

# A Note on Distributed Stable Matching

Alex Kipnis    Boaz Patt-Shamir  
School of Electrical Engineering  
Tel Aviv University  
Tel Aviv 69978  
Israel

## Abstract

We consider the distributed complexity of the stable marriage problem. In this problem, the communication graph is undirected and bipartite, and each node ranks its neighbors. Given a matching of the nodes, a pair of unmatched nodes is called *blocking* if they prefer each other to their assigned match. A matching is called *stable* if it does not induce any blocking pair. In the distributed model, nodes exchange messages in each round over the communication links, until they find a stable matching. We show that if messages may contain at most  $B$  bits each, then any distributed algorithm that solves the stable marriage problem requires  $\Omega(\sqrt{n/B \log n})$  communication rounds in the worst case, even for graphs of diameter  $O(\log n)$ , where  $n$  is the number of nodes in the graph. Furthermore, the lower bound holds even if we allow the output to contain  $O(\sqrt{n})$  blocking pairs. We also consider  $\varepsilon$ -stability, where a pair is called  $\varepsilon$ -blocking if they can improve the quality of their match by more than an  $\varepsilon$  fraction, for some  $0 \leq \varepsilon \leq 1$ . Our lower bound extends to  $\varepsilon$ -stability where  $\varepsilon$  is arbitrarily close to  $1/2$ . We also present a simple distributed algorithm for  $\varepsilon$ -stability whose time complexity is  $O(n/\varepsilon)$ .

## 1 Introduction

Stable marriage, ever since it was invented by Gale and Shapley [3], is considered to be one of the most interesting abstractions of markets: there are providers and customers (in the original paper: colleges and applicants, or, more provocatively, men and women); each customer has a ranking over the providers, and similarly each provider ranks the customers. Given a matching  $M$  that matches each customer with a provider, we say that a customer-provider pair  $(i, j)$  is *blocking* with respect to  $M$  if both  $i$  and  $j$  prefer each other to their matches under  $M$ . Clearly, a blocking pair is a destabilizing element for  $M$ , because members of

a blocking pair have an incentive to “elope,” i.e., break  $M$  by matching to each other. Therefore, a desired fixed point is a matching that does not induce any blocking pair; such a matching is said to be *stable*.

From the game-theoretic viewpoint, in stable matching (or stable marriage, as it is commonly known, using the metaphor of men and women), unlike classical games, cooperation between two players is a key ingredient. Consequently, the stability solution concept is robust against the cooperation of any couple, in contrast to the Nash equilibrium solution concept, whose motivation is that no lone player should have a reason to deviate from the solution state. Beyond game theory, it is also interesting to note that stable matching has applications in models without any hint of selfish agents, such as scheduling network switches [1].

Gale and Shapley proved that if the number of men is equal to the number of women, and if the preference of each player is a total order over the players of the opposite sex, then a stable marriage exists, and it can be found by a quadratic-time algorithm. Due to the many natural applications and the non-trivial algorithmic essence of the problem, much work has followed. The book by Gusfield and Irving [4] covers many variants of stable matching, such as the case where preferences may include ties, or the case where some pairs are unacceptable matches. Needless to say, most applications (like many scenarios considered in Game Theory) are inherently distributed in nature, namely players act locally, and are not activated by a central authority. It is therefore quite surprising that, to the best of our knowledge, until now no one studied the complexity of the problem from the distributed computation point of view, where communication between different players is accounted for. In this paper, we take a first step in this direction. Specifically, modeling the input as a bipartite graph, where each edge represents both a communication link and a possible match between its endpoints, we ask the question: how much time (namely, communication rounds) is required to reach a stable matching?

More concretely, we consider the standard synchronous model [10], where processors communicate in synchronous rounds with their neighbors according to an underlying graph  $G = (V, E)$ : in each round, each processor can send a message to each of its neighbors. For the distributed stable marriage problem (henceforth, DSM), the input to each node is a ranking of its neighbors. It is not hard to see that no algorithm for DSM can terminate in less time than required to cross the graph, i.e., the diameter of the graph. Intuitively, the reason is that input at one end of the graph may determine the output at another end of the graph. It is also easy to see that the following generic distributed algorithm can solve DSM: disseminate the whole input to all nodes by “flooding,” and then let each node apply a fixed centralized algorithm locally to find a global matching. This protocol is obviously unattractive as it just reduces the distributed problem to  $n$  replicas of the centralized problem, where  $n = |V|$ . But there are harder arguments against the generic protocol: even though no bit needs to traverse more than diameter hops, the number of bits each node needs to receive is  $O(\min(n^2, |E| \log n))$ , which should result in high communication time in a realistic model.

To capture this phenomenon, in this paper we use the CONGEST model [10], where a message may contain up to  $B$  bits, for some parameter  $B$  of the model. We note that typically, it is assumed that  $B = \Theta(\log n)$  (this means that each message may carry a constant number of node IDs and variables of polynomial magnitude).

**Our results.** The main result in this paper is that in the CONGEST model, even for graphs of diameter as small as  $\Theta(\log n)$ , the number of rounds required to reach a stable matching cannot be less than  $\Omega(\sqrt{n/B} \log n)$  in the worst case. The result holds even if we relax the notion of stability in two natural ways, namely approximation and  $\varepsilon$ -stability. To explain these relaxations, let us first define the standard notion of *blocking pairs*: a pair of players is called blocking with respect to a given matching  $M$  if these two players prefer each other to their matches under  $M$ . A matching is stable, then, if it does not induce any blocking pair. Our lower bound holds even if we allow the algorithm to produce a matching which induces up to  $O(\sqrt{n})$  blocking pairs. The second relaxation is the concept of  $\varepsilon$ -stability, where we consider a pair as  $\varepsilon$ -blocking only if by matching with each other they can improve the rank of their mate by more than  $\varepsilon$ , relatively speaking (see a precise definition in Section 2). Our lower bound holds for  $\varepsilon$ -stability as well.

We remark that the same technique extends to lower diameter graphs, at the price of weaker bounds: In the extreme case, we can show that there are graphs of diameter 6 that cannot be solved in fewer than  $\Omega((n/B)^{1/3})$  rounds. For  $\varepsilon$ -stability, this lower bound holds for diameter 10.

**More related work.** There are many papers devoted to Stable Marriage and its variants. Gusfield and Irving sur-

vey the state of the art up to 1989 in their book [4]. Some computational hardness results for stable marriage were obtained later [6]. Recently there is a renewed interest in the subject (e.g., a whole workshop was devoted to it [5] in 2008). There are a few new directions of research. For example, motivated by organ transplants donor scenarios, there are papers studying 3-way stable matchings: there are three sets of players, usually thought of as men, women and dogs. In one variant each player ranks the players of the other sets, and in another variant men rank only women, women rank only dogs, and dogs rank only men. The goal is to match them into families of man, woman and dog.

Regarding distributed computation, as mentioned above, no work that we are aware of studies the complexity of the problem from the distributed viewpoint. However, distributed matching is a well-studied problem. While maximum matching cannot be found in less than diameter time, good approximations can be found in logarithmic time even for weighted matching [9]. On the other hand, there are lower bounds on the time required to compute an approximate maximum matching even with unbounded-size messages [7].

A very relevant line of work is the lower bounds on distributed minimum-weight spanning tree (MST) computation. Peleg and Rubinfeld [11] showed that any protocol for MST in the CONGEST model takes  $\Omega(\sqrt{n}/B)$  time for graphs of diameter  $\Theta(\log n)$ . Lotker et al. [8] extended the ideas of [11] to a lower bound of  $\Omega(n^{1/3}/B)$  for graphs of diameter 4, and  $\Omega(n^{1/4}/\sqrt{B})$  for graphs of diameter 3. Elkin [2] unified and improved all the above results, and extended it to approximation algorithms. Specifically, Elkin shows a lower bound of  $T^{2+\frac{2}{D-2}} \cdot \rho = \Omega\left(\frac{n}{DB}\right)$ , where  $T$  is the running time,  $4 \leq D \leq O(\log n)$  is the diameter of the graph, and  $\rho$  is the approximation factor of the algorithm.

## 2 Model and Preliminaries

In this section we define the stable marriage problem, the CONGEST computational model, and an auxiliary problem called the Mailing Problem. The Mailing Problem will be used to prove our lower bounds in Sections 3 and 4.

**Stable Marriage.** We consider the following variant of the stable marriage problem, denoted SM. The input consists of an undirected bipartite graph  $G = (V, E)$ , and a preference list  $\text{Pref}_v$  for each node  $v \in V$ .  $\text{Pref}_v$  is an ordered list of all neighbors of  $v$ . We write  $\text{Pref}_v(u) = i$  for neighbors  $u, v$  if  $u$  is the  $i$ th node in  $\text{Pref}_v$ , and say that  $u$  is ranked  $i$  in  $v$ 's preference list.  $\text{Pref}_v(u) = 1$  means that  $u$  is the neighbor most preferred by  $v$ . If  $u$  is not a neighbor of  $v$ , we assume that  $\text{Pref}_v(u) = \infty$ . A *matching* is a set of disjoint edges. Given a matching  $M$  with  $(u, v) \in M$ , we denote  $M(v) = u$  and  $M(u) = v$ . If  $v$  is unmatched under  $M$ , we

assume that  $\text{Pref}_v(M(v)) = \infty$ . We can now define the key concept of blocking pairs.

**Definition 2.1** A pair of nodes  $(u, v)$  is said to be blocking with respect to  $M$  if  $\text{Pref}_v(u) < \text{Pref}_v(M(v))$  and  $\text{Pref}_u(v) < \text{Pref}_u(M(u))$ .

We note that in the terminology of [4], our variant is the one with “unacceptable partners,” because only pairs connected by an edge are eligible to the matching. Also, we allow the two parts of the node set to be of unequal size, but for convenience, we do not allow “indifference”, i.e., each node has a strict ordering over its neighbors.<sup>1</sup> We generalize the notion of blocking pairs as follows.

**Definition 2.2** Let  $0 \leq \varepsilon \leq 1$ . A pair of nodes  $(u, v)$  is said to be  $\varepsilon$ -blocking with respect to  $M$  if  $\text{Pref}_v(u) < \text{Pref}_v(M(v)) - \varepsilon|\text{Pref}_v|$  and  $\text{Pref}_u(v) < \text{Pref}_u(M(u)) - \varepsilon|\text{Pref}_u|$ .

Note that a blocking pair according to Definition 2.1 is a 0-blocking pair according to Definition 2.2. Intuitively, the notion of  $\varepsilon$ -blocking pair formalizes the notion that stability is violated only when there is a pair that can improve their matches “a lot” by matching to each other: a minor improvement might just not worth the trouble.

Finally, we can define a stable matching.

**Definition 2.3** A matching  $M$  is said to be stable (resp.,  $\varepsilon$ -stable for some  $0 \leq \varepsilon \leq 1$ ) if there are no blocking pairs (resp.,  $\varepsilon$ -blocking pairs) with respect to  $M$ .

A matching is a  $\rho$ -approximation to a stable marriage if it induces at most  $(1 - \rho)n$  blocking pairs.

**Computational Model.** We use the standard synchronous message passing model called CONGEST in [10]. Briefly, in the CONGEST model, an undirected graph  $G = (V, E)$  represents the system, where vertices represent processors and edges represent bidirectional communication links. We use the terms “processor,” “node” and “vertex” interchangeably. We denote  $|V| = n$ . Each vertex has a set of input variables, output variables, and some local variables. An assignment to these variables is the local state of the processor. It is assumed that each processor has a unique ID of  $O(\log n)$  bits. At time 0, the environment sets the state of the input variables, and the execution then proceeds in global synchronous rounds, where each round consists of the following. Processors first send messages to their neighbors, then receive messages sent to them in that round, and finally make a local computation. Each processor can halt its computation. The protocol is terminated when all local nodes have halted. The CONGEST model requires that no

<sup>1</sup>The graph must be bipartite, or otherwise it may be the case that no stable marriage exists.

message is longer than  $B$  bits, for some given parameter  $B$  (typically  $B = O(\log n)$ ).

In the Distributed Stable Marriage problems (denoted DSM and  $\varepsilon$ -DSM), the input to each processor is its preference list, and the output at each processor is a register that either points to its match or indicates that the processor is unmatched.

**The Mailing Problem.** To prove a lower bound on DSM, we shall use a lower bound on the following variant of the Mailing Problem [11]. The input consists of two disjoint sets of vertices  $S, R \subseteq V$ , referred to as the *senders* and the *receivers*, respectively. Each set contains  $b$  vertices for some given integer  $b \geq 1$ , and we denote  $S = \{s_1, \dots, s_b\}$  and  $R = \{r_1, \dots, r_b\}$ . For  $1 \leq j \leq b$ , the sender  $s_j$  has one input binary variable  $X_j^s$ , and the receiver  $r_j$  has one binary output variable  $X_j^r$ . An instance of the mailing problem is an assignment of one bit to the input variable of each sender. The requirement is that when the protocol terminates,  $X_j^r = X_j^s$  for all  $1 \leq j \leq b$ .

### 3 A Time Lower Bound on The DSM Problem

In this section, we prove our main result, namely a lower bound for the DSM problem. Intuitively, the idea is to use graphs that have a small diameter, but with a certain bottleneck structure. Our lower bound uses a construction introduced by Peleg and Rubinovich [11], followed by Lotker et al. [8], and extended by Elkin [2]. Specifically, we follow [2].

We first construct a family of graphs  $\mathcal{G}$  on which the mailing problem is hard, and then show that any algorithm for DSM can solve the mailing problem on  $\mathcal{G}$ .

#### 3.1 The Graph Family $\mathcal{G}$ and the Mailing Problem

The graph family  $\mathcal{G}$  is defined parametrically as follows. Let  $\Gamma$ ,  $m$  and  $p$  be positive integer parameters, such that  $m \geq 11$  is odd. The number of vertices in our graphs is  $n = (m + 1)\Gamma + \frac{(m+1)^{1+1/p}-1}{(m+1)^{1/p}-1} + \frac{m+1}{2}$ . Note that  $n = \Theta(\Gamma \cdot m)$ . Let  $d = (m + 1)^{1/p}$  (assume that  $d$  is an integer).

The graph family  $\mathcal{G}$  contains one  $n$ -vertex graph  $G_{\Gamma, m, p} = (V_{\Gamma, m, p}, E_{\Gamma, m, p})$  for every allowed combination of the parameters  $\Gamma$ ,  $m$  and  $p$ . The graph  $G_{\Gamma, m, p}$  comprises the following building blocks: see simplified version in Figure 1 (we note that technically,  $d$  cannot be odd). First,  $G_{\Gamma, m, p}$  contains  $\Gamma$  vertex-disjoint paths  $P_1, P_2, \dots, P_\Gamma$  with  $m + 1$  vertices each. Formally, the  $k$ 'th path  $P_k$  is defined by

$$\begin{aligned} V(P_k) &\stackrel{\text{def}}{=} \{v_{0,k}, v_{1,k}, \dots, v_{m,k}\}, \\ E(P_k) &\stackrel{\text{def}}{=} \{(v_{i,k}, v_{i+1,k}) : 0 \leq i < m\}. \end{aligned}$$

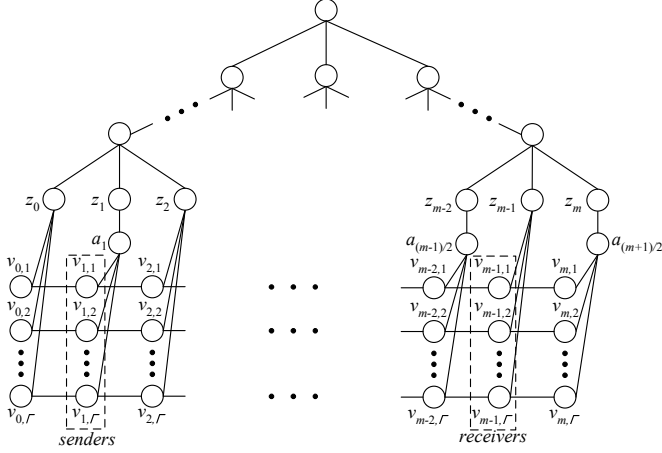


Figure 1. The graph family  $\mathcal{G}$ .

The second building block is a complete  $d$ -ary tree  $\tau$  of depth  $p$  and  $m + 1$  leaves. Its vertex set, denoted  $V(\tau)$ , is disjoint from  $\bigcup_{i=1}^{\Gamma} V(P_i)$ . The leaves of  $\tau$  are denoted by  $z_0, z_1, \dots, z_m$ . To make the graph bipartite, we add a set of  $\frac{m+1}{2}$  vertices  $A \stackrel{\text{def}}{=} \{a_1, a_2, \dots, a_{(m+1)/2}\}$ .

The components above are connected by the following edges.

$$\begin{aligned} & \{(z_i, v_{i,j}) : 1 \leq j \leq \Gamma, 0 \leq i \leq m, i \text{ is even}\} \\ & \{(z_i, a_{(i+1)/2}) : 0 \leq i \leq m, i \text{ is odd}\} \\ & \{(a_{(i+1)/2}, v_{i,j}) : 1 \leq j \leq \Gamma, 0 \leq i \leq m, i \text{ is odd}\} \end{aligned}$$

The following lemmas state straightforward properties of the construction. Their proofs can be found in the appendix.

**Lemma 3.1** Every graph  $G_{\Gamma,m,p} \in \mathcal{G}$  is bipartite.

**Lemma 3.2** The diameter of a graph  $G_{\Gamma,m,p} \in \mathcal{G}$  is  $2p + 4$ .

The key property of the graphs in  $\mathcal{G}$ , which follows directly from [2], is stated next.

**Lemma 3.3** Consider any deterministic protocol for the Mailing Problem running on  $G_{\Gamma,m,p}$ , where the senders are  $v_{1,1}, v_{1,2}, \dots, v_{1,\Gamma}$ , and the receivers are  $v_{m-1,1}, v_{m-1,2}, \dots, v_{m-1,\Gamma}$ . Let  $t < m$ . Then the total number of configurations of local states of the receiver nodes after  $t$  steps of the protocol is at most  $(2^{B+1} - 1)^{t \cdot p \cdot d}$ .

Informally, Lemma 3.3 says that in every communication round before time  $m$ , at most  $(B + 1) \cdot p \cdot d$  information bits can arrive at the receiver nodes. Intuitively, this is the bandwidth the tree  $\tau$  can provide; after  $m$  time units, information may also arrive on the paths, potentially delivering  $\Gamma \cdot B$  bits in each round.

We can now derive our result which is the anchor for our lower bounds on DSM. We do not attempt to optimize the constants here.

**Theorem 3.4** Let  $m = \left(\frac{n}{p \cdot B}\right)^{1/2 - \frac{1}{2(2p+1)}}$  and let  $0.8 < c \leq 1$  be a constant. Any deterministic protocol for the mailing problem in  $G_{\Gamma,m,p}$  that delivers correctly at least  $c \cdot \Gamma$  bits requires  $\Omega(m)$  rounds.

**Proof:** Suppose that a given protocol always delivers at least  $c \cdot \Gamma$  bits in  $t$  steps for some constant  $c > 0.8$ . Let  $\mathcal{R}$  denote the set of all possible configurations of the receiver nodes after  $t$  steps, for all possible inputs at the sender nodes. Since by assumption, at time  $t$  each receiver configuration encodes  $i$  correct bits for some  $c\Gamma \leq i \leq \Gamma$ , a receiver configuration may correspond to at most  $\sum_{i=c\Gamma}^{\Gamma} \binom{\Gamma}{i}$  sender configurations. Now, there are  $2^{\Gamma}$  possible sender inputs, and hence it must be the case that

$$\begin{aligned} |\mathcal{R}| & \geq \frac{2^{\Gamma}}{\sum_{i=c\Gamma}^{\Gamma} \binom{\Gamma}{i}} \geq \frac{2^{\Gamma}}{\binom{\Gamma}{c\Gamma} 2^{\Gamma - c\Gamma}} \geq \frac{2^{c\Gamma}}{2^{\Gamma H(c)}} \\ & = 2^{\Gamma(c - H(c))} = 2^{\Omega(\Gamma)}. \end{aligned} \quad (1)$$

The first inequality follows from the explanation above.<sup>2</sup> The second inequality follows from the fact that  $\sum_{i=k}^n \binom{n}{i} \leq \binom{n}{k} 2^{n-k}$ . The third inequality follows from the fact that for  $0 < c < 1$ ,  $\binom{n}{cn} \leq 2^{nH(c)}$ , where  $H(\cdot)$  is the entropy function. The last equality holds because  $H(c) < c$  for  $c > 0.8$ .

Combining Eq. (1) with Lemma 3.3 which says that  $|\mathcal{R}| \leq (2^{B+1} - 1)^{t \cdot p \cdot d}$ , we may conclude that

$$(2^{B+1} - 1)^{t \cdot p \cdot d} \geq 2^{\Omega(\Gamma)}.$$

Recalling that  $d = m^{1/p}$  and that  $\Gamma = \Theta(n/m)$ , we obtain  $t = \Omega(n / (Bpm^{1+1/p}))$ . Using  $m = \left(\frac{n}{p \cdot B}\right)^{1/2 - \frac{1}{2(2p+1)}}$ , the result follows. ■

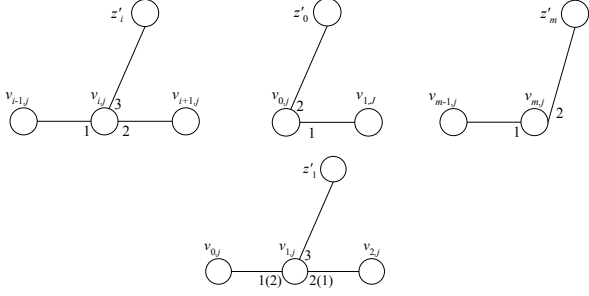
### 3.2 DSM in $\mathcal{G}^{DSM}$

We now extend the construction for the DSM problem. The family  $\mathcal{G}^{DSM}$  contains  $2^{\Gamma}$  DSM instances for every  $G_{\Gamma,m,p} \in \mathcal{G}$ . All of these  $2^{\Gamma}$  instances have the same vertex set  $V_{\Gamma,m,p}$  and the same edge set  $E_{\Gamma,m,p}$ . By Lemma 3.1, the graphs in  $\mathcal{G}$  are bipartite, and therefore can be legal DSM instances.

The main idea is to assign preference lists such that in a stable matching, each path  $P_j$  is an alternating path, defined as follows.

**Definition 3.5** Given a matching  $M$ , an alternating path is a path  $v_0, v_1, \dots, v_n$  in which the edges alternately belong to  $M$ , i.e., either  $\{(v_0, v_1), (v_2, v_3), \dots\} \subseteq M$  and  $\{(v_1, v_2), (v_3, v_4), \dots\} \cap M = \emptyset$  or vice versa.

<sup>2</sup>This is essentially the sphere-packing bound from the theory of error correcting codes, but for the reverse inequality.



**Figure 2. Left: preferences fixed at all  $2^\Gamma$  instances. The labels on the edges are the respective preferences. Right: possible preferences assigned to  $v_{1,j}$ . The actual preferences in these nodes depend on the particular instance.**

Since there are only two alternatives of stable matching for such a path, it can be used to implicitly transmit a bit of information. Therefore, by knowing that the chosen matchings are stable, we can deliver a string of bits from the senders to the receivers. Hence, we can reduce the mailing problem to the DSM problem.

Specifically, the preferences in  $\mathcal{G}^{DSM}$  are as follows. The following preference lists are fixed, i.e., they are assigned to all instances in  $\mathcal{G}^{DSM}$  (see Figure 2). Let  $z'_i = z_i$  for even  $i$ , and  $z'_i = a_{(i+1)/2}$  for odd  $i$ .

$$\begin{aligned} \text{Pref}_{v_{i,j}}(v_{i-1,j}) = 1, \text{ Pref}_{v_{i,j}}(v_{i+1,j}) = 2 \text{ for } 2 \leq i \leq m-1 \\ \text{Pref}_{v_{i,j}}(z'_i) = 3 \text{ for } 1 \leq i \leq m-1 \\ \text{Pref}_{v_{0,j}}(v_{1,j}) = 1, \text{ Pref}_{v_{0,j}}(z'_0) = 2 \\ \text{Pref}_{v_{m,j}}(v_{m-1,j}) = 1, \text{ Pref}_{v_{m,j}}(z'_m) = 2 \end{aligned}$$

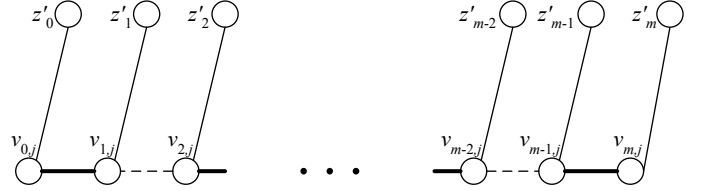
We assign the following preferences to the vertices of the tree  $\tau$ . Vertices with even distance from the root rank their leftmost child as their most preferred, and vertices with odd distance from the root rank their parent as their most preferred. These preferences ensure that in any stable matching, all nodes at even distance from the root are matched with their leftmost child (other rankings in the tree are immaterial).

The variable part in the instances in  $\mathcal{G}^{DSM}$  is the preferences of the vertices  $v_{1,j}$ , for  $1 \leq j \leq \Gamma$  (see Figure 2). Each node  $v_{1,j}$  may have  $\text{Pref}_{v_{1,j}}(v_{0,j}) = 1$  and  $\text{Pref}_{v_{1,j}}(v_{2,j}) = 2$ , or  $\text{Pref}_{v_{1,j}}(v_{0,j}) = 2$  and  $\text{Pref}_{v_{1,j}}(v_{2,j}) = 1$ . There is exactly one DSM instance in  $\mathcal{G}^{DSM}$  for each of the  $2^\Gamma$  possible combinations of  $v_{1,j}$ ,  $1 \leq j \leq \Gamma$ .

We now state simple properties of the construction.

**Lemma 3.6** *Under any stable matching, the path  $P_j$  is an alternating path for all  $1 \leq j \leq \Gamma$ .*

**Proof:** Let  $M$  be a stable matching. Obviously, two adjacent edges cannot belong to  $M$ . Assume for contradiction that for some  $1 \leq i \leq m-1$ , the edges  $(v_{i-1,j}, v_{i,j})$  and  $(v_{i,j}, v_{i+1,j})$  are not in  $M$ . Also, assume that  $(v_{i,j}, z'_i)$  are matched. There are two cases to consider: either  $i > 1$  or  $i = 1$ . In the first case,  $(v_{i,j}, v_{i+1,j})$  is a blocking pair. This can be seen from the preference lists defined earlier. Since  $\text{Pref}_{v_{i+1,j}}(v_{i,j}) = 1$ ,  $v_{i+1,j}$  prefers to be matched with  $v_i$  rather than with its current matching. Also, since  $\text{Pref}_{v_{i,j}}(v_{i+1,j}) < \text{Pref}_{v_{i,j}}(z'_i)$ ,  $v_{i,j}$  prefers to be matched with  $v_{i+1,j}$  rather than with its current matching. Hence,  $(v_{i,j}, v_{i+1,j})$  is a blocking pair. In the other case,  $i = 1$ ,  $\text{Pref}_{v_{0,j}}(v_{1,j}) = 1$  and  $\text{Pref}_{v_{2,j}}(1,j) = 1$  by construction. If  $\text{Pref}_{v_{1,j}}(v_{0,j}) = 1$ , then  $(v_{0,j}, v_{1,j})$  is a blocking pair. Otherwise,  $(v_{1,j}, v_{2,j})$  is a blocking pair. This is due to the simple fact that if  $\text{Pref}_u(v) = \text{Pref}_v(u) = 1$ , then  $(u, v)$  are matched in any stable matching. In the case where  $v_{i,j}$  is not matched in  $M$ , the statements above also apply. In either case, we get that  $M$  is not stable, in contradiction with the assumption. ■



**Figure 3. The matchings along each path  $P_j$  are either the dashed edges or the bold edges.**

From Lemma 3.6 it is evident that along each path  $P_j$ , there are only two possible matchings of its vertices. Namely,  $(v_{i,j}, v_{i+1,j})$  are matched either for every even  $i$  or for every odd  $i$ ,  $0 \leq i \leq m-1$ . The only thing that determines which of these two matchings is chosen, is the preference list of  $v_{1,j}$ . It is also evident that the matchings along each path do not depend on the preferences assigned to the vertices of the tree.

**Lemma 3.7** *For every  $1 \leq j \leq \Gamma$ , the preferences of  $v_{1,j}$  determine the matching of all vertices of  $P_j$  in any stable matching. Namely, if  $\text{Pref}_{v_{1,j}}(v_{0,j}) = 1$  then  $(v_{i,j}, v_{i+1,j})$  are matched for every even  $i$ ,  $0 \leq i \leq m-1$ . Otherwise, if  $\text{Pref}_{v_{1,j}}(v_{2,j}) = 1$ ,  $(v_{i,j}, v_{i+1,j})$  are matched for every odd  $i$ ,  $0 \leq i \leq m-1$ .*

**Proof:** As argued earlier, if  $\text{Pref}_{v_{1,j}}(v_{0,j}) = 1$  then  $(v_0, v_1)$  are matched in any stable matching. Since  $P_j$  is an alternating path, we get that  $(v_{i,j}, v_{i+1,j})$  are matched for every even  $i$ ,  $0 \leq i \leq m-1$ . Otherwise,  $(v_1, v_2)$  are matched in

any stable matching. We get that  $(v_{i,j}, v_{i+1,j})$  are matched for every odd  $i$ ,  $0 \leq i \leq m-1$ . ■

**Corollary 3.8** *Any deterministic algorithm for distributed stable marriage on graphs with diameter  $D$  requires at least  $\Omega(D)$  rounds.*

**Proof Sketch:** Consider a graph that consists of a single path  $P_j$  of length  $D$  as above. By Lemma 3.7, the match of  $v_{m-1}$  depends on  $\text{Pref}_{v_1}$ , which is indistinguishable at  $v_{m-1}$  until time  $D-1$ . ■

Note that the lower bound of Corollary 3.8 does not hold for approximate stable matching: a single blocking pair can flip the match at  $v_{m-1}$ .

We can now prove our main result, namely a time lower bound on DSM.

**Theorem 3.9** *Consider any deterministic  $\rho$ -approximation for distributed stable marriage on graphs with diameter  $D$  for  $D \in \{6, 8, 10, \dots\}$ , and assume that  $\rho = 1 - o(1/\sqrt{n})$ . Then the algorithm requires  $\Omega\left(\left(\frac{n}{D \cdot B}\right)^{\frac{1}{2} - \frac{1}{2(D-3)}}\right)$  communication rounds.*

**Proof:** The lower bound is proved by the following reduction of the Mailing problem on  $\mathcal{G}$  to DSM on  $\mathcal{G}^{DSM}$ . Given an input  $X_1^s, \dots, X_\Gamma^s$  to the mailing problem in  $G \in \mathcal{G}$ , we construct an instance of DSM as follows. Preferences of all nodes except the sender nodes are fixed as described above. We set the following preferences to the vertices  $v_{1,j}$ . If  $X_j^s = 1$ , then  $\text{Pref}_{v_{1,j}}(v_{0,j}) = 1$  and  $\text{Pref}_{v_{1,j}}(v_{2,j}) = 2$ ; and if  $X_j^s = 0$ , then  $\text{Pref}_{v_{1,j}}(v_{0,j}) = 2$  and  $\text{Pref}_{v_{1,j}}(v_{2,j}) = 1$ . This setting is performed locally by the vertices  $v_{1,j}$ , and therefore requires no distributed computation. When the DSM algorithm terminates, each node knows its match. To complete the reduction we define the output transformation by the rule that receiver nodes  $v_{m-1,j}$  set  $X_j^r = 1$  if and only if  $(v_{m-1,j}, v_{m,j})$  is in the matching.

Suppose now that we have a  $\rho$ -approximate stable matching  $M$  in the graph, for some  $\rho = 1 - o(1/\sqrt{n})$ . Then under  $M$ , there are at most  $(1-\rho)n = o(\sqrt{n})$  blocking pairs, and hence there are at least  $\Gamma - o(\sqrt{n})$  paths  $P_j$  without any blocking pair, which means that at least  $\Gamma - o(\sqrt{n})$  paths  $P_j$  are alternating paths under  $M$ , and hence, by Lemma 3.7, the output variables  $X_j^s$  in their receiver nodes are correct. Now, recall that by our choice of  $m$ , we have that  $\Gamma = \Theta(n/m) \gg \sqrt{n}$ , and hence the number of correct bits delivered using the reduction to DSM is  $\Gamma(1 - o(1))$ . In this case Theorem 3.4 applies, and we may conclude that the running time of the algorithm under the specific graphs described there is  $\Omega\left(\left(\frac{n}{D \cdot B}\right)^{\frac{1}{2} - \frac{1}{2(D-3)}}\right)$ . ■

By substituting  $D = \log n$ , we get the following.

**Corollary 3.10** *Any deterministic  $\rho$ -approximation for DSM on graphs with diameter  $\Theta(\log n)$  and  $\rho = o(1/\sqrt{n})$ , requires  $\Omega\left(\sqrt{\frac{n}{B \cdot \log n}}\right)$  rounds.*

By substituting  $D = 6$ , we get the following.

**Corollary 3.11** *Any deterministic  $\rho$ -approximation for DSM on graphs with diameter 6 and  $\rho = o(1/\sqrt{n})$ , requires  $\Omega\left(\left(\frac{n}{B}\right)^{\frac{1}{3}}\right)$  rounds.*

## 4 Extension to the $\varepsilon$ -DSM problem

In this section, we extend Theorem 3.9 to the relaxed model of  $\varepsilon$ -stability. We show that even for  $\varepsilon$  arbitrarily close to  $1/2$ , the lower bound deteriorates only by a constant factor.

We define a graph family  $\mathcal{G}^{\varepsilon-DSM}$  which is a generalization of  $\mathcal{G}^{DSM}$  from Section 3.2: we augment the graphs of  $\mathcal{G}^{DSM}$  with a few additional vertices as follows. For each  $v_{i,j}$ ,  $1 \leq i \leq m-1$ ,  $1 \leq j \leq \Gamma$ , we add  $2l$  pairs of vertices for some integer  $l$ . Formally, we add the following two sets of  $2l$  vertices each, denoted  $U_{i,j}$  and  $W_{i,j}$  (see Figure 4).

$$\begin{aligned} V(U_{i,j}) &= \{u_{i,j}^1, u_{i,j}^2, \dots, u_{i,j}^{2l}\} \\ V(W_{i,j}) &= \{w_{i,j}^1, w_{i,j}^2, \dots, w_{i,j}^{2l}\} \end{aligned}$$

These vertices are connected to the graph with the following edges.

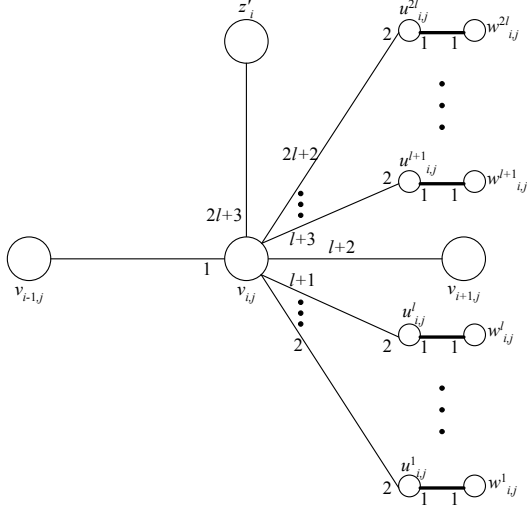
$$\{(v_{i,j}, u_{i,j}^k) : 1 \leq k \leq 2l\}, \text{ and } \{(w_{i,j}^k, u_{i,j}^k) : 1 \leq k \leq 2l\}$$

Note that this addition increases the number of vertices in the graph, roughly by a factor of  $4l$ . For  $1 \leq i \leq m-1$ ,  $1 \leq j \leq \Gamma$  and  $1 \leq k \leq 2l$ , we assign the following preference lists.

$$\begin{aligned} \text{Pref}_{w_{i,j}^k}(u_{i,j}^k) &= 1 \\ \text{Pref}_{u_{i,j}^k}(w_{i,j}^k) &= 1, \quad \text{Pref}_{u_{i,j}^k}(v_{i,j}) = 2 \\ \text{Pref}_{v_{i,j}}(z_i^l) &= 2l + 3 \\ \text{Pref}_{v_{i,j}}(u_{i,j}^k) &= k + 1 \text{ for } 1 \leq k \leq l \\ \text{Pref}_{v_{i,j}}(u_{i,j}^k) &= k + 2 \text{ for } l + 1 \leq k \leq 2l \end{aligned}$$

For  $2 \leq i \leq m-1$ , we assign  $\text{Pref}_{v_{i,j}}(v_{i-1,j}) = 1$  and  $\text{Pref}_{v_{i,j}}(v_{i+1,j}) = l + 2$ . For  $i = 1$ , we assign either  $\text{Pref}_{v_{1,j}}(v_{0,j}) = 1$  and  $\text{Pref}_{v_{1,j}}(v_{2,j}) = l + 2$ , or  $\text{Pref}_{v_{1,j}}(v_{0,j}) = l + 2$  and  $\text{Pref}_{v_{1,j}}(v_{2,j}) = 1$ , with one  $\varepsilon$ -DSM instance in  $\mathcal{G}^{\varepsilon-DSM}$  for each of the  $2^\Gamma$  combinations. Notice that  $(w_{i,j}^k, u_{i,j}^k)$  are matched in any  $\varepsilon$ -DSM for any  $\varepsilon < \frac{1}{2}$ . Therefore,  $(v_{i,j}, u_{i,j}^k)$  are never matched for such  $\varepsilon$ 's, for any  $1 \leq k \leq 2l$ .

**Lemma 4.1** *For every  $1 \leq j \leq \Gamma$  and for any  $\varepsilon < \frac{l+1}{2l+3}$ , the path  $P_j$  is an alternating path.*



**Figure 4. The additional two sets of  $2l$  vertices each, for some  $v_{i,j}$ . The bold edges are in the matching for any  $\varepsilon < \frac{1}{2}$ .**

**Proof:** Let  $M$  be an  $\varepsilon$ -SM. Obviously, two adjacent edges cannot belong to  $M$ . Assume for contradiction that for some  $1 \leq i \leq m-1$ , the edges  $(v_{i-1,j}, v_{i,j})$  and  $(v_{i,j}, v_{i+1,j})$  are not in  $M$ . Since  $(v_{i,j}, w^k_{i,j})$  are not matched for any  $1 \leq k \leq 2l$ ,  $v_{i,j}$  is either matched with  $z'_i$  or not matched at all. First consider the case where  $(v_{i,j}, z'_i)$  are matched. There are two cases to consider: either  $i > 1$  or  $i = 1$ . In the first case, we get the following.  $\text{Pref}_{v_{i,j}}(z'_i) - \text{Pref}_{v_{i,j}}(v_{i+1,j}) = l+1 = \frac{l+1}{2l+3} \cdot |\text{Pref}_{v_{i,j}}| > \varepsilon \cdot |\text{Pref}_{v_{i,j}}|$ . Also, if  $(i+1) \neq m$ , then  $\text{Pref}_{v_{i+1,j}}(M(v_{i+1,j})) - \text{Pref}_{v_{i+1,j}}(v_{i,j}) \geq l+1 = \frac{l+1}{2l+3} \cdot |\text{Pref}_{v_{i+1,j}}| > \varepsilon \cdot |\text{Pref}_{v_{i+1,j}}|$ . If  $(i+1) = m$ , then  $\text{Pref}_{v_{i+1,j}}(M(v_{i+1,j})) - \text{Pref}_{v_{i+1,j}}(v_{i,j}) \geq 1 = \frac{1}{2} \cdot |\text{Pref}_{v_{i+1,j}}| > \varepsilon \cdot |\text{Pref}_{v_{i+1,j}}|$ . From the above statements it can be seen that  $(v_{i,j}, v_{i+1,j})$  is an  $\varepsilon$ -blocking pair.

In the other case,  $i = 1$ ,  $\text{Pref}_{v_{0,j}}(v_{1,j}) = 1$  and  $\text{Pref}_{v_{2,j}}(v_{1,j}) = 1$  by construction. If  $\text{Pref}_{v_{1,j}}(v_{0,j}) = 1$ , then  $(v_{0,j}, v_{1,j})$  is an  $\varepsilon$ -blocking pair. Otherwise,  $(v_{1,j}, v_{2,j})$  is an  $\varepsilon$ -blocking pair. This can be seen by similar inequalities as in the first case. In the case where  $v_{i,j}$  is not matched in  $M$ , the statements above also apply. In summary, we get that in either case  $M$  is not an  $\varepsilon$ -SM, in contradiction with the assumption. ■

Once again we get that along each path  $P_j$  there are only two possible  $\varepsilon$ -stable matchings. The only thing that determines which of these two matchings is chosen, is the preference list of  $v_{1,j}$ . Therefore, the reduction of the mailing problem to the DSM problem can be applied for the  $\varepsilon$ -DSM problem as well, for any  $\varepsilon < \frac{l+1}{2l+3}$ .

We note that the additional vertices affect the lower

bound attained for the mailing problem, but do not affect its proof of correctness. This is due to the fact that these vertices are “dead-ends”, i.e., they are not on any simple path between any sender and any receiver. Therefore, they cannot contribute to any message passing between the senders and the receivers. We conclude with the following extension of Theorem 3.9 to  $\varepsilon$ -stability.

**Theorem 4.2** Consider any deterministic  $\rho$ -approximation for distributed  $\varepsilon$ -stable marriage on graphs with diameter  $D$  for  $D \in \{10, 12, 14, \dots\}$ , where  $\varepsilon < 1/2$  is constant and  $\rho = 1 - o(1/\sqrt{n})$ . Then the algorithm requires  $\Omega\left(D + \left(\frac{n}{D \cdot B}\right)^{\frac{1}{2} - \frac{1}{2(D-3)}}\right)$  communication rounds.

We remark that corollaries 3.10 and 3.11 extend to  $\varepsilon$ -stability as well, for any  $\varepsilon < 1/2$ .

## 5 Algorithm for the $\varepsilon$ -DSM problem

In this section, we show a distributed algorithm for the  $\varepsilon$ -DSM problem (DSM is just a special case of  $\varepsilon$ -DSM). Our algorithm is a distributed version of an extension of Gale and Shapley algorithm, augmented with a termination detection algorithm.

Let us first describe the basic algorithm. The nodes are divided into two sets, say *active* and *passive* according to the bipartition of the graph (we assume that each node knows whether it is a “man” or a “woman”). Active nodes may only propose, and passive nodes may only reject. The algorithm proceeds in double rounds. First, a rejected active node proposes to the top node on its preference list (initially, all active nodes consider themselves rejected, and their preferences are just the input). Then each passive node that has at least one proposal sends a “reject” message to all nodes (including to nodes that haven’t proposed to it) except for the most preferred proposer and for nodes whose ranking is better than the best current proposer by at least an  $\varepsilon$  fraction. More formally, if the best proposal that a passive node  $j$  received is from an active node  $i$  on  $j$ ’s list, then  $j$  sends reject messages to all nodes, except  $i$ , whose rank in  $j$ ’s list is at most  $\text{Pref}_j(i) - \varepsilon |\text{Pref}_j|$ . An active node  $i'$  that receives a reject message from a passive node  $j$  removes  $j$  from its preference list. This way an active node  $i$  proposes only to passive nodes that appear to be eligible matches to  $i$  at the time of the proposal. The algorithm terminates when there are no more rejected nodes.

Our idea to make this algorithm distributed is to add the following termination detection mechanism: Let  $D$  be the diameter of the graph. We require each rejected node to broadcast the fact that the protocol has not terminated yet, on a shortest-path tree. Thus, if  $D$  consecutive steps pass without any such broadcast received by a node  $v$ ,  $v$  can

safely conclude that no rejected nodes are in the system, and it can therefore safely halt its local computations.

Finding the diameter of a graph is easy to do in diameter time (see, e.g., [10]). We thus obtain the following result.

**Theorem 5.1**  $\varepsilon$ -DSM can be solved on any graph in time  $O(D + \min(|E|, \varepsilon^{-1}n))$ .

**Proof:** It takes  $O(D)$  time to find the value of  $D$  and to detect termination. Regarding the stable marriage computation, note first that before stabilization of the matching, in each round at least one passive node improves the rank of its match by at least a fraction  $\varepsilon$ . It follows that the total number of rounds until stabilization of the matching is bounded by  $O(\varepsilon^{-1}n)$ . In addition, note that in each round before stabilization at least one fresh active node is rejected by a passive node, and hence the total number of rounds to be stabilization is also bounded by  $|E|$ . The result follows. ■

For the exact case, we set  $\varepsilon = 1/n$  and the running time is  $O(|E|)$ . We remark that even though in  $O(|E|)$  time one can disseminate the complete input to all nodes, the algorithm above is more attractive for the following reasons. First, depending on the input, it may terminate well before  $\Omega(|E|)$  rounds have passed, while generic dissemination always takes this much time. And second, the local computation required by the algorithm above is quite trivial, while the generic dissemination algorithm requires each node to run the full centralized stable-marriage algorithm.

## 6 Conclusion

In this paper we considered the Stable Marriage problem in a distributed model where messages have bounded length. We showed that the number of communication rounds required to solve the problem in this model cannot be smaller than  $\Omega(\sqrt{n/B \log n})$ , even if the graph has diameter  $\Theta(\log n)$ . On the other hand, our best upper bound for the problem is  $O(|E|)$  communication rounds, or  $O(\varepsilon^{-1}n)$  for  $\varepsilon$ -DSM. While we narrowed the gap, there is still work to do before our original question is settled: what is the true distributed complexity of stable marriage?

## References

- [1] S.-T. Chuang, A. Goel, N. McKeown, and B. Prabhakar. Matching output queueing with a combined input output queued switch. *IEEE Journal on Selected Areas in Communications*, 17(6):1030–1039, June 1999.
- [2] M. Elkin. An unconditional lower bound on the time-approximation tradeoff for the minimum spanning tree problem. *SIAM J. Comput.*, 36(2):463–501, 2006.
- [3] D. Gale and L. S. Shapley. College admissions and the stability of marriage. *American Math. Monthly*, 69:9–15, 1962.
- [4] D. Gusfield and R. W. Irving. *The Stable Marriage Problem: Structure and Algorithms*. MIT Press, Cambridge, MA, USA, 1989.
- [5] M. M. Halldórsson, R. Irving, K. Iwama, and D. Manlove, editors. *MATCH-UP: Matching Under Preferences*, Reykjavk, Iceland, Mar. 2008. Satellite workshop of ICALP 2008. <http://www.dcs.gla.ac.uk/research/algorithms/workshop/>.
- [6] M. M. Halldórsson, K. Iwama, S. Miyazaki, and Y. Morita. Inapproximability results on stable marriage problems. In *LATIN '02: Proc. 5th Latin American Symp. on Theoretical Informatics*, pages 554–568, London, UK, 2002. Springer-Verlag.
- [7] F. Kuhn, T. Moscibroda, and R. Wattenhofer. The price of being near-sighted. In *Proc. 17th Ann. ACM-SIAM Symposium on Discrete Algorithms*, pages 980–989, New York, NY, USA, 2006. ACM.
- [8] Z. Lotker, B. Patt-Shamir, and D. Peleg. Distributed MST for constant diameter graphs. *Distributed Computing*, 18(6):453–460, 2006.
- [9] Z. Lotker, B. Patt-Shamir, and S. Pettie. Improved distributed approximate matching. In *Proc. ACM Symposium on Parallelism in Algorithms and Architecture*, pages 129–136, New York, NY, USA, 2008. ACM.
- [10] D. Peleg. *Distributed Computing: A Locality-Sensitive Approach*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2000.
- [11] D. Peleg and V. Rubinfeld. Near-tight lower bound on the time complexity of distributed MST construction. *SIAM J. Comput.*, 30:1427–1442, 2000.

## Appendix

**Proof of Lemma 3.1:** We give an explicit partition  $V_{\Gamma,m,p} = V_1 \cup V_2$  such that all edges in  $E_{\Gamma,m,p}$  are between  $V_1$  and  $V_2$ . Partition the tree  $\tau$  by assigning  $z_i$  to  $V_1$  for  $0 \leq i \leq m$ ,  $a_i$  to  $V_2$  for  $0 \leq i \leq \frac{m+1}{2}$ , and assign the remaining vertices inductively, level by level. We divide the rest of the vertices as follows. For every  $0 \leq i \leq m$  and  $1 \leq j \leq \Gamma$ , if  $i$  is even then  $v_{i,j} \in V_2$ , and otherwise  $v_{i,j} \in V_1$ .

It remains to show that all edges connect nodes in  $V_1$  with nodes in  $V_2$ . There are three cases to consider. First, consider an edge  $(v_{i,j}, v_{i+1,j})$ . According to the chosen partition, if  $i$  is even then  $v_{i,j} \in V_2$  and  $v_{i+1,j} \in V_1$ , and otherwise  $v_{i,j} \in V_1$  and  $v_{i+1,j} \in V_2$ . Second, consider an edge  $(z_i, v_{i,j})$ . By construction of  $G_{\Gamma,m,p}$ , this edge exists only if  $i$  is even, and we get that  $v_{i,j} \in V_2$  and  $z_i \in V_1$ . Finally, the case of an edge  $(a_{i+1/2}, v_{i,j})$  is analogous to the second case, and the result follows. ■

**Proof of Lemma 3.2:** We first show that the distance between any two vertices in the graph is not greater than  $2p + 4$ . Consider some vertices  $z_i$  and  $v_{i,j}$ . By construction of the tree  $\tau$ , the distance between  $z_i$  and the root of the tree is  $p$ . The distance between  $v_{i,j}$  and  $z_i$  is either 1 or 2. Thus, the total distance between  $v_{i,j}$  and the root of the tree is not greater than  $p + 2$ , and therefore the distance between any two vertices in the graph is not greater than  $2p + 4$ . To see that the diameter is not smaller than  $2p + 4$ , note that the shortest path between  $v_{1,1}$  and  $v_{m,\Gamma}$  goes through the tree  $\tau$ , and therefore the distance between them is  $2p + 4$ . ■

**Proof of Lemma 3.3:** Our construction is nearly identical to Elkin's [2], and our Lemma 3.3 is derived from Lemma 3.4 in [2]. Instead of reproducing the (complex) proof from [2], let us just review the differences here. First, in [2] there is a single sender and single receiver; and second, we introduced extra nodes  $a_i$  that make sure that the graph is bipartite. The extra nodes are clearly insignificant for the construction (they only increase the diameter by an additive constant term). Regarding the single sender and receiver, we note that the single sender  $z_0$  in [2] is connected to all our sender nodes  $v_{1,j}$ , and the single receiver  $z_m$  in [2] is connected to all our receiver nodes  $v_{m-1,j}$ . Obviously, a string at a node can be sent to all nodes connected to it in a single step (one bit per node), and reversely, a set of bits at residing at different nodes can be collected in a single step by a node connected to all of them. It follows that if after  $T$  steps there are at most  $q$  possible states at the single-sender, single-receiver problem, then after  $T - 2$  steps there are at most  $q$  possible states at our version of the problem. ■