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

Telikepalli Kavitha[1⋆], Kurt Mehlhorn[1⋆], Dimitrios Michail[1⋆], Katarzyna Paluch[2⋆⋆]

[1] Max-Planck-Institut für Informatik, Saarbrücken, Germany.
{kavitha, mehlhorn, michail}@mpi-sb.mpg.de
[2] Institute of Computer Science, University of Wroclaw, Poland.
abraka@ii.uni.wroc.pl.

Abstract. An instance of the stable marriage problem is an undirected bipartite graph $G = (X \dot\cup W, E)$ with linearly ordered adjacency lists; ties are allowed. 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 is either 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 $O(m(|R| + \sum_{h \in H} p_h))$ algorithm for computing a strongly stable matching in the hospitals-residents problem, where $|R|$ is the number of residents and p_h is the quota of a hospital h. The previous best algorithm had running time $O(m^2)$.

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, respectively.[3] Each person seeks to be assigned to a person of the opposite sex and his/her preference is given by the ordering of his/her 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'. We use n for the number of vertices and m for the number of edges. A stable

⋆ Partially supported by the Future and Emerging Technologies programme of the EU under contract number IST-1999-14186 (ALCOM-FT).
⋆⋆ Work done while the author was at MPII supported by Marie Curie Doctoral Fellowship.

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

marriage problem is called *complete* if there are an equal number of men and women and G is the complete bipartite graph; thus $m = (n/2)^2$.

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 and a the partner of b. A matching M is *strongly stable* if there is no edge $(a, b) \in E \setminus M$ (called *blocking edge*) such that by becoming matched with to each other, one of a and b (say, a) is better off and b is not worse off. For a being better off means that it is either unmatched in M or strictly prefers b to his/her partner in M, and a being not worse off means that it is either better off or is indifferent between b and his/her partner in M. In other words, a would prefer to match up 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 its situation and as b's situation would not get worse, it might yield to a. If there exists no such edge, then M can be considered to be reasonably stable since no two vertices a and b such that (a, b) in $E \setminus M$ gain by changing their present state and getting matched with each other. Observe that not every instance of the stable marriage problem has a strongly stable solution.

There are two more notions of stability in matchings. The matching M is said to be weakly stable (or, super strongly 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 proved to be NP-hard [IMMM99]. There is a simple $O(n^2)$ algorithm [Irv94] to determine if a super strongly stable matching exists or not and it computes one, if it exists.

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 valid assignment of residents to hospitals, defined more formally as a set of edges no two of which share the same resident and at most p_h of the 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 match up 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. 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.

Our Contributions: In this paper we give an $O(nm)$ algorithm to determine a strongly stable matching for the classical stable marriage problem. We also give an $O(m(|R|+\sum_{h\in H} p_h))$ algorithm to compute a strongly stable matching in the hospitals-residents problem. 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 instances. 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 specialisation 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 reduces the running time of the algorithm to $O(nm)$. We also use the above ideas in the hospitals-residents problem and obtain an improvement over [IMS03].

The stable marriage problem has great practical significance [Irv98,Roth84], [CRMS]. The classical results in stable marriage (no ties and the lists are complete) are the Gale/Shapley theorem and algorithm [GS62]. Gusfield and Irving [GI89] covers plenty of results obtained in the area of stable matchings.

Organisation of the paper: 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 $O(m(|R| + \sum_{h\in H} p_h))$ algorithm for the hospitals-residents problem.

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. Figure 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 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 it. Let $\mathcal{E}_i$ consist of the top choices[4] in E' of each free man. We say, that every

[4] Recall that $E' \subseteq E$ and that adjacency lists are linearly ordered with ties allowed. The top choices for a man x are the set of women tied for first place.

free man 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 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[5] all lowest ranked edges in $E_c \cup E'$ incident to it. 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 is isolated in G_c since the edges incident to it were removed when we searched for an augmenting path starting from it.

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 algorithm is free with respect to M. [6]

We refer the reader to [Irv94,Man99] for the proof of correctness of this algorithm. The algorithm runs in $O(m^2)$ time since the cost summed over all phases is $O(m \cdot (1 + \text{number of successful augmenting path computations}))$ and since the number of augmenting path computations is at most m. The latter claim follows from the fact that a matched man becomes free only if the matching edge incident to it is deleted.

2.2 The New Algorithm

We now 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 a strongly stable matching is actually the time spent on looking for augmenting paths. The notion of the level of an edge and the level of a vertex help us to search for augmenting paths in a streamlined manner. The vertices with higher levels are given precedence when searching for

[5] It is here, where we slightly deviate from Irving's algorithm. 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.

[6] For complete instances, it is particularly easy to decide whether the final matching M is stable. M is stable if it is a perfect matching in G.

```
Set phase number i = 1, E' = E and E_c = ∅.
M = ∅
repeat
  while ∃ a free man x do
    move all top choice edges e = (x, w) of x in E' to E_c and delete all edges
    (x', w) from E' ∪ E_c which w ranks strictly after e.
  end while
  Let 𝓔_i be the edges moved to E_c.
  for all 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 alter-
      nating path and let p be an alternating path from x to w
      M = M ⊕ 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;
      for all women w ∈ N(Z) do
        delete all lowest ranked edges in E_c ∪ E' incident to w;
      end for
    end if
  end for
  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.
```

Fig. 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 $O(nm)$.

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 1. *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 $\mathcal{E}_1 \cup \mathcal{E}_2 \cup \ldots \mathcal{E}_r$, where r is the total number of phases in the algorithm. Note that r can be as large as m.

Definition 2. *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 3. *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.*

Lemma 1. *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.*

Proof. Obvious.

Lemma 2. *A matching M is level-maximal iff there is no alternating path from a free woman to a woman of lower level.*

Proof. Observe that 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 3. *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 thereby by Lemma 2 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, $l(w') \leq l(w'')$.

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

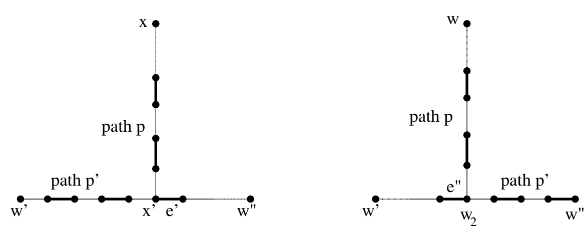

Fig. 2. The thick edges belong to the matching N

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 and not belonging to p, cannot belong to the matching N and we started the traversal of p' from the unmatched woman. So, e' is matched. 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 at its other end, that was free in M. Since $M \oplus p = N$, the matched edges of path p were exactly vice versa before the augmentation, in the sense that those edges, that are now present in the matching N, were not present in the matching M previously and the other way round. It means that w' was reachable by an alternating path from x in M. Thus $l(w') \leq l(w)$.

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 its one end and does not contain e'', we will notice that it has the woman w at its other end. Thus in M there existed an alternating path from the free woman w to the matched woman w'' and hence, by Lemma 2, $l(w) \leq l(w'')$.

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

Lemma 4. *M is a level-maximal matching at all times of the execution.*

Proof. We use induction on time. 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 neighbours. 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 3, the resulting matching is level-maximal.

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 maximal level free woman w 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. 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 neighbour of x). This contradicts the level-maximality of the matching.

We organise 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 Hungarian 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 initialise the bucket structure by putting the neighbours 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 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. Thus for a single 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 assigned 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 m. The total cost of successful searches is therefore bounded by $O(nm)$.

Theorem 1. *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 single woman sums to at most m over all phases. So, if $|W| \ll |X|$ or $|X| \ll |W|$ (then 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), then 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. We give an $O(m(|R|+\sum_{h \in H} p_h))$ 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. We give an outline of our approach here and the full version of the paper has all the proofs and details.

3.1 The Algorithm in [IMS03]

We first review a variant of the algorithm in [IMS03] and then present our modified algorithm. The algorithm in [IMS03] generalises the ideas used for computing strongly stable matchings in [Irv94] to the hospitals-residents problem.

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 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* according as it is provisionally assigned a number of residents greater than, less than, or equal to, its quota.
- 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 graph 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 for strongly stable marriage, 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 neighbours 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 top most 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 $G_r^{(i)}$. 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 hospital that is not filled up to its quota is considered an augmenting path.

We iterate over the free residents in 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 it 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 assignees in M' equal to its degree in G_c. We refer the reader to [IMS03] for the proof of correctness of this algorithm.

3.2 Our Modifications

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

Definition 4. *Define the level of an edge e, $l(e)$, to be the phase that e is added to the reduced assignment graph G_r.* [7]

Definition 5. *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, its level is undefined.*

Definition 6. *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 7. *A matching M is level-maximal if $l(M) \geq l(M')$ for any matching M' which matches the same residents.*

The following lemmas show how to maintain a level maximal matching. The proofs are available in the full version of this paper.

[7] 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.

Lemma 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.*

Lemma 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.*

Lemma 7. *M is a level-maximal matching at all times of the execution.*

3.3 The Running Time

The search for augmenting paths in G_r is implemented as in the classical stable marriage problem. Using similar arguments, one can see that the cost of unsuccessful searches is $O(m)$ and of successful searches is $O((\sum_{h \in H} p_h)m)$. Furthermore, with an appropriate representation of the graphs, all changes of G_r can be done in time $O(|R|m)$.

Theorem 2. *Strongly stable matchings for the hospitals-residents problem can be computed in time $O(m(|R| + \sum_{h \in H} p_h))$.*

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 $\sum_{h \in H} p_h = |R|$, in which case we get a total running time $O(|R|m)$.

References

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

[GS62] D. Gale and L.S. Shapley. College admissions and the stability of marriage. *American Mathematical Monthly*, 69, pages 9–15, 1962.

[GI89] D. Gusfield and R.W. Irving. The Stable Marriage Problem: Structure and Algorithms. (MIT Press, Boston, MA, 1989).

[IMS03] 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.

[Irv98] 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.

[Irv94] R.W. Irving. Stable marriage and indifference. *Discrete Applied Mathematics*, pages 261–272, 1994.

[IMMM99] 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.

[Man99] D.F. Manlove. Stable marriage with ties and unacceptable partners. Technical report, University of Glasgow, 1999.

[Roth84] 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.