2, h_1), (r_3, h_2), (r_4, h_4), (r_5, h_6), (r_6, h_5)\}$ and $\{(r_2, h_1), (r_3, h_2), (r_4, h_5), (r_5, h_4), (r_6, h_6)\}$, and also admits weakly stable matchings of sizes 4 and 6, namely $\{(r_2, h_1), (r_3, h_2), (r_4, h_4), (r_6, h_6)\}$ and $\{(r_1, h_1), (r_2, h_2), (r_3, h_3), (r_4, h_4), (r_5, h_6), (r_6, h_5)\}$ respectively. It is therefore of interest to consider the question of the relative sizes of strongly stable matchings and weakly stable matchings in a given instance of HRT. Scott [21] has shown that, given an HRT instance I, where M is a strongly stable matching in I and M' is a maximum cardinality weakly stable matching in I, the inequality $M' \leq \frac{3}{2}|M| - \frac{1}{2}u_M$ holds, where

r_1: h_1	h_1: r_2 r_1
r_2: $(h_1\ h_2)$	h_2: $(r_2\ r_3)$
r_3: h_2 h_3	h_3: r_3
r_4: $(h_4\ h_5)$	h_4: $(r_4\ r_5)$
r_5: $(h_4\ h_6)$	h_5: $(r_4\ r_6)$
r_6: $(h_5\ h_6)$	h_6: $(r_5\ r_6)$
Resident's preferences	Hospital's preferences

Figure 3: An HRT instance.

$u_M = \sum_{h \in H*} f_h$, also H^* is the set of hospitals that are under-subscribed in M, and f_h is the number of posts that $h \in H^*$ fills in M.

Finally, we conclude with a remark regarding the structure of strongly stable matchings, given an instance of HRT. For a given instance of SMT, it is known that the set of strongly stable matchings forms a distributive lattice, when the set is partitioned by a suitable equivalence relation [16]. Malhotra [14] has extended this result to the many-to-many Stable Marriage problem with ties, which includes HRT as a special case.

References

[1] Canadian Resident Matching Service. How the matching algorithm works. Web document available at `http://www.carms.ca/matching/algorith.htm`.

[2] P. Fishburn. Preference structures and their numerical representations. *Theoretical Computer Science*, 217:359–383, 1999.

[3] H.N. Gabow. An efficient reduction technique for degree-constrained subgraph and bidirected network flow problems. In *Proceedings of STOC '83: the 15th Annual ACM Symposium on Theory of Computing*, pages 448–456. ACM, 1983.

[4] D. Gale and L.S. Shapley. College admissions and the stability of marriage. *American Mathematical Monthly*, 69:9–15, 1962.

[5] D. Gale and M. Sotomayor. Some remarks on the stable matching problem. *Discrete Applied Mathematics*, 11:223–232, 1985.

[6] D. Gusfield and R.W. Irving. *The Stable Marriage Problem: Structure and Algorithms*. MIT Press, 1989.

[7] J.E. Hopcroft and R.M. Karp. A $n^{5/2}$ algorithm for maximum matchings in bipartite graphs. *SIAM Journal on Computing*, 2:225–231, 1973.

[8] R.W. Irving. Stable marriage and indifference. *Discrete Applied Mathematics*, 48:261–272, 1994.

[9] R.W. Irving. Matching medical students to pairs of hospitals: a new variation on a well-known 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.

[10] R.W. Irving, D.F. Manlove, and S. Scott. The Hospitals/Residents problem with Ties. In *Proceedings of SWAT 2000: the 7th Scandinavian Workshop on Algorithm Theory*, volume 1851 of *Lecture Notes in Computer Science*, pages 259–271. Springer-Verlag, 2000.

[11] R.W. Irving, D.F. Manlove, and S. Scott. Strong stability in the Hospitals/Residents problem. In *Proceedings of STACS 2003: the 20th Annual Symposium on Theoretical Aspects of Computer Science*, volume 2607 of *Lecture Notes in Computer Science*, pages 439–450. Springer-Verlag, 2003.

[12] 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.

[13] T. Kavitha, K. Mehlhorn, D. Michail, and K. Paluch. Strongly stable matchings in time $O(nm)$ and extension to the Hospitals-Residents problem. In *Proceedings of STACS 2004: the 21st International Symposium on Theoretical Aspects of Computer Science*, volume 2996 of *Lecture Notes in Computer Science*, pages 222–233. Springer-Verlag, 2004.

[14] V.S. Malhotra. On the stability of multiple partner stable marriages with ties. In *Proceedings of ESA '04: the Twelfth Annual European Symposium on Algorithms*, volume 3221 of *Lecture Notes in Computer Science*, pages 508–519. Springer-Verlag, 2004.

[15] D.F. Manlove. Stable marriage with ties and unacceptable partners. Technical Report TR-1999-29, University of Glasgow, Department of Computing Science, January 1999.

[16] D.F. Manlove. The structure of stable marriage with indifference. *Discrete Applied Mathematics*, 122(1-3):167–181, 2002.

[17] D.F. Manlove, R.W. Irving, K. Iwama, S. Miyazaki, and Y. Morita. Hard variants of stable marriage. *Theoretical Computer Science*, 276(1-2):261–279, 2002.

[18] M. Mukerjee. Medical mismatch. *Scientific American*, 276:40–41, 1997.

[19] 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.

[20] A.E. Roth. On the allocation of residents to rural hospitals: a general property of two-sided matching markets. *Econometrica*, 54:425–427, 1986.

[21] S. Scott. *A study of stable marriage problems with ties*. PhD thesis, University of Glasgow, Department of Computing Science, 2005.