Gossiping in One-Dimensional Synchronous Ad Hoc Wireless Radio Networks

Avery Miller
University of Toronto
Toronto, Canada
a4miller@cs.toronto.edu

ABSTRACT

Consider a set of n processors traveling with bounded speed along continuous trajectories on a line and suppose that each processor must share a piece of information with all other processors in the set. This is known as the *gossiping task*. Each processor has a radio transmitter with transmission radius R and interference radius $R' \geq R$. We present a deterministic algorithm for the gossiping task, under certain network density assumptions, that is provably correct and terminates within $O(n)$ time slots.

Categories and Subject Descriptors

C.2.1 [**Computer-Communication Networks**]: Network Architecture and Design—*Wireless communications*; F.2.2 [**Analysis of Algorithms and Problem Complexity**]: Nonnumerical Algorithms and Problems

General Terms

Algorithms

Keywords

gossiping, mobile ad hoc radio networks

1. INTRODUCTION

In the gossiping task, each processor $p_i \in \{p_1, \ldots, p_n\}$ has a message m_i that it wishes to share with all processors in the network. Gossiping is useful for performing any aggregate computation that requires information from each processor. This information can be internal, such as a processor's planned trajectory, or external, such as environmental conditions. We consider processors that move along arbitrary, continuous trajectories on a one-dimensional line and travel distance at most σ in one time slot. The processors transmit messages wirelessly on a shared communication channel: a processor p receives a message from a transmitting processor q during time slot t if and only if q is within distance R from p *for the entirety* of slot t and no other transmitting processor q' is within distance R' from p *at any point* during slot t. If p does not receive a message during slot t due to interfering transmissions from q and q', then we say a *collision* has occurred at p during slot t; however, we assume that no processor is aware that such a collision has occurred. The processors operate in a distributed manner, that is, they each execute a local algorithm and only receive communication from other processors that are nearby, rather than from a central coordinator. For some applications, such as vehicular ad hoc networks (VANETs), a small probability of error or a rare communication delay means that human life is at risk, so we restrict our attention to deterministic solutions to the gossiping task.

To our knowledge, ours is the first deterministic algorithm that solves the gossiping task in such a mobile ad hoc network (MANET). An important feature of our solution is that all processors terminate their local algorithm in the same time slot. Therefore, our solution is particularly useful as a subroutine in more complex algorithms, since the processors can all begin executing a subsequent algorithm at the same time. Moreover, this means that our gossiping algorithm can be executed repeatedly without any delay between executions. This can be useful for communicating periodic updates to the entire network, for example, in a sensor network where each processor regularly takes new measurements. Similarly, processors can send out updates to the network when they make a change in their planned trajectory, a fact that we use to answer an open question raised in Ellen, Welch, and Subramanian [8].

Our algorithm is based on a collision-free transmission schedule created by Ellen, Welch, and Subramanian (EWS) [8] for maintaining neighbourhood information. In Section 4.1, we describe this schedule and then use it to form our gossiping algorithm, presented in Section 4.2. In Section 4.3, we give an overview of the analysis of our algorithm. In particular, we use a new "window" technique that we feel could be useful for providing clear and rigorous analyses of information dissemination algorithms in mobile networks where processors move along continuous trajectories. In Section 4.4, we show that our gossiping algorithm terminates within $O(n)$ slots.

TADDS '12, December 17 2012, Roma, Italy
Copyright 2012 ACM 978-1-4503-1849-5/12/12 ...$15.00.

2. RELATED WORK

The deterministic gossiping task has been studied extensively for static networks [10]. More relevant to our work are the results pertaining to geometric radio networks, that is, networks where processors are positioned in physical space and each has a transmission radius of R [7, 9, 13]. Less is known about mobile ad hoc networks (MANETs), where processors move within a physical environment and all processors have the same transmission radius R. Mohsin [14] provides a survey of broadcasting algorithms in MANETs and also discusses various mobility models and their effect on broadcasting algorithms.

Many broadcasting and gossiping algorithms require that the topology of the network must remain static for significant periods of time, at regular intervals [3, 5, 12, 16, 17, 18]. In other algorithms [15], processors are required to be located on the points of a one-dimensional grid at the beginning of each round and can only move to one adjacent grid-point per round. Although arbitrarily small grid sizes can model continuous trajectories, it comes at the cost of requiring nodes to move extremely slowly.

Some algorithms for information dissemination tasks are designed for models without collisions, assuming the MAC layer deals with channel contention. Unfortunately, existing MAC layer implementations do not guarantee efficient message delivery in highly dynamic networks. Moreover, an algorithm that seems efficient in the absence of collisions might result in costly situations for the MAC layer. Essentially, the high complexity of the algorithm is pushed to a different layer and ignored.

In this paper, we consider a model that includes transmission collisions and where processors move through the environment along arbitrary, continuous trajectories with bounded speed. In this model, Ellen et al. [8] present an algorithm for maintaining neighbourhood information in a one-dimensional network. This result relies on the specification of a collision-free schedule, which we use in this paper. Their schedule is adapted for the plane by Viqar and Welch [19] and for road networks by Chung, Viqar, and Welch [4]. Anta and Milani [2] provide solutions to the geocasting task, in which a designated source needs to send a message to all nodes within a specified geographic region. Anta et al. [1] compare different classes of algorithms (oblivious, quasi-oblivious, and adaptive) for information dissemination tasks, such as geocasting and broadcasting. They provide separations between these classes in terms of the number of slots used by information dissemination algorithms, and, they provide necessary restrictions on how often and how long an informed processor must be within the communication range of an uninformed processor.

3. SET-UP AND ASSUMPTIONS

We assume that each processor has a unique identifier (ID). Also, we assume that the network is sufficiently dense, that is, there is a constant $L \leq R$ such that no two consecutive processors on the line are farther than L units apart. Given initial input constants K and m, the algorithm divides the environment into *segments* of length K, partitions the set of segments into m *segment classes*, and partitions the set of time slots into *phases* of length $m-1$. Specifically, segment S_i is $[iK, (i+1)K)$, segment class $\mathbb{S}_\ell = \{S_i \mid i \equiv \ell \pmod{m}\}$, and phase π_j consists of the time slots $jm, jm+1, \ldots, (j+1)m-1$. Each processor knows its location in the environment, so, at all times, processors know in which segment they are located. The values of K and m are chosen according to a system of constraints given by three inequalities. The assumption that a feasible solution exists places further restrictions on our model, namely, on the relationship between the values of R, R', σ and L (though the authors of [8] show that these constraints are not difficult to satisfy). The three constraints are: (C1) $K > (m-1)\sigma$; (C2) $R+R' \leq (K-2\sigma)(m-1)$; (C3) $L \leq [R-3(m-1)\sigma-3K]/2$. Constraint (C1) implies that a processor can cross at most one segment boundary within a single phase of the transmission schedule (although it may cross that boundary many times within the phase). Constraint (C2) implies that any two processors that are scheduled to transmit during the same slot are guaranteed to be far enough away from each other such that there is no possibility of a transmission collision occurring. Constraint (C3) implies that, between the leftmost and rightmost processors, there is never an interval of length $[R-3(m-1)\sigma-3K]/2$ that contains no processors. This ensures that there are always enough processors nearby that will propagate a transmitted message.

For the purpose of analysis, we define a constant $W = \lceil L/K \rceil$. From the definition of K and constraints (C1)-(C3), we get the following useful guarantees about the density of the processors in the network.

Observation 1. *At all times, each contiguous block of W segments in the network contains at least one processor. Further, $W < (m-3)/4$ (which implies that $m \geq 8$), and $R > (2W+1)K + 3(m-1)\sigma$.*

Finally, we assume that, at the start of the algorithm, each processor knows its own planned trajectory for the first $6n+13$ phases. While it may seem unrealistic that each processor knows a lot about its future trajectory, it seems very difficult to weaken this assumption: to find out about new neighbours so that transmissions can be coordinated, it seems necessary that planned trajectory information is received before processors become neighbours. We also assume that, for each processor q within distance $R+2(m-1)\sigma$ of p at the start of the algorithm, p knows the planned trajectory of q for the first $6n+13$ phases. This information could be learned by first running a two-hop neighbourhood discovery algorithm as an initial step. Although we are not aware of any deterministic neighbourhood discovery algorithms for our model, we could use the algorithm in [6] on top of a reliable MAC layer, or the algorithm in [11] if the processors remained stationary during this initial step.

4. ALGORITHM DESCRIPTION

4.1 The EWS Schedule

The task of maintaining neighbourhood information requires that, at all times, each processor in the network has an up-to-date list of all of its neighbours and their locations. It is assumed that, initially, each processor has this information and the trajectories of its neighbours. The EWS schedule is defined in [8] to solve the neighbourhood maintenance task and is designed to avoid transmission collisions. To do this, it provides a method for choosing a leader for each segment at the beginning of each phase of the transmission schedule, and, specifies when each segment leader is allowed to transmit within each phase. At the beginning of each phase, each processor determines a leader for its segment, namely, the processor in its segment with smallest ID. For each transmission slot, the transmission schedule specifies a single segment class, and all leaders of segments in this segment class are allowed to transmit. Note that, since the number of slots per phase is one less than the number of segment classes, some segment leaders do not get to transmit. In [8], the authors prove that, at the beginning of each subsequent phase, each processor knows the identity and trajectory of each processor that is distance at most $R + 2(m-1)\sigma$. It follows that, at all times, each processor knows the identity and trajectory of all of its neighbours. In Appendix A, we present more properties of the EWS schedule. These additional properties will be useful when proving results about the speed of information propagation in the network.

4.2 Gossiping Algorithm

In our gossiping algorithm, each processor follows the EWS schedule, and, when chosen as a segment leader, transmits all of the information it knows. This includes its own message, the gossiping messages it has received, and trajectory information.

In the EWS algorithm, it is possible that a processor with large ID will never get to transmit a message if there is always a processor with smaller ID located in its segment at the beginning of each phase. This motivates a modified leader selection method for our gossiping algorithm: each processor, at the beginning of each phase, checks if it has received the gossiping message of each processor in its segment and has transmitted its own message. If so, it chooses as leader the processor with smallest ID in the segment; otherwise, it chooses as leader the processor with smallest ID among all processors whose gossiping messages it has not received or transmitted. In Appendix B, we show that, at the start of each phase, all processors in the same segment choose the same leader.

Next, let LM and RM denote the indices of the leftmost and rightmost segments that contain processors at the beginning of phase π_0. Due to the relationship between R and W from Observation 1 (i.e., $R > (2W+1)K$): a processor is in one of the $W+1$ leftmost (rightmost) segments of the network if and only if there is a block of W contiguous empty segments in its neighbourhood to the left (right) of its current location. Further, each processor knows the location of all processors in its neighbourhood, so it can determine if it or one of its neighbours is in one of the $W+1$ leftmost (rightmost) segments of the network, and, hence, can determine the value of LM (RM). We would like all processors in the network to know the values of LM and RM, so, in phase π_0, each leader that is located in the leftmost (rightmost) $2W+1$ segments of the network includes in its transmission a variable whose value is the index of the leftmost (rightmost) segment. The EWS schedule guarantees that at least one such processor transmits in phase π_0.

Finally, we set out to define an appropriate termination condition for the algorithm. When a gossiping message m_i is transmitted for the first time, it may take a while for the message to reach all processors in the network. Thus, two processors that are far away from one another may receive m_i during different phases. This makes it difficult to make sure that all processors terminate at the same time. So, when a processor p_i transmits m_i for the first time, it attaches a timestamp to its message, i.e., the phase number a during which it performed the transmission. Along with these timestamps, the processors use their knowledge of LM and RM to determine whether or not it is possible that there is a gossiping message that it has not received yet. Since a processor can cross at most one segment boundary per phase, we know that, at the beginning of phase π_b, the segment containing the leftmost processor has index at least $LM - b$ and the segment containing the rightmost processor has index at most $RM + b$. Using an upper bound on the amount of time it takes for a message transmitted by a processor at one edge of the network to reach the other (see Corollary 7 of Section 4.3), we know that any message transmitted during phase number a is received by all processors by the end of phase number $a + 2(RM - LM + 4a)/(m-2) + 11$. This helps us determine the minimum number of phases that a processor must wait for new messages to arrive before it can safely terminate. Specifically, if processor p has received LM and RM, then if a' is the latest timestamp it has received and it receives no message with later timestamp by the end of phase number $k = a' + 2(RM - LM + 4a')/(m-2) + 16$, then p terminates at the end of phase π_k. In Appendix C, we show how this termination condition guarantees that gossiping has been completed before any processor terminates and that all processors terminate at the same time.

4.3 Analysis

In this section, we outline the techniques used to analyze the running time and correctness of our gossiping algorithm. Due to space limitations, the proofs of the results in this section have been provided in Appendices D and E.

We focus on finding an upper bound on the time it takes for a transmitted message to reach all processors in the network. It is important for our algorithm that each processor can calculate this bound locally, since it is used to decide when to terminate. To derive the desired upper bound, we created a simple and useful technique for analyzing information dissemination algorithms when processors are traveling along continuous trajectories with bounded speed.

At a high-level, the technique consists of three parts: (1) divide the physical environment into regions large enough so that processors cannot pass through them quickly. Namely, the region size will depend on the known upper bound σ on the distance a processor can move in one time slot; (2) create one or more windows, each with size the same as one region, and define how they jump from region to region. Then, give an upper bound on the number of time slots that elapse before each processor has been located within some window; (3) prove a window "invariant", which is a statement of the form *if processor p is located within a window at time t, then p satisfies property Z by time t'*. Examples of property Z include "has received all messages" or "has terminated its local algorithm". Combining the window invariant with the fact that every processor is eventually located within some window implies that the desired property Z eventually holds for all processors in the network.

Partitioning the Environment.

We partition the environment into equal-sized convex *tiles*. Then, we combine disjoint sets of adjacent tiles into convex *supertiles*. These supertiles also tile the environment. For each supertile T, we associate a *region* consisting of T along with any tiles that share a boundary with a tile in T.

To analyze our gossiping algorithm, we take our tiles to be the set of segments used by the EWS algorithm. Next, for all $j \in \mathbb{Z}$, we define supertile T_j to be the union of m contiguous tiles, namely, $T_j = \bigcup\{S_i \mid i \in \{jm, \ldots, (j+1)m-1\}\}$. Then, for all $j \in \mathbb{Z}$, we define region $X_j = S_{jm-1} \cup T_j \cup S_{(j+1)m}$. Note that each region overlaps its neighbouring regions by two tiles each. Finally, we partition the set of time slots into *superphases*: for $a \geq 0$, define superphase ρ_a to be the union of phases π_{2a} and π_{2a+1}. Note that each superphase ρ_a contains $H = 2m-2$ slots numbered $aH, aH+1, \ldots, (a+1)H-1$. By constraint (C1), a processor can cross at most two different segment boundaries during a single superphase.

Defining the Windows.

We now define the moving windows and give an upper bound on the number of superphases that elapse before each processor has been located in a window. We consider an arbitrary tile S_i located in an arbitrary region X_j. We define one window that moves rightward, and one that moves leftward, both starting in region X_j at the beginning of an arbitrary superphase ρ_a. Each window has size equal to the size of one region. The rightward-moving window jumps one region rightward at the beginning of each successive superphase, while the leftward-moving window jumps one region leftward at the beginning of each successive superphase. More formally, we say that, at the end of superphase $\rho_{a+\phi}$, *q is located in the rightward-moving window* if it is located in region $X_{j+\phi}$. Similarly, at the end of phase $\rho_{a+\phi}$, *q is located in the leftward-moving window* if it is located in region $X_{j-\phi}$.

We now provide an upper bound on the time that elapses before an arbitrary processor q is located within a window. At the end of superphase ρ_a, an arbitrary processor q is either found in S_i, to the right of S_i, or to the left of S_i. In what follows, we consider the processors that are located at or to the right of tile S_i at the end of superphase ρ_a. We prove that no such processor is ever located to the left of the rightward-moving window without first being located within the rightward-moving window at the end of some superphase.

LEMMA 2. *Consider two tiles S_i and $S_{i'}$, with $i \leq i'$, and suppose that processor q is located in $S_{i'}$ at the end of superphase ρ_a. Further, suppose that S_i is contained in supertile T_j. For any fixed $\beta \geq 0$, if q is not located in region $X_{j+\beta}$ at the end of superphases $\rho_a, \ldots, \rho_{a+\beta}$, then q is not located to the left of region $X_{j+\beta}$ at the end of superphase $\rho_{a+\beta}$.*

Next, we prove that the rightward-moving window eventually passes any processor q that is located in or to the right of tile S_i at the end of superphase ρ_a. Along with Lemma 2, this gives the desired upper bound on the time that elapses before processor q is located within the rightward-moving window.

LEMMA 3. *Consider two tiles S_i and $S_{i'}$, with $i \leq i'$, and suppose that processor q is located in $S_{i'}$ at the end of superphase ρ_a. Further, suppose that S_i is located in supertile T_j. For any integer $\beta > (i' - jm + 3)/(m-2)$, q is located to the left of region $X_{j+\beta}$ at the end of superphase $\rho_{a+\beta}$.*

COROLLARY 4. *Consider two tiles S_i and $S_{i'}$, with $i \leq i'$, and suppose that processor q is located in $S_{i'}$ at the end of superphase ρ_a. Further, suppose that S_i is located in supertile T_j. There exists an integer $0 \leq \phi \leq \lceil (i' - jm + 2)/(m-2) \rceil$ such that q is located in the rightward-moving window at the end of superphase $\rho_{a+\phi}$.*

The analogous result for the leftward-moving window is similar. The resulting range for ϕ is $0 \leq \phi \leq \lceil (m(j+1) - i' + 1)/(m-2) \rceil$.

Window Invariant.

The last part of the technique involves proving a property Z about all processors that are found within a window at the end of a given superphase. We have already shown that each processor in the network will eventually be found within one of the windows, so we will be able to guarantee that property Z eventually holds for all processors in the network.

Our current goal is to find upper bounds on the time it takes for a specific message M to reach all processors in the network. So, our property Z will concern the amount of time that elapses between the time M is first transmitted and the time when all processors found within the window receive M. More concretely, the property will take the form "if message M is first transmitted during superphase ρ_a, and q is located in the window at the end of superphase $\rho_{a+\phi}$, the q will receive M by the end of superphase $\rho_{a+\phi+1}$".

LEMMA 5. *Suppose that, during superphase ρ_a, p transmits M on behalf of segment S_i in supertile T_j. Suppose that, at the end of superphase ρ_a, processor q is located in segment $S_{i'}$, with $i' \geq i$ ($i' \leq i$). If, for $\phi \geq 0$, q is found in the rightward-moving (leftward-moving) window at the end of superphase $\rho_{a+\phi}$, then q has received M by the end of superphase $\rho_{a+\phi+1}$.*

From Lemma 5 and an upper bound on the amount of time that elapses before each processor has been found in the window (i.e., Corollary 4 and its leftward analogue), we get an upper bound on the amount of time that elapses before all processors have received a transmitted message M.

THEOREM 6. *Suppose that, during superphase ρ_a, p transmits M on behalf of segment S_i in supertile T_j. Suppose that, at the end of superphase ρ_a, processor q is located in segment $S_{i'}$. Then, q receives M by the end of superphase $\rho_{a+\phi+1}$, where $\phi \leq \max\{\lceil (i' - jm + 2)/(m-2) \rceil, \lceil (m(j+1) - i' + 1)/(m-2) \rceil\}$.*

This upper bound can be greatly simplified if the indices of the leftmost and rightmost segments of the network at the end of superphase ρ_a are known. This is because these indices provide bounds on the values of i', i, and j. If LM and RM are the indices of the leftmost and rightmost segments of the network at the beginning of superphase ρ_0, then, at the end of superphase ρ_a, the leftmost segment has index at least $LM - 2(a+1)$ and the rightmost segment has index at most $RM + 2(a+1)$. This is because a processor can cross at most two different segment boundaries during a single superphase. Therefore, we know that $i, i' \in \{LM - 2(a+1), \ldots, RM + 2(a+1)\}$, which we use to prove the following result.

COROLLARY 7. *Suppose that, during phase π_a, p transmits M on behalf of segment S_i in supertile T_j. Then, every processor receives M by the end of phase $\pi_{a+2\phi+11}$, where $\phi \leq (RM - LM + 4a)/(m-2)$.*

4.4 Running Time

In this section, we provide an upper bound on the running time of our gossiping algorithm. From the termination condition presented in Section 4.2, we know that all processors terminate their local algorithm at the end of phase number $a' + 2(RM - LM + 4a')/(m - 2) + 16$, where a' is the largest timestamp received by the processors. From Lemma 15 (found in Appendix C), $a' \leq 2n - 1$, so, all processors terminate their local algorithm by the end of phase number $2n + 2(RM - LM)/(m - 2) + 8(2n - 1)/(m - 2) + 15$. Next, from Observation 1, we know that every block of W contiguous segments contains at least one processor, $4W + 3 < m$, and $m \geq 8$. It follows that $(n - 1)W \geq RM - LM$ and $8/(m - 2) \leq 3/2$. Therefore, the algorithm terminates by the end of phase number $6n + 13$.

5. CONCLUSIONS

Our gossiping algorithm can be used to weaken the assumptions about trajectory knowledge in the EWS neighbourhood maintenance algorithm. Rather than requiring that all processors know their *entire* future trajectory (as well as those of their neighbours), gossiping can be used to communicate trajectory updates. By running the gossiping algorithm every $6n + 13$ phases and taking each processor's gossiping message to be its trajectory for the next $12n + 26$ phases, we can guarantee (using a simple induction argument) that the trajectory information required by the EWS algorithm is always known.

The window technique presented in Section 4.3 is a conceptual tool for analyzing algorithms in networks with processors that travel along arbitrary, continuous trajectories. We have also used it to provide a slightly weaker, but simpler analysis of the geocasting algorithm presented in [2], even though their algorithm does not divide the environment into segments, nor the set of time slots into phases.

The most important open problem is the acquisition of the neighbourhood knowledge that our algorithm assumes is initially known. For our algorithm to be fully deterministic and reliable, a deterministic neighbourhood learning algorithm for MANETs is needed. This is the focus of our current research. Also, the assumptions and constraints needed by our algorithm are quite strong. Although this poses a problem from a practical standpoint, we feel that, for deterministic gossiping where processors follow arbitrary continuous trajectories, these kinds of constraints are needed. Therefore, we are currently investigating impossibility results to show the inherent difficulty of the problem.

Acknowledgments

The author wishes to thank his advisor, Faith Ellen, whose helpful insights and guidance led to a vastly improved presentation of this research. Also, the author appreciates the invaluable comments and suggestions from the anonymous reviewers of this paper. The author was supported by the Natural Sciences and Engineering Research Council of Canada.

References

[1] A. Anta, A. Milani, M. Mosteiro, and S. Zaks. Opportunistic information dissemination in mobile ad-hoc networks: The profit of global synchrony. In *DISC*, pages 374–388, 2010.

[2] A. F. Anta and A. Milani. Bounds for deterministic reliable geocast in mobile ad-hoc networks. In *OPODIS*, pages 164–183, 2008.

[3] S. Basagni, I. Chlamtac, and D. Bruschi. A mobility-transparent deterministic broadcast mechanism for ad hoc networks. *IEEE/ACM Trans. Netw.*, 7(6):799–807, 1999.

[4] H. C. Chung, S. Viqar, and J. Welch. Neighbor knowledge of mobile nodes in a grid network. In *ICDCS*, pages 486–495, 2012.

[5] A. E. F. Clementi, A. Monti, and R. Silvestri. Distributed broadcast in radio networks of unknown topology. *Theor. Comput. Sci.*, 302(1-3):337–364, 2003.

[6] A. Cornejo, S. Viqar, and J. L. Welch. Reliable neighbor discovery for mobile ad hoc networks. In *DIALM-POMC*, pages 63–72, 2010.

[7] A. Czumaj and X. Wang. Fast message dissemination in random geometric ad-hoc radio networks. In *ISAAC*, pages 220–231, 2007.

[8] F. Ellen, J. L. Welch, and S. Subramanian. Maintaining information about nearby processors in a mobile environment. In *ICDCN*, pages 193–202, 2006.

[9] Y. Emek, L. Gasieniec, E. Kantor, A. Pelc, D. Peleg, and C. Su. Broadcasting in udg radio networks with unknown topology. *Distributed Computing*, 21(5):331–351, 2009.

[10] L. Gasieniec. On efficient gossiping in radio networks. In *SIROCCO*, pages 2–14, 2009.

[11] L. Gasieniec, A. Pagourtzis, I. Potapov, and T. Radzik. Deterministic communication in radio networks with large labels. *Algorithmica*, 47(1):97–117, 2007.

[12] S. K. S. Gupta and P. K. Srimani. An adaptive protocol for reliable multicast in mobile multi-hop radio networks. In *WMCSA*, pages 111–122, 1999.

[13] K. Krzywdzinski. Fast construction of broadcast scheduling and gossiping in dynamic ad hoc networks. In *IMCSIT*, pages 879–884, 2010.

[14] M. Mohsin. *Reliable Communication in Mobile Ad Hoc Networks*. PhD thesis, The University of Texas at Dallas, 2006.

[15] M. Mohsin, D. Cavin, Y. Sasson, R. Prakash, and A. Schiper. Reliable broadcast in wireless mobile ad hoc networks. In *HICSS*, 2006.

[16] E. Pagani and G. P. Rossi. Reliable broadcast in mobile multihop packet networks. In *MOBICOM*, pages 34–42, 1997.

[17] K. Sinha, S. Ghose, and P. K. Srimani. Fast deterministic broadcast and gossiping algorithms for mobile ad hoc networks. *J. Parallel Distrib. Comput.*, 68(7):922–938, 2008.

[18] P. K. Srimani and B. P. Sinha. Mobility tolerant broadcast in mobile ad hoc networks. In *IWDC*, pages 435–446, 2004.

[19] S. Viqar and J. L. Welch. Deterministic collision free communication despite continuous motion. In *ALGOSENSORS*, pages 218–229, 2009.

APPENDIX

A. SCHEDULE PROPERTIES

In this section, we present various properties of the EWS schedule that we will use repeatedly in later proofs. These properties are easily verified by inspecting the provided schedule diagrams.

In each phase of the EWS schedule, there is exactly one segment class that is not scheduled to transmit. So, we have to be careful when proving properties about our algorithm: we may show that, at the beginning of a given phase, there is a processor in a certain part of the network that is ready to transmit a message M, but the processor is located in a segment that is not scheduled to transmit during that phase. However, we notice from the schedule that it is only segment classes $\mathbb{S}_0$ and $\mathbb{S}_{\lfloor m/2 \rfloor}$ that may not be scheduled. Therefore, as the next result states, for any two segments that are close enough together, we know that at least one of them is scheduled to transmit.

Observation 8. *Consider any two segments S_i and S_j. If $|i-j| < \lfloor m/2 \rfloor$, then at least one of $\{S_i, S_j\}$ is scheduled to transmit during every phase.*

From Observation 8 and the fact that a processor can cross at most one segment boundary in one phase, it follows that every processor p is very often located in a segment that is scheduled to transmit (i.e., at least once every two phases).

Observation 9. *For any processor p and any phase π_a, either:*

- *at the beginning of phase π_a, p is located in a segment that is scheduled to transmit in phase π_a, or,*
- *at the beginning of phase π_{a+1}, p is located in a segment that is schedule to transmit in phase π_{a+1}.*

Next, we recall some definitions and establish some new terminology. Recall that superphase ρ_a consists of phases π_{2a} and π_{2a+1} of the EWS schedule. We will call phase π_{2a} the *first half of superphase ρ_a* and π_{2a+1} the *second half of superphase ρ_a*. Also, recall that supertile T_j consists of segments numbered $jm, jm+1, \ldots, j(m+1)-1$. The segments $\{S_{jm}, \ldots, S_{jm+\lfloor m/2 \rfloor -1}\}$ will be called the *left half of supertile T_j* and the segments $\{S_{jm+\lfloor m/2 \rfloor}, \ldots, S_{j(m+1)-1}\}$ will be called the *right half of supertile T_j*. The following fact follows from the relationship between m and W in Observation 1.

Observation 10. *If $jm \le i \le jm+W-1$, then segment S_i is in the left half of supertile T_j. If $jm+m-W \le i \le (j+1)m-1$, then segment S_i is in the right half of supertile T_j.*

The most important property of the EWS schedule is what can be thought of as the 'directionality' of transmissions. This has nothing to do with the actual physical direction of a transmission: all antennae are omnidirectional. Instead, we are referring to the fact that the message will continue to be propagated in a certain direction in the network without significant delay. The following definitions make this more concrete.

Definition 1. A transmission that is scheduled during a slot t on behalf of a segment S_i is called a *rightward transmission* (*leftward transmission*) if the segment S_{i+1} (S_{i-1}) is scheduled to transmit in slot $t+1$ or in slot $t+2$.

Note that a transmission can be both rightward and leftward. Using these definitions, we can augment our schedule diagrams to show the directionality of transmissions, as shown in Figure 1.

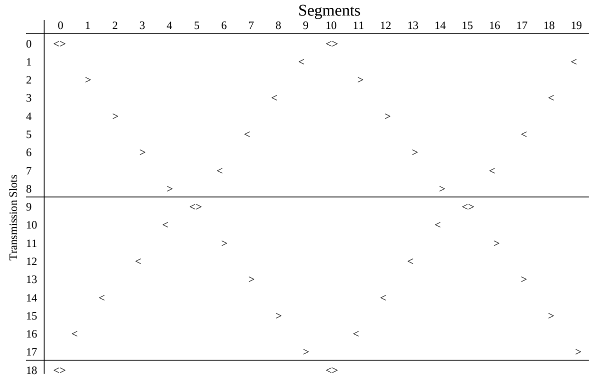

Figure 1: A prefix of the transmission schedule when m=10 augmented with the directionality of transmissions

From the augmented schedule diagram, it is easy to characterize when certain segments are scheduled for rightward versus leftward transmissions.

Observation 11. *In the first half of every superphase, each segment in the left (right) half of T_j is scheduled to perform a rightward (leftward) transmission. In the second half of every superphase, each segment in the right (left) half of T_j is scheduled to perform a rightward (leftward) transmission.*

Since each segment is scheduled to transmit at most once per phase of the schedule, we observe that there is a clear order in which the segments in a given supertile are scheduled to transmit (as demonstrated by the 'diagonal lines' of transmissions in the schedule diagram).

Observation 12. *Suppose that segment S_i in supertile T_j is scheduled to make a rightward (leftward) transmission in slot t of phase π_b. For all segments $S_{i'}$ with $i' > i$ ($i' < i$) in the same half of supertile T_j as S_i, $S_{i'}$ is scheduled to make a rightward (leftward) transmission in phase π_b after slot t.*

Observation 13. *Suppose that in phase π_b, segment S_i is scheduled to make a leftward (and not rightward) transmission. Then, for each $j \in \{i-1, i, i+1\}$, if segment S_j is scheduled to transmit in phase π_{b+1}, then its transmission will be rightward.*

The following fact follows from Observation 8 and the relationship between m and W in Observation 1 in Section 4.1.

Observation 14. *Suppose that in phase π_b, segment S_i is not scheduled to perform a transmission. Then, segments $S_{i+1}, \ldots, S_{i+W}$ are all scheduled to perform a rightward transmission during phase π_{b+1}.*

B. LEADER SELECTION

In this section, we prove that, for any segment S and any phase π, the modified leader selection method in our gossiping algorithm satisfies the following two properties:

1. If there exists a processor in segment S at the beginning of phase π that has never transmitted before the beginning of phase π, then the leader chosen for S for phase π is a processor that has never transmitted before the beginning of phase π.

2. All processors that are in the same segment S at the beginning of a phase π choose the same leader.

Recall the specification of the leader selection algorithm: each processor in segment S, at the beginning of phase π, checks if it has received the gossiping message of each processor in its segment. If so, it chooses as leader the processor with smallest ID in the segment; otherwise, it chooses as leader the processor with smallest ID among all processors whose gossiping messages it has not received.

To see that Property 1 is satisfied, note that the gossiping message of a processor q that has not transmitted before the beginning of phase π is not received by any processor. Therefore, by our leader selection method, each processor in the same segment as q at the beginning of phase π will pick as leader the processor p with smallest ID among all processors whose gossiping messages it has not received. By Lemma 17 (found in Appendix D), processor p did not transmit before the beginning of phase π.

By the definition of the neighbourhood maintenance task, the EWS schedule ensures that each processor knows all of its neighbours at the beginning of phase π. By constraint (C3) in the choice of EWS algorithm parameters, we know that the length of a segment is smaller than the transmission radius. So, we conclude that all processors in segment S at the beginning of phase π know about each other. To prove that Property 2 is satisfied, we consider the following two cases:

1. *Each processor located in S at the beginning of phase π has received the gossiping messages of all other processors located in S at the beginning of phase π.* Since all processors have unique IDs, they will all pick the unique processor with smallest ID, as required.

2. *There is a processor q located in S at the beginning of phase π that has not received the gossiping message of some processor p that is located in S at the beginning of phase π.* Without loss of generality, assume that q chooses processor p as leader according to our leader selection algorithm. Namely, of all processors in S at the beginning of phase π whose gossiping message was not received by q, p has the smallest ID. By Lemma 17, it follows that p did not transmit its gossiping message before the beginning of phase π. Therefore, no processor located in S at the beginning of phase π has received p's gossiping message by the beginning of phase π, so all processors have p as a potential candidate for leader. Next, for any processor p' in S at the beginning of phase π with ID smaller than p's, it must be the case that q received the gossiping message belonging to p' before phase π (by the choice of p). By Lemma 17, all processors located in S at the beginning of phase π received the gossiping message belonging to p' before phase π. Therefore, all processors choose p as leader, as required.

C. TERMINATION

In this section, we show how the termination condition described in Section 4.2 guarantees that gossiping has been completed before any processor terminates and that all processors terminate at the same time. We will need the following result, which guarantees that, as long as there are processors that have never transmitted before, at least one new gossiping message is transmitted every two phases.

LEMMA 15. *Suppose that there is a processor p_i that has not transmitted its message m_i before the beginning of phase π_a. Then, there exists some processor p_j that transmits m_j for the first time in phase π_a or π_{a+1}.*

PROOF. Let π_b, with $b \geq a$, be the first phase such that, at the beginning of phase π_b, p_i is located in a segment S that is scheduled to transmit in phase π_b. By Observation 9, $b \in \{a, a+1\}$. Clearly, p_i has not transmitted its message m_i before the beginning of phase π_b, so, by property 1 of our leader selection method (as discussed in Appendix B), the leader p_j chosen for segment S for phase π_b has never transmitted before the beginning of phase π_b. Therefore, p_j will transmit m_j for the first time in phase π_b, with $b \in \{a, a+1\}$. $\square$

Now, recall the specification of the termination condition: if processor p has received LM and RM, then, if a' is the latest timestamp received by processor p and if no message with later timestamp is received by p by the end of phase number $k = a' + 2(RM - LM + 4a')/(m-2) + 16$, then p terminates at the end of phase π_k.

To prove that no processor terminates before gossiping has been completed, we assume otherwise and show that a contradiction arises. Namely, assume that some processor p terminates at the end of some phase π_z such that there exists a processor q that has not received the gossiping message of some processor u by the end of phase π_z. Let a' be the latest timestamp received by processor p before it terminated. By the termination condition, $z = a' + 2(RM - LM + 4a')/(m-2) + 16$. Suppose that u transmitted for the first time during phase π_b. If $b \leq a'$, then, by Corollary 7, q received u's message by the end of phase $b + 2(RM - LM + 4b)/(m-2) + 11 \leq a' + 2(RM - LM + 4a')/(m-2) + 11 < z$, a contradiction. So, in what follows, we assume that $b > a'$. In other words, u has not transmitted by the end of phase $\pi_{a'}$. By Lemma 15, in either phase $\pi_{a'+1}$ or phase $\pi_{a'+2}$, some processor v that never transmitted before phase $\pi_{a'+1}$ will transmit for the first time. In this transmitted message, v will attach timestamp $c \in \{a'+1, a'+2\}$. By Corollary 7, p will receive v's message by the end of phase $c + 2(RM - LM + 4c)/(m-2) + 11 \leq a' + 2 + 2(RM - LM + 4(a'+2))/(m-2) + 11 \leq a' + 2(RM - LM + 4a')/(m-2) + 2 + 16/(m-2) + 11$. From Observation 1, $m \geq 8$, so $a' + 2(RM - LM + 4a')/(m-2) + 2 + 16/(m-2) + 11 \leq a' + 2(RM - LM + 4a')/(m-2) + 16 \leq z$. This means that, before p terminated, it received a timestamp $c > a'$, a contradiction.

The fact that each processor eventually terminates follows from the following observations:

- the values of LM and RM are transmitted during phase π_0, and, by Corollary 7, these values are received by all processors by the end of phase number $2(RM - LM)/(m-2) + 11$,

- since the number of processors is finite, Lemma 15 implies that all processors transmit at least once by the end of some phase number t', and,
- since no messages contain a timestamp greater than t', all processors will eventually terminate.

To see that all processors terminate at the same time, suppose that a' is the last phase during which some processor transmits for the first time. Since no processor terminates before gossiping is completed, all processors receive a message containing timestamp a' before they terminate. Since no processor will receive a message with a timestamp greater than a', every processor will terminate at the end of phase number $k = a' + 2(RM - LM + 4a')/(m-2) + 16$, as required.

D. SPEED OF MESSAGE PROPAGATION

In this section, we set out to prove the window invariant specified in Lemma 5 of Section 4.3. We focus on proving the result for the rightward-moving window, as the proof for the leftward-moving window is analogous (by symmetry). In what follows, we suppose that there is a single message M that has been transmitted by some source processor p. The goal is to calculate an upper bound on the amount of time that elapses before M reaches an arbitrary processor that is located to the right of p when it transmits.

The first useful result shows that if processor p transmits its message during a phase π_α, then the processors that are close enough to p at the beginning or end of phase π_α will receive p's transmission during phase π_α.

LEMMA 16. *Suppose that a processor p transmits M during a phase π_α as the leader of segment S_i. If processor q is located in $S_{i'}$ with $i' \in \{i-2W, \dots, i+2W\}$ at the beginning or end of phase π_α, then q received M during p's transmission.*

PROOF. Suppose that processor p is located at point x in segment S_i at the beginning of phase π_α. As the length of π_α is $m-1$ transmission slots, it follows that, at any time during phase π_α, p's location must be in the range $[x-(m-1)\sigma, x+(m-1)\sigma]$. Similarly, if q is located at point y in segment $S_{i'}$ at the beginning or end of phase π_α, then, at any time during phase π_α, q's location must be in the range $[y-(m-1)\sigma, y+(m-1)\sigma]$. Therefore, at all times during phase π_α, the distance between p and q is bounded above by $|y-x| + 2(m-1)\sigma$.

Without loss of generality, we assume that $y \geq x$. Since $x \in [iK, (i+1)K)$ and $y \in [i'K, (i'+1)K)$, it follows that $y - x + 2(m-1)\sigma < (i'+1)K - iK + 2(m-1)\sigma \leq (i+2W+1)K - iK + 2(m-1)\sigma < (2W+1)K + 3(m-1)\sigma < R$ by Observation 1. Thus, we have shown that, at all times during phase π_α, the distance between processors p and q is less than R, which implies that q receives the transmission by p. $\square$

Next, we consider whether or not a processor p's transmitted message is received by the processors in a certain section of the network before the arrival of p into that section. This result is crucial for showing the correctness of our modified leader selection method: the processors in a given segment want to choose as leader the processor in the segment with smallest ID that has not transmitted yet. If we can guarantee that all processors in the segment know who else in the segment has already transmitted at least once, then they will all choose the same leader.

LEMMA 17. *Suppose that p is located in segment S_i at the beginning of phase π_a and that p has transmitted a message M before the beginning of phase π_a. If q is located in segment S_j with $j \in \{i-W, \dots, i+W\}$ at the beginning of phase π_a, then q received M before the beginning of phase π_a.*

PROOF. Consider the smallest a such that, at the beginning of phase π_a, p is located in a segment S_i, q is located in a segment S_j with $j \in \{i-W, \dots, i+W\}$, p has transmitted M before the beginning of phase π_a, but q has not received M by the beginning of phase π_a. There are two cases to consider:

1. p transmits during phase π_{a-1}. Then, p is the leader of some segment $S_{i'}$ during phase π_{a-1}. Since p can cross at most one segment boundary per phase, it follows that $i' \in \{i-1, i, i+1\}$, so, at the beginning of phase π_a (i.e., at the end of phase π_{a-1}) q is located in S_j with $j \in \{i'-W-1, \dots, i'+W+1\} \subseteq \{i'-2W, \dots, i'+2W\}$. By Lemma 16 with $\alpha = a-1$ and $i = i'$, q receives p's transmission of M during phase π_{a-1}, which contradicts the assumption that q did not receive M by the beginning of phase π_a.
2. p does not transmit during phase π_{a-1}. Let $S_{i'}$ and $S_{j'}$ be the segments in which p and q are located at the beginning of phase π_{a-1}, respectively. Since a processor can cross at most one segment boundary during a phase, $i' \in \{i-1, i, i+1\}$ and $j' \in \{j-1, j, j+1\} \subseteq \{i-W-1, \dots, i+W+1\}$. It follows that $j' \in \{i'-W-2, \dots, i'+W+2\}$. Since q has not received M by the beginning of π_{a-1}, then, by our choice of a, $j' \notin \{i'-W, \dots, i'+W\}$. So, either $j' \in \{i'-W-2, i'-W-1\}$ or $j' \in \{i'+W+1, i'+W+2\}$. Without loss of generality, assume $j' \in \{i'-W-2, i'-W-1\}$. By Lemma 1, there exists a processor in a segment $S_{i''}$ with $i'' \in \{i'-W, \dots, i'-1\}$ at the beginning of phase π_{a-1}. By Observation 8 (in Appendix A) and Observation 1, the leader of at least one of $\{S_{i'}, S_{i''}\}$ transmits during phase π_{a-1}. Let p' be such a leader, transmitting on behalf of segment S_k, where $k \in \{i', i''\} \subseteq \{j-W, \dots, j+W\}$. By the choice of a, p' received M before the beginning of phase π_{a-1}.
   Finally, since $j' \in \{j-1, j, j+1\}$ and $j' \in \{i'-W-2, i'-W-1\}$, it follows that $j \leq i'-W$. Also, since $i' \in \{i-1, i, i+1\}$ and $j \geq i-W$, it follows that $j \geq i'-W-1$. Finally, since $i'-W \leq i'' \leq i'-1$, it follows that $i'-W \leq k \leq i'$, so $k-2W \leq j \leq k+2W$. By Lemma 16 with $\alpha = a-1$, $p = p'$, and $i = k$, q received the transmission of M by p' during phase π_{a-1}, which contradicts the assumption that q did not receive M before the beginning of phase π_a.

$\square$

The next result shows that as long as p is not located near the right edge of its supertile when it makes a rightward transmission, M will be transmitted again during the same superphase by the leader of a segment to the right of p.

LEMMA 18. *Suppose that, during superphase ρ_a, p performs a rightward transmission of message M on behalf of segment S_i in supertile T_j. If $i \leq jm + m - W - 1$, then M will also be transmitted via a rightward transmission later in superphase ρ_a by the leader of segment $S_{i'}$ in supertile T_j for some $i' > i$.*

PROOF. There are two cases to consider:

1. *p's transmission occurs during the second half of superphase ρ_a.* By Observation 11 (in Appendix A, since p performs a rightward transmission on behalf of S_i, it follows that S_i is located in the right half of supertile T_j. By Observation 1, there is some segment $S_{i'}$ with $i' \in \{i+1, \ldots, i+W\}$ that contains a processor at the beginning of phase π_{2a+1}. Let p' be the leader of $S_{i'}$ for phase π_{2a+1}. By Lemma 16 with $q = p'$ and $\alpha = 2a+1$, p' receives p's transmission of M. Since $i < i' \leq i + W \leq jm + m - 1$, it follows that $S_{i'}$ is located in the right half of supertile T_j. By Observation 12 (in Appendix A, p' is scheduled to perform a rightward transmission in phase π_{2a+1} after p does. Thus, we have shown that p' transmits M via a rightward transmission on behalf of a segment $S_{i'}$ in supertile T_j for some $i' > i$.

2. *p's transmission occurs during the first half of superphase ρ_a.* By Observation 11, since p performs a rightward transmission on behalf of S_i, it follows that S_i is located in the left half of supertile T_j. There are two sub-cases to consider:

   (a) $i \leq jm + \lfloor m/2 \rfloor - W - 1$. By Observation 1, there is some segment $S_{i'}$ with $i' \in \{i+1, \ldots, i+W\}$ that contains a processor at the beginning of phase π_{2a}. Let p' be the leader of $S_{i'}$ for phase π_{2a}. By Lemma 16 with $q = p'$ and $\alpha = 2a$, p' receives p's transmission of M. Since $i < i' \leq i + W \leq jm + \lfloor m/2 \rfloor - 1$, it follows that $S_{i'}$ is located in the left half of supertile T_j. By Observation 12, p' is scheduled to perform a rightward transmission in phase π_{2a} after p does. Thus, we have shown that p' transmits M via a rightward transmission on behalf of a segment $S_{i'}$ in supertile T_j for some $i' > i$.

   (b) $i \geq jm + \lfloor m/2 \rfloor - W$. By Observation 1, there is some segment $S_{i'}$ with $i' \in \{i+W, \ldots, i+2W-1\}$ that contains a processor at the beginning of phase π_{2a+1} (equivalently, at the end of phase π_{2a}). Let p' be the leader of $S_{i'}$ for phase π_{2a+1}. By Lemma 16 with $q = p'$ and $\alpha = 2a$, p' receives p's transmission of M. Since $i' \geq i+W \geq jm+\lfloor m/2 \rfloor$, it follows that $S_{i'}$ is located in the right half of supertile T_j. By Observation 11, p' is scheduled to perform a rightward transmission in phase π_{2a+1}, namely, after p's transmission in phase π_{2a}. Thus, we have shown that p' transmits M via a rightward transmission on behalf of a segment $S_{i'}$ in supertile T_j for some $i' > i$.

□

We have just shown that p's rightward transmission in superphase ρ_a gets propagated rightward during the same superphase until it reaches the right edge of p's supertile. We now use this to show that all processors in the same region as p during superphase ρ_a receive M by the end of superphase ρ_a. This is essentially the base case of the window invariant: processors found in the rightward-moving window at the end of superphase ρ_a receive M by the end of superphase ρ_a.

LEMMA 19. *Suppose that, during superphase ρ_a, p performs a rightward transmission of message M on behalf of segment S_i in supertile T_j. Suppose that, at the end of superphase ρ_a, q is located in $S_{i'}$ in region X_j, with $i' \geq i$. Then, q has received M by the end of superphase ρ_a.*

PROOF. Consider the maximum value of i such that, during superphase ρ_a, p performs a rightward transmission of message M on behalf of segment S_i in supertile T_j and q does not receive M by the end of superphase ρ_a.

First, suppose that $i \geq jm+m-W$. Then, by Observation 10 (in Appendix A, S_i is in the right half of supertile T_j. By Observation 11, p's transmission occurs during the second half of superphase ρ_a. Since $S_{i'}$ is in region X_j, it follows that $i' \leq (j+1)m < jm+m+W \leq i+2W$. Since q is located in segment $S_{i'}$ at the end of phase π_{2a+1}, Lemma 16 implies that q received p's transmission of M. This contradicts the choice of i.

Next, suppose that $i \leq jm+m-W-1$. By Lemma 18, M will also be transmitted via a rightward transmission later in superphase ρ_a by the leader of a segment $S_{i'}$ in supertile T_j for some $i' > i$. This contradicts the choice of i, so it must be the case that q receives M by the end of superphase ρ_a.

□

Now we consider what happens across supertile boundaries. If p transmits M via a rightward transmission from a supertile T_k, and supertile T_{k+1} contains a processor at the beginning of the next superphase, then M will be transmitted again by a leader near the left edge of supertile T_{k+1} during the first half of the next superphase.

LEMMA 20. *Suppose that, during superphase ρ_b, message M is transmitted via a rightward transmission on behalf of a segment in supertile T_k. If supertile T_{k+1} contains a processor at the beginning of superphase ρ_{b+1}, then there exists a processor that performs a rightward transmission of M on behalf of a segment in $\{S_{(k+1)m}, \ldots, S_{(k+1)m+W-1}\}$ during phase $\pi_{2(b+1)}$.*

PROOF. Consider the last rightward transmission of M on behalf of a segment in supertile T_k during superphase ρ_b. Suppose that this transmission is performed by processor p on behalf of segment S_i. By Lemma 18, it follows that $i \geq km + m - W$. Further, by Observation 10, S_i is in the right half of supertile T_k. By Observation 11, p's transmission occurs in the second half of superphase ρ_b.

Assume that supertile T_{k+1} contains a processor at the beginning of superphase ρ_{b+1}. Our goal is to show that there is a processor q located in a segment $S_{i'}$ with $i' \in \{(k+1)m, \ldots, (k+1)m + W - 1\}$ at the beginning of superphase ρ_{b+1}. First, we know that p was chosen as leader of S_i in supertile T_k at the beginning of phase π_{2b+1}. Therefore, $i \leq (k+1)M - 1$. Since p can cross at most one segment boundary per phase, it follows that, at the beginning of superphase ρ_{b+1}, p is in or to the left of segment $S_{(k+1)M}$. If p is in $S_{(k+1)M}$ at the beginning of superphase ρ_{b+1}, then set $q = p$ and we are done. Otherwise, we proceed with the assumption that p is located to the left of supertile T_{k+1} at the beginning of superphase ρ_{b+1}. By assumption, we know that there is at least one processor located in supertile T_{k+1} at the beginning of superphase ρ_{b+1}. If we assume that all such processors are to the right of the segment with index $(k+1)m + W - 1$, then the W segments with indices $(k+1)m, \ldots, (k+1)m + W - 1$ are all empty. These segments are all in the network since p is to the left of them and there exists at least one processor to the right of them. Therefore, Lemma 1 is violated, and we conclude that there is a processor q located in a segment $S_{i'}$ with $i' \in \{(k+1)m, \ldots, (k+1)m + W - 1\}$ at the beginning of superphase ρ_{b+1}. By Observation 10, $S_{i'}$ is in the left half of supertile T_{k+1}. By Observation 11, q will perform a rightward transmission on behalf of $S_{i'}$ in the first half of superphase ρ_{b+1}.

Finally, since S_i is in supertile T_k, $i < (k+1)m \leq i'$. Also, $i' - i \leq ((k+1)m + W - 1) - (km + m - W) = 2W - 1$. Since $i' \in \{i+1, \ldots, i+2W-1\}$, then, by Lemma 16 with $\alpha = 2b + 1$, q received M during p's transmission during phase π_{2b+1}. Hence, q transmits M during its transmission in superphase ρ_{b+1}. □

So far, we have shown that a rightward transmission of M during a superphase ρ_a gets propagated within the same supertile during ρ_a (Lemma 19) and into the next supertile during ρ_{a+1} (Lemma 20). So, using an induction argument, we now show that message M continues to be propagated until it gets to the right edge of the network.

LEMMA 21. *Suppose that, during superphase ρ_a, message M is transmitted via a rightward transmission on behalf of a segment in supertile T_j. For all $\phi > 0$, if supertile $T_{j+\phi}$ contains a processor at the beginning of superphase $\rho_{a+\phi}$, then there exists a processor that performs a rightward transmission of M on behalf of a segment in $\{S_{(j+\phi)m}, \ldots, S_{(j+\phi)m+W-1}\}$ during phase $\pi_{2(a+\phi)}$.*

PROOF. The proof is by induction on ϕ. For the base case, set $\phi = 1$. By assumption, message M is transmitted via a rightward transmission on behalf of a segment in supertile T_j during superphase ρ_a. If supertile T_{j+1} contains a processor at the beginning of superphase ρ_{a+1}, then, by Lemma 20, there exists a processor that performs a rightward transmission of M on behalf of a segment in $\{S_{(j+1)m}, \ldots, S_{(j+1)m+W-1}\}$ during phase $\pi_{2(a+1)}$.

As induction hypothesis, for $\phi > 1$, assume that if supertile $T_{j+\phi-1}$ contains a processor at the beginning of superphase $\rho_{a+\phi-1}$, then there exists a processor that performs a rightward transmission of M on behalf of a segment in $\{S_{(j+\phi-1)m}, \ldots, S_{(j+\phi-1)m+W-1}\}$ during phase $\pi_{2(a+\phi-1)}$.

For $\phi > 1$, suppose that supertile $T_{j+\phi}$ contains a processor p at the beginning of superphase $\rho_{a+\phi}$. We show that supertile $T_{j+\phi-1}$ contains a processor at the beginning of superphase $\rho_{a+\phi-1}$. First, consider the location of p at the beginning of superphase $\rho_{a+\phi-1}$. Since p can cross at most two segment boundaries during a single superphase and each supertile has at least 8 segments (by Observation 1), it follows that p is located in or to the right of supertile $T_{j+\phi-1}$ at the beginning of superphase $\rho_{a+\phi-1}$. So we proceed with the assumption that p is located to the right of supertile $T_{j+\phi-1}$ at the beginning of superphase $\rho_{a+\phi-1}$. Next, from the Lemma statement, there is a processor p' located in a segment S_i of supertile T_j at the beginning of superphase ρ_a. By the definition of T_j, $i \leq (j+1)m - 1$. Since p' can cross at most two segment boundaries in one superphase, it follows that, at the beginning of superphase $\rho_{a+\phi-1}$, p' is in or to the left of the segment with index $i + 2(\phi-1) \leq (j+1)m + 2(\phi-1)$. The leftmost segment of supertile $T_{j+\phi-1}$ has index $(j+\phi-1)m = (j+1)m + m(\phi-2)$, which is strictly greater than $(j+1)m + 2(\phi-1)$ since $m \geq 8$ (by Observation 1). Therefore, we have shown that, at the beginning of superphase $\rho_{a+\phi-1}$, p' is located to the left of supertile $T_{a+\phi-1}$. Thus, all segments in supertile $T_{j+\phi-1}$ are part of the network at the beginning of superphase $\rho_{a+\phi-1}$, so, by Lemma 1, any contiguous block of W segments in the supertile contains a processor, as required.

By the induction hypothesis, there exists a processor that performs a rightward transmission of M on behalf of a segment in $\{S_{(j+\phi-1)m}, \ldots, S_{(j+\phi-1)m+W-1}\}$ during phase $\pi_{2(a+\phi-1)}$. In particular, during superphase $\rho_{a+\phi-1}$, message M is transmitted via a rightward transmission on behalf of a segment in supertile $T_{j+\phi-1}$. Since $T_{j+\phi}$ contains a processor at the beginning of superphase $\rho_{a+\phi}$, it follows from Lemma 20 that there exists a processor that performs a rightward transmission of M on behalf of a segment in $\{S_{(j+\phi)m}, \ldots, S_{(j+\phi)m+W-1}\}$ during phase $\pi_{2(a+\phi)}$.

□

In Lemma 19, we basically proved the "base case" of the window invariant: processors located in the rightward-moving window at the end of superphase ρ_a receive M by the end of superphase ρ_a. We now prove the induction step for the case when p's transmission is rightward.

LEMMA 22. *Suppose that, during superphase ρ_a, p performs a rightward transmission of M on behalf of segment S_i in supertile T_j. Suppose that, at the end of phase ρ_a, processor q is located in segment $S_{i'}$, with $i' \geq i$. If, for $\phi \geq 0$, q is found in region $X_{j+\phi}$ at the end of superphase $\rho_{a+\phi}$, then q has received M by the end of superphase $\rho_{a+\phi}$.*

PROOF. We proceed by induction on ϕ. When $\phi = 0$, the result follows by Lemma 19. As induction hypothesis, assume that, for $\phi > 0$, if q is found in region $X_{j+\phi-1}$ at the end of superphase $\rho_{a+\phi-1}$, then q has received M by the end of superphase $\rho_{a+\phi-1}$.

Now, suppose that q is found in region $X_{j+\phi}$ at the end of superphase $\rho_{a+\phi}$. There are two cases to consider:

1. *At the beginning of superphase $\rho_{a+\phi}$, q is located to the left of supertile $T_{j+\phi}$.* At the end of superphase $\rho_{a+\phi}$, q is in region $X_{j+\phi}$, that is, q is located in a segment with index at least $(j+\phi)m - 1$. Since q can cross at most 2 segment boundaries during superphase $\rho_{a+\phi}$, it follows that, at the beginning of superphase $\rho_{a+\phi}$ (or, equivalently, at the end of superphase $\rho_{a+\phi-1}$) q is located in a segment with index at least $(j+\phi)m - 3 \geq (j+\phi-1)m$. In particular, this means that q is located in region $X_{j+\phi-1}$ at the end of superphase $\rho_{a+\phi-1}$. By the induction hypothesis, q has received M by the end of superphase $\rho_{a+\phi-1}$.
2. *At the beginning of superphase $\rho_{a+\phi}$, q is located in or to the right of supertile $T_{j+\phi}$.* Then it follows that supertile $T_{j+\phi}$ is non-empty at the beginning of superphase $\rho_{a+\phi}$. By Lemma 21, there exists a processor p'' that performs a rightward transmission of M on behalf of a segment $S_{i''}$ with $i'' \in \{(j+\phi)m, \ldots, (j+\phi)m + W - 1\}$ during phase $\pi_{2(a+\phi)}$. If $i' \geq i''$, then, by Lemma 19, q receives M by the end of superphase $\rho_{a+\phi}$. Otherwise, $i' < i'' \leq (j+\phi)m + W - 1$. Since q can cross at most two segment boundaries during superphase $\rho_{a+\phi}$, it follows that, at the beginning of superphase $\rho_{a+\phi}$, q is located to the left of segment with index $(j+\phi)m + W + 1 \leq i'' + W + 1$. But, by assumption, at the beginning of superphase $\rho_{a+\phi}$, q is located

in or to the right of segment $(j+\phi)m \geq i'' - W + 1$. It follows that $i' \in \{i''-W+1, \ldots, i''+W\} \subseteq \{i''-2W, \ldots, i''+2W\}$. By Lemma 16 with $p = p''$, $i = i''$ and $\alpha = 2(a+\phi)$, we know that q received the transmission of M by p'' during phase $\pi_{2(a+\phi)}$.

□

Finally, we must consider what happens if p's transmission is not rightward. In this case, there is a small delay before M is transmitted via a rightward transmission. Once this rightward transmission occurs, M is propagated rightward relatively quickly, as we have already shown. In fact, a processor found in the rightward-moving window must wait one more superphase for M to arrive than in the case where the initial transmission of M was rightward.

LEMMA 23. *Suppose that, during superphase ρ_a, p performs a leftward (but not rightward) transmission of M on behalf of segment S_i in supertile T_j. Suppose that, at the end of phase ρ_a, processor q is located in segment $S_{i'}$, with $i' \geq i$. If, for $\phi \geq 0$, q is found in region $X_{j+\phi}$ at the end of superphase $\rho_{a+\phi}$, then q has received M by the end of superphase $\rho_{a+\phi+1}$.*

PROOF. Suppose that p transmits during phase π_b in superphase ρ_a. Suppose that, at the end of phase π_b, q is located in segment $S_{i''}$.

First, consider the case where $i \leq i'' \leq i+2W$. At the end of phase π_b, q is located in $S_{i''}$ with $i'' \in \{i-2W, \ldots, i+2W\}$, so, by Lemma 16, q receives p's transmission of M during phase π_b. Thus, q receives M by the end of superphase ρ_a.

So, in what follows, we assume that $i'' \geq i + 2W + 1$. At the end of phase π_b, p is located in some segment S_k with $k \in \{i-1, i, i+1\}$ since p can cross at most one segment boundary per phase. Therefore, $i'' \geq k + 2W$. By Lemma 1, there must be a leader p' in a segment with index $k' \in \{k+1, \ldots, k+W\}$ at the beginning of phase π_{b+1}. Also, we know that p is in segment S_k at the beginning of phase π_{b+1}. Thus, by Observation 8, at least one of p' or the chosen leader for S_k will transmit during phase π_{b+1}. Further, since $k, k' \in \{k-W, \ldots, k+W\}$, by Lemma 17, the transmitting processor has already received M before the beginning of phase π_{b+1}, so M will be transmitted during phase π_{b+1}.

Next, we show that this transmission of M is rightward. If segment S_k is scheduled to transmit in phase π_{b+1}, then, by Observation 13 (in Appendix A), segment S_k is scheduled to perform a rightward transmission in phase π_{b+1}. Otherwise, segment S_k is not scheduled to transmit in phase π_{b+1}, so, by Observation 14 (in Appendix A), segment $S_{k'}$ is scheduled to perform a rightward transmission in phase π_{b+1}.

Finally, we consider how long it takes before q receives M. There are two cases to consider:

1. Suppose that $b = 2a$. It follows that, at the end of phase π_{2a}, q is located in segment $S_{i''}$ with $i'' \geq i + 2W + 1$. Since q can cross at most one segment boundary per phase, it follows that, at the end of phase π_{2a+1}, q is located in segment S_ℓ with $\ell \geq i + 2W$. But, the transmission of M during phase π_{b+1} is on behalf of a segment $S_{k''}$ with $k'' \in \{k, \ldots, k+W\} \subseteq \{i-1, \ldots, i+W+1\}$. So, we know that, at the end of phase π_{2a+1} (namely, at the end of superphase ρ_a), q is located in a segment with index $\ell \geq k$. By Lemma 22, q will receive M by the end of superphase $\rho_{a+\phi}$.
2. Suppose that $b = 2a + 1$. It follows that, at the end of phase π_{2a+1}, q is located in segment $S_{i''}$ with $i'' \geq i + 2W + 1$. Since q can cross at most one segment boundary per phase, it follows that, at the end of phase π_{2a+3}, q is located in segment $S_{k'}$ with $k' \geq i + 2W - 1$. But, the transmission of M during phase π_{b+1} is on behalf of a segment S_k with $k \in \{j, \ldots, j+W\} \subseteq \{i-1, \ldots, i+W+1\}$. So, we know that, at the end of phase π_{2a+3} (namely, at the end of superphase ρ_{a+1}), q is located in a segment with index k', where either $k' = k - 1$ or $k' \geq k$.

   If $k' = k-1$, then, since q can cross at most one segment boundary per phase, q is located in a segment with index in $\{k-2, k-1, k\} \subseteq \{k-2W, \ldots, k+2W\}$ at the end of phase π_{2a+2}. Therefore, by Lemma 16, q receives the transmission of M during phase $\pi_{2a+2} = \pi_{b+1}$, that is, before the end of superphase ρ_{a+1} (which is by the end of $\rho_{a+\phi+1}$ since $\phi \geq 0$).

   Otherwise, if $k' \geq k$, then, by Lemma 22 with $a = a+1$, q will receive M by the end of superphase $\rho_{a+\phi+1}$.

□

Taken together, Lemmas 22 and 23 imply the window invariant (in the rightward-moving case) described in Lemma 5 of Section 4.3, as desired.

E. PROOFS FOR THE WINDOW TECHNIQUE

Proof of Lemma 2.

PROOF. The proof is by induction on β. First, consider the case when $\beta = 0$. Since the leftmost segment of T_j has index jm and S_i is contained in T_j, q must be located in a tile with index at least jm at the end of superphase ρ_a. But, the leftmost segment of region X_j is S_{jm-1}, so q is not located to the left of region X_j at the end of superphase ρ_a.

Now, consider the case when $\beta > 0$. Suppose q is not located in $X_{j+\beta}$ at the end of superphases $\rho_a, \ldots, \rho_{a+\beta}$. Consider the location of q at the end of superphase $\rho_{a+\beta-1}$. By the induction hypothesis, q is not located to the left of region $X_{j+\beta-1}$. Since q is not located in region $X_{j+\beta-1}$, q is located to the right of region $X_{j+\beta-1}$. Since the rightmost tile in region $X_{j+\beta-1}$ has index $(j+\beta)m$, q must be located in a tile with index at least $(j+\beta)m+1$. During superphase $\rho_{a+\beta}$, q can cross at most two different tile boundaries. Thus, q must be located in a tile with index at least $(j+\beta)m - 1$ at the end of superphase $\rho_{a+\beta}$. But, $S_{(j+\beta)m-1} \subseteq X_{j+\beta}$. Hence, q is not located to the left of region $X_{j+\beta}$ at the beginning of superphase $\rho_{a+\beta}$, as required. □

Proof of Lemma 3.

PROOF. At the end of superphase $\rho_{a+\beta}$, q is located at or to the left of segment $S_{i'+2\beta}$, since q can cross at most 2 different segment boundaries during one superphase. Rearranging the inequality $\beta > (i' - jm + 1)/(m-2)$ gives $i' + 2\beta < (j+\beta)m - 1$. But, the leftmost segment of $X_{j+\beta}$ is $S_{(j+\beta)m-1}$, so segment $S_{i'+2\beta}$ is located to the left of region $X_{j+\beta}$. □

Proof of Corollary 7.

PROOF. The result follows from Theorem 6 by finding a suitable upper bound for $\max\{\lceil(i'-jm+2)/(m-2)\rceil, \lceil(m(j+1)-i'+1)/(m-2)\rceil\}$. First, consider $\lceil(i'-jm+2)/(m-2)\rceil$. Since S_i is in supertile T_j, we know that $i \leq (j+1)m$, so $jm \geq i-m$. Therefore, $\lceil(i'-jm+2)/(m-2)\rceil \leq \lceil(i'-(i-m)+2)/(m-2)\rceil$. Next, since $i, i' \in \{LM-2(a+1), \dots, RM+2(a+1)\}$, it follows that $\lceil(i'-(i-m)+2)/(m-2)\rceil \leq \lceil((RM-LM+4a)+(m+6))/(m-2)\rceil$. By Observation 1, $m \geq 8$, so $\lceil((RM-LM+4a)+(m+6))/(m-2)\rceil \leq (RM-LM+4a)/(m-2)+4$. A similar argument shows that $\lceil(m(j+1)-i'+1)/(m-2)\rceil \leq (RM-LM+4a)/(m-2)+4$. Therefore, by Theorem 6, for an arbitrary processor q, q receives p's transmission by the end of superphase $\rho_{a+\phi+1}$, where $\phi \leq (RM-LM+4a)/(m-2)+4$. Since each superphase consists of two phases, this means that q receives the transmission by the end of phase $\pi_{a+2\phi+11}$, as required.

□