

Zvi Lotker · Boaz Patt-Shamir · David Peleg

Distributed MST for constant diameter graphs

Received: 1 December 2003 / Accepted: 14 February 2005 / Published online: 3 May 2006
© Springer-Verlag 2006

Abstract This paper considers the problem of distributively constructing a minimum-weight spanning tree (MST) for graphs of constant diameter in the bounded-messages model, where each message can contain at most B bits for some parameter B . It is shown that the number of communication rounds necessary to compute an MST for graphs of diameter 4 or 3 can be as high as $\Omega(\sqrt[3]{n}/\sqrt{B})$ and $\Omega(\sqrt[4]{n}/\sqrt{B})$, respectively. The asymptotic lower bounds hold for randomized algorithms as well. On the other hand, we observe that $O(\log n)$ communication rounds always suffice to compute an MST deterministically for graphs with diameter 2, when $B = O(\log n)$. These results complement a previously known lower bound of $\Omega(\sqrt[3]{n}/B)$ for graphs of diameter $\Omega(\log n)$.

Keywords Distributed algorithm · Minimum-weight spanning tree

1 Introduction

The communication complexity of distributed network algorithms is often determined by *size* parameters such as the number of vertices or edges in the network. In contrast, the *time* efficiency of distributed solutions for various problems on networks is sometimes largely affected by other types of network parameters. For many natural distributed network problems, the time complexity is controlled by *locality* measures such as the network's diameter D (namely, the maximum length of a shortest path between any two vertices,

measured in hops). This holds, for example, for leader election and related problems [17]. Yet other problems are independent of (or sublinearly dependent on) global parameters. Maximal independent set computations and related problems fall into that category [13].

In this paper we study the time complexity of the problem of distributively computing a minimum spanning tree (abbreviated MST) of the underlying communication graph. Informally, in the MST problem, we are given a communication network, where each edge has a weight; each node initially knows the weight of its incident edges, and the task is to choose a minimal-weight spanning tree of the network, i.e., a connected set of edges of minimal total weight that spans all nodes in the system. It is not hard to see that $\Omega(D)$ time is necessary for distributed MST construction in the worst case, since in some cases, the weights of some edges must be communicated across the system [19].

It is less clear, however, if $O(D)$ time *suffices* for distributed MST construction. Trivially, if the model allows for transmitting messages of unbounded size in a single time unit (cf. [13]), then any function of the distributed inputs, including MST computation, can be solved in $O(D)$ time, simply because all inputs can be disseminated to all nodes in $O(D)$ time.

In this paper we consider a more interesting (and more realistic) model, called below the *B-model*, each link can carry at most B bits at each time unit, for some specified parameter $B > 0$. It is usually assumed that $B = \Theta(\log n)$, where n is the number of nodes in the system: this means that messages can carry node IDs and variables of polynomial size. The classical distributed MST construction algorithms of [8] and [3, 6] are optimal in terms of the total number of communicated bits; however, their time complexity in the *B* model, with $B = \Theta(\log n)$, is $O(n \log n)$ and $O(n)$, respectively. The time complexity was subsequently improved [9, 11], and the best upper bound currently known is $O(D + \sqrt{n} \log^* n)$ [11].

Regarding lower bounds, it was shown in [19] that the time complexity of any deterministic algorithm for MST is $\Omega(\sqrt{n}/(B \log n))$. However, this lower bound was shown

An extended abstract of this work appears in Proceedings of 20th ACM Symposium on Principles of Distributed Computing, August 2001.

Z. Lotker (✉) · B. Patt-Shamir
Department of Electrical Engineering, Tel Aviv University,
Tel Aviv 69978, Israel
E-mail: {zvilo, boaz}@eng.tau.ac.il

D. Peleg
Department of Computer Science, Weizmann Institute,
Rehovot 76100, Israel
E-mail: peleg@wisdom.weizmann.ac.il

only for graphs whose diameter is $\Omega(\log n)$. The time complexity of distributed MST computation for graphs of smaller diameter (i.e., $o(\log n)$) was so far unknown.

In this paper we present negative and positive results for constant-diameter graphs. On the negative side, we extend the lower bound of [19] by presenting constructions that show that the time complexity of distributed MST is $\Omega(\sqrt[3]{n}/\sqrt{B})$ for graphs of diameter 4, and $\Omega(\sqrt[4]{n}/\sqrt{B})$ for graphs of diameter 3. We also show that all lower bounds hold for randomized algorithms as well. On the positive side, we show that if the diameter of the graph is 2, then MST can be constructed deterministically in $O(\log n)$ time. This fact is a consequence of a much more general observation, namely that the B model with network diameter 2 is at least as strong (in terms of asymptotic time complexity) as the CRCW PRAM model. Thus, the existence of $O(\log n)$ time PRAM algorithm for MST computation (e.g., [9]) demonstrates a wide gap in the time complexity of the B model between the cases of network diameters 2 and 3.

The remainder of this paper is organized as follows. In Sect. 2 we define the basic terms we use. In Sect. 3 we present our lower bounds. In Sect. 4 we present the simulation of CRCW PRAM by the B model with diameter 2. We conclude in Sect. 5.

2 Preliminaries

The computation model we consider in this paper, called the B -model, is the standard synchronous message passing model with one important exception: messages have bounded size. In this section, we briefly describe the system and define the MST problem. We also define the mailing problem which is key to our lower bounds.

2.1 The B -model

We assume that the system is described by the standard synchronous message passing model. The reader is referred to standard texts (e.g., [2, 15]) for a full formal exposition of the model. We give a brief summary here. In the synchronous message passing model, there is an undirected graph $G = (V, E)$, where nodes represent processors and edges represent bidirectional communication links. Each node has a set of read-only input variables, write-once output variables, and some read-write local variables. At time 0, the environment is assumed to initialize the input variables, and the protocol sets the initial state of the local variables. Thereafter, computation proceeds in synchronous rounds, where in each round, nodes first send messages to their neighbors, then receive messages sent to them in that round, and finally make an arbitrary (potentially randomized) local computation. Thus, the state of each node is a (possibly randomized) function of its input values and the messages received so far. The protocol is said to terminate when all output variables

have been set. When we refer to the state at time t , we refer to the state after t communication rounds. The time complexity of a protocol is its worst-case number of rounds from initialization to termination.

So far we have described the standard synchronous system model. Now we introduce the important deviation from that model (see [18]): we require that at most one B -bit message can be sent on each link in each direction in every round. Our algorithms assume that B is large enough to allow the transmission of an edge weight and a node ID in a single message (this assumption is not used for the lower bounds). To avoid subtleties that do not affect the asymptotic time complexity, we require that each node transmits at each round exactly B bits on each incident link. (Otherwise, silence can be interpreted as useful information, reducing the time complexity by a constant factor.)

2.2 The distributed MST problem

The *minimum spanning tree (MST)* problem is defined as follows (see, e.g., [12]). We are given a graph $G = (V, E)$, and a *weight function* ω that assigns a nonnegative integer weight $\omega(e)$ to each edge $e = (u, v) \in E$. A *spanning graph* of G is a connected set of edges such that each node in v is incident to at least one edge in the set. The MST task is to find a set of edges $\text{mst}(G) \subseteq E$ whose total weight, $\omega(\text{mst}(G)) \stackrel{\text{def}}{=} \sum_{e \in \text{mst}(G)} \omega(e)$, is minimal (obviously this set is a tree).

In the distributed version of the MST problem, we assume that each node has a unique ID, and that the input at a node consists of the node's ID, the IDs of its neighbors, and the weights of its incident edges. We require that the computation will result with each node knowing exactly which of its incident edges in the MST. Formally, the set of output variables at a node comprises a boolean variable for each incident edge, and the requirement is that when the execution terminates, the variable corresponding to an edge e has value 1 if $e \in \text{mst}(G)$ and 0 otherwise.

2.3 The mailing problem

To prove a lower bound on the MST problem, we use the *mailing problem*, defined as follows [19]. We are given a graph $G = (V, E)$ with two distinct nodes $s, r \in V$, referred to as the *sender* and the *receiver*, respectively. Intuitively, the mailing problem requires copying a bit string from s to r . Formally, the sender s has b input variables X_1^s, \dots, X_b^s for some given integer $b \geq 1$, and the receiver r has a set of b output variables X_1^r, \dots, X_b^r for the same b . An instance of the problem is an assignment of bits to the input variables at s , and the requirement is that when the protocol terminates, $X_i^r = X_i^s$ for all $1 \leq i \leq b$. Given a graph G with sender s and receiver r , and given a bit string $X = x_1 \dots x_b$, let $\text{Mail}(G, s, r, X)$ denote the mailing problem with initial assignment $X_i^s = x_i$ for $1 \leq i \leq b$. We refer to the

class of problem instances containing all strings X of length b as $\text{Mail}(G, s, r, b)$. The notation $T_A(G, s, r, X)$ denotes the time required by an algorithm A to solve the problem instance $\text{Mail}(G, s, r, X)$, and $T_A(G, s, r, b)$ denotes the worst-case time complexity of an algorithm A over the class of problem instances $\text{Mail}(G, s, r, b)$.

3 Lower bounds on distributed MST construction

In this section we prove lower bounds on distributed MST construction. The structure proof is as follows. First, we prove a lower bound on the time complexity of the mailing problem in certain topologies; then we argue that for each topology there is a collection of edge weight assignments such that under these assignments, the mailing problem can be reduced to MST. This reduction shows that any lower bound on the mailing problem applies to the MST problem, yielding the promised result.

3.1 Diameter 4 graphs

Fix a parameter $m \geq 2$. We start by defining an unweighted graph F_m , using two basic building blocks (see Fig. 1). First, the graph F_m contains m^2 copies of an m -node path, denoted P_1, \dots, P_{m^2} . Formally, the k th copy P_k is defined by

$$V(P_k) = \{v_{1,k}, v_{2,k}, \dots, v_{m,k}\}$$

$$E(P_k) = \{(v_{i,k}, v_{i+1,k}) : i = 1, 2, \dots, m - 1\}.$$

The second building block is a star graph. There are two kinds of stars in F_m . The first kind is called a *level star*: for each $i = 1, 2, \dots, m$, add a node u_i called *level i center*. For each path P_k , connect $v_{i,k}$ to u_i . Formally, define

$$E(C_i) = \{(u_i, v_{i,k}) : k = 1, 2, \dots, m^2\}.$$

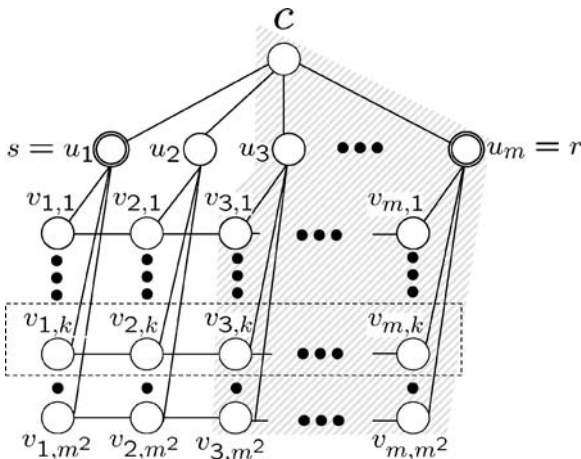


Fig. 1 The graph F_m used in the lower bound for diameter 4. The dashed box highlights a typical path P_k , and the nodes in the shaded area are in the tail set Z_2 (see text)

The second kind of star is the *center star*: add a single node c and connect it to all the level center nodes. Formally,

$$E(C) = \{(c, u_i) : i = 1, 2, \dots, m\}.$$

In summary, the graph F_m is defined by

$$V(F_m) = \{v_{i,k} : 1 \leq i \leq m, 1 \leq k \leq m^2\} \cup \{u_i : 1 \leq i \leq m\} \cup \{c\}$$

$$E(F_m) = \left(\bigcup_{k=1}^{m^2} E(P_k)\right) \cup \left(\bigcup_{i=1}^m E(C_i)\right) \cup E(C).$$

We denote the node u_1 by s and the node u_m by r .

We will now show that for any algorithm A , $T_A(F_m, s, r, m^2) = \Omega(\sqrt[3]{n}/\sqrt{B})$, i.e., that sending an m^2 -bit string from s to r on F_m cannot be done in $o(\sqrt[3]{n}/\sqrt{B})$ communication rounds. The main idea is as follows. Consider the set of possible states a node v may be in at a given time t of the execution of a mailing algorithm on X . The state of a node at a time t during the execution of a deterministic protocol is a function of nothing but its initial state, local input, and the messages received from its neighbors up to time t . Therefore, when execution starts at time 0, each non-sender node is in one specific initial local state, since non-sender nodes have no local input. The sender s has 2^{m^2} possible initial states, determined by the value of the input string X . Upon termination, the string X resides in the output variables of the receiver r , hence r may be in one of at least 2^{m^2} states. Our argument is based on analyzing the growth rate of the number of possible states in the system, and showing that at least $\Omega(m/\sqrt{B})$ time must elapse until r 's set of possible states is of size 2^{m^2} . The argument does not depend on the representation of the states or messages: it hinges on the *number* of states, which is limited by the number of different messages that can be delivered in a time unit.

For the formal analysis, we first state the following obvious properties (the bound on diameter follows from the fact that the distance between any node and c is at most 2).

Lemma 3.1 *For all $m \geq 2$, the number of nodes in F_m is $m^3 + m + 1$ and its diameter is at most 4.*

For $0 \leq i \leq m$, define the *tail set* of the graph F_m , denoted Z_i , as the subgraph induced by the nodes $\{v_{\ell,j} : i + 1 \leq \ell \leq m, 1 \leq j \leq m^2\} \cup \{u_\ell : i + 1 \leq \ell \leq m\} \cup \{c\}$. The shaded area in Fig. 1 shows the tail set Z_2 .

Now fix a deterministic mailing algorithm A . Let φ_X denote the execution of A on an m^2 -bit input X in the graph F_m with sender s and receiver r . Let $C_t(X)$ denote the vector of states of nodes in the tail set Z_t at time t in execution φ_X . $C_t(X)$ is called the *configuration* of Z_t on X at time t . Define $\mathcal{C}_t = \{C_t(X) : X \text{ is a legal input}\}$, and let $\rho_t = |\mathcal{C}_t|$. Namely, ρ_t is the number of distinct reachable configurations of Z_t at time t .

Our main lemma is the following.

Lemma 3.2 *For every $t = 1, \dots, m - 1$, we have $\rho_t \leq 2^{Bt(t+1)/2}$.*

Proof First, note that at time 0, the input string X is known only to the sender s . The rest of the nodes are found in some initial state, which is independent of X . Thus in particular $\rho_0 = 1$. Next, we claim that for all $1 \leq t \leq m - 2$,

$$\rho_{t+1} \leq \rho_t \cdot 2^{Bt} \tag{1}$$

Inductive application of Eq. (1), combined with the fact that $\rho_0 = 1$, proves the lemma.

To prove Eq. (1), let $t \geq 0$, and fix a configuration $\hat{C} \in \mathcal{C}_t$. The tail set Z_{t+1} is connected to the rest of the graph by the m^2 edges $(v_{t+1,k}, v_{t+2,k})$, $1 \leq k \leq m^2$ and by the t edges (u_ℓ, c) , $1 \leq \ell \leq t$. Let us count the number of different configurations in \mathcal{C}_{t+1} that may result of the configuration \hat{C} . Observe that since the state of each node $v_{t+1,k}$ is determined by \hat{C} , the message sent by $v_{t+1,k}$ to $v_{t+2,k}$ is also determined by \hat{C} , and thus it does not introduce any divergence in the execution. The same applies to all the edges internal to Z_{t+1} and to the edge (u_{t+1}, c) . As for the t edges (u_ℓ, c) , $1 \leq \ell \leq t$, the nodes u_ℓ are not in the set Z_t , hence they may be in any one of many possible states, and the value passed into the set Z_{t+1} over the edges connecting them to c is not determined by \hat{C} . However, since by the B -model assumption each message contains B bits, the number of distinct messages c receives at round t is at most 2^{Bt} . Equation (1), and hence the lemma, follow. \square

Lemma 3.3 *For any mailing algorithm A in the B model and any $m \geq 2$, $T_A(F_m, s, r, m^2) = \Omega(m/\sqrt{B})$.*

Proof Let T denote the time it takes algorithm A to complete the mailing. As argued earlier, there must be at least 2^{m^2} possible different states for the receiver r at time T . Since $r \in Z_t$ for any $0 \leq t \leq m - 1$, we conclude that either $T \geq m$ or else $\rho_T \geq 2^{m^2}$. If $T \geq m$ we are done; otherwise, from Lemma 3.2 we get that $2^{BT(T+1)/2} \geq 2^{m^2}$, i.e., $T = \Omega(m/\sqrt{B})$ and the lemma follows. \square

As $m = \Omega(\sqrt[3]{n})$ in F_m , we get:

Lemma 3.4 *For any mailing algorithm A in the B model and $m \geq 2$, $T_A(F_m, s, r, m^2) = \Omega(n^{1/3}/\sqrt{B})$.*

The lower bound for the MST construction problem can now be established by a reduction from MST construction to the mailing problem. Essentially, the reduction shows that for the graph F_m , it is possible to construct a corresponding family of weighted graphs \mathcal{F}_m , such that if the mailing problem requires $\Omega(t)$ time on F_m in the B model, then any distributed MST construction algorithm requires $\Omega(t)$ time on some graphs of \mathcal{F}_m in the B model. More specifically, the weight assignment is as follows. All graphs in \mathcal{F}_m have the following edge weights.

$$\omega(e) = \begin{cases} 0, & \text{if } e \in E(P_k) \\ 0, & \text{if } e \in E(C) \\ 1, & \text{if } e \in E(C_m) \\ 10, & \text{if } e \in E(C_i) \text{ for } 2 \leq i \leq m - 1 \end{cases}$$

The only difference between different graphs in \mathcal{F}_m is the weights of edges in $E(C_1)$, i.e., edges of the type $(u_1, v_{1,i})$, for $1 \leq i \leq m^2$. These edges take as weights either 0 or 2, with one graph in \mathcal{F}_m for each of the 2^{m^2} combinations.

Consider now the MSTs of graphs in \mathcal{F}_m . The MST of any graph in \mathcal{F}_m contains $(\cup_{k=1}^{m^2} E(P_k)) \cup E(C)$ (this can be seen, for example, by Kruskal's algorithm that picks the lightest edges so long as they do not close a cycle). However, for each $1 \leq k \leq m^2$, exactly one of the edges $(u_1, v_{1,k})$ and $(u_m, v_{m,k})$ is in the MST of a graph in \mathcal{F}_m . Moreover, $(u_1, v_{1,k})$ is in the MST if and only if $\omega(u_1, v_{1,k}) = 0$, and $X_k^1 = 1$ if and only if $(u_m, v_{m,k})$ is in the MST.

It therefore follows that any MST algorithm that works for all graphs in \mathcal{F}_m solves the mailing problem where $X_k^s = 1$ if and only if $\omega(u_1, v_{1,k}) = 2$. We conclude with the following result.

Theorem 3.5 *For every $m \geq 2$, there exists a family \mathcal{F}_m of graphs with diameter at most 4 and $n = O(m^3)$ such that every distributed MST construction algorithm requires $\Omega(n^{1/3}/\sqrt{B})$ time on some graph of \mathcal{F}_m in the B model.*

3.2 Diameter 3 graphs

In this section we modify the construction of Sect. 3.1 to prove a weaker lower bound for graphs with diameter 3. Intuitively, the idea is to replace the center star C in F_m with a clique, and optimize the parameters for the new topology (see Fig. 2).

Formally, we proceed as follows. Fix $m \geq 2$. We define a graph H_m as follows. First, take m^3 copies of an m -node path P_k defined as in Sect. 3.1: the nodes in P_k are $v_{i,k}$ for $1 \leq i \leq m$ and $1 \leq k \leq m^3$ and

$$V(P_k) = \{v_{i,k} : 1 \leq i \leq m, 1 \leq k \leq m^3\},$$

$$E(P_k) = \{(v_{i,k}, v_{i+1,k}) : 1 \leq i < m, 1 \leq k \leq m^3\}.$$

Next, add nodes u_i for $1 \leq i \leq m$, and connect u_i with each node $v_{i,k}$ for $1 \leq k \leq m^3$. Formally, let

$$E(C_i) = \{(u_i, v_{i,k}) : k = 1, 2, \dots, m^3\}.$$

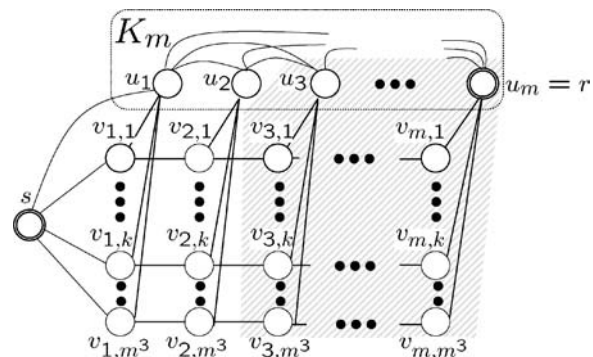


Fig. 2 The graph H_m used in the lower bound for diameter 3. The dashed box contains a complete graph on nodes u_1, \dots, u_m , and the shaded area is the tail set Z_2 (see text)

Let $C = \{u_1, u_2, \dots, u_m\}$. Now make C fully connected, by adding the edges

$$E(C) = \{(u_i, u_j) : 1 \leq i < j \leq m\}.$$

Finally, add a sender node s , and connect it to the beginning of each path, and to the first level center node u_1 . Formally, add the edges

$$E(S) = \{(s, v_{1,k}) : k = 1, 2, \dots, m^3\} \cup (s, u_1).$$

Now, the graph H_m is defined formally by

$$\begin{aligned} V(H_m) &= \{v_{i,k} : 1 \leq i \leq m, 1 \leq k \leq m^3\} \cup \{u_i : 1 \leq i \leq m\} \cup \{s\} \\ E(H_m) &= \left(\bigcup_{i=k}^{m^3} E(P_k)\right) \cup \left(\bigcup_{i=1}^m E(C_i)\right) \cup E(C) \cup E(S). \end{aligned}$$

The receiver node r is the node u_m .

Directly from the construction, we have the following properties. (The bound on the diameter follows from the fact that every node is at distance at most one from a node in C , and that the distance between any two nodes in C is at most 1.)

Lemma 3.6 *For any $m \geq 2$, the number of nodes in H_m is $m^4 + m + 1$ and its diameter is at most 3.*

We now show that solving the mailing problem on H_m with an m^3 -bit string X requires $\Omega(\sqrt[4]{n}/\sqrt{B})$ time units. The proof follows the lines of the Sect. 3.1, with the following changes. For $0 \leq i \leq m$, we now define the tail set of the graph H_m , denoted Z_i , as the subgraph induced by the nodes

$$\{v_{\ell,k} : i+1 \leq \ell \leq m, 1 \leq k \leq m^3\} \cup \{u_\ell : i+1 \leq \ell \leq m\}.$$

Fix a mailing algorithm A for H_m . As before, \mathcal{C}_t is the set of possible states at the nodes of Z_t at time t , and $\rho_t = |\mathcal{C}_t|$. We now prove the main lemma, which bounds ρ_t from above.

Lemma 3.7 *For every $1 \leq t \leq m-1$, $\rho_t \leq 2^{Bt^2m}$.*

Proof Similar to Lemma 3.2, this lemma is proved by showing that at time t , each configuration in \mathcal{C}_{t-1} branches off into at most $2^{Bt(m-t)}$ different configurations of \mathcal{C}_t . Below, we use the convention that the sender s is also denoted as u_0 , and as $v_{0,k}$ for any $1 \leq k \leq m^3$.¹

To prove the lemma, we first claim that for all $1 \leq t \leq m-1$,

$$\rho_t \leq \rho_{t-1} 2^{Bt(m-t)}. \quad (2)$$

To prove Eq. (2), consider \mathcal{C}_t for some $t > 0$. The tail set Z_t is connected to the rest of the graph by the m^3 edges $(v_{t,k}, v_{t+1,k})$ for $1 \leq k \leq m^3$, and by the $t(m-t)$ edges $(u_\ell, u_{\ell'})$, for $1 \leq \ell \leq t$ and $t+1 \leq \ell' \leq m$.

¹ The shift of indexes w.r.t. Lemma 3.2 is due to the fact that in the graph F_m , the sender is u_1 while in H_m , the sender is u_0 ; the reason for that will become apparent when we discuss the reduction of the mailing problem to MST.

We now count the number of different configurations in \mathcal{C}_t that may result of some fixed configuration $\hat{C} \in \mathcal{C}_{t-1}$. Starting from \hat{C} , each node $v_{t,k}$, as well as the node u_t , is restricted to a single state, hence it does not introduce any divergence in the execution. The same applies to messages sent over all the edges internal to Z_{t-1} and the edges $(u_t, u_{\ell'})$ for $t+1 \leq \ell' \leq m$. As for the $t(m-t)$ edges $(u_\ell, u_{\ell'})$, where $1 \leq \ell \leq t$ and $t+1 \leq \ell' \leq m$, the nodes u_ℓ are not in the set Z_t , hence the values they pass into the set Z_t (to the nodes $u_{\ell'}, t+1 \leq \ell' \leq m$) are not determined by the configuration \hat{C} . However, in the B -bounded-message model, this can cause the configuration \hat{C} to branch off into at most $2^{Bt(m-t)}$ possible configurations in \mathcal{C}_t . Equation (2) follows.

To complete the proof of the lemma, we first note that $\rho_0 = 1$, since $s \notin Z_0$ and the state of all other nodes is uniquely defined by the algorithm at time 0. Applying Eq. (2) inductively, we get that

$$\begin{aligned} \rho_t &\leq \prod_{\tau=1}^t 2^{B\tau(m-\tau)} = 2^{B \sum_{\tau=1}^t \tau(m-\tau)} \leq 2^{Bm \sum_{\tau=1}^t \tau} \\ &\leq 2^{Bt^2m}, \end{aligned}$$

and the lemma follows. \square

Lemma 3.8 *For any mailing algorithm A in the B model and $m \geq 2$, $T_A(H_m, s, r, m^3) = \Omega(n^{1/4}/\sqrt{B})$.*

Proof Let T denote the time it takes algorithm A to complete the mailing. We prove that $T = \Omega(m/\sqrt{B})$, which implies the lemma, since $m = \Omega(\sqrt[4]{n})$. If $T \geq m$ we are done. Otherwise $T < m$, and then by Lemma 3.7, $\rho_T \leq 2^{BT^2m}$. On the other hand, since at time T there are at least 2^{m^3} possible different states at the receiver r , and since $r \in Z_T$ for $T < m$, we have that $\rho_T \geq 2^{m^3}$. It therefore follows that $2^{BT^2m} \geq 2^{m^3}$, i.e., $T^2 \geq m^2/B$ and the lemma follows. \square

The lower bound on the time complexity of algorithms for the MST problem follows by a direct reduction to the mailing problem, similarly to Sect. 3.1.

Theorem 3.9 *For every $m \geq 2$, there exists a family \mathcal{H}_m of graphs with diameter 3 and $n = O(m^4)$ nodes, such that every distributed MST construction algorithm requires $\Omega(n^{1/4}/\sqrt{B})$ time on some graph of \mathcal{H}_m in the B model.*

Proof Define a family of 2^{m^3} weighted graphs \mathcal{H}_m , where the unweighted underlying graph for each $G \in \mathcal{H}_m$ is just H_m as defined above. Define the weights of edges by

$$\omega(e) = \begin{cases} 0, & \text{if } e \in E(P_k) \\ 0, & \text{if } e = (u_1, u_i) \text{ for } 2 \leq i \leq m \\ 0, & \text{if } e = (s, u_1) \\ 1, & \text{if } e \in E(C_m) \\ 10, & \text{if } e \in E(C_i) \text{ for } 2 \leq i \leq m-1 \end{cases}$$

In addition, if $e = (s, v_{1,k})$ for some $1 \leq k \leq m^3$, then $\omega(e) \in \{0, 2\}$, such that each possible 2^{m^3} combination appears exactly once in \mathcal{H}_m . By the weights, we have that all edges of all P_k are members in the MST of any graph in \mathcal{H}_m , as well as the edges (u_1, u_i) for $1 < i \leq m$ and the edge (s, u_1) . However, an edge $(s, v_{1,k})$ is in the MST if and only if its weight is 0 (otherwise, $(u_m, v_{m^2,k})$ will be in the MST). It follows that an algorithm computing an MST on all graphs of \mathcal{H}_m solves the mailing problem on H_m , and the theorem follows from Lemma 3.7. \square

3.3 A lower bound for randomized algorithms

A randomized algorithm is a deterministic algorithm that has access to an infinite tape of random bits (in addition to the standard inputs). A randomized Las-Vegas algorithm is said to solve a given problem if for any instance of the problem, the algorithm produces the correct output in finite expected time (where the expectation is taken over the space of random tapes). In this section we show, using standard techniques, that all Las-Vegas algorithms for MST admit the same asymptotic bounds we proved for deterministic algorithms.

Let us concentrate on the diameter 4 graph F_m of Sect. 3.1; the proof for diameter 3 graphs is similar. We start by fixing a deterministic distributed mailing algorithm A and establishing a slightly stronger claim than Lemma 3.3.

Lemma 3.10 *For every $m \geq 2$ and for at least half of the possible m^2 -bit input strings X of the mailing problem, $T_A(F_m, s, r, X) \geq \alpha \cdot m/\sqrt{B}$ for some constant $\alpha > 0$.*

Proof Similar to the proof of Lemma 3.3 except that, defining τ_{mid} to be the minimum integer such that $T_A(F_m, s, r, X) \leq \tau_{\text{mid}}$ for at least half the possible input strings, we get

$$2^{B\tau_{\text{mid}}(\tau_{\text{mid}}+1)/2} \geq \rho_{\tau_{\text{mid}}} \geq 2^{m^2-1},$$

and the claim follows. \square

A lower bound on the expected time complexity of any randomized (Las-Vegas) distributed algorithm for the mailing problem over the graph F_m now follows by Yao's method ([21], cf. [16]). By this method, in order to bound this complexity (over any distribution D_1 on random input strings X), it suffices to find some "difficult" distribution D_2 on the inputs, and prove that on D_2 , every deterministic mailing algorithm requires $\Omega(n^{1/3}/\sqrt{B})$ expected time. Thus applying Yao's lemma we get the following.

Lemma 3.11 *For every $m \geq 2$, and for every randomized Las-Vegas algorithm A , there exists some input string X of the mailing problem such that Algorithm A requires $\alpha \cdot m/2\sqrt{B}$ expected time for solving the instance $\text{Mail}(F_m, s, r, X)$ in the B model.*

The reduction from MST to mailing applies without change, hence we get

Theorem 3.12 *There exists a family of n -node graphs of diameter 4 on which any randomized Las-Vegas distributed algorithm for the MST problem in the B model requires $\Omega(n^{1/3}/\sqrt{B})$ expected time.*

In a similar manner, the lower bound of [19] for graphs of arbitrary diameter can also be extended to randomized algorithms by applying Yao's lemma to the graphs of [19].

4 PRAM simulation by diameter 2 graphs

In this section, we complement the lower bounds of Sect. 3 with an upper bound, that demonstrates the existence of an exponential gap in the time complexity of MST in the B model: while graphs with diameter 3 or more may require a polynomial number of steps, this section shows that graphs with diameter 2 can always be solved in a logarithmic number of steps. This fact is a consequence of a simple but general result we prove here: if the graph has diameter at most 2, then any PRAM program can be simulated by the B model with the same asymptotic time complexity. Our simulation technique is similar to those of [1, 20], and our result can be thought of as a generalization of the result of [1] from graphs of diameter 1 to graphs of diameter 2 (more about that in Sect. 4.1 below.) The existence of $O(\log n)$ time PRAM algorithms for MST computation now implies the gap.

Let us briefly review the classical PRAM model [7]. There are n processors with shared global memory. Execution proceeds in synchronized global steps. For simplicity, let us assume that each step consists of three substeps: in the first substep, all processors write; in the second step, all processors read, and in the third substep, each processor may perform any arithmetic or logical operation as in the sequential RAM model. There are several variants of the PRAM model, depending on the concurrency allowed in memory accesses. In this section we consider the strongest variant, called CRCW PRAM, that allows concurrent reads and writes. In this model, one must specify what is the outcome of simultaneous writes to the same register. This specification is usually called *contention resolution rule*. In this paper, all we assume is that the contention resolution rule is *associative*, i.e., the result of multiple writes does not depend on the order in which the resolution rule is applied to the writes. (Some examples of associative rule include *min*, where the smallest of the values is written, or *equal*, which writes a value only if all proposed values are equal.)

To simulate a PRAM program on the B model, we shall assume that B is large enough so that a message can contain any value stored by any register used in the PRAM model. We also assume that B is large enough so that we can encode in a message memory location indexes and program counter values.

We now describe the simulation. Suppose we are given a PRAM algorithm. To simulate this program in the B model, each PRAM processor will be simulated by a B model processor. To simulate the PRAM shared memory, each processor p records, for each shared register a of the PRAM program, a value $v_p(a)$ and a timestamp $t_p(a)$. Intuitively, $v_p(a)$ is the latest value of a processor p knows about, and it was recorded at time $t_p(a)$. This interpretation is fulfilled as follows. Consider first a step in which processor p writes a value v to address a . This is simulated in the B model by letting p send $\langle a, v \rangle$ to all its neighbors. Each processor applies the contention resolution rule for each register, and records the resulting values in its local memory, along with the time in which they were received.

Next, consider a *read* step: suppose that in the PRAM program, processor p reads a value from register a . In the B model, p sends a message to all its neighbors the contents a . Each neighbor q sends $\langle v_q(a), t_q(a) \rangle$ to p . Processor p then finds what is the latest value written to a , and if there are several such values it applies the resolution rule to compute the outcome of the simulated read operation.

Thus we have the following.

Theorem 4.1 *Any program that can be run in T steps on an n -processor CRCW PRAM with an associative resolution rule, with m registers of size w can be executed in the B model in $O(T)$ steps on any topology with diameter at most 2 and $B = \Omega(w + \log m + \log T)$.*

Proof For the simulation above, all we need to observe is that the bandwidth requirement is not violated, and that result of each *read* is correct. For the bandwidth, note that since a processor in the PRAM model writes to at most one location in each step, there is no problem in the simulation. For a write step, note that each processor sends just one response to each other processor, and each response is of length $O(w + \log T)$. The correctness of the simulation follows from the assumption that the diameter of the network is 2 and that the conflict resolution rule is associative: Consider a read by processor p . This read needs to return a value previously written by some processor q . There must be some processor r which is a neighbor both of p and q , and thus r holds the result of the write of q , and when r gets the read request of p , it will send the correct value to p . \square

To establish our result for MST, we would like to simulate the CRCW PRAM algorithm of [4] (which uses the *min* contention resolution rule). We note that Theorem 4.1 cannot be readily used to this end, as the algorithm of [4] employs $O(n + m)$ processors on n -vertex, m -edge graphs. Fortunately, the algorithm of [4], as well as some other natural PRAM algorithms for graph problems, has some additional convenient properties that allow us to simulate them in our model using only one processor per vertex, i.e., giving up the processors associated with edges in the PRAM algorithm. In particular, each register r used by the PRAM algorithm of [4] can be mapped to some vertex v_r of the network, so that r is written to (in each round) only by the vertex v_r and its neighbors in the network. Thus any PRAM

write step of this algorithm can be simulated by the neighboring vertices sending their intended write value to v_r in a single communication round, followed by v_r resolving the concurrent write according to the *min* rule. We can therefore conclude the following.

Lemma 4.2 *MST can be solved in $O(\log n)$ rounds in the B model if the diameter of the network is at most 2 and $B = \Omega(\log n)$.*

4.1 Related results

The PRAM simulation presented above uses several well-known ideas, such as using read/write sets with non-empty intersections and globally consistent timestamps. These ideas can be traced back to early work on concurrency control, replicated databases and quorum-based distributed computing (cf. [10] and others). More specifically, our simulation method is technically similar to those of [1, 20].

In [20], a PRAM computation is simulated by the Module Parallel Computer (MPC). This result cannot be translated directly into our model, however, as the MPC model is different from our B model in some significant aspects. For instance, in the MPC model, memory is decoupled from processors, in the sense that every processor has access to every memory cell, without going through any other intermediate processor. This is not the case in our model, in which each memory cell is resident at some processor, and can be accessed by other processors only by communicating with the host processor.

In [1], an asynchronous shared memory computation is simulated in a message-passing setting, in a model similar to ours. However, the network is assumed to be a complete graph. Hence our result can be thought of as a generalization of the result of [1] from complete graphs to graphs of diameter 2. (Another technical difference is that the simulation techniques of [1] use majorities for ensuring intersection, while our method relies on intersecting adjacency lists in a graph of diameter 2.)

5 Conclusion

In this paper we have proved that the complexity of distributed MST computation changes exponentially when the diameter changes from 2 to 3. The significance of diameter 2 may be explained by the fact that the only bottlenecks in graphs of diameter 2 may be articulation *nodes*: nodes, however, may relay linear number of bits in a single time step. This property does not hold for graphs of diameter 3, where the bottleneck may be an edge, that can relay only B bits in a single time step.

There have been some interesting developments since our preliminary report was published. First, it was shown that for graphs of diameter 1 (cliques), MST can be computed in $O(\log \log n)$ communication rounds using messages of size $O(\log n)$ [14]. Second, Elkin [5] has recently

improved the lower bounds of [19] and the current paper by a factor of $\sqrt{B \log n}$. However, none of these bounds is known to be tight.

Acknowledgements We are grateful to the anonymous referees for helpful comments and, in particular, for pointing out some inaccuracies in an earlier draft of this paper.

References

1. Attiya, H., Bar-Noy, A., Dolev, D.: Sharing memory robustly in message-passing systems. *J. ACM* **42**(1), 124–142 (1995)
2. Attiya, H., Welch, J.: *Distributed Algorithms*. McGraw-Hill Publishing Company, UK (1998)
3. Awerbuch, B.: Optimal distributed algorithms for minimum weight spanning tree, counting, leader election and related problems. In: *Proceedings of 19th ACM Symposium Theory of Computing*, pp. 230–240 (May 1987)
4. Awerbuch, B., Shiloach, Y.: New connectivity and MSF algorithms for the shuffle-exchange network and PRAM. *IEEE Trans. Comput.* **C-36**, 1258–1263 (1987)
5. Elkin, M.: Unconditional lower bounds on the time-approximation tradeoffs for the distributed minimum spanning tree problem. In: *Proceedings of 36th ACM Symposium on Theory of Computing* (May 2004)
6. Faloutsos, M., Molle, M.: Optimal distributed algorithm for minimum spanning trees revisited. In: *Proceedings of 14th ACM Symposium on Principles of Distributed Computing*, pp. 231–237 (1995)
7. Fortune, S., Wyllie, J.: Parallelism in random access machines. In: *Proceedings of 19th ACM Symposium on Theory of Computing*, pp. 230–240 (May 1987)
8. Gallager, R.G., Humblet, P.A., Spira, P.M.: A distributed algorithm for minimum-weight spanning trees. *ACM Trans. Prog. Lang. Syst.* **5**(1), 66–77 (1983)
9. Garay, J., Kutten, S., Peleg, D.: A sub-linear time distributed algorithm for minimum-weight spanning trees. *SIAM J. Comput.* **27**, 302–316 (1998)
10. Gifford, D.K.: Weighted Voting for Replicated Data. In: *Proceedings of 7th Symposium on Operating System Principles*, pp. 150–162 (1979)
11. Kutten, S., Peleg, D.: Fast distributed construction of small k -dominating sets and applications. *J. Algorithm.* **28**, 40–66 (1998)
12. Lawler, E.: *Combinatorial Optimization: Networks and Matroids*. Holt, Rinehart and Winston, New York (1976)
13. Linial, N.: Locality in distributed graph algorithms. *SIAM J. Comput.* **21**, 193–201 (1992)
14. Lotker, Z., Patt-Shamir, B., Pavlov, E., Peleg, D.: MST construction in $O(\log \log n)$ communication rounds. In: *Proceedings of 2003 ACM Symposium on Parallelism in Algorithms and Architecture*, pp. 94–100 (2003)
15. Lynch, N.: *Distributed Algorithms*. Morgan Kaufmann, San Mateo, CA (1995)
16. Motwani, R., Raghavan, P.: *Randomized Algorithms*. Cambridge University Press (1995)
17. Peleg, D.: Time-optimal leader election in general networks. *J. Parallel Distrib. Comput.* **8**, 96–99 (1990)
18. Peleg, D.: *Distributed Computing: A Locality-Sensitive Approach*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA (2000)
19. Peleg, D., Rubinovich, V.: Near-tight lower bound on the time complexity of distributed MST construction. *SIAM J. Comput.* **30**, 1427–1442 (2000)
20. Upfal, E., Wigderson, A.: How to share memory in a distributed system. *J. ACM* **34**(1), 116–127 (1987)
21. Yao, A.C.: Probabilistic computations: Toward a unified measure of complexity. In: *Proceedings of 18th ACM Symposium on Theory of Computing*, pp. 222–227 (1977)