A Mobility-Transparent Deterministic Broadcast Mechanism for *Ad Hoc* Networks

Stefano Basagni, *Student Member, IEEE*, Danilo Bruschi, and Imrich Chlamtac, *Fellow, IEEE*

***Abstract*— Broadcast (distributing a message from a source node to all other nodes) is a fundamental problem in distributed computing. Several solutions for solving this problem in mobile wireless networks are available, in which mobility is dealt with either by the use of randomized retransmissions or, in the case of deterministic delivery protocols, by using conflict-free transmission schedules. Randomized solutions can be used only when unbounded delays can be tolerated. Deterministic conflict-free solutions require schedule recomputation when topology changes, thus becoming unstable when the topology rate of change exceeds the schedule recomputation rate. The deterministic broadcast protocols we introduce in this paper overcome the above limitations by using a novel mobility-transparent schedule, thus providing a delivery (time) guarantee without the need to recompute the schedules when topology changes. We show that the proposed protocol is simple and easy to implement, and that it is optimal in networks in which assumptions on the maximum number of the neighbors of a node can be made.**

***Index Terms*— *Ad hoc* networks, broadcast protocols, distributed algorithms, mobile computing.**

I. Introduction

BROADCAST is the task initiated by any of the nodes of a network, called *source*, whose goal is to send a message m to all other network nodes. In addition to disseminating data, a broadcast mechanism represents the starting point for the implementation of group communication primitives and various system services in distributed environments. With the renewed interest in *ad hoc* networks, combined with the need to support multimedia and real-time applications, the need arises to have in the mobile network a mechanism for reliable dissemination of control and data. It is necessary to have broadcast protocols that meet the combined requirements of these networks—*delivery-guarantee* and *mobility transparency*.

The term *ad hoc*, or *multi-hop mobile* radio network, refers to a set of geographically dispersed nodes which may be stationary or mobile, in which each node is willing to forward packets for other nodes that cannot communicate directly with each other (namely, *each* node is also a switch). These networks successfully fill a role in applications in which a wired backbone is not viable. Emergency services, whether commercial or tactical, all rely on this type of networks to provide communication. Law enforcement, disaster recovery, as well as *ad hoc* networks to assist in administration and control of entertainment events, shows, etc., are all cost-effectively best served by multi-hop networks.

One of the basic characteristic of these networks is the use of shared transmission channels. Thus, selective transmission is impossible: whenever a node transmits, all its neighbors (nodes within transmission range) will receive the message, and a collision may occur if some transmissions overlap, preventing correct message reception. In this paper, we are interested in broadcast protocols with "collision resolution," that is, protocols in which the broadcast algorithm itself guarantees the delivery of the message in the presence of collisions. For references to protocols that use "collision detection" mechanisms, e.g., see [1]. (These protocols, however, do not guarantee that a node can sense all collisions [2]; thus, media where no collision detection is performed are usually considered.)

The broadcast problem has been extensively studied for multi-hop networks. In particular, several solutions have been presented in which the broadcast *time complexity* is investigated in detail.[1] *Optimal* solutions were obtained for the case when each node knows the topology of the entire network (*centralized* broadcast). The broadcast protocol introduced in [3] completes the broadcast of a message in $O(D\log^2 n)$ steps. From the result proved in [4], this protocol is optimal for networks with constant diameter. For networks with a larger diameter, a protocol by Gaber *et al.* [5] completes the broadcast within $O(D+\log^5 n)$ time slots, and it is optimal for networks with $D \in \Omega(\log^5 n)$. These solutions are *deterministic* and guarantee a bounded delay on message delivery, but the requirement that each node must know the entire network topology is a strong condition, impossible to maintain in *ad hoc* mobile environments.

The mobility limitation of the above protocols can be overcome by sacrificing delivery guarantees by using *randomized* protocols. In [1], a *randomized* broadcast protocol which works in $O(D\log n+\log^2 n)$ time slots was given. Recently, Kushilevitz *et al.* [6] proved that such an algorithm is optimal for any network with $D \leq n^{1-\epsilon}$, where ϵ is any constant >0. A suite of randomized protocols based on [1] is presented in [7] where, by means of simulations, the time

Manuscript received December 9, 1996; revised December 8, 1997; approved by IEEE/ACM TRANSACTIONS ON NETWORKING Edior K. Sabnani. This work was supported in part by the Army Research Office under Contract DAAG55-97-1-0312.

S. Basagni and I. Chlamtac are with the Center for Advanced Telecommunications Systems and Services, University of Texas at Dallas, Richardson, TX, 75083-0688 USA (e-mail: basagni@utdallas.edu; chlamtac@utdallas.edu).

D. Bruschi is with the Department of Computer Science, Universitá degli Studi di Milano, Milan, Italy.

Publisher Item Identifier S 1063-6692(99)09710-1.

[1] We express a protocol time complexity as the number of *time slots*, or steps, required by the protocol for broadcasting a message. The parameters used for expressing such a complexity measure are n, the number of nodes in the network, and D, the diameter of the network.

complexity of one of the protocol of the suite is demonstrated to improve the theoretical results in many network topologies. Randomized solutions, however, can be applied to non time-dependent applications, i.e., when unbounded delays can be tolerated during the broadcast process, and cannot be used as a mechanism for control information dissemination.

Deterministic broadcast protocols that do not require the knowledge of the entire network topology have been introduced in, e.g., [8], [9]. They operate by maintaining an updated *broadcast spanning tree*. In order to adapt to topology changes, these protocols require that each node knows the identity of its current neighbors and recomputes the transmission schedule accordingly (using an independent control channel). Furthermore, the rate of node mobility must not exceed the rate at which updating of topological information can occur. Another broadcast protocol characterized by both topology and time localization of execution has been introduced in [10]. However, the protocol requires nodes to exchange information about the topology over a control channel, and to execute, prior to each transmission, a distributed algorithm to decide which nodes are going to transmit. A broadcast protocol for "cellular" type *ad hoc* topologies, termed multi-cluster architectures, has also been proposed in [11]. In this case, nodes are organized in clusters, mimicking the organization of cellular networks. The underlying virtual cellular organization in mobile networks has to be updated continuously, and the proposed overlayed broadcast protocol is influenced by the speed of the nodes: when the rate of change of the network topology becomes too high, the protocol switches to flooding. This heavily affects its time complexity (which can even be quadratic in n), and provides no guarantees on delivery.

Given the increased interest in mobility, we are interested in designing a broadcast solution which overcomes the above mentioned mobility related limitations, while providing guaranteed delivery. Specifically, we are interested in a solution which is the following.

1) *Mobility independent*, in that the correct forwarding of a message is always guaranteed independently of the current node's neighbors and of their rates of mobility.
2) *Deterministic*, so that an *a priori* known bound on the maximum delay for broadcast completion can be determined.
3) *Distributed*, in the sense that the protocol can be executed at each node without the *a priori* knowledge not only of the entire network topology, but also of the identity of the neighbors.
4) *Simple and easy to implement*, i.e., no computational overhead is associated with the transmission of a message and no periodical recomputation of the transmission schedule is needed. Moreover, each node can compute its own transmission schedule efficiently.

We observe that this type of mechanism is the required basis for low-level protocols in all those wireless and mobile situations in which network state and control information have to be efficiently disseminated among all (or part of) the nodes of the network *without* depending on the network state itself and on the rate of mobility of the nodes.

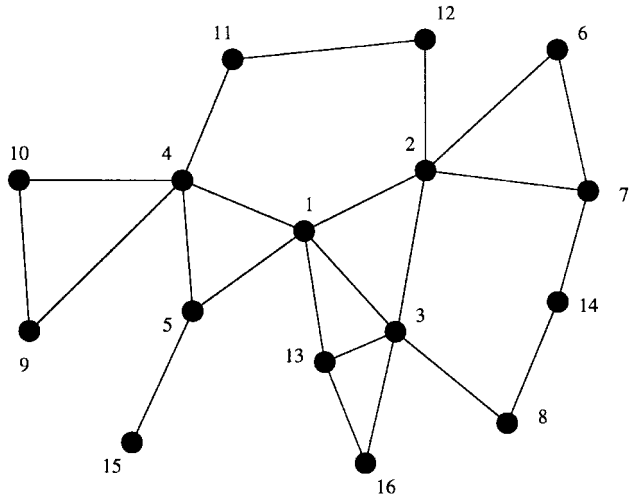


Fig. 1. A multi-hop network with 16 nodes, $D = 4$ and $\Delta = 5$.

In this paper, we present a general algorithmic scheme for devising broadcast protocols with the above-mentioned properties. In particular, we introduce a new broadcast protocol in which each node computes its own transmission schedule *once and for all* at network initialization depending only on global network parameters, such as n and the *degree* of the network, namely, the maximum number of neighbors that a node can have. Thus, the proposed protocol is mobility independent in the sense defined above. Moreover, our solution completes the broadcast of a message in polylogarithmic time. All these results are obtained by characterizing the broadcast problem as a combinatorial problem for the solution of which we propose a novel, explicit, and deterministic method. Furthermore, when assumptions can be made on the degree of the network, our characterization of the broadcast problem allows us to show that our solution is optimal.

II. Preliminaries

We model a multi-hop network by an undirected graph $G = (V, E)$ in which $V = \{p_1, \cdots, p_n\}$ is the set of (radio) nodes and there is an edge $(p_i, p_j) \in E$ if and only if p_j is in the *hearing range* (namely, can hear the transmissions) of p_i and vice versa. In this case, we say that p_i and p_j are neighbors. Due to mobility, the graph may change in time.

The set of the neighbors of a node p will be indicated by $\Gamma(p)$ and its cardinality, $\delta(p) = |\Gamma(p)|$, is called the *degree* of p. As usual, $\Delta = \max\{\delta(p)\colon p \in V\}$ indicates the maximum *degree* of the network G. The *distance* $d(p_i, p_j)$ between two nodes p_i and p_j, $1 \leq i, j \leq n$, is defined as the length of the shortest path (minimum number of hops) between p_i and p_j. The maximum distance between any pair of nodes is called the *diameter* D of the network. Given the source s of a message, all the nodes p such that $d(s, p) = \ell \leq D$ are said to belong to the ℓth *layer* of the network, $0 \leq \ell \leq D$. Every node in the network is assigned a unique ID which we assume denoted 1–n. As an example, the topology of a simple multi-hop network is shown in Fig. 1.

A *deterministic distributed broadcast protocol* Π *for multihop networks* is a protocol which is executed *at each node* in the network in the following way:

a) Time of execution is considered to be slotted and the time slots, or *rounds*, are numbered $0, 1, \cdots$. At round 0, a specific node s, called the *source*, transmits a message m.
b) In each round, a node acts either as a transmitter or as a receiver. A node receives a message m in a specific round if, and only if, in that round it acts as a receiver and *exactly* one of its neighbors acts as a transmitter. In this case m is the same message transmitted by the neighbor.
c) The action of a node in a specific round is deterministically determined by its initial input, i.e., its own ID (my_ID), n, and the degree Δ of the network.
d) The broadcast is *completed* at round t if all the nodes have correctly received the message m at one of the rounds $0, 1, \cdots, t$.

Thus, the broadcast proceeds according to a *schedule* $L_\Pi = \langle T_1, \cdots, T_t \rangle$, i.e., according to a list of *transmissions* (*transmission sets*) which specifies for each round i the set of nodes which act as (potential) transmitters, $1 \leq i \leq t$.

During the broadcast process, the nodes that in a given round have received a message m are said to be *covered* by the broadcast. The nodes that have not received m are said to be *uncovered*. Given a node p, $\Gamma_c(p)\,(\Gamma_u(p))$ will indicate its (un)covered neighborhood. Finally, a set H of covered nodes is said to be a *conflicting set* if $\cap_{p \in H}\ \Gamma_u(p) \neq \emptyset$. In other words, H is a conflicting set when there is at least a neighbor common to all the nodes in H that has not received a message from them yet. Finally, in the case of multi-hop networks with mobile nodes, we assume that at least one node from a conflicting set remains in the hearing range of any neighboring uncovered node. Notice that the network does *not* have to be static during the entire broadcast process, but it is required always to be *connected*, i.e., each uncovered node in the network must be able to receive a message.

III. A General Broadcast Scheme

The problem of distributed broadcast as stated in the previous section is that of scheduling, in a deterministic way, the transmissions of the covered nodes in order to guarantee the correct delivery of the message independently of the possibility of collisions. To this end, we assume that the time axis is divided into units called (transmission) *frames*. Each frame is made up of rounds, numbered 1–τ, where $\tau > 0$ is the frame *length*. We assume that the nodes are synchronized on a frame basis (namely, we assume that each node has a counter which is set to 1 at the beginning of each frame and that is incremented by 1 with each subsequent round) and that the round length is the same for each node.

Each node that either generated or received a message m is allowed to transmit it only in certain rounds in a frame. The node calculates these slots (at the set up of the network, or any time the number of the nodes in the network changes) by means of the following procedure that takes as input the number of nodes in the network and its degree and returns a set of integers

PROCEDURE *Round_Numbers* (n, Δ);
begin
  Transm := *Get_The_Rounds* (n, Δ)
end;

The set Transm $\subseteq \{1, \cdots, \tau\}$ will contain the rounds in which the node is allowed to transmit the message. By specifying the function *Get_The_Rounds*, we get different broadcast protocols.

As soon as a node either has a message m ready for transmission or receives m, it waits for the beginning of a new frame (this frame will be its own transmission frame). At that time, it starts to check when it can transmit m. Specifically, the node tests if the current value of the counter belongs to Transm, and when this is the case, the node sends m.

By a simple inductive argument, it is possible to show that the described scheme achieves the broadcast of m in a layer by layer fashion.

Proposition 1: Each node p such that $d(s,p) = \ell$ transmits the message m issued by s *after* m has been transmitted by all the nodes in the layer $\ell - 1$ and *before* each node in the layer $\ell + 1$ will transmit it, $0 \leq \ell < D$.

It is easy to see that such a scheme completes the broadcast in $t \leq D\tau$ rounds as long as we can find a suitable function *Get_The_Rounds* which allows us to prove that m is correctly forwarded from a given layer to the subsequent one in the τ rounds of a frame. Thus, what we need is a (deterministic) method to distribute the nodes to the transmission sets in a frame in such a way that it is *always* guaranteed that at least one set will contain only *one* node from *any* conflicting set $H \subseteq V$ (i.e., in the round corresponding to that transmission set, no collision will occur). More than that, we want such a method to generate a schedule that is independent of the *local* current conditions of the network, i.e., such that each node has no need to know the identity of its current neighbors to be guaranteed of the correct delivery of the message (distributivity).

Thus, the described scheme will have the following desirable properties.

1) *Parallel Broadcast:* More than one message issued by different source nodes can be traversing the network at any given time. The distributed nature of the method used to generate the broadcast schedule guarantees the correct reception of a message sent by *any* neighbor of a given node. These neighbors may have to send different messages. In this case, since a node is allowed to forward only one message per frame, a message may need to be temporarily buffered (and thus, delayed) at one node. The time bounds we present in this paper refer to messages that are at least τ rounds from each other. Indeed, it is easy to see that if any node either receives or generates no more than one message m for each frame, then m can be forwarded in the following frame (i.e., it is never subject to buffering delays).

2) *Mobility:* Given the schedule independence of the current neighborhood of a node, the topology of the network may change without affecting the broadcast process. Moreover, every node that has moved from an uncovered neighborhood to a covered one during the broadcast must at some time be the neighbor of a node which has already received the broadcasted message m, and will receive m from it using a *failsafe* recovery procedure such as in [12], [13].
3) *Scalability:* Anytime we want to add a new node to the network, each newly inserted node can issue a broadcast message requesting to update the Transm set of each node according to the new value of n. Each node, then, has only to execute the above procedure *Round_Numbers*.

In the following section, we illustrate an application of our layer-to-layer broadcast mechanism by describing a simple linear protocol that works in any multi-hop mobile network.

IV. A Linear Broadcast Algorithm

One of the simplest possible broadcast algorithms that meets the requirements/properties listed in the previous sections is obtained using the following:

```
FUNCTION Get_The_Rounds (n, Δ): integer;
begin
  output my_ID
end;
```

Each node is allowed to transmit just once in a frame: when the value of its counter equals its own ID. This simple method generates a layer-to-layer schedule for which $\tau = n$. Due to the uniqueness of the node ID's, it is clear that, at most, one node will transmit in a round, so that no collision can ever occur. Such a broadcast protocol Π has a schedule $L_\Pi = \langle T_1, \cdots, T_t \rangle$ such that each T_j, $1 \leq j \leq t$, is a singleton and t is bounded by Dn.

Remark 1: According to the general scheme presented in the previous section, the algorithm obtained using the previous *Get_The_Rounds* function is an *off-line* algorithm: each node calculates in advance the round in which it will transmit the message, and keeps this information in the set Transm (which, in this case, is an integer variable). The following is an equivalent (with respect to the time complexity) *on-line* version of the same algorithm.

When a node v receives a message m, it sets a timer to

$$\text{wakeup\_time} = n - u + v$$

where u is the ID of the first sender from which v has received m. The timer is decremented with each round. When the timer equals 0, v transmits m.

It is easy to see that:

1) due to the uniqueness of the ID of each node, no collision will occur, i.e., no more than one node will transmit in the same round;
2) a node of level ℓ will transmit the message in round j, $(\ell-1)n < j \leq \ell n$, $1 \leq \ell \leq D$. This implies that no node in layer ℓ will transmit before a node in layer $\ell-1$, $1 \leq \ell \leq D$, or after a node in layer $\ell+1$, $1 \leq \ell < D$, i.e., the broadcast proceeds in a layer by layer fashion and it takes $\ell n - (\ell-1)n = n$ rounds to forward m from layer ℓ to the next layer, $1 \leq \ell < D$.

The total number of rounds t required by the previous algorithm to complete the broadcast is bounded by the following expression:

$$t \leq \sum_{\ell=1}^{D} n = n \sum_{\ell=1}^{D} 1 = Dn.$$

$\star$

The described protocols work in multi-hop networks with maximum degree $\Delta = n-1$, i.e., they always complete the broadcast in any multi-hop network. (It is clear from the code of the *Get_The_Rounds* function that this linear protocol *does not* depend on the degree of the network.)

In [14], it is shown that the simple algorithms given above are optimal if a broadcast protocol Π is not able to use a schedule L_Π^ℓ for the nodes of layer ℓ which uses *all* the nodes of layer ℓ and *all* the uncovered nodes. For these "restricted" protocols, an $\Omega(Dn)$ lower bound for the deterministic distributed broadcast of a message m in a (mobile) multi-hop network is proved. Moreover, the construction used in [14] is general, i.e., for multi-hop networks with any diameter D, and immediately obtains, as a special case, the $\Omega(n)$ lower bound presented in [1] for networks with constant diameter.

In the following section, we propose a distributed broadcast algorithm which maintains the property of being deterministic and mobility independent, i.e., we prove that it is always possible to correctly forward a message m from any layer to the following one within a deterministically bounded frame length and without depending either on the knowledge of the current neighbors or on the rate of their mobility.[2] This algorithm completes the broadcast in polylogarithmic time, and in *sparse* networks (i.e., in networks with a "small" or with a constant maximum degree Δ), it has to be preferred to the linear protocols just described.

V. Polylogarithmic Broadcast

As noticed in Section III, the problem of the correct forwarding of a message between any two consecutive layers of a multi-hop network is that of distributing the nodes to the transmission sets in a frame in such a way that in at least one transmission set there are no two nodes from the same conflicting set. In the previous section, we have introduced a simple protocol which solves this problem by avoiding the transmission of more than one node per round. Here, we prove that this condition is not necessary, i.e., we show that by allowing more than one node to transmit in a round we can still guarantee the mobility independence property while correctly forwarding any message. Moreover, depending on the degree of the network, we show how to obtain polylogarithmic frame lengths, i.e., frame lengths shorter than that of the linear solution. We obtain these results by presenting a novel

[2] For the sake of simplicity on the description of the method, we consider here that only one message m is traversing the network.

combinatorial method (*division method*) that, given a non-empty set of integers P, distributes the elements of *any* non-empty set $R \subseteq P$ in a family $\mathcal{F}$ of τ subsets of P, so that there exists at least a set $T \in \mathcal{F}$ such that $|T \cap R| = 1$. We can think of this family as made up of those (transmission) sets of nodes allowed to transmit in a specific round, so as to always guarantee the correct delivery of a message (i.e., we guarantee that at a given round, only *one* node from *any* set of nodes transmits. In that round, no collision occurs).

The method is completely deterministic and constructive, and it can be executed *at each node* p in order to find the numbers of the rounds in which p is allowed to transmit. Each node needs only to know the number of the nodes in the network n and the degree of the network Δ. Starting from the set $P = \{1, \cdots, n\}$ (of the ID's) of the nodes of the network, each node executes the function *Get_The_Rounds* that generates the family $\mathcal{F}$ of sets with the mobility independence property. The set of integers P is initially *divided* into $2\log|P|$ sets in such a way that each element of P belongs to $\log|P|$ sets. The crucial property of this division is that, for each non-empty set $R \subseteq P$, there always exist two sets T_1 and T_2 among the $2\log|P|$ sets, such that $T_1 \cap R$ and $T_2 \cap R$ are a partition of R. In terms of the broadcast schedule, this means that, given any conflicting set, we can always distribute its nodes into two different non-empty subsets so that if we let the two groups of nodes transmit in different rounds, no collision will occur between the corresponding transmissions. An iterated application of this division to subsequently divided sets leads to the distribution of the elements of R in several subsets of P so that there is at least one of these sets, say T, such that $|T \cap R| = 1$. In other words, if R is *any* conflicting set of nodes, then the schedule (list of τ transmission sets) obtained by applying the division method to the set of nodes P always contains a set to which only one of the nodes in R belongs. In the round corresponding to that set all the nodes in the common uncovered neighborhood of the nodes in R receive the message correctly. Once $\mathcal{F}$ is locally generated, for each $T \in \mathcal{F}$ each node checks if its own ID (my_ID) belongs to T, and, when this is the case, the round number corresponding to T is added in Transm.

In the remaining part of this section, we first describe the division method, then present a family of broadcast algorithms based on the method and prove their correctness. For the sake of simplicity, in this section we consider n and $|P|$ powers of 2. All logarithms are to be considered to be base 2. Finally, throughout the section we use the asymptotic notation to express the time complexity bounds of our protocols (e.g., see [15]), but it is easy to see that for all the results presented here this notation does not hide any significant constant.

A. The Division Method

Consider a non-empty set of integers P. In this section, we describe a general method for deriving a family of subsets of P that *hits* any non-empty set $R \subseteq P$, i.e., a family $\mathcal{F} \subseteq 2^P$ such that there exists at least a set $T \in \mathcal{F}$ for which $|T \cap R| = 1$. In the following, with the operator $\longrightarrow$ we will partition a set of integers I into two subsets I_1 and I_2 with the same cardinality.

The method is based on the following procedure that given a set of integers I, $|I| \geq 2$, divides I into $2\log|I|$ distinct sets $T^1, \cdots, T^{2\log|I|}$

```
PROCEDURE Divide(I);
begin
  I ⟶ o_1^1, e_1^1;
  T^1 := o_1^1;
  T^2 := e_1^1;
  for i := 2 to log|I| do
    begin
    T^{2i-1} := T^{2i} := ∅;
    for j := 1 to 2^{i-2} do
      begin
      o_j^{i-1} ⟶ o_{2j-1}^i, e_{2j-1}^i;
      e_j^{i-1} ⟶ o_{2j}^i, e_{2j}^i;
      T^{2i-1} := T^{2i-1} ∪ o_{2j-1}^i ∪ o_{2j}^i;
      T^{2i} := T^{2i} ∪ e_{2j-1}^i ∪ e_{2j}^i
      end
    end
end;
```

An application of the procedure *Divide* is explained in the following example.

Example 1: Consider $P = \{1, \cdots, 16\}$. The following is the output of the *Divide* procedure called on P. Starting from

$$P = \underbrace{\{1,2,3,4,5,6,7,8}_{o_1^1}, \underbrace{9,10,11,12,13,14,15,16\}}_{e_1^1}$$

we will have

$$T^1 = \{\underbrace{1,2,3,4}_{o_1^2}, \underbrace{5,6,7,8}_{e_1^2}\}$$

$$T^2 = \{\underbrace{9,10,11,12}_{o_2^2}, \underbrace{13,14,15,16}_{e_2^2}\}$$

$$T^3 = \{\underbrace{1,2}_{o_1^3}, \underbrace{3,4}_{e_1^3}, \underbrace{9,10}_{o_2^3}, \underbrace{11,12}_{e_2^3}\}$$

$$T^4 = \{\underbrace{5,6}_{o_3^3}, \underbrace{7,8}_{e_3^3}, \underbrace{13,14}_{o_4^3}, \underbrace{15,16}_{e_4^3}\}$$

$$T^5 = \{\underbrace{1}_{o_1^4}, \underbrace{2}_{e_1^4}, \underbrace{5}_{o_2^4}, \underbrace{6}_{e_2^4}, \underbrace{9}_{o_3^4}, \underbrace{10}_{e_3^4}, \underbrace{13}_{o_4^4}, \underbrace{14}_{e_4^4}\}$$

$$T^6 = \{\underbrace{3}_{o_5^4}, \underbrace{4}_{e_5^4}, \underbrace{7}_{o_6^4}, \underbrace{8}_{e_6^4}, \underbrace{11}_{o_7^4}, \underbrace{12}_{e_7^4}, \underbrace{15}_{o_8^4}, \underbrace{16}_{e_8^4}\}$$

$$T^7 = \{1,3,5,7,9,11,13,15\}$$

$$T^8 = \{2,4,6,8,10,12,14,16\}.$$

♢

It is easy to verify that for each i, $1 \leq i \leq \log|P|$, (a) $T^{2i-1} \cap T^{2i} = \emptyset$, $T^{2i-1} \cup T^{2i} = P$, (b) $|T^{2i-1}| = |T^{2i}| = |P|/2$. This implies that after h, $1 \leq h \leq \log|P|$, calls of the *Divide* procedure every time on a set of its output, the

last output will be sets with cardinality $|P|/2^h$. Furthermore, (c) for each j, $1 \le j \le 2^{i-1}$, is $|o_j^i| = |P|/2^i$ $(= |c_j^i|)$. Another useful property of the *Divide* procedure is stated in the following lemma.

Lemma 1: Given a subset R of a set of integers P, $|R| \ge 2$, there always exists an i, $1 \le i \le \log|P|$, such that R is partitioned by the procedure call Divide(P) into two non-empty subsets, R_1 and R_2, such that $R_1 \subseteq T^{2i-1}$ and $R_2 \subseteq T^{2i}$.

Proof: We show that as far as $R \subseteq T^{2i-1}(T^{2i})$, $1 \le i \le \log|P| - 1$, then there exists a j, $1 \le j \le 2^{i-1}$, such that $R \subseteq o_j^i$ (c_j^i) and that as soon as this is no longer true we have the thesis. We proceed by induction on the number i of subsequent partitions of the set P. Let $R \subseteq o_1^1 = T^1$ (the case $R \subseteq c_1^1 = T^2$ is symmetric. The case in which $(R_1 =) R \cap T^1 \neq \emptyset \neq R \cap T^2$ $(= R_2)$ is obvious). The base case of induction $(i = 1)$ is then trivial. Now, suppose that for a j, $1 \le j \le 2^{i-2}$, we have $R \subseteq o_j^{i-1}$ (the case with $R \subseteq c_j^{i-1}$ is analogous). We know that in the next iteration of the main loop in the *Divide* procedure we will have

$$o_j^{i-1} \longrightarrow o_{2j-1}^i, e_{2j-1}^i$$

and we have the following two cases: either $R \subseteq o_{2j-1}^i$ (e_{2j-1}^i) or R is partitioned into two non-empty subsets R_1 and R_2 such that $R_1 \subseteq o_{2j-1}^i$ and $R_2 \subseteq e_{2j-1}^i$, which implies the thesis ($R_1 \subseteq T^{2i-1}$ and $R_2 \subseteq T^{2i}$).

Notice that the second case always occurs when

$$|o_j^{i-1}| \ge R \ge (|o_j^{i-1}|/2) = |o_{2j-1}^i|$$

(see (c) above), whence the thesis in $\log|P|$ steps. □

An immediate consequence of the previous lemma is that each set $R \subseteq P$ such that $2 \le |R| \le 3$ is partitioned into two subsets, such that at least one is a singleton. In broadcast terms, this means that the nodes of any conflicting set with cardinality ≤ 3 are scheduled to transmit in such a way that in at least one round *only one* node transmits m. In that slot all their uncovered neighbors receive m correctly. A repeated application of the procedure *Divide* to subsequently divided sets, allows to hit any non-empty set $R \subseteq P$. This is achieved by the following function (where $c \ge 1$ is an integer variable that takes values $\le |I|$ and $h \in \{0, \cdots, \log c\}$):

```
FUNCTION Smash (I);
begin
  if |I| = c / 2^h
  then output I
  else
    begin
      Divide (I);
      for j := 1 to 2log|I| do Smash (T^j)
    end
end;
```

Example 2: Consider $P = \{1, \cdots, 16\}$ and $c = |P| = 16$. Then, when $h = 0$, the function called *Smash*(P) returns P as output. When $h = 1$, the same call will output the 8 sets as in Example 1. When $h = 2$, we obtain the 48 sets output by the *Divide* procedure successively called on the sets $T^1, \cdots, T^8$. For example, *Divide*(T^1) gives the 6 sets $T^1 = \{1,2,3,4\}$, $T^2 = \{5,6,7,8\}$, $T^3 = \{1,2,5,6\}$, $T^4 = \{3,4,7,8\}$, $T^5 = \{1,3,5,7\}$, $T^6 = \{2,4,6,8\}$; *Divide*(T^3) gives the 6 sets $T^1 = \{1,2,3,4\}$, $T^2 = \{9,10,11,12\}$, $T^3 = \{1,2,9,10\}$, $T^4 = \{3,4,11,12\}$, $T^5 = \{1,3,9,11\}$, $T^6 = \{2,4,10,12\}$, and *Divide*(T^4) gives the 6 sets $T^1 = \{5,6,7,8\}$, $T^2 = \{13,14,15,16\}$, $T^3 = \{5,6,13,14\}$, $T^4 = \{7,8,15,16\}$, $T^5 = \{5,7,13,15\}$ and $T^6 = \{6,8,14,16\}$. ♢

It is easy to verify (see also (b) above) that, in general, for a set of integers P and $c = |P|$, the function called *Smash*(P) outputs $\tau = 2^h \Pi_{j=0}^{h-1} \log(|P|/2^j)$ sets, each one with cardinality $|P|/2^h$, $0 \le h \le \log|P|$.[3] Furthermore, when $c = |P|$, the depth of the recursion is h. It is worth noticing that h also indicates the number of subsequent applications of the *Divide* procedure to subsequently halved sets.

We show now that given a non-empty set of integers P and $c = |P|$, for each h, $0 \le h \le \log|P|$, the function call *Smash*(P) returns a family of subsets of P that hits any non-empty set $R \subseteq P$, such that $|R| < 2^{h+1}$.

Theorem 1: Given a non-empty set of integers P and the integer $c = |P|$, for each h, $0 \le h \le \log|P|$, the Smash function called on P returns a family $\mathcal{F} \subseteq 2^P$ such that for any non-empty set $R \subseteq P$, $|R| < 2^{h+1}$, $\mathcal{F}$ hits R.

Proof: We proceed by induction on h, the depth of the recursive calls of the *Smash* function on subsequently halved sets. When $h = 0$, then any non-empty set $R \subseteq P$ such that $|R| < 2$ is a singleton. In this case, the *Smash*(P) call returns P (Example 2) and $\mathcal{F} = \{P\}$ clearly hits any singleton $R \subseteq P$.

Suppose now that, after $h - 1$, $h > 0$, recursive calls of the *Smash* function on a set of integers $I \subseteq P$, we obtain a family of subsets of I that hits any subset of I whose cardinality is $< 2^h$. Consider now a set $R \subseteq P$ such that $|R| < 2^{h+1}$. After the first call of the *Divide* ($h > 0$ ensures that we have at least a call of the *Divide* procedure) we know that there exists an i, $1 \le i \le \log|P|$, such that R is partitioned into two non-empty subsets $R_1 \subseteq T^{2i-1}$ and $R_2 \subseteq T^{2i}$ (Lemma 1). Now, $2^{h+1} > |R| = |R_1| + |R_2|$ implies that between the two conditions $|R_1| < 2^h$ and $|R_2| < 2^h$ at least one holds. Without loss of generality, let $|R_1| < 2^h$. As noted above, being $c = |P|$, the depth of recursion is h and therefore, by inductive hypothesis, the remaining $h - 1$ recursive calls triggered by *Smash*(T^{2i-1}) give the thesis. □

Example 3: Consider again $P = \{1, \cdots, 16\}$, $c = |P| = 16$ and the set $R = \{3,5,6\}$. When $h = 1$, after the first call of the *Divide* procedure on the set $\{1, \cdots, 16\}$ we know that there exists an i, $1 \le i \le 4$, such that R is partitioned into two non-empty subsets R_1 and R_2 with $R_1 \subseteq T^{2i-1}$ and $R_2 \subseteq T^{2i}$ (Lemma 1). Indeed, when $i = 2$, we have $R_1 = \{3\} \subseteq T^3$ (namely, T^3 hits R) and $R_2 = \{5,6\} \subseteq T^4$ (see Example 1). Let us now consider $h = 2$ and the set $R = \{1,2,3,4,5,6,7\}$. Using the same argument as above, we find that $R_1 = \{1,2,3,4\} \subseteq T^3$ and $R_2 = \{5,6,7\} \subseteq T^4$. Then, the previous theorem guarantees that the *Divide*(T^4)

[3] When $h = 0$, we stipulate that $\Pi_{j=0}^{h-1} \log(|P|/2^j) = 1$.

call will distribute the elements of R_2 in such a way that in at least one of the sets there will be only one element of R_2 (see Example 2: the set T^6 generated by the *Divide*(T^4) procedure call, hits R). ♢

Remark 2: When $h>1$, the *Smash* function called on a set of integers P with $c=|P|$ returns $\tau = 2^h \Pi_{j=0}^{h-1} \log(|P|/2^j)$ sets which *are not* all pairwise different (see Example 2: the division of T^1 and T^3 yields the same set $T^1=\{1,2,3,4\}$). Therefore, the actual cardinality of the family $\mathcal{F}$ is $<\tau$. This is evident when $h=\log|P|$ ($|P|>1$): when $c=|P|$, the *Smash*(P) call returns $|P|\Pi_{j=0}^{\log|P|-1}\log(|P|/2^j)>|P|$ sets, each one with cardinality 1. Thus, it is clear that $|\mathcal{F}|=|P|$ (and indeed the family of all singletons of P hits any non-empty subset $R\subseteq P$), but the *Smash* function returns all the τ sets unaware that sometimes it outputs a set already produced. This consideration on the *redundancy* of the division method leaves room for improvements both from the combinatorial point of view and for the complexity of distributed broadcast, but it is not further investigated in this paper. ⋆

B. The "Polylog" Broadcast Algorithm

The *Get_The_Rounds* function used by each node (by the means of the *Round_Numbers* procedure, see Section III) uses a slightly modified version of the *Smash* function described in the previous section. Instead of returning a set, the following *Find_Rounds* procedure divides the set $P=\{1,\cdots,n\}$ of nodes' ID's into $\tau = 2^h\Pi_{j=0}^{h-1}\log(n/2^j)$ sets, $1\le h<\log n$,[4] and checks if the ID, my_ID, of the node that executes the procedure *Round_Numbers* belongs to the resulting sets

FUNCTION *Get_The_Rounds* (n,Δ): Set of integers;
begin
  $c := n$;
  $h = \lfloor \log \Delta \rfloor$;
  $\rho := 2^{h-1} \prod_{j=1}^{h-1} \log \frac{n}{2^j}$;
  Temp $:= \emptyset$;
  Find_Rounds $(\{1,\cdots,n\},0,0)$;
  output Temp
end;

where

PROCEDURE *Find_Rounds* (I,x,y);
begin
  if $|I| = \frac{c}{2^h}$
    then if my_ID $\in I$
      then Temp $:=$ Temp $\cup \{\rho(x-1)+y\}$
    else begin
      Divide (I);
      if $x=0$
        then for $j := 1$ **to** $2\log|I|$
          do *Find_Rounds* $(T^j,j,1)$
        else for $j := 1$ **to** $2\log|I|$
          do *Find_Rounds* (T^j,x,j)
    end
end;

Example 4: Suppose that node p with my_ID $=2$ executes the procedure *Round_Numbers* $(16,\Delta)$. When $\Delta=2,3$ ($h=1$), the previous *Get_The_Rounds* function will output the set $\{1,3,5,8\}$, i.e., during a frame of length $\tau=8$, node p is allowed to transmit the message in rounds 1, 2, 3 and 8. Indeed, if we consider $n=16$, the transmission sets $T_1,\cdots,T_8$ of a frame are the 8 sets output by the *Divide* procedure called on $\{1,\cdots,16\}$. More precisely: $T_i=T^i$, $1\le i\le 8$ (Example 1). When $4\le\Delta<8$ ($h=2$), we have Transm $=\{1,3,6,7,9,12,13,15,18,43,45,47\}$, i.e., during a frame of length $\tau=48$, node p is allowed to transmit the message in rounds $1,3,6,7,9,12,13,15,18,43,45$ and 47. The transmission sets $T_1,\cdots,T_{48}$ of the frame are the 48 sets output by the *Divide* procedure called on the sets $T_1,\cdots,T_8$ (Example 2). ♢

Now we prove that the *Get_The_Rounds* function described in this section allows a mobility-independent forwarding of a message m between any two consecutive layers.

Proposition 2: Consider a multi-hop network with n nodes and maximum degree $\Delta=2^{h+1}-1$, $1\le h<\log n$, in which each node p executes the Round_Numbers procedure with the previous Get_The_Rounds function. Then, in a frame of length $\tau=2^h\Pi_{j=0}^{h-1}\log(n/2^j)$, the message m is correctly forwarded between any two consecutive layers.

Proof: The proof that any conflicting set $R\subset P$ is distributed among the transmission sets of a frame in such a way that no collision occurs in at least one round, relies on Theorem 1, noticing that the two integer parameters x and y used by the *Find_Rounds* procedure do not interfere with the division method: they just deal with the problem of the correct attribution of the rounds to the node p. As for this last problem, we note that when the *Get_The_Rounds* function calls the *Find_Rounds* procedure with the actual parameters $P=\{1,\cdots,n\}$, $x=0$, and $y=0$, the **else** branch of the outermost **if** is executed ($c=n$ and for each h, $1\le h<\log n$, $n\ne(n/2^h)$). Then, being $x=0$, the **then** branch of the innermost **if** is executed (this is the unique time in the whole recursive execution of the procedure) and the *Find_Rounds* procedure is recursively called on the $2\log n$ sets returned by the *Divide* $(\{1,\cdots,n\})$ call. If $h=1$, each one of the $2\log n$ calls of the *Find_Rounds* procedure (*Find_Rounds* $(T^j,j,1)$, $1\le j\le 2\log n$) executes the **then** branch of the outermost **if** and if the membership condition is satisfied, the set Temp (i.e., the set Transm of the node p) is updated with a round number. More precisely, in this case the round number is $\rho(x-1)+y=x-1+1=x$, and being $x=j$, $1\le j\le 2\log n$, each one of the $2\log n$ rounds is correctly assigned.[5] If $h>1$, then each one of the $2\log n$

[4] Notice that, considering networks with maximum degree Δ, $2\le\Delta\le n-1$, we now let h range over $\{1,\cdots,\log n-1\}$.

[5] {Notice that, when $h=1$, we have $\rho=1$ provided that we define $\Pi_{j=1}^{h-1}\log(n/2^j)=1$ (see also Note 3).

calls of the *Find_Rounds* procedure executes again the **else** branch of the outermost **if**, but this time the condition of the innermost **if** is no longer verified ($x = j, 1 \leq j \leq 2\log n$) and the successive recursive calls of the *Find_Rounds* procedure are of the form *Find_Rounds* (T^j, x, j), $1 \leq j \leq 2\log|I|$ and $1 \leq x \leq 2\log n$**.** After $h-1$ such recursive calls, the *Find_Rounds* procedure finally executes the **then** branch of the outermost **if**, and if the membership condition is verified the set Temp is updated. As noticed in Section V-A, each one of the initial sets output by the first call of the *Divide* procedure is in turn divided into $\rho = 2^{h-1}\Pi_{j=1}^{h-1}\log(n/2^j)$ transmission sets whose round number is $\rho(x-1)+y$, $1 \leq x \leq 2\log n$ and $1 \leq y \leq 2\log(n/2^{h-1})$. The total number of rounds needed is thus $\tau = \rho 2\log n = 2^h\Pi_{j=0}^{h-1}\log(n/2^j)$. □

According to the general scheme of Section III, this section is summed up by the following:

Theorem 2: The broadcast algorithm obtained using the previous function Get_The_Rounds completes the broadcast in $t \in O(D2^h\log^h n)$ rounds in multi-hop networks with n nodes and maximum degree $\Delta = 2^{h+1}-1$, $1 \leq h < \log n$.

Remark 3 (Optimality): Due to the $\Omega(D\log n)$ lower bound on deterministic distributed broadcast protocol in (mobile) multi-hop networks proved in [16], when $h = 1$ (namely, for networks with maximum degree $\Delta = 3$) the bound proved in Theorem 2 is tight. Furthermore, due to the lower bound on the broadcast of a message in networks with constant diameter presented in [4], when $h = 2$ (namely, for networks with maximum degree $\Delta = 7$) and D is a constant, the presented algorithm is optimal.

When $h = \log n - 1$, i.e., in multi-hop networks with the maximum possible degree, the above presented algorithm completes the broadcast in $O(Dn\log^{\log n} n)$ rounds, so that the linear algorithm presented in Section III is to be preferred. In general, in networks with $\Delta = 2^{h+1}-1$, the above family is to be used when $h \in O((\log n/\log\log n))$, as stated by the following result.

Proposition 3: For each integer h, $0 \leq h < (\log n/\log\log n + 1)$ is $(2\log n)^h < n$.

Proof: Starting from $(2\log n)^h < n$ we have

$$\begin{aligned} \log(2\log n)^h &< \log n \\ h(\log 2\log n) &< \log n \\ h(\log\log n + \log 2) &< \log n \\ h(\log\log n + 1) &< \log n \end{aligned}$$

and thus, $h < (\log n/\log\log n + 1)$. □

Being $(2\log n)^h \geq \tau = 2^h\Pi_{j=0}^{h-1}\log(n/2^j)$, $0 \leq h \leq \log n$ (the proof is easily obtained by induction on h), Proposition 3 says that for networks with maximum degree $\Delta = 2^{\lfloor \log n/\log\log n+1\rfloor+1}-1$, the broadcast algorithm presented in this section is faster than the linear one.

It is worth noticing that, for all those values of Δ for which $\tau < n$, the corresponding protocols perform almost as well when given a polynomial upper bound (denoted N) in n (instead of the actual number of nodes). This bound yields the same complexity up to a constant factor (since complexity is logarithmic in N).

Remark 4: The protocol described in this section depends on the maximum degree Δ of the network: Once Δ is known, h is easily derived and consequently the set Transm is computed. The completion of the broadcast is then guaranteed when $\Delta < 2^{h+1}$. This condition can be globally insured in static multi-hop networks, or in mobile networks in which it is always possible to know the number of nodes in a given area (e.g., networks of satellites). In the general case of ad hoc networks, a mechanism that provides each node with the current degree of the network is needed. Here we assume, similar to past solutions, the existence of a separate control channel by which each node senses its neighbors and communicates its degree through the network. It is worth noticing that each node can always compute its transmission schedules in advance (one transmission schedule for each Δ for which the frame length is $<n$), and it chooses the transmission schedule to use according to the current maximum degree. Thus no on-line recomputation of the schedule is needed when the maximum degree changes.

VI. Conclusion

In this paper, we have presented a new mechanism for disseminating data and control information in ad hoc mobile networks. The proposed solution is deterministic and independent of the mobility of the nodes. We have shown that our protocols overcome the limitations induced in other solutions by the use of randomized techniques or, in deterministic delivery protocols, by the need to periodically recompute the transmission schedule. The mobility independence of our protocol is achieved by allowing each node to compute its own transmission schedule once and for all at network initialization, depending only on n and on the degree Δ of the network. Moreover, the broadcast of a message is proven to be guaranteed in polylogarithmic time. Obtaining these results became possible by characterizing the broadcast problem as a combinatorial problem for the solution of which we proposed a novel, explicit and completely deterministic method. Finally, when an assumption can be made on the degree of the network, we showed that our solution is optimal.

Acknowledgment

The authors wish to thank E. Rosti for her valuable suggestions.

References

[1] R. Bar-Yehuda, O. Goldreich, and A. Itai, "On the time-complexity of broadcast in multi-hop radio networks: An exponential gap between determinism and randomization," *J. Comput. Syst. Sci.*, vol. 45, pp. 104–126, Aug. 1992.

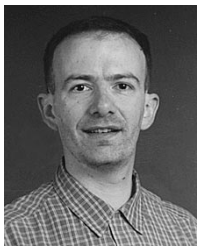
[2] W. F. Lo and H. T. Mouftah, "Collision detection and multitone tree search for multiple-access protocols on radio channels," *IEEE J. Select. Areas Commun.*, vol. SAC-5, pp. 1035–1040, July 1987.

[3] I. Chlamtac and O. Weinstein, "The wave expansion approach to broadcasting in multihop radio networks," *IEEE Trans. Commun.*, vol. 39, pp. 426–433, Mar. 1991.

[4] N. Alon, A. Bar-Noy, N. Linial, and D. Peleg, "A lower bound for radio broadcast," *J. Comput. Syst. Sci.*, vol. 43, pp. 290–298, Oct. 1991.

[5] I. Gaber and Y. Mansour, "Broadcast in radio networks," in *Proc. 6th Annu. ACM-SIAM Symp. Discrete Algorithms*, San Francisco, CA, Jan. 1995, pp. 577–585.

[6] E. Kushilevitz and Y. Mansour, "An $\Omega(D \log n/D)$ lower bound for broadcast in radio networks," *SIAM J. Computing*, vol. 27, pp. 702–712, June 1998.
[7] C. Lee, J. E. Burns, and M. H. Ammar, "Improved randomized broadcast protocols in multi-hop radio networks," in *Proc. 1993 Int. Conf. Network Protocols*, San Francisco, CA, pp. 234–241.
[8] I. Chlamtac and S. Kutten, "On broadcasting in radio networks—Problem analysis and protocol design," *IEEE Trans. Commun.*, vol. COM-33, pp. 1240–1246, Dec. 1985.
[9] ———, "Tree-based broadcasting in multihop radio networks," *IEEE Trans. Comput.*, vol. C-36, pp. 1209–1223, Oct. 1987.
[10] I. Chlamtac and O. Weinstein, "Distributed 'wave' broadcasting in mobile multi-hop radio networks," in *Proc. 7th Int. Conf. Distributed Computing Systems*, Berlin, Germany, 1987, pp. 82–89.
[11] E. Pagani and G. P. Rossi, "Reliable broadcast in mobile multi hop packet networks," in *Proc. 3rd Annu. ACM/IEEE Int. Conf. Mobile Computing and Networking (MobiCom'97)*, Budapest, Hungary, pp. 34–42.
[12] P. Merlin and A. Segall, "A failsafe distributed routing protocol," *IEEE Trans. Commun.*, vol. COM-27, pp. 1280–1287, Sept. 1979.
[13] A. Segall and M. Sidi, "A failsafe distributed routing protocol for minimum delay routing," *IEEE Trans. Commun.*, vol. COM-29, pp. 689–695, May 1981.
[14] S. Basagni, *On the Broadcast and Clustering Problems in Peer-To-Peer Networks*. Ph.D. dissertation, Univ. degli Studi di Milano, Milano, Italy, May 1998.
[15] T. H. Cormen, C. E. Leiserson, and R. L. Rivest, *Introduction to Algorithms*. Cambridge, MA: MIT Press, and New York: McGraw-Hill, 1990.
[16] D. Bruschi and M. Del Pinto, "Lower bounds for the broadcast problem in mobile radio networks," *Distributed Computing*, vol. 10, no. 3, pp. 129–135, Apr. 1997.



Stefano Basagni (S'96) received the B.Sc. degree in computer science from the Università degli Studi di Pisa, Pisa, Italy, in 1991, and the Ph.D. degree in computer science from the Università degli Studi di Milano, Milan, Italy, in 1998.

He is currently a post-doctoral Researcher at the Center for Advanced Telecommunications Systems and Services, The University of Texas at Dallas. His research mainly concerns algorithms for network systems, with special emphasis on routing, multicast, and broadcast protocols for wireless networks.

Dr. Basagni was awarded the Texas Public Education Grant in 1997/98 and 1998/99, and the TxTEC Consortium Fellowship in 1998/1999.

Danilo Bruschi received the B.Sc. and Ph.D. degrees in computer science from the Università degli Studi di Milano, Milan, Italy, in 1985 and in 1989, respectively.

In 1990, he spent a year at the University of Wisconsin at Madison. In 1991, he joined the Computer Science Department, Università degli Studi di Milano, where he is currently an Associate Professor. His current research interests include distributed systems, mobile systems, and computer and network security.

Imrich Chlamtac (M'86–SM'86–F'93), for photograph and biography, see p. 9 of the February 1999 issue of this TRANSACTIONS.