Strongly Stable Matchings in Time $O(nm)$ and Extension to the Hospitals-Residents Problem

TELIKEPALLI KAVITHA
Indian Institute of Science, Bangalore, India
and
KURT MEHLHORN, DIMITRIOS MICHAIL and KATARZYNA E. PALUCH
Max-Planck-Institut für Informatik, Saarbrücken, Germany

An instance of the stable marriage problem is an undirected bipartite graph $G = (X \dot\cup W, E)$ with linearly ordered adjacency lists with ties allowed in the ordering. A matching M is a set of edges no two of which share an endpoint. An edge $e = (a, b) \in E \setminus M$ is a *blocking edge* for M if a is either unmatched or strictly prefers b to its partner in M, and b either is unmatched or strictly prefers a to its partner in M or is indifferent between them. A matching is strongly stable if there is no blocking edge with respect to it. We give an $O(nm)$ algorithm for computing strongly stable matchings, where n is the number of vertices and m is the number of edges. The previous best algorithm had running time $O(m^2)$. We also study this problem in the hospitals-residents setting, which is a many-to-one extension of the above problem. We give an $O(m \sum_{h \in H} p_h)$ algorithm for computing a strongly stable matching in the hospitals-residents problem, where p_h is the quota of a hospital h. The previous best algorithm had running time $O(m^2)$.

Categories and Subject Descriptors: G.2.2 [**Discrete Mathematics**]: Graph Theory—*Graph algorithms*

General Terms: Algorithms, Design

Additional Key Words and Phrases: Bipartite matching, level maximal, stable marriage, strong stability, ties

1. INTRODUCTION

An instance of the *stable marriage problem* is an undirected bipartite graph $G = (X \dot\cup W, E)$ where the adjacency lists of vertices are linearly ordered with ties allowed. As is customary, we call the vertices of the graph men and women, respec-

A preliminary version of this paper appeared in the Proc. 21st International Symposium on Theoretical Aspects of Computer Science (STACS), 2004 [KMMP04].
Partially supported by the Future and Emerging Technologies programme of the EU under contract number IST-1999-14186 (ALCOM-FT).
Authors' addresses: Telikepalli Kavitha, CSA Department, Indian Institute of Science, Bangalore 560012, India ; email: kavitha@csa.iisc.ernet.in; Kurt Mehlhorn, Dimitrios Michail, Katarzyna E. Paluch, Max-Planck-Institut für Informatik, Stuhlsatzenhausweg 85, 66123 Saarbrücken, Germany; email: {mehlhorn, michail, paluch}@mpi-inf.mpg.de.

x_1:	w_1, w_2	w_1:	x_2, x_1
x_2:	$\{w_1, w_2\}$	w_2:	x_2, x_1

Fig. 1. Both women prefer x_2 to x_1. The man x_1 prefers w_1 to w_2 and x_2 is indifferent between the women. The instance is complete and hence any strongly stable matching must match all men and all women. The matching $\{(x_1, w_1), (x_2, w_2)\}$ is not strongly stable since w_1 prefers x_2 to x_1 and x_2 is indifferent between w_1 and w_2. The matching $\{(x_1, w_2), (x_2, w_1)\}$ is not strongly stable since w_2 prefers x_2 to x_1 and x_2 is indifferent between w_1 and w_2.

tively[1]. Each person seeks to be assigned to a person of the opposite sex and her/his preference is given by the ordering of her/his adjacency list. In a's list, if the edges (a, b) and (a, b') are tied, we say that a is *indifferent* between b and b', and if the edge (a, b) strictly precedes (a, b'), we say that *a prefers b to b'*. Preference lists are not necessarily complete, meaning that each person may rank only a subset of the persons of the opposite sex. Note that preference lists are *consistent* in the sense that a person a appears in b's list if and only if b appears in a's list. Consistency of preference lists will be assumed throughout. A pair (a, b) is *acceptable* if each of a and b are on the other's preference list. All edges $e \in E$ correspond to acceptable pairs. We use n for the number of vertices and $m = |E|$ for the number of edges. A stable marriage problem is called *complete* if G is the complete bipartite graph.

A *matching* M is a set of edges no two of which share an endpoint. If $(a, b) \in M$ we call b the *partner* of a. A matching M is *strongly stable* if there is no edge $(a, b) \in E \setminus M$ (called a *blocking edge*) such that by becoming matched with each other, one of a and b (say, a) is better off and b is not worse off. A person a is better off means that she/he is either unmatched in M or strictly prefers b to her/his partner in M, and a is not worse off if she/he is either better off or is indifferent between b and her/his partner in M. In other words, a would prefer to be matched with b and b would not object to the change.

In this paper, we consider the problem of computing a strongly stable matching. One of the motivations for this form of stability is the following. Suppose we have a matching M and there exists a blocking edge (a, b). Suppose it is a that becomes better off by becoming matched to b. It means that a is willing to take some action to improve her/his situation and as b's situation would not get worse, she/he might yield to a. If there exists no such edge, then M can be considered to be stable since no two vertices a and b, where (a, b) is in $E \setminus M$, improve by changing their present state and becoming matched with each other. Observe that not every instance of the stable marriage problem has a strongly stable solution. Figure 1 shows an instance of the stable marriage problem with no strongly stable solution.

The stable marriage problem can also be studied in the more general context of hospitals and residents. This is a many-to-one extension of the classical men-women version. An instance of the hospitals-residents problem is again an undirected bipartite graph $(R \dot\cup H, E)$ with linearly ordered (allowing ties) adjacency lists. Each resident $r \in R$ seeks to be assigned to exactly one hospital, and each hospital $h \in H$ has a specified number p_h of posts, referred to as its *quota*. A matching M is a set of edges, no two of which share the same resident, and at most p_h of the

[1]We use x, x', x'' to denote men, w, w', w'' to denote women, a, a', b, b' to denote persons of either sex.

edges in M can share the hospital h.

A blocking edge to a matching is defined similarly as in the case of men-women. An edge $(a, b) \in E \setminus M$ is a blocking edge to M if a would prefer to be matched with b and b would not object to the change. A matching is strongly stable if there is no blocking edge with respect to it. Again we assume that the preference lists are consistent. We also consider the problem of computing a strongly stable matching in this setting. Observe that the classical stable marriage problem is a special case of this general problem by setting $p_h = 1$ for all hospitals.

In this paper we give an $O(nm)$ algorithm to determine a strongly stable matching for the classical stable marriage problem or report that no such matching exists. We also give an $O(m \sum_{h \in H} p_h)$ algorithm to compute a strongly stable matching in the hospitals-residents problem or report that no such matching exists. The previous results for computing strongly stable matchings are as follows. Irving [Irv94] gave an $O(n^4)$ algorithm for computing strongly stable matchings for men-women in complete stable marriage instances where there are equal number of men and women. In [Man99] Manlove extended the algorithm to incomplete bipartite graphs; the extended algorithm has running time $O(m^2)$. In [IMS03] an $O(m^2)$ algorithm was given for computing a strongly stable matching for the hospitals-residents problem.

Our new algorithm for computing a strongly stable matching for the classical stable marriage problem can be viewed as a specialization of Irving's algorithm, i.e., every run of our algorithm is a run of his, but not vice versa. We obtain the improved running time by introducing the concept of *levels*. Every vertex has a level associated with it; the level of a vertex can change during the algorithm. We use the levels of vertices to search for special matchings which are *level-maximal* and this makes the running time of the algorithm $O(nm)$. We also use the above ideas in the hospitals-residents problem and obtain an improvement over [IMS03]. We do not assume that the input graphs are complete bipartite graphs.

The stable marriage problem has great practical significance. The hospitals-residents problem has widespread application to matching schemes that match graduating medical students to hospital posts [Irv98; Roth84; CRMS]. The classical results in stable marriage are the Gale/Shapley theorem and algorithm [GS62]. In these instances, it is assumed that there are no ties and each person ranks all the members of the opposite sex in strict order of preference. Then there always exists at least one stable matching and the Gale/Shapley algorithm finds one in $O(n^2)$ time. However, when ties are allowed, finding a stable matching seems more difficult. There are, in fact, two more notions of stability in matchings. The matching M is said to be *weakly stable* (or, *super-stable*) if there does not exist a pair $(a, b) \in E \setminus M$ such that by becoming matched to each other both a and b are better off (respectively, neither of them is worse off). The problem of finding a weakly stable matching of maximum size was recently proven to be NP-hard [IMMM99]. There is a simple $O(n^2)$ algorithm [Irv94] to determine if a super-stable matching exists or not and to compute one, if it exists. Gusfield and Irving [GI89] covers plenty of results obtained in the area of stable matchings.

In Section 2 we present our $O(nm)$ algorithm for strongly stable matchings for the classical stable marriage problem. In Section 3 we present our $O(m \sum_{h \in H} p_h)$

algorithm for the hospitals-residents problem. We conclude with some open problems in Section 4.

2. THE ALGORITHM FOR STRONGLY STABLE MARRIAGE

We review a variant of Irving's algorithm [Irv94] in Section 2.1 and then describe our modifications in Section 2.2. Algorithm 1 contains a concise write-up of our algorithm.

2.1 Irving's algorithm

We review a variant of Irving's algorithm for strongly stable matchings. The algorithm proceeds in phases and maintains two graphs G' and G_c; G' and G_c are subgraphs of G. G_c is the current graph of edges E_c in which we compute maximum matchings and G' is the graph of edges E' not considered relevant yet. In each phase, a certain subset of the edges of G' is moved to G_c. Also edges get deleted from G' and G_c. We use $\mathcal{E}_i$ to denote the edges moved in phase i and $\mathcal{E}_{\le i}$ to denote the edges moved in the first i phases. Initially, we have $G' = G$ and $E_c = \emptyset$.

At the beginning of phase i, $E_c \subseteq \mathcal{E}_{<i}$ and we have a maximum matching M in G_c. Also, if a man is free with respect to M then no edges of E_c are incident to him. Let $\mathcal{E}_i$ consist of the most preferred edges[2] in E' of each free man. We say that every free man in succession proposes to all women currently at the top of his list. When a woman receives a proposal from a man x, she deletes all strict successors of x from E' and E_c. This may also remove edges in M.

Observe, that the rules for adding and deleting edges guarantee that if $(a, b) \in E_c$ and $(a, b') \in E_c$ then a is indifferent between b and b'. For a free man x, all his top choices are moved to E_c and hence edges in E' go to strictly inferior women. A woman keeps only the best proposals made to her and hence edges in E' go either to strictly superior men or to men tied with her choices in E_c.

Next we extend M to a maximum matching in E_c. During this process, further edges may be deleted. We iterate over the free men in an arbitrary order. Let x be any free man. If there is an augmenting path starting at x, we use it to increase the cardinality of the matching. Otherwise, let Z be the set of men reachable from x by alternating paths and let $N(Z)$ be the set of women adjacent to Z in E_c. For each woman $w \in N(Z)$ we delete[3] all *lowest ranked* edges in $E_c \cup E'$ (i.e, all edges tied for the last place in the list restricted to $E_c \cup E'$) incident to her. This is at least one edge $(x'', w) \in E_c$ and zero or more edges $(x', w) \in E'$.

At the end of the phase, we have a maximum matching in E_c. Also, every free man x is isolated in G_c since the edges incident to x were removed when we searched for an augmenting path starting from x.

The algorithm terminates when all free men have run out of proposals. Let M be the final matching and let G_c be the final graph. Then M is a maximum matching in G_c and all free men are isolated in G_c and G'. M is a strongly stable matching in G if no woman that was ever non-isolated in G_c during the execution of the

[2]Recall that $E' \subseteq E$ and that adjacency lists are linearly ordered with ties allowed.
[3]We slightly deviate from Irving's algorithm here. We delete edges whenever we identify a free man which cannot be matched. Irving first computes a maximum matching in E_c and then deletes edges.

```
Set phase number $i = 1$, $E' = E$ and $E_c = \emptyset$ and each man to be free
$M = \emptyset$
repeat
    while $\exists$ a free man $x$ who has a non-empty list do
        move all top choice edges $e = (x, w)$ of $x$ in $E'$ to $E_c$ and delete all
        edges $(x', w)$ from $E' \cup E_c$ which $w$ ranks strictly after $e$
    end
    Let $\mathcal{E}_i$ be the edges moved to $E_c$
    forall free men $x$ w.r.t. $M$ do
        if an alternating path from $x$ to a free woman exists then
            let $w$ be a free woman [of maximal level] reachable from $x$ by an
            alternating path and let $p$ be an alternating path from $x$ to $w$
            $M = M \oplus p$
        else
            let $Z$ be the set of men reachable from $x$ by alternating paths and
            let $N(Z)$ be the women adjacent to them in $E_c$
            forall women $w \in N(Z)$ do
                delete all lowest ranked edges in $E_c \cup E'$ incident to $w$
            end
        end
    end
    $i = i + 1$
until all free men have run out of proposals
declare $M$ strongly stable if every woman that was ever non-isolated in $G_c$
during the execution of the algorithm is matched in $M$. Otherwise, there is no
strongly stable matching
```

Algorithm 1: Two algorithms for strongly stable marriage. The algorithms differ by the phrase [*of maximal level*]. Without the phrase, the algorithm may augment the current matching along any augmenting path and the running time is $O(m^2)$. With the phrase, an augmenting path to a woman of maximal level (see Section 2.2) must be used. The running time improves to $O(nm)$.

algorithm is free with respect to M.

The following Lemmas were shown in [Irv94; Man99].

Lemma 2.1. *If an edge $e = (x, w)$ is deleted during the execution, it cannot block any matching output by the algorithm.*

Proof. Let M be a matching output by our algorithm. Suppose e was deleted in phase i. This means that w had some edge incident to her during phase i in G_c. So, w has to be matched in M. Otherwise, our algorithm would have declared that there is no strongly stable matching. So, in M, w has to be paired to a man she prefers to x. And hence, e is non-blocking. □

Lemma 2.2. *If the algorithm returns a matching, the matching is strongly stable.*

Proof. Assume otherwise, i.e., the algorithm returns a matching M and yet a blocking edge $e = (x, w)$ exists. Then e was never deleted during the execution by Lemma 2.1 and hence $e \in E_c \cup E'$ when the algorithm terminates. By the termination condition and since M is declared strongly stable, x must be matched in M. Since e blocks M, x cannot prefer his partner in M to w, and since the edge connecting x with his partner in M is not deleted, x cannot prefer w to his partner in M. Thus x is indifferent between w and his partner in M. When x proposed to his partner in M, he also proposed to w. At this time (x, w) was added to G_c. Thus w was non-isolated node during the execution of the algorithm and hence is matched in M. The partner of w is tied with x in w's preference list and hence e is non-blocking. □

Let us call an edge *strongly stable*, if it belongs to some strongly stable matching.

LEMMA 2.3. *No strongly stable edge is ever deleted during the execution of the algorithm.*

PROOF. Assume otherwise, and let $e = (x, w)$ be the first strongly stable edge deleted by the algorithm. Let M be a strongly stable matching containing e. We distinguish cases according to why e was deleted.

Assume first that e is deleted because w receives a better proposal, say from x'. Then w strictly prefers x' to x. When x' proposes to w, all edges (x', w') with w' strictly preceding w in x''s preference list have been discarded already and hence none of them can be strongly stable. We conclude that the partner of x' in M is either tied with w or after w in the preference list of x', and x' is before x in w's preference list. Thus (x', w) blocks M, a contradiction.

Assume next that at some point of the execution we have a free man x_0 from which no augmenting path starts. Let E_c be the edge set of the current graph when we search for augmenting paths from x_0 and let M_c be the current matching. Let Z be the set of men reachable from x_0 by alternating paths and let $N(Z)$ be the women adjacent to them in E_c. Then $|Z| = |N(Z)| + 1$ and every woman in $N(Z)$ is matched with a man in Z by M_c. Also $w \in N(Z)$ and e is a lowest ranked edge incident to w. Let $\widetilde{E}$ be the set of lowest ranked edges incident to women in $N(Z)$, i.e., $\widetilde{E}$ is the set of edges which the algorithm removes after discovering the fact that no augmenting path starting at x_0 exists.

Consider $M \cap \widetilde{E}$, let U' be their female endpoints and let Z' be their male endpoints. Then $w \in U'$ and hence $U' \neq \emptyset$. Also, M matches men in Z' and women in U' with each other, Thus $|Z'| = |U'| \leq |N(Z)| < |Z|$.

We next show the existence of an edge $e' = (x', w') \in E_c$ with $x' \in Z \setminus Z'$ and $w' \in U'$ and then show that it blocks M. Assume that M_c contains no such edge. Then M_c pairs the women in U' with men in Z' and since $|Z'| = |U'|$, M_c pairs the men in Z' with the women in U'. We conclude $x_0 \in Z \setminus Z'$ as x_0 is free. Consider the alternating path (with respect to M_c) from x_0 to w and let $e' = (a, b)$ be the first edge on this path with $b \in U' \cup Z'$. If $b \in Z'$, e' is a matching edge and hence $a \in U'$, contradicting that (a, b) is the first edge on this path with $b \in U' \cup Z'$. Thus $b \in U'$ and $a \in Z \setminus Z'$ and we have shown the existence of the desired edge.

Since $w' \in U'$, this implies that w' is matched in M to a man tied with x' in her preference list. And in M, x' is either unmatched or matched to a woman strictly after w' in his preference list. To see this note that x' cannot be matched in M with any edge that has already been deleted, by the assumption that (x, w) is the first strongly stable edge deleted. The only remaining case is if x' is matched in

M with a woman w'' who is tied with w' in his preference list and edge (x', w'') is not already deleted. But then edge $(x', w'') \in \widetilde{E}$, a contradiction to the fact that $x' \in Z \setminus Z'$. We conclude that (x', w') blocks M, a contradiction. □

Lemma 2.4. *Assume a strongly stable matching exists. Then the algorithm returns a strongly stable matching. Also all strongly stable matchings have the same cardinality and match the same set of men and women.*

Proof. Let M_0 be a strongly stable matching. By Lemma 2.3, no edge of M_0 is ever deleted and hence $M_0 \subseteq E_c \cup E'$ at termination. Let M be the final matching computed by the algorithm. We show first that $|M| = |M_0|$ and that both matchings match the same set of men and women.

All men free in M are isolated in $E_c \cup E'$ and hence are free in M_0. Thus every man matched by M_0 is also matched by M and hence $|M_0| \leq |M|$. Assume that there is a woman w which is non-isolated in G_c and free with respect to M_0. Let $(x, w) \in E_c$ be an edge incident to w. Then (x, w) blocks M_0 since x cannot strictly prefer its partner in M_0 to w. Thus M_0 matches all non-isolated woman of G_c and M certainly cannot match more women. Thus $|M| \leq |M_0|$. We conclude that both matchings have the same cardinality and match exactly the non-isolated nodes of G_c. This proves the second claim.

Assume next that the algorithm declares that M is not stable. By the above, M matches all non-isolated nodes of G_c.

Suppose there is a woman w which is free in M and was non-isolated during the execution of the algorithm. Then w is also free in M_0. Let $e = (x, w)$ be the last edge incident to w during the execution. After e was deleted, no edge in $E_c \cup E$ incident to x is strictly before e in the preference list of x. Thus e blocks M_0. □

The algorithm runs in $O(m^2)$ time since the cost summed over all phases is $O(m \cdot (1 + \text{number of successful augmenting path computations}))$. The number of augmenting path computations is at most m since a matched man becomes free only if the matching edge incident to him is deleted. For a detailed analysis we refer the interested reader to [IMS03].

2.2 The New Algorithm

We show how to modify the algorithm so that it runs in time $O(nm)$. Our method maintains *level-maximal* matchings and uses level-maximal augmenting paths.

The running time of the algorithm for finding a strongly stable matching is actually the time spent on looking for augmenting paths. We introduce the notion of the *level* of an edge and the *level* of a vertex which will help us to search for augmenting paths in a streamlined manner. The vertices with higher levels are given precedence when searching for augmenting paths. When we search for augmenting paths with this precedence and we succeed in finding one, then we can show that the level numbers of all the edges traversed are at least the level number of the unmatched vertex at the end of the augmenting path. This allows us to bound the total number of edges traversed in our search for augmenting paths.

Definition 2.5. Let $\mathcal{E}_i$ be the edges added to G_c in phase i and define the *level* $l(e)$ of an edge to be the phase when this edge was first added to G_c. Edges never added to G_c have no level assigned to them.

So, the set of edges ever added to G_c consists of the disjoint union $\mathcal{E}_1 \cup \mathcal{E}_2 \cup ... \mathcal{E}_r$, where r is the total number of phases in the algorithm. Note that $r \leq m$.

Definition 2.6. Define the *level* $l(v)$ of a vertex v to be the minimum level of the edges in G_c incident to v. The level of an isolated vertex is undefined.

Definition 2.7. The *level* $l(M)$ of a matching M is the sum of the levels of the matched women. A matching M is *level-maximal* if $l(M) \geq l(M')$ for any matching M' which matches the same men.

The following lemma is immediate.

LEMMA 2.8. *For a man, all incident edges in G_c have the same level. All women adjacent to a man of level i have level at most i. When a woman loses an incident edge in E_c she loses all her incident edges in E_c.*

We next characterize level-maximal matchings and show how to maintain them.

LEMMA 2.9. *A matching M is level-maximal iff there is no alternating path from a free woman to a woman of lower level.*

PROOF. If there is such an alternating path, then the endpoint of the path is a matched woman. Augmentation increases the level of the matching and does not change the set of matched men.

For the converse, assume that M is not level-maximal. Let M' be level-maximal and matching the same men. Then $M \oplus M'$ is a set of alternating paths and cycles. Augmenting a cycle does not change the level sum. Thus there must be at least one path whose augmentation to M increases the level sum. Since the degree of every man in $M \oplus M'$ is either zero or two, the path must connect two women, one free in M and one free in M'. □

LEMMA 2.10. *If M is level-maximal, x is a free man with respect to M, w is a woman of maximal level reachable from x by an augmenting path and p is an augmenting path from x to w, then $N = M \oplus p$ is level-maximal.*

PROOF. Let us look at an alternating path p' from a free woman w' to a matched woman w'' (all with respect to N). We will show that $l(w') \leq l(w'')$ and therefore by Lemma 2.9 it follows that N is level-maximal.

If p' does not contain any edge from p, then p' was an alternating path from a free woman w' to a matched woman w'' in M. Since M is level-maximal, by Lemma 2.9, $l(w') \leq l(w'')$.

Let us then assume that p' contains some edge(s) from p.

Let x' denote the first vertex on the path p' that belongs to p, which we meet while traversing p' from w'. Let e' denote the first edge belonging to p (Figure 2). The vertex x' must be a man because all edges incident to vertices on p that do not belong to p cannot belong to the matching N - so the edge of p' that leads to x' is unmatched by N and recall that we started the traversal of p' from the unmatched woman. So, e' is matched by N. Let us now look at this part of p that has x' at its one end and does not contain e'. It has the man x, that was free in M, at its other end (possibly $x = x'$). Since $M \oplus p = N$, the matched edges of path p were exactly the other way around before the augmentation, that is, those edges that

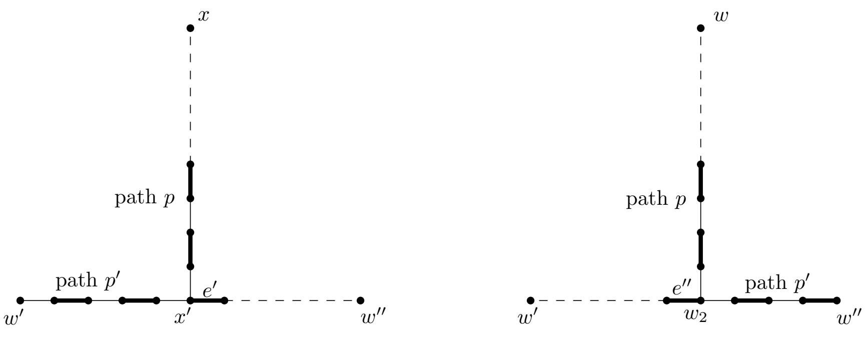

Fig. 2. The thick edges belong to the matching N

are now present in the matching N were not present in the matching M and vice versa. It means that w' was reachable by an alternating path from x in M. Since w is a woman of maximal level reachable from x in M, $l(w') \leq l(w)$.

Analogously, let w_2 denote the first vertex on the path p' that belongs to p which we meet when we traverse p' beginning from the matched woman w''. Let e'' denote the first edge belonging to p (Figure 2). It is not difficult to notice that w_2 must be a woman. Now, if we look at that part of p that has w_2 at one end and does not contain e'', then we notice that it has the woman w at its other end (possibly $w = w_2$). Thus in M there existed an alternating path from the free woman w to the matched woman w'' and hence, by Lemma 2.9, $l(w) \leq l(w'')$.

Combining the observations, we get that $l(w') \leq l(w'')$. $\square$

Lemma 2.11. *M is a level-maximal matching at all times of the execution.*

Proof. We use induction on the steps of the algorithm. Initially, M is empty and therefore level-maximal. For the induction step assume that M is level-maximal at the beginning of phase i.

First, every free man proposes to the women at the top of his list. This introduces the edge set $\mathcal{E}_i$. The level of non-isolated women does not change, the level of women previously isolated and not isolated anymore is set to i. M is still level-maximal. Assume otherwise, then there must be an alternating path from a free woman to a woman of lower level. This path must use one of the new edges. The new edges are incident to free men, a contradiction.

Every woman keeps only her best proposals. For a particular woman w this has one of two effects: either she does not drop any incident edge or she keeps only edges in $\mathcal{E}_i$ (not necessarily all of them). The matching M may be reduced in size. Let us use M' to denote the resulting matching. We claim that it is level-maximal. Assume otherwise, then there must an alternating path p from a free woman to a woman of lower level. It cannot use any of the new edges since new edges are incident to free men. Thus p can use only old edges. Also p cannot start at a woman of level i since only new edges are incident to such a woman. Thus p starts at a woman of level less than i and hence the woman is free with respect to M. Since $M' \subseteq M$, p is alternating with respect to M, a contradiction to the level-maximality of M.

Next, we consider the free men in turn and search for augmenting paths. Let x

be a free man.

If no augmenting path starting at x exists, let Z be the set of men reachable by alternating paths from x and let $N(Z)$ be their neighbors. Then $|Z| > |N(Z)|$. We delete all lowest rank edges incident to the women in $N(Z)$. This may decrease the size of the matching. The matching clearly stays level-maximal.

If an augmenting path exists, let p be an augmenting path to a woman of maximal level. We use p to increase the cardinality of the matching. By Lemma 2.10, the resulting matching is level-maximal. $\square$

2.3 The Search for Augmenting Paths and the Analysis

We come to the implementation of the search for augmenting paths and the analysis.

Let x be a free man. We need to determine a free woman w of maximal level that is reachable from x and an augmenting path from x to w. Let p be such an augmenting path. Then all women on this path have level at least $l(w)$ by Lemma 2.9. Note that $l(w) \leq l(x)$. This is because all the women adjacent to x have level at most $l(x)$, so if w is adjacent to x, then $l(w) \leq l(x)$. If w is not adjacent to x and if $l(w) > l(x)$, then p contains an alternating path from a free woman of higher level (that is, w) to a matched woman of lower level (the neighbor of x). This contradicts the level-maximality of the matching. Note that the same holds for each man x' in p.

We organize the search in rounds $l(x)$, $l(x) - 1$, $l(x) - 2$, ... In round j, we explore all augmenting paths starting in x and exploring only edges out of vertices of level j or larger. We stop in round j when a free woman of level j is reached by the search or if the alternating tree rooted at x has reached its full size. In the former case, j is the maximal level of a woman reachable from x by an augmenting path. In the latter case, no free woman is reachable from x. If the search has not stopped yet, the frontier of the search consists of women of level less than j. In the next round, we continue the search from all women of level $j - 1$ in the frontier.

In order to find these women, we maintain an array A of buckets (= linear lists) which implements a simple priority queue. All buckets are initially empty. At the beginning of round j, bucket B_l, $l \leq j$ contains the women of level l in the frontier. We also keep an (unsorted) list of the non-empty buckets and the total number of women contained in the buckets. We initialize the bucket structure by putting the neighbors of x into the appropriate buckets and setting j to $l(x)$. In round j, we continue the search from the women in bucket j. If the bucket is empty and the number of unexplored women is positive, we decrease j by one. If the bucket is empty and the number of unexplored women is zero, we stop. There is no augmenting path starting at x (failure). If the bucket is non-empty, let w be a woman in the bucket. We remove w from the bucket. If w is free, we stop (success): w is the highest ranked woman reachable from x. If w is matched, we explore alternating paths from w (starting with matched edges) until a woman of level less than j is reached. These women are then added to their appropriate buckets. When the search stops, we empty all buckets using the list of non-empty buckets.

Let $j(x)$ be the minimal bucket index from which we remove a woman. In the case of failure this is the minimal level of a woman reachable from x and in the case of success this is the maximum level of a woman reachable from x by an augmenting

r_1:	$\{h_1, h_2\}$	h_1:	r_2, r_1
r_2:	$\{h_1, h_2\}$	h_2:	r_2, r_1

Fig. 3. When h_1 and h_2 are regarded as the same hospital h with quota 2, then there is a strongly stable matching which matches both r_1 and r_2 to h. However, when two copies of h are made, say, h_1 and h_2 as shown here, then this instance becomes similar to Figure 1 where there is no strongly stable matching.

path.

The time for the search from x is proportional to the number k of edges explored in the search plus $l(x) - j(x) + 1$. We charge this cost as follows:

In the case of failure we charge one unit each to each edge deleted (this accounts for k) and we charge $l(x) - j(x) + 1$ to the minimum level woman w reachable from x. The first kind of charges adds up to m since every edge is deleted at most once. The second kind of charge is less than the difference of the current level of w and the next level of w. Since the maximum level is m, for any woman the total charge of the second kind is bounded by m. We conclude that the total cost of unsuccessful searches is $O(nm)$.

In the case of success, we charge both costs to w. Observe that all edges explored have level at least $l(w)$ $(= j(x))$ and at most i (= the phase number) and that the level of w jumps to at least $i + 1$ if it ever becomes free again. Thus every edge can be charged at most once to w. Also $l(x) - j(x) + 1$ is bounded by the difference between the current level of w and the next level of w. Thus the total charge to w is bounded by $O(m)$. The total cost of successful searches is therefore bounded by $O(nm)$.

Theorem 2.12. *Strongly stable matchings for the classical stable marriage problem can be computed in time $O(nm)$.*

Note that the running time of our strongly stable matching algorithm is actually $O(|W|m)$ since the total cost of all unsuccessful searches and augmentations is shared by women and the cost charged to a particular woman sums to at most m over all phases. So, if $|W| \ll |X|$ or $|X| \ll |W|$ (we reverse the roles of men and women and it is free women who propose in every phase and it is men who pay for the augmentations), we can bound the running time of our algorithm by $O(\min(|X|, |W|) \cdot m)$.

3. EXTENSION TO HOSPITALS-RESIDENTS

Recall that the hospitals-residents problem is a many-to-one extension of the classical stable marriage problem. Observe that replacing each hospital h with p_h women with preference lists identical to h and running the strongly stable matching algorithm for men-women does not work. This is because there might be a strongly stable matching in the hospitals-residents problem but no strongly stable matching in the reduced men-women instance. Figure 3 contains such an instance.

We present an $O(m \sum_{h \in H} p_h)$ algorithm for computing a strongly stable matching for the hospitals-residents problem. Our algorithm is based on the algorithm in [IMS03] which is an $O(m^2)$ algorithm. We obtain the improved running time by restricting again all augmentations to result in level-maximal matchings.

3.1 The Algorithm of Irving et al.

We first review a variant of the algorithm in [IMS03] and then present our modified algorithm. The algorithm in [IMS03] generalizes the ideas used for computing strongly stable matchings in [Irv94; Man99] to the hospitals-residents problem. A summary of the algorithm along with our modifications is presented in Algorithm 2.

As in the case of the stable marriage problem, the algorithm proceeds in phases. In any phase, every free resident proposes to all hospitals currently at the top of her/his list and residents become *provisionally assigned* to hospitals. Each hospital h can accommodate up to p_h residents, and it needs to keep only the best p_h proposals made to it but if there is a tie in the last place of its list (called the *tail*), then h can be provisionally assigned to $> p_h$ residents. We introduce a few terms:

—A hospital is said to be *over-subscribed*, *under-subscribed* or *fully subscribed* if it is provisionally assigned a number of residents greater than, less than, or equal to its quota, respectively.

—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).

—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.

The algorithm maintains two graphs G' and G_c which are subgraphs of G. G_c is called the *provisional assignment graph* with edge set E_c and G' is the graph of edges E' not considered yet. During the execution of the algorithm, residents become provisionally assigned to hospitals which means that edges are moved from G' to G_c. The algorithm proceeds in the same way as the algorithm in Section 2.1, by deleting edges $e = (r, h)$ which cannot belong to any strongly stable matching.

Reduced Assignment Graph. We maintain a graph $G_r \subseteq G_c$, called the *reduced assignment graph*. The residents who appear in G_r are those that are not bound to any hospital (we call such residents unbound). So, for any hospital h, the edges incident to h in G_r are to the unbound residents, and hence are at the tail of h's list. Each hospital h has a reduced quota p'_h in the reduced assignment graph, which is the difference between the original quota p_h and the number of residents bound to h. So, the vertices of G_r are the set of unbound residents and the set of hospitals which are the neighbors of the unbound residents. The reduced assignment graph of phase i is denoted as $G_r^{(i)}$.

Now the algorithm is very similar to the algorithm for strongly stable marriage, except that we compute maximum matchings in the reduced assignment graph. Initially, $G' = G$; $E_c = \emptyset$; all the residents are free and $G_r^{(0)}$ is the empty graph. At the beginning of phase i, we have a maximum matching M in $G_r^{(i-1)}$. If a resident is free with respect to M, then he is isolated in $G_r^{(i-1)}$. Then we move the edges corresponding to the topmost choices of every free resident from E' to E_c. This denotes free residents being provisionally assigned to hospitals. Whenever a hospital h becomes fully or over-subscribed, then we delete all edges (r, h), where r is dominated on h's list, from G' and G_c. The reduced assignment graph $G_r^{(i)}$ is computed from $G_r^{(i-1)}$. Observe that an edge (r, h) can change state from bound (r is bound to h) to unbound (r is not bound to h) but not vice versa. If a new edge that gets added to G_c corresponding to one of the top choices of a free resident in $G_r^{(i-1)}$ is a bound edge, then it could cause some bound edges to become unbound or it could cause some edges to get deleted. Any edge of $G_r^{(i-1)}$ that is not deleted from G_c continues to remain in $G_r^{(i)}$. The change of state of an edge (r, h) from bound to unbound need not make the resident r unbound unless (r, h) was the only bound edge incident to r and now (r, h) has changed state to unbound. Then r, which was not present in $G_r^{(i-1)}$, starts appearing in $G_r^{(i)}$. (We discuss the cost of computing $G_r^{(i)}$ from $G_r^{(i-1)}$ in Section 3.3.) Then we extend M in $G_r^{(i)}$ to match all the unmatched residents.

Augmenting path. In the hospitals-residents setting, a hospital h is considered free in G_r if it is not matched up to its reduced quota p'_h. An alternating path from a free resident to a free hospital is considered an augmenting path.

We iterate over the free residents in an arbitrary order. Let r be any free resident. If there is an augmenting path starting at r, we use it to increase the cardinality of the matching. Otherwise, let Z be the set of residents reachable from r by alternating paths and let $N(Z)$ be the set of hospitals adjacent to Z in E_c. For each hospital $h \in N(Z)$ we delete all lowest ranked edges in $E_c \cup E'$ incident to it.

At the end of the phase, we have a maximum matching M in $G_r^{(i)}$. Also, every free resident is isolated in G_c since the edges incident to her/him were removed when we searched for an augmenting path starting from it. When all free residents have run out of proposals, we need to find a *feasible* matching M' in G_c which contains the maximum matching M in G_r and matches every bound resident r to a hospital that r is bound to. M' is a strongly stable matching if a hospital that was fully or over-subscribed at some point in the execution of the algorithm is fully matched in M' or a hospital that was always under-subscribed has a number of assignees in M' equal to its degree in G_c. The correctness of this algorithm follows from similar proofs of correctness as in Section 2.1 and we refer the reader to [IMS03] for the exact proofs.

3.2 Our Modifications

Let us extend our definitions in order to capture the somewhat different structure of the hospitals-residents problem.

Definition 3.1. Define the level of an edge e, $l(e)$, to be the phase that e is added to the reduced assignment graph G_r. [4]

Definition 3.2. Define the level of a vertex v, $l(v)$, to be the minimum level of the edges incident to v. If v does not belong to G_r, then its level is undefined.

Definition 3.3. Define the level of a matching M, $l(M)$, to be the sum over all hospitals of the level of a hospital multiplied by the number of edges that this hospital is matched with.

Definition 3.4. A matching M is level-maximal if $l(M) \geq l(M')$ for any matching M' which matches the same residents.

[4]Note that an edge appears in G_r at some phase which might not necessarily be the phase that this edge appeared in G_c.

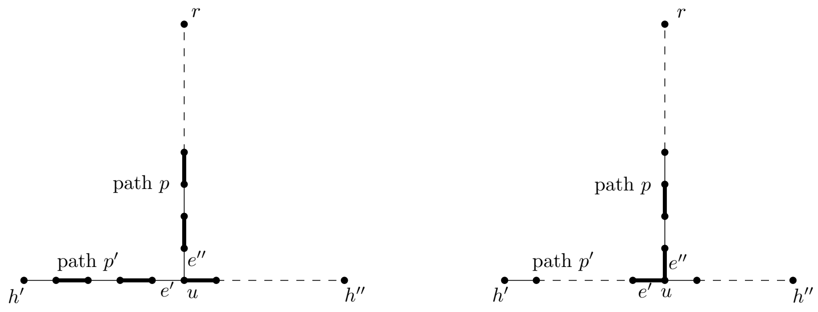

Fig. 4. The two cases corresponding to whether u is a resident or a hospital.

The following lemmas show how to maintain a level maximal matching.

Lemma 3.5. *A matching M is level-maximal iff there is no alternating path starting with an unmatched edge from a free hospital to a hospital of lower level.*

Proof. If such a path exists, then observe that the endpoint of the path is a hospital matched with a resident. Augmentation increases the level of the matching and does not change the set of matched residents.

For the converse, assume that M is not level-maximal. Let M' be a level-maximal matching that matches the same residents. We can decompose $M \oplus M'$ into a set of alternating paths and cycles, all of them edge-disjoint. Augmenting a cycle does not change the level sum. Thus there must be at least one path whose augmentation to M increases the level sum. Since the degree of every resident in $M \oplus M'$ is either zero or two, the path must connect two hospitals, one free in M and one free in M'. □

The following lemma and its proof are very similar to Lemma 2.10.

Lemma 3.6. *If M is level-maximal, r is a free resident with respect to M, h is a hospital of maximal level reachable from r by an augmenting path and p is an augmenting path from r to h, then $N = M \oplus p$ is level-maximal.*

Proof. Let us look at an alternating path p' starting with an unmatched edge from a free hospital h' to a hospital h'' (all with respect to N). We will show that $l(h') \le l(h'')$ and therefore (by Lemma 3.5) N is level-maximal.

If p' does not contain any edge from p, then p' was an alternating path starting with an unmatched edge from a free hospital h' to a hospital h'' in M. Since M is level-maximal, by Lemma 3.5, $l(h') \le l(h'')$.

Let us then assume that p' contains some edge(s) from p.

Let u denote the first vertex on the path p' that belongs to p, which we meet while traversing p' from h'. We cannot argue here like in the proof of Lemma 2.10 that u has to be a resident. However, it does not matter. If u is a resident (possibly $r = u$), it means that the edges e' and e'' (Figure 4) are unmatched in N and if u is a hospital, it means that both e' and e'' are matched in N. Let us look at that part of p that has u at its one end and the resident r at its other end, that was free in M. Since $M \oplus p = N$, the matched edges of path p were exactly the other way around before the augmentation, that is, those edges that are now present in the matching N were not present in the matching M previously and vice versa. Independent of whether u is a resident or a hospital, it follows that h' was reachable by an alternating path from r in M. Since h is a hospital of maximal level reachable from r by an augmenting path, it follows that $l(h') \leq l(h)$.

Analogously, let u' denote the first vertex on the path p' that belongs to p which we meet when we traverse p' beginning with a matched edge from the hospital h''. It is not difficult to see that u' must be a hospital (possibly $u' = h$). Let us look at that part of p which has u' at one end and the hospital h at its other end. This path added to the portion of p' between u' and h'' produces an alternating path in M starting with a free edge from the free hospital h to the hospital h''. Hence, by Lemma 3.5, $l(h) \leq l(h'')$.

Combining the observations, we get that $l(h') \leq l(h'')$. $\square$

Lemma 3.7. *M is a level-maximal matching at all times of the execution.*

Proof. We use induction on the steps of the algorithm. Initially, M is empty and therefore level-maximal. For the induction step assume that M is level-maximal at the beginning of phase i. Free residents propose, and this can introduce two types of edges in G_r: (a) edges from G', (b) residents who were bound might become *unbound* and so they appear now in $G_r^{(i)}$. Hence, edges which were already in G_c may appear in $G_r^{(i)}$. Both types of edges appear in this phase with level i. Assume edge $e = (r, h)$ appears in G_r. If $h \in G_r^{(i-1)}$, then its level cannot change. This is because h was over-subscribed and therefore all proposals worse than its current ones (in phase $i-1$) have been dominated. On the other hand, if $h \notin G_r^{(i-1)}$ then h gets a level equal to i. We claim that M is still level-maximal. Otherwise, it follows from Lemma 3.5 that there must be an alternating path from a free hospital to a hospital of lower level. This path must use new edges, but the new edges are incident to free residents, a contradiction.

By the new proposals residents might become dominated, and therefore edges might be deleted from G_c and G_r. The matching $M \in G_r^{(i-1)}$ may be reduced in size. Let M' be this resulting matching in $G_r^{(i)}$. Assume that M' is not level-maximal, then there is an alternating path p from a free hospital to a hospital of lower level. It cannot use any of the new edges, since they are incident to free residents. Thus p must use only old edges. Also p cannot start at a hospital of level i since only new edges are incident to such a hospital. Thus, p starts at hospital of level less than i and hence that hospital is free in M. Since $M' \subseteq M$, p is an alternating path with respect to M, a contradiction to the level maximality of M.

Next, we consider the free residents in G_r in turn and search for augmenting paths. Let r a free resident. If no augmenting path starting from r exists, let Z be the set of residents reachable by alternating paths in G_r from r and $N(Z)$ be their neighbors. Then $|Z| > \sum_{h \in N(Z)} p'_h$. We delete the edges in the tail of the hospitals in $N(Z)$. This may decrease the size of the matching, but the matching stays clearly level-maximal. If an augmenting path exists, let p be an augmenting path to a hospital of maximal level. We use p to increase the cardinality of the matching. By Lemma 3.6, the resulting matching is level-maximal. $\square$

```
Set phase number $i = 1$, $E' = E$, $E_c = \emptyset$, $M = \emptyset$ and each resident to be free
repeat
    while $\exists$ a free resident $r$ who has a non-empty list do
        move all top choice edges $e = (r, h)$ of $r$ in $E'$ to $E_c$
        if $h$ is fully-subscribed or over-subscribed then
            delete all edges $(r', h) \in E' \cup E_c$ such that $r'$ is dominated in $h$'s list
        end
        [update $G_r$]
    end
    Let $\mathcal{E}_i$ be the edges moved to $E_c$
    Let $G_r^{(i)}$ be the reduced assignment graph at this phase
    forall free residents $r \in G_r^{(i)}$ w.r.t $M$ do
        if an alternating path from $r$ to a free hospital exists in $G_r^{(i)}$ then
            let $h$ be the free hospital [of maximal level] reachable from $r$ by an
            alternating path and let $p$ be an alternating path from $r$ to $h$
            $M = M \oplus p$
        else
            let $Z$ be the set of residents reachable from $r$ by alternating paths
            in $G_r^{(i)}$ and let $N(Z)$ be the hospitals adjacent to them in $E_r$
            forall hospitals $h \in N(Z)$ do
                delete all lowest ranked edges in $E_c \cup E'$ incident to $h$
                [update $G_r$]
            end
        end
    end
    $i = i + 1$
until all free residents have run out of proposals
Let $G_c$ by the final provisional assignment graph
Let $M'$ be a feasible matching in $G_c$
if a hospital that was fully or over-subscribed at some point in the execution of
the algorithm is not full in $M'$ or a hospital that was always under-subscribed
has fewer assignees in $M'$ than its degree in $G_c$ then
    no strongly stable matching exists
else
    output the strongly stable matching $M'$
end
```

Algorithm 2: Two algorithms for strongly stable Hospitals/Residents matching. The algorithms differ by the phrase [*of maximal level*]. Without the phrase, the algorithm may augment the current matching along any augmenting path and the running time is $O(m^2)$. With the phrase, an augmenting path to a hospital of maximal level must be used. The running time improves to $O(m \sum_{h \in H} p_h)$. The phrase [*update* G_r] bounds the maintenance cost of G_r to $O(m)$.

3.3 The Running Time

The search for augmenting paths in the reduced assignment graph is implemented as in the classical strongly stable marriage problem. The time spent on the searches is calculated similarly. Thus, let r be a free resident of level $l(r)$, for whom we want to determine a free hospital h of maximal level reachable from r via an augmenting path. The search is organized, exactly as before, in rounds and using buckets. Let $j(r)$ be the minimal bucket index from which we remove a hospital. The time for the search is proportional to the number k of edges explored plus $l(r) - j(r) + 1$. The cost is charged as previously, in the case of success: everything to the found free hospital h and in the case of failure: one unit each to each edge deleted (this accounts for k) and $l(r) - j(r) + 1$ to the minimum level hospital reachable from r.

Reasoning as earlier, we get that the cost of unsuccessful searches is $O(|H|m)$. As for the successes, let us observe that all edges explored during the search have level at least $l(h)$ and at most i (= the phase number) and that the level of h jumps to at least $i+1$, if it becomes free again. This means that the set of edges explored during a successful search ending on h while it has some level $l_1(h)$ is disjoint from the set of edges explored during a successful search ending on h when h has some different level $l_2(h)$. The maximum number of times a successful augmenting path search ends at h, while the level of h is fixed at $l_1(h)$, is at most p_h. So the cost of all successful searches ending at h is $O(p_h m)$, hence the cost of all successful searches is $O((\sum_{h \in H} p_h)m)$.

3.3.1 *Implementation details about the graphs.* In order to maintain and update the provisional assignment graph and reduced assignment graph efficiently, we keep the useful information about these graphs in the following manner.

The adjacency list of every vertex is sorted according to the preferences of that vertex. The edges of a resident r that are currently in the provisional assignment graph are kept in two lists called *bound* or *unbound*. An edge (r, h) is in the bound list if r is bound to h, otherwise it is in the unbound list.

Every hospital h keeps the information about its quota p_h, the number of residents bound to it b_h, and the number of residents incident to it that are not bound to it u_h. The reduced quota of h is p'_h, which is equal to $p_h - b_h$. Observe that there are three types of residents incident to h: residents bound to h, resident not bound to h but bound to some other hospital h' and residents not bound to any hospital (the unbound residents). We assume that h maintains these three types of residents in three separate lists.

The residents that belong to the reduced assignment graph are those whose bound list is empty and unbound list nonempty. The hospitals that belong to the reduced assignment graph are those whose reduced quota is greater than zero. The edges that belong there are the unbound ones. We assume that every edge keeps the information about whether it is in a bound list or unbound list and pointers to all the lists it belongs to.

When the edge (r, h) is moved from E' to E_c, we check whether it is bound or unbound and update the following information: first we update b_h or u_h. If the added edge is in the tail of h's list and is unbound and if there are any bound edges in the tail, they change state to unbound. (Such a situation can arise when before adding this edge, there were exactly p_h bound edges incident on h and no unbound ones incident on it.) Because of the addition of the new edge (r,h) if there are residents that are now dominated in h's list, then those edges should be deleted.

Whenever an edge (r,h) gets added to the provisional assignment graph, gets deleted or changes state from bound to unbound we are able to update the information about r and h in constant time (including moving it into or away from the reduced assignment graph). Also, if an edge e appears in $G_r^{(i)}$, then it remains in $G_r^{(j)}, j > i$ until it gets deleted or the algorithm terminates. Since every edge can be added and deleted only once, as well as it can change its state only from being bound to being unbound, the total cost of updating and maintaining G_c and G_r is $O(m)$.

THEOREM 3.8. *Strongly stable matchings for the hospitals-residents problem can be computed in time $O(m \sum_{h \in H} p_h)$.*

We conclude that in the worst case $\sum_{h \in H} p_h$ can be as large as m, in which case we get a running time of $O(m^2)$, but in any practical application, we expect that $\sum_{h \in H} p_h = O(|R|)$, in which case we get a total running time of $O(|R|m)$.

4. CONCLUSIONS AND OPEN PROBLEMS

We have presented an $O(nm)$ algorithm for computing a strongly stable matching or reporting that none exists, given an instance of the stable matching problem. Furthermore, we extended this to an $O(m \sum_{h \in H} p_h)$ algorithm for the corresponding problem in the hospitals-residents context. The algorithms are variants of previous known algorithms with running time $O(m^2)$. We close with two open problems.

First, it is conceivable that Irving's algorithm runs in time $O(nm)$ but the analysis is not tight enough to show this. Can the analysis be improved or is there a family of instances which, under a particular sequence of augmenting paths, force the algorithm to spend time $\Omega(m^2)$? A similar question applies in the hospitals-residents context.

Second, can our $O(m \sum_{h \in H} p_h)$ algorithm for computing a strongly stable matching in the hospitals-residents $(R \dot{\cup} H, E)$ setting be improved to an $O(nm)$ algorithm?

However, there is a lower bound presented in [IMS03] that if f is any function on n, where $f = \Omega(n^2)$, then the existence of an $O(f(n))$ algorithm for deciding whether a strongly stable matching exists in an instance $(X \dot{\cup} W, E)$ with n vertices implies an $O(f(n))$ algorithm for deciding whether a given bipartite graph admits a perfect matching. This reduction may be adapted easily to the many-one case to show that the existence of an $O(f(n))$ algorithm for deciding whether a strongly stable matching exists, given an instance of the hospitals-residents problem, implies an $O(f(n))$ algorithm for deciding whether $(R \dot{\cup} H, E)$ (where $|R| = |H|$) admits a degree-constrained subgraph G' that satisfies all degree bounds exactly (i.e., each resident has exactly one edge incident to it in G' and each hospital h has exactly p_h edges incident to it). To the best of our knowledge, the current fastest algorithm for this problem has running time $O(m\sqrt{\sum_{h \in H} p(h)})$ [Gab83], where $m = |E|$.

Acknowledgements. We wish to thank the referees for their detailed comments and suggestions.

REFERENCES

Canadian Resident Matching Scheme. How the matching algorithm works. http : //www.carms.ca/matching/algorith.htm

H. N. Gabow An efficient reduction technique for degree-constrained subgraph and bidirected network flow problems. *Proceedings of STOC '83*, pages 448–456. ACM, 1983.

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

D. Gusfield and R.W. Irving. The Stable Marriage Problem: Structure and Algorithms. (MIT Press, Boston, MA, 1989).

R.W. Irving, D.F. Manlove, and S. Scott. Strong stability of the hospitals/residents problem. In *STACS , LNCS 2607*, pages 439–450. Springer Verlag, 2003.

R. W. Irving. Matching medical students to pairs of hospitals: a new variation of a well-known theme. In *ESA, LNCS 1461*, pages 381–392. Springer Verlag, 1998.

R.W. Irving. Stable marriage and indifference. *Discrete Applied Mathematics*, pages 261–272, 1994.

K. Iwama, D. Manlove, S. Miyazaki, and Y. Morita. Stable Marriage with incomplete lists and ties. In *ICALP , LNCS 1644*, pages 443–452. Springer Verlag, 1999.

T. Kavitha, K. Mehlhorn, D. Michail, and K. E. Paluch. Strongly Stable Matchings in Time $O(nm)$ and Extension to the Hospitals-Residents Problem. In *STACS, LNCS 2996*, pages 222–232. Springer Verlag, 2004.

D.F. Manlove. Stable marriage with ties and unacceptable partners. Technical report, University of Glasgow, 1999.

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), pages 991–1016, 1984.