Algorithms **2014**, *7*, 189-202; doi:10.3390/a7020189

OPEN ACCESS

algorithms

ISSN 1999-4893

www.mdpi.com/journal/algorithms

Article

Faster and Simpler Approximation of Stable Matchings

Katarzyna Paluch

Institute of Computer Science, University of Wrocław, Joliot-Curie 15, 50–383 Wrocław, Poland;
E-Mail: abraka@cs.uni.wroc.pl; Tel.: +48-71-37-57-821

Received: 5 September 2013; in revised form: 22 March 2014 / Accepted: 27 March 2014 / Published: 4 April 2014

Abstract: We give a $\frac{3}{2}$-approximation algorithm for finding stable matchings that runs in $O(m)$ time. The previous most well-known algorithm, by McDermid, has the same approximation ratio but runs in $O(n^{3/2}m)$ time, where n denotes the number of people and m is the total length of the preference lists in a given instance. In addition, the algorithm and the analysis are much simpler. We also give the extension of the algorithm for computing stable many-to-many matchings.

Keywords: stable matching; stable marriage; approximation algorithm

---

1. Introduction

In the paper, we consider a variant of the problem called **Stable Matchings**, known also in the literature as the **Stable Marriage** problem. The problem is defined as follows. We are given two sets W and U of women and men. Each woman w of W has a preference list L_w of a subset of men, and similarly, each man m of U has a preference list L_m of a subset of women. The preference lists are linearly ordered lists of **ties**, which are subsets of men (or respectively, women), who are equally good for a given woman (respectively, man). Ties are disjoint and can contain also one person, appropriately a man or a woman. Thus if m and m' are on list L_w of woman w, then either (1) w prefers m to m', or in other words, m is better for w than m'; or (2) m and m' are in a tie on L_w, and then we say that w is indifferent between m and m' or that m and m' are equally good for her; or (3) w prefers m' to m. Man m and woman w are said to be **mutually acceptable** to each other if they belong to each other's preference lists. The most preferred person(s) is (are) at the top of the preference lists. A **matching** is a set of pairs (m, w) such that $m \in U, w \in W$ and m and w are mutually acceptable, and each man/woman belongs to at most one pair. If (m, w) belongs to a certain matching M, then we write $M(m) = w$, which means that in M woman w is a partner of m, and analogously that $M(w) = m$. If man m (or woman w) is not contained in any

pair of a matching M, then we say that m (or w) is **unmatched** or **free** in M. A matching M is called **stable** if it does not admit a **blocking pair**. A pair (m, w) is blocking for M if (1) m and w are mutually acceptable; (2) m is unmatched or prefers w to $M(m)$; and (3) w is unmatched or prefers m to $M(w)$. Each instance of the problem can be represented by a bipartite graph $G = (U \cup W, E)$ with vertices of U representing men, vertices of W representing women and edges E connecting all mutually acceptable pairs of men and women. The problem we are interested in is that of finding a stable matching that has the largest cardinality. The version in which there are no ties in the preference lists of men and women has been long known and an algorithm by Gale and Shapley [3] solves it exactly in $O(m)$ time, where m denotes the number of edges in the underlying graph. In the version without ties a stable matching always exists, and every stable matching has the same cardinality. If we allow ties, as in the problem we consider in this paper, then a stable matching also always exists and can be found via the Gale-Shapley algorithm by breaking ties arbitrarily. However, the sizes of stable matchings can vary considerably, and the problem of finding a stable matching of maximum cardinality is NP-hard, which was shown by Manlove *et al.* in [14]. Therefore it is desirable to devise an approximation algorithm for the problem.

Previous and related results Previous approximation algorithms were presented in [7–9,11,14]. Currently the best approximation algorithm is by McDermid [15] and achieves the approximation guarantee $\frac{3}{2}$. Its running time is $O(n^{3/2}m)$, where n denotes the number of vertices and m the number of edges. Inapproximabilty results were shown in [4,5,17]. For a variant of the Stable Matchings problem, in which ties are allowed on one side only, an approximation algorithm achieving the ratio $\frac{25}{17}$ has been given in [10]. A slightly different linear time $\frac{3}{2}$-approximation algorithm based on an earlier version of this paper was given by Király in [12]. The conference version of this paper appeared also in the proceedings of WAOA'11 (Workshop on Approximation and Online Algorithms) [16].

Our results While constructing approximation algorithms, the goal is not only to achieve a good approximation guarantee but also good running time. We give a 3/2-approximation algorithm that runs in $O(m)$ time and additionally is significantly simpler than that of McDermid. In devising the algorithm we were led by the observation that it suffices to find a stable matching that will not create a dangerous path, which is defined later. We also give the extension of the algorithm for computing stable many-to-many matchings, which runs in $O(m \log c)$ time, where c denotes the minimum of the maximal capacities in each side of the bipartition. In particular, it means that we give an $O(m)$-time algorithm for the Hospitals-Residents problem, improving on an $O(d^{5/2}n^{3/2}m)$ time algorithm given by McDermid, where d denotes the maximal capacity of a hospital. McDermid's algorithm follows from the reduction of the Hospitals-Residents problem to the Stable Matchings problem by "cloning" hospitals. The approach by cloning does not work if the vertices on both sides of the bipartition are allowed to have capacities larger than 1. Since these problems have many practical applications (see [1,2,6], for example), we believe our algorithms will be of help.

2. Algorithm

For a given instance of the problem let M_{opt} denote an optimal (*i.e.*, largest) stable matching and let M, M' be any two matchings. We say that e is an M-edge if $e \in M$. A path P or a cycle C is called **alternating (w.r.t.** M**)** if its edges alternate between M-edges and edges of $E \backslash M$. It is well

known from matching theory (see [13] for example) that $M \oplus M'$ disintegrates into a set of alternating paths and alternating cycles. (For two sets X, Y, the set $X \oplus Y$ denotes $(X \backslash Y) \cup (Y \backslash X)$.) Let S denote a set of maximal alternating paths and cycles of $M \oplus M_{opt}$. Consider any alternating cycle C of S or any alternating path P of even length of S. Then both C and P contain the same number of M-edges and M_{opt}-edges. Consider an alternating path P of length $2k+1$ of S. Then either $\frac{|M_{opt} \cap P|}{|M \cap P|} = \frac{k+1}{k}$ or $\frac{|M \cap P|}{|M_{opt} \cap P|} = \frac{k+1}{k}$. Therefore, if M is stable and S does not contain a path of length 3 with the middle edge being an M-edge, then $|M_{opt}| \leq \frac{3}{2}|M|$ and M is a $\frac{3}{2}$-approximation of M_{opt}. To achieve a $\frac{3}{2}$-approximation, we will eliminate such potential paths of length 3 of $M \oplus M_{opt}$.

Accordingly, we define a **dangerous path w.r.t. a matching** M to be an alternating path $P = (w, m_1, w_1, m)$ such that w and m are unmatched in M (which means that (m_1, w_1) is in M and $(w, m_1), (w_1, m)$ do not belong to M) and (m_1, w_1) is not a blocking pair for the matching $N = \{(w, m_1), (w_1, m)\}$. We need to concern ourselves only with dangerous paths w.r.t. a matching M that can be extended to a stable matching,*i.e.*, M is such that there exists a stable matching M' with $M \subseteq M'$. Therefore, if $P = (w, m_1, w_1, m)$ is a dangerous path w.r.t. M, then w and m_1 do not form a blocking pair for M, which means that w_1 is at least as good for m_1 as w, and similarly, w_1 and m do not form a blocking pair for M, hence m_1 is at least as good for w_1 as m. Since (m_1, w_1) is not a blocking pair for the matching $N = \{(w, m_1), (w_1, m)\}$, either m_1 is indifferent between w and w_1 and then we say that P is a **masculine dangerous path**, or w_1 is indifferent between m and m_1 and then we say that P is a **feminine dangerous path**. A path P can of course be both a masculine and a feminine dangerous path.

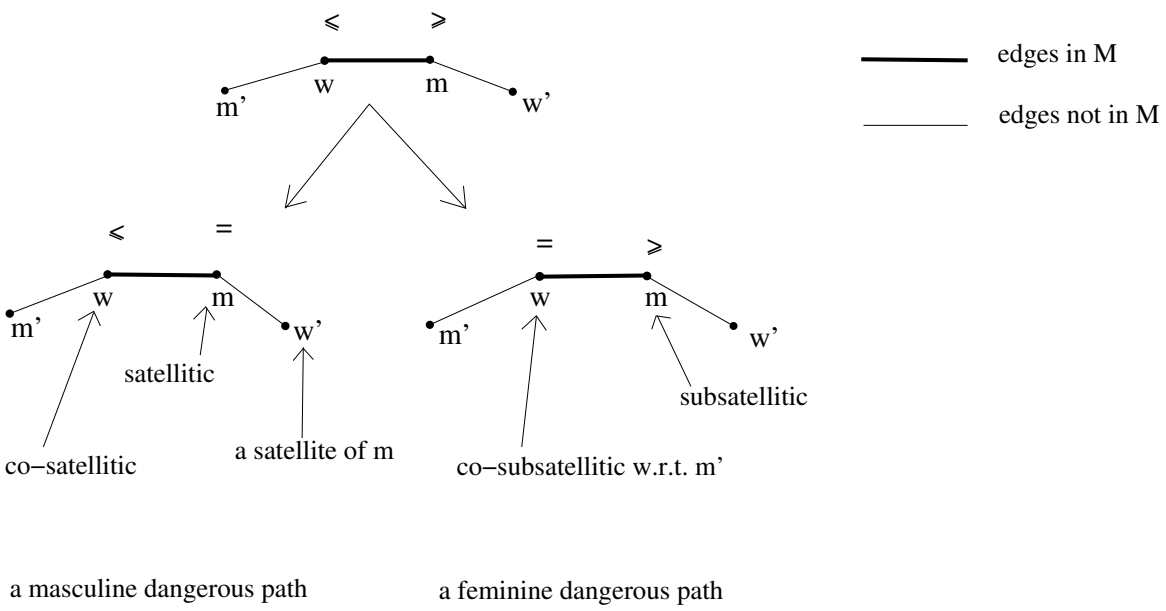
We also introduce the following terminology.

- If man m is matched to woman w and there is at least one free woman w_1 such that w and w_1 are equally good for m, then we say that w_1 is a **satellite** of m and m is **satellitic**.
- If woman w is matched to a satellitic man m, then we say that w is **co-satellitic**.
- If $e = (m, w)$ is such that w is free and there is at least one free woman $w_1 \neq w$ such that w and w_1 are equally good for m, then e is called **special**, and consequently, (m, w_1) is also called special.
- If man m has at least one free woman incident with him, then he is said to be **subsatellitic**.
- Woman w matched to a subsatellitic man m and not co-satellitic is said to be **co-subsatellitic w.r.t.** m' if m and m' are equally good for her.

Figure 1 illustrates the notions introduced above.

Let us notice that if a free man m has a co-satellitic woman w on his list, then it is possible that he belongs to a masculine dangerous path, and analogously, if a free man m has a woman w on his list who is co-subsatellitic with respect to him, then it is possible that he belongs to a feminine dangerous path.

Figure 1. The description of a dangerous path and its two types: a feminine dangerous path and a masculine dangerous path.



2.1. Description of Algorithm GS Modified

Algorithm GS Modified given further on is to some extent modeled on the Gale-Shapley algorithm in which men propose to women on their lists and women dispose. In the course of running the algorithm, preference lists L_m will diminish and some additional lists L'_m will be built. If at some point a free man m has a nonempty list L_m, it means that he has not yet proposed to all women on his list L_m and potentially belongs to a blocking pair or a masculine dangerous path. If a free man m has a nonempty list L'_m, it means that he potentially belongs to a feminine dangerous path.

Whenever it is man m's turn to propose and $L_m \neq \emptyset$, he would like to get matched to the best possible woman on his list L_m without creating a blocking pair (as in GS algorithm) but also ensure that he does not belong to any masculine dangerous path. To this end, m proposes to the woman w to whom he has not yet proposed and who is as high on L_m as possible. If w is free or matched to someone worse for her than m, she accepts m and rejects her current partner if she had one. If w is co-satellitic, which means that she is matched to some man m' such that there is a free woman w' who is equally good for m' as w, then it means that m currently belongs to a masculine dangerous path (m, w, m', w'). In this case w does not care whether m is better for her than m' and accepts him while rejecting m', and immediately afterwards m' proposes to w', who accepts him. This operation can be very well viewed as though m' proposed to w' without having proposed to w first, and some time later m proposed to w. (Here edge (m', w) was special at the moment m' proposed to w for the first time and that's why if m' gets rejected by w', he will propose to w again, because in this case w was not removed from $L_{m'}$. Let us notice that if m' were allowed to remove w from his list $L_{m'}$ at the moment he proposed to w for the first time and the edge (m', w) was special, then after being rejected by w' later on, he would not propose to w for the second time, and as a result a blocking pair involving m' and w might arise.) To avoid multiple operations of this kind concerning one woman, we will assume that given a tie, a man proposes to unmatched women

before proposing to matched ones. This ensures that in the above scenario, m cannot become satellitic after being matched with w. If a woman w to whom m proposes is matched to man m' equally good for her as m, and w is co-subsatellitic w.r.t. m, meaning that m' has some free women on his list, then at the current moment m belongs to a feminine dangerous path. What happens now is that w rejects m but m adds w to his list L'_m. (Woman w does not accept m because m may be subsatellitic as well.) In every other case w rejects m.

If man m has proposed to all women on his list L_m and remained free, but his list L'_m is nonempty, he will propose to women on L'_m starting from the top. If he proposes to w, and w is matched to some man m' who is equally good for her as m, and additionally, m' is subsatellitic, then w accepts m and rejects m'. Otherwise w rejects him. Let us notice that it cannot happen that w prefers m to m', because man m had proposed to all women on his list L_m before proceeding to list $L'_m \subseteq L_m$. Therefore he must have been rejected by w at some point, which means that w must be now matched to someone at least as good for her as m. (This will also be formally proved in the next section.)

Algorithm GS Modified

Each man m's preference list L_m is organized in such a way that if L_m contains a tie t, then free women in t come before matched women in t. At the beginning all women are free and ties on men's lists are broken arbitrarily, and in the course of running the algorithm, whenever woman w becomes matched for the first time, say to man m, we move her to the end of every tie she belongs to.

while there exists a free man m with a nonempty list L_m or a nonempty list L'_m
  if $L_m \neq \emptyset$, **then**
    $w \leftarrow$ woman at the top of m' s list L_m
    if (m, w) is not special, **then** remove w from L_m
    if w is free, **then** $M \leftarrow M \cup (m, w)$
    else if w is co-satellitic, **then**
      let w' be a satellite of $M(w)$
      if $(M(w), w')$ is not special, **then** remove w' from $L_{M(w)}$
      $M \leftarrow M \cup \{(m, w), (M(w), w')\} \backslash (w, M(w))$
    else if w prefers m to $M(w)$, **then** $M \leftarrow M \cup (m, w) \backslash (w, M(w))$
    else if w is co-subsatellitic w.r.t. m, **then** add w to the end of list L'_m
  else
    $w \leftarrow$ woman at the top of m' s list L'_m
    remove w from L'_m
    if w is co-subsatellitic w.r.t. m, **then** $M \leftarrow M \cup (m, w) \backslash (w, M(w))$

First we show how Algorithm GS Modified runs on the following example. Suppose the preference lists of men m_1, m_2, m_3, m_4 and women w_1, w_2, w_3, w_4 are as follows. The brackets indicate ties.

$m_1:$	(w_1, w_2)	w_3			$w_1:$	m_1	m_2	m_3
$m_2:$	w_1	w_3	w_4		$w_2:$	m_3	m_1	m_2
$m_3:$	w_2	w_1	w_3		$w_3:$	m_1	(m_2, m_4)	m_3
$m_4:$	w_3				$w_4:$	m_2		

Suppose that man m_1 starts. He proposes to woman w_1 and gets accepted ((m_1, w_1) is a special edge and w_2 is a satellite of m_1). Now suppose that it is m_2's turn to propose. (It might also be m_3 or m_4.) He proposes to woman w_1 and gets accepted, because w_1 is co-satellitic. Man m_1 gets matched with woman w_2. Next m_3 proposes to woman w_2 and gets accepted. Now m_1 proposes to w_1 (as (m_1, w_1) was a special edge) and gets accepted. Then m_2 proposes to w_3 and gets accepted. Next m_4 proposes to w_3 and gets rejected, but w_3 is co-subsatellitic w.r.t. m_4, and m_4 adds w_3 to his list L'_{m_4}. Afterwards m_4 proposes to w_3 again, this time from L'_{m_4}, and gets accepted. Finally m_2 proposes to w_4 and gets accepted.

3. Correctness of Algorithm GS Modified

In this section we prove the correctness of Algorithm GS Modified.

If $w \in L_m$ and m proposes to w, then we will sometimes say that m proposes from L_m (to w). If $L_m = \emptyset,\ w \in L'_m$ and m proposes to w, then we will sometimes say that m proposes from L'_m (to w).

Lemma 1. *1. If woman w becomes matched, she will stay matched.*

2. *Woman w can become co-satellitic only the first time someone, say m, proposes to her and only if at the time of the proposal edge (m, w) is special. If a co-satellitic woman w receives a proposal, she always accepts it and is no longer co-satellitic.*
3. *If woman w is matched to man m and is not co-satellitic, she can accept man m' only if m' is at least as good for her as m. Moreover, if m' is better for her than m, she always accepts him. If m' is equally good for her as m, then she accepts him only if she is co-subsatellitic w.r.t. m' and m' proposes from $L'_{m'}$.*
4. *If woman w matched to man m is not co-satellitic and changes m for m', who is equally good for her as m, then m is subsatellitic and m' is not.*

Proof. Statements 1 and 3 follow directly from the description of Algorithm GS Modified.

Let us prove statement 2. If w is matched and m proposes to her, then there is no free woman w' incident with m who is equally good for m as w (because then m would propose to w' before proposing to w). As a result, if w becomes matched to m, she will not become co-satellitic and she will cease to be co-satellitic if she was before.

It remains to prove statement 4. If w changes m for m' who is equally good for her as m (and w is not co-satellitic), then by statement 3, m' proposes from $L'_{m'}$ and m is subsatellitic. Man m' proposing from $L'_{m'}$ does not have any free women incident with him. □

Lemma 2. *Let M denote a matching computed by Algorithm GS Modified. Then the underlying graph does not contain blocking pairs and dangerous paths w.r.t. M.*

Proof. Suppose that (m, w) is a blocking pair. Man m is either free or $M(m)$ is worse for him than w. It means that at some point m proposed to w from L_m when edge (m, w) was not special. (Clearly, at some point m proposed to w from L_m. Assume that at that point edge (m, w) was special. Then w was free and accepted m. However, m got rejected later and therefore proposed to w from L_m again, when edge (m, w) was no longer special.) If w rejected him then, then by claim 2 of Lemma 1, w was not co-satellitic and, by claim 3 of Lemma 1, was matched to someone at least as good for her as m and thus would have stayed matched to someone as good for her as m. If w accepted m, then after getting matched to m she was not co-satellitic, and by claim 3 of Lemma 1, would have stayed matched to someone at least as good for her as m. Either way we get a contradiction.

Suppose now that the graph contains a masculine dangerous path (m', w, m, w') such that $m = M(w)$. Thus m is satellitic and w is co-satellitic. Since she is co-satellitic, it means that the only proposal she ever got was from m, but m' must have proposed to her too, a contradiction.

Finally, suppose that the graph contains a feminine dangerous path (m', w, m, w') such that $m = M(w)$. Thus m is subsatellitic and w is co-subsatellitic w.r.t. m', also m' is not subsatellitic. At some point, m' proposed to w from $L_{m'}$ when (m', w) was not special. If he got accepted at that moment, then later on he could not become rejected, because by claim 3 of Lemma 1, after accepting m' woman w was not co-satellitic, and by claim 4 of Lemma 1, could not accept a subsatellitic man equally good for her as her current partner. Therefore he was rejected then, and w was already matched with m (by claims 3 and 4 of Lemma 1). Hence w was co-subsatellitic w.r.t. m' (because m was subsatellitic), and m' added w to the end of list $L'_{m'}$. Thus later m' proposed to w from $L'_{m'}$. According to the algorithm, w would have accepted him and could not later on become matched to someone subsatellitic and equally good for her as m'. A contradiction. □

Theorem 1. *Algorithm GS Modified computes a stable matching M which is a $\frac{3}{2}$- approximation of the optimal solution. Algorithm GS Modified runs in $O(m)$ time.*

Proof. By Lemma 2, matching M computed by Algorithm GS Modified is stable and does not contain dangerous paths. Therefore M is a $\frac{3}{2}$-approximation of the optimal solution.

The running time of the algorithm is proportional to the sum of the lengths of lists L_m and L'_m. By the length of list L_m or L'_m we mean the number of people on it and not the number of ties. Each edge of L_m is scanned at most twice–twice only if the first time it was scanned, it was special, and each edge of L'_m is scanned at most once. Checking whether a given woman w is co-satellitic or co-subsatellitic can be done in constant time as follows. Suppose that w is matched to man m. We look at the tie at the top of list L_m. If this tie contains woman w and some free woman w', then w is co-satellitic, otherwise she is not. (Recall that the free women proceed matched ones in a tie.) To facilitate checking whether a given man is subsatellitic, each man has the counter of free women incident with him, and whenever a woman becomes matched and moves herself to the end of the ties, men decrease their respective counters. The number of movements in the ties is upperbounded by the sum of the lengths of lists L_m as once a woman becomes matched, she stays so. □

Let us finally make the following remark.

If we break ties and run the classic Gale-Shapley algorithm, then the cardinality of the computed matching depends on the order in which we break ties. Algorithm GS Modified outputs a matching that would have been output by the GS algorithm if ties were broken as follows. Men's lists would be identical to those at the beginning of Algorithm GS Modified but for one thing: if at some point during running Algorithm GS Modified, man m proposes to a co-satellitic woman w, and as a result m gets matched to w, and w's partner $M(w)$ gets matched to his satellite w', then a tie on $L_{M(w)}$ would be broken in such a way that w' comes before w. Every tie t on a woman w's list would be first broken into $(m_1, m_2, \ldots, m_s)$ in such a way that m_1 denotes the first man of t to whom w got matched without becoming co-satellitic, and assuming that it happened at some step S, m_2 denotes the first man of t who proposed to w after step S, m_3 denotes the second man of t, who proposed to w after step S and so on. Next we would make the following alterations on women's lists: if at some point, man m proposes from L'_m to a co-subsatellitic woman w matched to $M(w)$, then a tie on L_w would be broken in such a way that m comes before $M(w)$.

4. Extension to Bipartite Stable b-Matchings

Suppose we have a simple bipartite graph $G = (V, E)$, where $V = U \cup W$ and U, W are disjoint sets, and a function b: $V \to \mathbb{N}$. Then a subset $M \subseteq E$ is called a b-matching if for each $v \in V$ it is $deg_M(v) \leq b(v)$, where $deg_M(v)$ denotes the degree of vertex v in the graph $G_M = (U \cup W, M)$. We refer to the vertices of U and W as U-agents and W-agents, respectively. Each U-agent u of U has a preference list L_u of a subset of W-agents, and analogously, each W-agent w has a preference list L_w of a subset of U-agents. The preference lists are linearly ordered lists of ties. The majority of the terminology for stable matchings goes through for stable b-matchings. Instead of saying that some agent or vertex is free, we will use the term **unsaturated**: agent v is unsaturated in a b-matching M if $deg_M(v) < b(v)$, and if $deg_M(v) = b(v)$, then we will say that v is **saturated**. For any agent v we will denote the set $\{w \in U \cup W{:}(v, w) \in M\}$ by $M(v)$. A pair (u, w) is **blocking** for a b-matching M if (1) u and w are mutually acceptable; (2) u is unsaturated or prefers w to one of the W-agents of $M(u)$; and (3) w is unsaturated or prefers u to one of the U-agents of $M(w)$. A b-matching M is said to be stable if it does not admit a blocking pair. As previously, we are interested in finding a stable b-matching of largest size. Let us also note that if for each u in U we have $b(u) = 1$, then the problem is known under the name Hospitals-Residents problem or one-to-many stable matching problem.

Alternating paths and cycles are defined for b-matchings in an analogous way as for matchings, but we do not require paths and cycles to be simple, *i.e.*, an **alternating path** P **w.r.t.** a b-matching M is defined as any sequence of edges $(v_1, v_2), (v_2, v_3), \ldots, (v_{k-1}, v_k)$ such that the edges alternate between M-edges and edges of $E \backslash M$, and an **alternating cycle** C **w.r.t.** M is defined as an alternating path (w.r.t. M) that ends and begins with the same vertex, *i.e.*, the sequence of edges has the form $(v_1, v_2), (v_2, v_3), \ldots, (v_k, v_1)$. As before, for any two b-matchings

M and M', the symmetric difference $M \oplus M'$ can be partitioned into maximal alternating paths and cycles (*i.e.*, no two alternating paths from the partition can be combined to form one alternating path). A given stable b-matching M might not be a 3/2-approximation of M_{opt}, where M_{opt} denotes a stable b-matching of maximum size, only if the underlying graph contains a dangerous path defined as follows. If M is a stable b-matching, then a path $P = (w, u_1, w_1, u)$ is called **dangerous with respect to** M if u and u_1 are U-agents, w and w_1 are W-agents, (u_1, w_1) is in M, $(w, u_1), (w_1, u)$ are not in M, w and u are unsaturated in M, u_1 and w_1 are saturated in M, and (u_1, w_1) is not a blocking pair for the b-matching $M' = (M \setminus (u_1, w_1)) \cup \{(w, u_1), (w_1, u)\}$. Since (w, u_1) is not blocking for M, w is not better for u_1 than any of the W-agents currently belonging to $M(u_1)$, and analogously, u is not better for w_1 than any of the U-agents currently belonging to $M(w_1)$. Thus if P is dangerous, then either w and w_1 are equally good for u_1 and then P is called a **masculine dangerous path**, or u and u_1 are equally good for w_1 and then P is called a **feminine dangerous path**.

An approximation algorithm for finding stable b-matchings is constructed analogously to the algorithm for finding stable matchings. U-agents play the role of men and W-agents play the role of women. For convenience, we shall refer to a U-agent as "he" and to a W-agent as "she". We adapt the terminology from the one-to-one setting to the current one as follows.

- If a U-agent u is in $M(w)$, and there is at least one unsaturated W-agent $w_1 \notin M(u)$ different from w such that w and w_1 are equally good for u, then we say that w_1 is a **satellite of** u **w.r.t.** w and u is **satellitic w.r.t.** w.
- W-agent w belonging to $M(u)$ such that u is satellitic w.r.t. w is said to be **co-satellitic**.
- If $e = (u, w)$ is such that w is unsaturated and there is at least one unsaturated W-agent $w_1 \neq w$ such that w and w_1 are equally good for u, then e is called **special**, and consequently, (u, w_1) is also called special.
- If a U-agent u has at least one unsaturated W-agent $w \notin M(u)$ incident with him, then he is called **subsatellitic**.
- A saturated and not co-satellitic W-agent $w \in M(u)$ such that u is subsatellitic is said to be **co-subsatellitic w.r.t.** u' if $u' \notin M(w)$ and u and u' are equally good for her.
- By a **worst** U**-agent belonging to** $M(w)$ we will mean any U-agent $u \in M(w)$ such that there is no other U-agent $u' \in M(w)$ who is worse for w than u.

Algorithm ASBM (short for *Approximate Stable b-Matching*)

Each U-agent u's preference list L_u is organized in such a way that if L_u contains a tie t, then unsaturated W-agents in t come before saturated W-agents in t. At the beginning, all W-agents are unsaturated and ties on U-agents's lists are broken arbitrarily, and in the course of running the algorithm, whenever W-agent w becomes saturated for the first time, we move her to the end of every tie she belongs to.

while there exists an unsaturated U-agent u with a nonempty list L_u or a nonempty list L'_u

if $L_u \neq \emptyset$, **then**
$w \leftarrow W$-agent at the top of u' s list L_u
if (u, w) is not special, **then** remove w from L_u
if w is unsaturated, **then** $M \leftarrow M \cup (u, w)$
else if w is co-satellitic, **then**
let w' be a satellite of a U-agent $u' \in M(w)$ w.r.t. w
if (u', w') is not special, **then** remove w' from $L_{u'}$
$M \leftarrow M \cup \{(u, w), (u', w')\} \backslash (w, u')$
else if w prefers u to a worst U-agent in $M(w)$, **then**
let u' denote any worst U-agent belonging to $M(w)$
$M \leftarrow M \cup (u, w) \backslash (w, u')$
if w is co-subsatellitic w.r.t. u', **then** add w to the end of list $L'_{u'}$
else if w is co-subsatellitic w.r.t. u, **then** add w to the end of list L'_u.
else
$w \leftarrow W$-agent at the top of u' s list L'_u
remove w from L'_u
if w is co-subsatellitic w.r.t. u, **then**
let u' denote a subsatellitic U-agent in $M(w)$ equally good for w as u
$M \leftarrow M \cup (u, w) \backslash (w, u')$
if w is co-subsatellitic w.r.t. u', **then** add w to the end of $L'_{u'}$

5. Correctness of Algorithm ASBM

The correctness of Algorithm ASBM is proved in a very similar way as the correctness of Algorithm GS Modified.

Lemma 3. *1. If a W-agent becomes saturated, she will stay saturated.*

2. *A co-satellitic W-agent w accepts every proposal. Once a saturated W-agent is not co-satellitic, she cannot become co-satellitic later.*
3. *A W-agent $w \in M(u)$ can reject u only if w is saturated and a) u is satellitic w.r.t. w (w is co-satellitic), or b) w is not co-satellitic, u is one of the worst U-agents belonging to $M(w)$, and w receives a proposal from u' who is better for w than u, or c) w is not co-satellitic and u is one*

of the worst U-agents belonging to $M(w)$, u is subsatellitic, and w is co-subsatellitic w.r.t. u' who proposes from $L'_{u'}$

4. *A saturated W-agent w who is not co-satellitic can accept a U-agent u' only if u' is at least as good for w as a worst U-agent $u \in M(w)$; moreover if u' is equally good for w as u, then w accepts u' only if w is co-subsatellitic w.r.t. u' and u' proposes from $L'_{u'}$.*

The proof of Lemma 3 is very similar to that of Lemma 1 and follows directly from the description of Algorithm ASBM.

Theorem 2. *Let M denote a b-matching computed by Algorithm ASBM. Then M is a $3/2$-approximation of an optimal stable b-matching.*

Proof. We will show that the underlying graph does not contain blocking pairs and dangerous paths with respect to M. The non-existence of a dangerous path for the stable matching found guarantees the approximation ratio $3/2$.

Suppose that (u, w) is a blocking pair. Agent u is either unsaturated or there exists $w' \in M(u)$ worse for u than w. It means that at some point u proposed to w from L_u when edge (u, w) was not special. If u's proposal to w was rejected, then at that point w was saturated and not co-satellitic, and the worst $u' \in M(w)$ was at least as good as u for w (by Lemma 3), and thus (also by claim 4 of Lemma 3) w could not later become matched to some u'' who is worse for w than u. Therefore u got accepted then and later got rejected. Since at the moment of that proposal edge (u, w) was not special, u was not satellitic w.r.t. w (and clearly could not become satellitic later.) By claim 3 of Lemma 3, W-agent w was not co-satellitic at the moment of rejecting u, and the worst U-agent belonging to $M(w)$ was u. Therefore by claim 4 of Lemma 3, w could not later become matched to some u'' who is worse for her than u. A contradiction.

Suppose now that the graph contains a masculine dangerous path (u', w, u, w') such that $u \in M(w)$. Thus u is satellitic w.r.t. w and w is co-satellitic. It means that at some point u' proposed to w from $L_{u'}$ when edge (u', w) was not special. Then w was either co-satellitic, because she is co-satellitic now, or unsaturated. Therefore u' got accepted. Later on he was clearly rejected. However, by claim 3 of Lemma 3 and the description of the algorithm ABSM, this is impossible, because a co-satellitic w rejects only U-agents who are satellitic w.r.t. w.

Finally, suppose that the graph contains a feminine dangerous path (u', w, u, w') such that $u \in M(w)$. Thus w is co-subsatellitic w.r.t. u', and u is subsatellitic. At some point u' proposed to w from $L_{u'}$ when edge (u', w) was not special. If he got rejected, then w was not co-satellitic, and a worst U-agent $u'' \in M(w)$ was equally good for her as u' and hence as u. By the description of the algorithm, at that point w was co-subsatellitic w.r.t. u', and u' added w to the end of list $L'_{u'}$. If he was accepted (*i.e.*, when he was proposing from $L_{u'}$ and edge (u', w) was not special), then later he was rejected, and by the description of the algorithm, he also must have added w to the end of list $L'_{u'}$. When u' proposed to w from $L'_{u'}$, w was still co-subsatellitic w.r.t. u' (because w is co-subsatellitic w.r.t. u' now), hence u' was accepted (because u' proposing from $L'_{u'}$ is not subsatellitic) and could not have got rejected later if there were still subsatellitic U-agents matched with w who were equally good as u' for w. A contradiction. □

6. Data Structures and Running Time

Each agent a (either a U-agent or a W-agent) has a preference list L_a, which is a list of lists, *i.e.*, we have a list for each tie. For each list we have access to both its first and last element.

Each agent has a pointer to their position in every tie they belong to (here a 1-element list is also considered a tie). Whenever W-agent w gets saturated for the first time, w goes over her whole list L_w and moves herself to the end of every tie she belongs to, as explained in the description of Algorithm ASBM. This operation takes $O(|L_w|)$ time.

Every W-agent w stores information about U-agents currently belonging to $M(w)$ in a priority queue. U-agents belonging to $M(w)$ who are equally good for w are kept in one list, thus the priority queue contains lists. Each such list is associated with a priority value, which is the position the U-agents on that list have on the initial list L_w, *i.e.*, the U-agents that are at the top tie of list L_w have priority equal to 1. We implement a priority queue as a balanced tree with an additional pointer to the maximum element. We will use the following operations:

- finding the maximum element (takes $O(1)$ time)
- deleting a given element (takes $O(\log n)$ time, where n denotes the number of elements in a priority queue)
- inserting a given element (takes $O(\log n)$ time)
- finding an element having a certain priority value (takes $O(\log n)$ time).

After deleting or inserting an element we update the pointer to the maximum element, which takes $O(\log n)$ time.

Each U-agent u keeps track of W-agents currently belonging to $M(u)$ in a list.

Each U-agent u has the counter of the number of unsaturated W-agents incident with him, and whenever a saturated W-agent moves herself to the end of the ties, U-agents also decrease respective counters. Therefore checking if u is subsatellitic takes constant time.

Each W-agent w has a separate list S_w of satellitic U-agents w.r.t. w belonging to $M(w)$. Every time w gets matched to some new U-agent u who is satellitic with respect to w, she adds u to S_w. When we want to check if w is co-satellitic, we go over S_w and for each $u \in S_w$ check if u is still satellitic w.r.t. w. Checking if a U-agent u is satellitic w.r.t. w can be done as follows. W-agent w has a pointer to her position in a tie t on L_u. We also have access to the first element of t. If the first element of t is unsaturated, then it means that u is satellitic w.r.t. w. This way we can also find a satellite of a satellitic U-agent in $M(w)$. If u is not satellitic w.r.t. w, we remove u from S_w, otherwise we do an appropriate exchange. Once u is removed from S_w, he will not be added to S_w again. It is so since once u has no unsaturated W-agents equally good for him as w on his list, it will stay so. Hence the overall time Algorithm ASBM spends on S_w is $O(|L_w|)$.

Each list describing a tie within the priority queue of U-agents belonging to $M(w)$ is organized in such a way that subsatellitic U-agents proceed U-agents that are not subsatellitic. Whenever a U-agent u ceases to be subsatellitic, we move him to the end of every list in every priority queue he is in. Moving u to the end of every such list takes $O(|M(u)|)$ time. Every u ceases to be subsatellitic at most once in the course of running the algorithm. Therefore the total time spent on moving U-agents who ceased to be subsatellitic to the ends of priority queues does not exceed $O(m)$. This way, to see if w is co-subsatellitic

w.r.t. u, we look at the list in the priority queue containing U-agents in $M(w)$ that are equally good for w as u and see if the first U-agent on this list is subsatellitic.

Each U-agent u makes a proposal to every W-agent on L_u at most twice and to every W-agent on L'_u at most once. The second statement follows from the observation that whenever an agent u proposes from L'_u, he does not have any unsaturated W-agents on L'_u. After each such proposal, Algorithm ASBM performs at most two insertions and at most one deletion on a suitable priority queue. Each insertion and each deletion into a priority queue containing U-agents belonging to $M(w)$ takes at most $O(\log b(w))$ time.

Summing all the arguments together, we get that the running time of Algorithm ASBM is $O(m \max\{1, \log \max\{b(w) : w \in W\}\})$, where m denotes the number of edges in G. If $\max\{b(u) : u \in U\} < \max\{b(w) : w \in W\}$, then we can swap the roles of U-agents and W-agents. Therefore we can state the following.

Theorem 3. *The running time of Algorithm ASBM is $O(m \max\{1, \log c\})$, where $c = min\{max\{b(v) : v \in U\}, max\{b(v) : v \in W\}\}$ and m denotes the number of the edges.*

Acknowledgments

I would like to thank anonymous referees for many helpful comments.

Conflicts of Interest

The authors declare no conflict of interest.

References

1. Abdulkadiroglu, A.; Pathak, P.A.; Roth, A.E. Strategy-proofness *vs.* Efficiency in matching with indifferences: Redesigning the NYC high school match. *Am. Econ. Rev.* **2009**, *99*, 1954–1978.
2. Erdil, A; Haluk, E. *What's the Matter with Tie-Breaking? Improving Efficiency in School Choice*. *Am. Econ. Rev.* **2008**, *98*, 669–689.
3. Gale, D.; Shapley, L.S. College admissions and the stability of marriage. *Am. Math. Mon.* **1962**, *69*, 9–15.
4. Halldórsson, M.M.; Irving, R.W.; Iwama, K.; Manlove, D.; Miyazaki, S.; Morita, Y.; Scott, S. Approximability results for stable marriage problems with ties. *Theor. Comp. Sci.* **2003**, *306*, 431–447.
5. Halldórsson, M.M.; Iwama, K.; Miyazaki, S.; Yanagisawa, H. Improved approximation results for the stable marriage problem. *ACM Trans. Algorithms* **2007**, *3*, 30.
6. Irving, R.W.; Manlove, D. Finding large stable matchings. *ACM J. Exp. Algorithmics* **2009**, *14*, article 2.
7. Iwama, K.; Miyazaki, S.; Okamoto, K. A $(2 - c\frac{\log n}{n})$-approximation algorithm for the stable marriage problem. *IEICE Trans.* **2006**, *89-D*, 2380–2387.
8. Iwama, K.; Miyazaki, S.; Yamauchi, N. A $(2 - c\frac{1}{\sqrt{n}})$-approximation algorithm for the stable marriage problem. *Algorithmica* **2008**, *51*, 342–356.

9. Iwama, K.; Miyazaki, S.; Yamauchi, N. A 1.875-approximation algorithm for the stable marriage problem. In Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA, New Orleans, January 7–9, 2007; Bansal, N., Pruhs, K., Stein, C., Eds.; pp. 288–297.
10. Iwama, K.; Miyazaki, S.; Yanagisawa, H. A $25/17$-Approximation algorithm for the stable marriage problem with one-sided ties. *Lect. Notes Comp. Sci.* **2010**, *6347*, 135–146.
11. Király, Z. Better and simpler approximation algorithms for the stable marriage problem. *Algorithmica* **2011**, *60*, 3–20.
12. Király, Z.*Approximation of Maximum Stable Marriage*; Technical Report TR-2011-0; The Egerváry Research Group: Budapest, Hungary, 2011.
13. Lovász, L.; Plummer, M.D. Matching Theory. In Annals of Discrete Mathematics 29, North-Holland, Amsterdam, 1986.
14. Manlove, D.; Irving, R.W; Iwama, K.; Miyazaki, S.; Morita, Y. Hard variants of stable marriage. *Theor. Comput. Sci.* **2002**, *276*, 261–279.
15. McDermid, E.: A 3/2-approximation algorithm for general stable marriage. In Proceedings of the Automata, Languages and Programming, 36th International Colloquium, ICALP, Rhodes, Greece, 2009; Albers, S., Marchetti-Spaccamela, A., Matias, Y., Nikoletseas, S.E., Thomas, W., Eds.; pp. 689–700.
16. Paluch, K.E. Faster and simpler approximation of stable matchings. In Proceedings of the WAOA '11: 9th Workshop on Approximation and Online Algorithms, Saarbruecken, Germany, 2011; Solis-Oba, R., Persiano, G., Eds.; pp. 176–187.
17. Yanagisawa, H. Approximation Algorithms for Stable Marriage Problems. Ph.D. Thesis, Kyoto University, Kyoto, Janpan, 2007.