On A Self-organizing MANET Event Routing Architecture with Causal Dependency Awareness

Guanhong Pei
Dept. of ECE
Virginia Tech, USA
guanhong@vt.edu

Binoy Ravindran
Dept. of ECE
Virginia Tech, USA
binoy@vt.edu

E.D. Jensen
The MITRE Corporation
Bedford, MA 01730, USA
jensen@mitre.org

Abstract

Publish/subscribe (P/S) is a communication paradigm of growing popularity for information dissemination in large-scale distributed systems. The strong decoupling between information producers and consumers in P/S systems is attractive for loosely coupled and dynamic network infrastructures such as mobile ad hoc networks (MANETs). However, achieving end-to-end timeliness and reliability properties when P/S events are causally dependent is an open problem in MANETs. In this paper, we propose an architecture design for event routing in MANET that can effectively support timely and reliable event delivery with awareness of event causal dependencies. Our design features a two layer structure, including novel distributed algorithms and mechanisms for P/S tree construction and maintenance with self-organization capabilities. Our simulation-based experimental studies illustrate the architecture's effectiveness.

1 Introduction

The P/S paradigm [9] communicates on the basis of either the message content or the message source being of interest to destinations – as opposed to the source specifying the recipient(s). P/S systems can be considered to be a form of event-based systems, in the sense that the information injected to and propagated through the system can be treated as *events*. A unit in the system can act either or both as information producers (*publishers*) and consumers (*subscribers*). The subscribers declare their interests by *subscriptions* to certain events, most commonly specified by the content or the topic of the events (with different expressive power), and publishers produce events of information to the system. The *event routing* mechanism implemented in the P/S system (usually middleware) then takes charge of the event delivery according to the subscription knowledge.

The strong decoupling between information producers' and consumers' identities in a P/S system is appealing in loosely coupled network background such as mobile ad hoc networks (MANETs) [5], because of the ease with which components can be added, removed or changed at runtime. A MANET is a collection of mobile devices with dynamically changing membership and multi-hop topologies composed of wireless links. By its nature, a MANET is a self-organizing adaptive network, and thus needs to be formed and maintained in a distributed manner without centralized support or fixed infrastructures [24]. However, the potential advantages of the P/S interaction model atop MANETs are not fully realized by the state of the art predominant products and research solutions, such as TIB/RV [18], SIENA [10], Gryphon [11], and RTI's DDS [12].

Many of the previous works have focused primarily on throughput and overhead. Timeliness and reliability, which are important for many MANET-based applications have not been well addressed. Furthermore, MANETs are subject to significant run-time uncertainties and resource constraints, unlike traditional fixed-infrastructure networks, including: (1) frequent link breakages and temporary network disconnections; (2) temporary node unavailability and node joins or departures at unpredictable times; and (3) mobility-induced resource constraints on the overall architecture, such as limits on bandwidth and energy consumption.

In addition, an important property of many emerging MANET-based applications that are suited for P/S-style communication is *event causal dependencies*. This refers to the existence of multiple publication and subscription hops that are causally related (e.g., topic-wise), resulting in an event causal chain or event causal graph. We illustrate this through a motivating scenario from the military domain (similar scenarios can be found in other application domains such as automotive [16] and e-health [15] systems).

As illustrated by Figure 1, a commander either makes (i.e., invokes) or publishes a service request to "agents" to find out something about the situation in a particular combat region. Naturally, there is a mission-critical time constraint for him to obtain that information — the information's util-

ity to the mission degrades after a certain amount of time following his service request. To obtain that information:

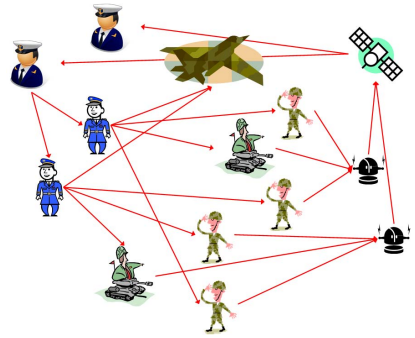


Figure 1. Example Scenario

two forms of imagery are collected by subscription from surveillance platforms, and fused by recipient agents; re-scoped down to the geographical region of interest (perhaps by publishing it to a service which does that); re-sized to fit on soldiers' PDA screen (perhaps by publishing the re-scoped information to a service which re-sizes it); published to, and annotated by, one or more soldiers in that region; then sent by those soldiers to a ground-to-space relay and from there to a satellite and then to the commander making the original service request (and probably other interested officers). Most, if not all, of these communications can effectively be publish/subscribes.

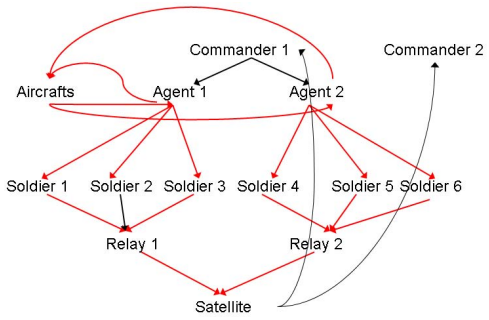


Figure 2. Example Causal Graph

Figure 2 shows the underlying causal relationship graph of the Figure 1 scenario. The timeliness and reliability of the data delivery from the causal event initiator (commander in Figure 1) to the last publisher in the whole chain (satellite in Figure 1), is critical and is challenging due to the MANET dynamics.

This scenario raises fundamental yet unsolved problems: (1) What interconnection architecture is appropriate for P/S service in MANETs with awareness of causal dependencies? (2) How to self-organize mobile nodes into an interconnection topology with support for reliable and timely delivery of causally-dependent messages? (3) How to reliably and timely route causally-dependent messages?

In this paper, we answer these questions. We consider a MANET, where every node in the network has mobility and can access P/S service (e.g., by running P/S middleware). Based on elaborate analysis, we present an event routing architecture called SOMER (Self-Organizing MANET Event Routing architecture) that supports timely and reliable data delivery across causally-dependent event chains. Key aspects of SOMER include algorithms and mechanisms for P/S tree construction and maintenance with self-organization and self-configuration capabilities. Our simulation-based experimental results show that SOMER achieves 30%–100% timeliness improvement and up to 30% reliability improvement over previous solutions.

To the best of our knowledge, this is the first work to consider causally-dependent P/S systems on MANETs. Thus, the paper's contribution is the SOMER architecture.

The remainder of this paper is organized as follows. Section 2 discusses related work and identifies SOMER's unique aspects. Then we first illustrate the advantages of a hierarchical tree-based interconnection through an analytical model in Section 3. Section 4 presents SOMER's basic design. In Section 5, we discuss how we design architectural support for causal awareness. Section 6 reports our experimental results, including an extensive comparison study. Section 7 concludes the paper.

2 Related Work

Typically, in MANETs, a structured interconnection topology is used; most recent representative works [13, 20, 3, 17] present algorithms for constructing and maintaining an event routing tree as the interconnection topology under a filter-based routing scheme. There are also structureless approaches—e.g., [1] employs a form of informed flooding-based event routing based on Euclidean distances.

It is important to note the various approaches used in P/S systems atop wired networks. For example, a distributed hash table configuration is used to facilitate the interconnection topology [22, 25, 6]. Some approaches do not maintain any deterministic data structure on the topology at a peer. In this case, event routing is neither filtering-based nor rendezvous based, like in [7, 8]; they employ probabilistic flooding and gossiping, respectively.

Tree-based vs. Non-tree Approaches. We refer to a *tree-based interconnection topology* as an approach to interconnect mobile nodes with an acyclic structure. We refer to *non-tree approaches* as the ones that use either rendezvous-based or flooding/gossip-based approaches. The fundamental difference between the tree-based topology and the non-tree approaches is the interconnection topology of non-tree approaches tends to poorly match the underlying physical

topology, because of the loose coupling among the nodes. The impact on the message delivery is that latency can be very high and sometimes unpredictable as information might pass across many nodes some of which might be slow as well as have long physical paths between them in the underlying network.

Also, large periods of message flooding may occur in some peer-to-peer networks which can cause congestions and inefficient use of capacity. By contrast, a tree-based topology can more efficiently and effectively meet the end-to-end timeliness requirement [14, 4]. However, in order to meet the demands of large scale MANETs, we need a more effective architecture to handle node failures, link breakage, node joins and departures, while still maintaining the same or increased timely message delivery.

Existing MANET Tree-based P/S Architecture. To the best of our knowledge, for MANETs, most P/S-tree-based interconnection approaches employ a single P/S tree structure in the whole network or build one P/S tree for each publisher. The latter case can only accommodate a limited number of designated publishers and suffers from the lack of scalability. The single-tree strategy is typically used; and due to the resemblance between multicast and P/S in MANETs, in [17] the P/S tree is derived from multicast tree construction and maintenance mechanisms of MAODV (Multicast Ad hoc On-Demand Distance Vector Routing Protocol) [23]. In large scale MANETs, as the average distance increases between the root and a node in the P/S tree, it becomes much harder to handle failures and topology changes. Moreover, although the single-tree architecture is fully based on the geographical topology, there still exists a possibility that a publisher's message may not reach some of its subscribers via the physically shortest route, and the length of the path is hard to be bounded and is independent of the length of the physically shortest route.

A hierarchical tree-based interconnection in MANET self-organizes all the nodes into non-overlapping trees and has specific mechanisms for inter-tree communication. The roots of the trees function as the inter-tree brokers. In Section 3, we will show the advantages of a hierarchical tree-based interconnection topology over a single P/S tree approach through both qualitative and quantitative analysis.

Motivated by all the above observations, we consider a tree-based approach and explore a hierarchical architecture for causally related event delivery. The idea of using hierarchical architecture in ad hoc networks is not new. For example, [21] presents a scalable P/S system called SensTrac based on a tree-based hierarchical structure. SOMER is uniquely novel and different from [21] in several aspects including: (1) *Tree Construction*. SensTrac uses static clustering via nodes' geographic coordinates using GPS. In contrast SOMER uses dynamic clustering allowing nodes to be clustered on-the-fly; (2) *Inter-tree Communication*. SensTrac uses AODV to find routes among root nodes (leaders), and use gossip to disseminate information among root nodes, whereas SOMER uses flooding among root nodes which is more suitable for a mobile environment; (3) In SensTrac, the query node (subscriber) subscribe to the info of its area of interest (AOI) which is bounded by a given square. In SOMER, the subscribers' subscription is more general; (4) In SensTrac, the query node (subscriber) subscribe to the information of its arera of interest (AOI) which is bounded by a small square, whereas our subscription model is more general; (5) In SensTrac, the query node is the only subscriber, which does not act as a broker, and the publishers are only the sensors in the AOI, whereas we do not have limitations on subscribers or publishers; (6) In SensTrac, nodes are stationary and only the query node can move, whereas we allow every unit to be mobile; and (7) SensTrac does not support the causal dependencies. (its static node clustering mechanism has no similar capability to handle causal dependencies as that of SOMER).

3 Hierarchical Tree-based Model

Suppose that the system (i.e., a MANET) has already completed self-organization and remains stationary. To build an analytical model from which we get our first motivations, we make the following assumptions: (1) All nodes are homogeneous and evenly distributed within a 2-dimensional area; and (2) Each node has the same number of immediate neighbors and thus each tree has the same number of immediate neighboring trees. Let the number of neighbors of any node be $N_{Neighbor}$.

Graph Theory Preliminaries. An r-tree is a tree with *root* r. Let $T(r)$ denote such a tree. The *level* of a vertex v in $T(r)$ is the length of the path rTv. Each edge of $T(r)$ joins vertices on consecutive levels, and it is convenient to think of these edges as being oriented from the lower to the higher level, so as to form a *branching*. Each vertex except r on the path rTv, is called an *ancestor* of v, and each vertex of which v is an *ancestor* is a *descendant* of v. Two vertices are *related* in T if one is an ancestor of the other. The immediate ancestor of v is its *predecessor* or *parent*, denoted $p(v)$, and the vertices whose *predecessor* is v are its *successors* or *children*. A *leaf* of a tree is the node who has no successors. We refer to the process of going from a non-root node to the root by way of its parent as "going upward," and going the reverse way as "going downward."

Suppose $N_{Neighbor}$ is subject to a normal distribution with mean 6 and standard deviation of 4. Obviously, only 4 and 6 are the possible number of neighbors that can produce an evenly distributed physical topology. E.g., if $N_{Neighbor} = 5$, we cannot find a topology to maintain the equality of the distances between any two neighboring

nodes, thus violating the even distribution assumption.

$$N_{Neighbor} = 4, 6 \tag{1}$$

The network can be represented by a connectivity graph $N(V, E, P)$, where $V = \{v_1, ..., v_n\}$ is the set of nodes, E is the set of links, and $P = \{p_1, ..., p_k\}$ is the set of publishers. Also, let n denote the number of nodes and k denote the number of publishers. Let D_T denote the density of root nodes of all the nodes and $T = \{t_1, ...t_t\}$ denote the set of trees. Let the longest distance from the tree root to a node in its territory be R (i.e., the radius). Let the distance between any pair of neighboring nodes be denoted as $dist_{Min}$. We assume that $0.5dist_{CST} < dist_{Min} < dist_{CST}$, where $dist_{CST}$ is the one-hop wireless transmission range (carrier sensing range), such that a node should only receive the signal from an immediate neighbor.

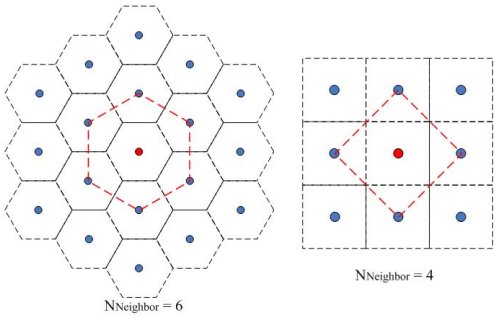


Figure 3. Network Model Structures

From Equation 1, we can derive that the network connection links form either a grid-like ($N_{Neighbor} = 4$) or a beehive-like ($N_{Neighbor} = 6$) structure. Figure 3 illustrates this, where the solid dots denote nodes in the network, and the red hexagons and squares with nodes at the angles cover the set of immediate neighbors of the center node.

Assume that a straight path can be established between any pair of neighboring trees' roots. It can be noticed that for any root node to reach another root node, if their trees are not neighboring, it needs a route by way of other root nodes, and this route takes at most one turn along the route. Let β denote the angle of the turn if there is one. Now,

$$\beta = \begin{cases} 120^o, \text{ for } N_{Neighbor} = 6 \\ 145^o, \text{ for } N_{Neighbor} = 4 \end{cases} \tag{2}$$

Let $s(p_i, v_j)$ be one of $p_i \in P$'s subscriber which is associated to the tree rooted at $v_j \in V$. We estimate the average distance from a node to its tree root to be $\frac{R}{2}$. Now, the length of the path between p_i and $s(p_i, v_j)$ is:

$$\begin{aligned} &length(p_i, s(p_i, v_j)) = length(p_i, r(p_i)) + \\ &\quad length(r(p_i), v_j) + length(v_j, s(p_i, v_j)) \\ &= \tfrac{R}{2} + length(r(p_i), v_j) + \tfrac{R}{2} \\ &= R + length(r(p_i), v_j) \end{aligned} \tag{3}$$

Let $dist(u, v)$ denote the straight-line distance between two nodes u and v. We can bound the length of the path between p_i and $s(p_i, v_j)$ as:

$$\begin{aligned} &length(r(p_i), v_j) \leq \tfrac{dist(r(p_i), v_j)}{|\sin\beta|} \Rightarrow length(p_i, s(p_i, v_j)) \\ &\leq \begin{cases} R + \frac{2}{\sqrt{3}} dist(r(p_i), v_j), \text{ for } N_{Neighbor} = 6 \\ R + \frac{2}{\sqrt{2}} dist(r(p_i), v_j), \text{ for } N_{Neighbor} = 4 \end{cases} \end{aligned} \tag{4}$$

For a large-scale MANET, $dist(p_i, s(p_i, v_j))$ is approximately $dist(r(p_i), v_j)$. Therefore,

$$\begin{aligned} &length(p_i, s(p_i, v_j)) \\ &\leq \begin{cases} R + \frac{2}{\sqrt{3}} dist(p_i, s(p_i, v_j)), \text{ for } N_{Neighbor} = 6 \\ R + \frac{2}{\sqrt{2}} dist(p_i, s(p_i, v_j)), \text{ for } N_{Neighbor} = 4 \end{cases} \\ &= R + \frac{2\sqrt{2}}{\sqrt{N_{Neighbor}}} dist(p_i, s(p_i, v_j)) \end{aligned} \tag{5}$$

Path Overhead Ratio. We define the *path overhead ratio* (POR) as the extra number of hops required by the path over that of the straight line distance. The number of hops is proportional to the path length, and hence:

$$\begin{aligned} POR_{SOMER} &\leq \tfrac{length(p_i, s(p_i, v_j)) - dist(p_i, s(p_i, v_j))}{dist(p_i, s(p_i, v_j))} \\ &= \tfrac{2\sqrt{2}}{\sqrt{N_{Neighbor}}} + \tfrac{R}{dist(p_i, s(p_i, v_j))} - 1 \end{aligned} \tag{6}$$

By applying this POR in a large-scale network where $R \ll dist(p_i, s(p_i, v_j))$, we obtain:

$$POR_{SOMER} \leq \frac{2\sqrt{2}}{\sqrt{N_{Neighbor}}} - 1 \tag{7}$$

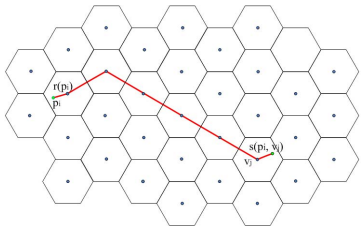


Figure 4. Advantage over Single-tree Model

Figure 4 shows the path from p_i to $s(p_i, v_j)$ with red lines. In the figure, the blue solid nodes are root nodes and the solid hexagons denote the abstract territories of the trees rooted at the center nodes.

This analysis shows that for the hierarchical tree-based architecture, POR can be bounded and is small ($\sqrt{2} - 1$ or $\frac{2}{\sqrt{3}} - 1$); whereas for single-tree-based and structure-less approaches, the length of the path cannot be bounded and sometimes could be very high. Thus, the benefits of the hierarchical tree-based architecture include the following: (1) event delivery can be completed in a shorter and more deterministic time; (2) short and bounded path length implies lower probability for a delivery to fail, despite frequent

link breakages and node failures; and (3) the multi-tree structure gives enough leeway to devise multi-path event delivery schemes through the use of multiple neighboring trees and multiple border leaf nodes among the trees.

4 Basic Architecture Design

SOMER's design employs the hierarchical tree-based interconnection as shown in Figure 5.

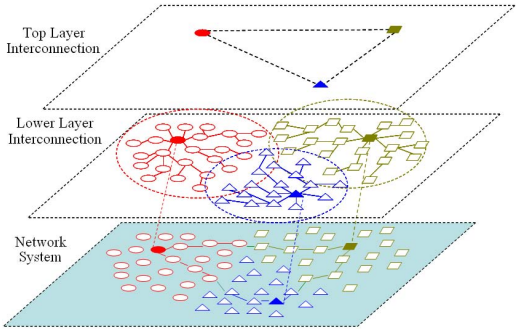


Figure 5. Architecture Overview

At the lower layer (*inner-tree level*), nodes are self-organized into multiple P/S trees rooted at several *root nodes* across the network. At the top layer (*inter-tree level*), root nodes act as super publisher/subscriber nodes and constitute an overlay network.

4.1 Inner-tree Level Interconnection

The *publish/subscribe tree* (PST) is constructed distributively in a request/reply fashion. Any node v_0 that has not joined a PST broadcasts the *join request* message and waits for its immediate neighbors that are currently in a P/S tree to reply with a *join reply* message. After a timeout for the neighbors' reply expires, if the node gets any replies, it greedily selects the best candidate neighbor from the replies as its parent according to the *parent evaluation metric* (PEM); otherwise, it continues to broadcast new *join request* messages. Each node in the system can only join one PST.

Inner-tree Subscription and Publication. Each node v_0 has its own subscription interest $s(v_0)$, termed *inherent subscription*. On joining the tree, node v_0 sends a *subscription* message containing $s(v_0)$ upward toward the *grafting node* for each of its ancestors to match, and subsequently publish information to v_0, which is now a leaf in that tree. We define a non-leaf node u_0's effective subscription $S(u_0)$ as the "combined" subscription formed by merging u_0's inherent subscriptions with all of its descendants' subscriptions. Thus, the grafting node of node v_0 is the closest node (in terms of number of hops) in the upward path, whose effective subscription overlaps with that of v_0. In this way, each non-leaf node maintains data structures for its successors' subscription interests, and in u_0's parent $p(u_0)$'s view, u_0 is a child that has the subscription interest of $S(u_0)$.

Parent Evaluation Metrics (PEM). PEM is the key mechanism for constructing and maintaining the PST. There are various PEMs, including: (1) Shortest-path Metric (SM); and (2) Publication-overhead-aware Metric (PM).

SM is a most straightforward metric, in which a node chooses the neighbor at lowest level as its parent. PM usually makes use of the subscription information of the other nodes in the tree to decrease the total number of messages used to complete the delivery of one publication. [13] proposes a PM that requires a' priori knowledge of the exact rates of invocation (publication) for each category of subscription; the lack of that knowledge will cause high inaccuracy while the rates of invocation can vary over time or even can be random as is the case in our motivating scenarios. The PM in [3] requires *periodic* messages to go upwards in the tree through a probabilistically chosen candidate parent until it reaches a new grafting node. However, in a large-scale network where subscription interests vary largely, especially for a content-based system, finding a grafting node may require traversing a long distance and incur much higher overhead than SM. In [3], the authors claim an advantage of less "overhead" over MAODV-based PST approach. However, by "overhead," they only include the messages used for publication without considering the considerable overhead incurred for a better path detection initiated periodically by each node other than the root or level-1 nodes.

Based on these observations, we choose SM, like the MAODV-based PST approaches and [21]. Obviously, SM inherently favors the timing property of event delivery.

4.2 Inter-tree Level Interconnection

A natural question that arises for this distributed architecture is that which nodes should be the roots, given no centralized administration. Another question is how the inter-tree communication mechanism is designed to facilitate event routing.

4.2.1 Root Node Selection and Tree Merging

A node's role (either root or non-root) in the system is designated when the P/S application is initiated.

Root Selection. We use a random-number-based strategy for distributed root selection. Recall that D_T denotes the density of roots. In fact, D_T has a significant influence on the performance of the architecture, as shown in Section 6. Given a range of the network size of a MANET, we can select the best D_T based on experimental studies, which is reasonable. A uniform random number $Rand$ within the

range of 0 to $Rand_{Max}$ is produced by the P/S middleware on each node. For a node v, that is:

$$\begin{cases} \text{If } 0 < Rand(v) \leq D_T \cdot Rand_{Max}, v \text{ is a root node} \\ \text{Otherwise, } v \text{ is a non-root node} \end{cases}$$

Tree Merging. To deal with the degradation of performance caused by roots that lie geographically too close (e.g., one or two hops away from each other), a *tree merging* mechanism is employed. A commonly used tree maintenance mechanism is a periodic *refresh* message broadcast with a sequence number, from the root throughout the tree. Every node rebroadcasting this message replaces the *level value* field with its own level value. Whenever a node v_0 receives a foreign tree node u_0's *refresh* message, it calculates the root-to-root distance between $r(v_0)$ and $r(u_0)$ from the level values of v_0 and u_0. If the root-to-root distance falls below a threshold $Merging\_Thres$, either v_0 or u_0 sends a *merging request* message to the root, and the merging process is initiated. The tree merging process is similar to the partition merging in MAODV [23], in which the root with a higher ID remains its role and takes over the other tree.

New Root Node Selection. New root node selection is required based on the following considerations. To better accommodate MANET dynamics, it is important to observe that the topology changes due to node mobility and failures can cause the distance among the roots to change. Such dynamics together with *tree merging* can cause the number of trees (i.e., the number of root nodes) to decrease. Thus, schemes must be designed for handling root node failures, mobility, and for controlling the tree sizes to be below a certain upper threshold. The basic idea is hence to select a new root to overcome those adverse conditions.

When a node v's level is above a certain predefined threshold New_Root_Thres, or it has not been a member of any tree for a certain time frame Out_Period, the node checks the possibility of advancing itself to a root. A new random number will be produced raising the probability of the node being accepted as a root. After the new root's "birth", it broadcasts invitations to "crop" tree members from other trees, until the level value of its leaf nodes is no smaller than that of at least one of the leaf nodes' neighboring foreign nodes.

Due to space constraints, we omit detailed procedural-style descriptions of the algorithms for the above strategies; these can be found in [19]. Note that in a reliable wireless network with little topology change, our tree merging and new root selection strategies for system maintenance would hardly introduce any additional traffic overhead.

4.2.2 Inter-tree communication

Inter-tree Route Establishment. Similar to the mechanism for tree merging, whenever a node v_0 receives a foreign tree node u_0's *refresh* message, it calculates the root-to-root distance between $r(v_0)$ and $r(u_0)$ from the level values of v_0 and u_0. If the new root-to-root distance is smaller than the current value, node v_0 records u_0 as the next hop destination for inter-tree routing, and send a *route report* message upwards. On receiving the route report message, v_0's parent $p(v_0)$ checks its local root-to-root distance value. If the new value reported is better, $p(v_0)$ will also update its record and send a route report message upwards. In this way, the root will always keep the most fresh feasible route for inter-tree communication. Further, a node only registers the next hop destination and the corresponding root-to-root distance for inter-tree routing, requiring only small memory usage. When u_0 receives an inter-tree message (e.g., a publication or notification message) from a foreign node, u_0 will simply forward this message upward to its root. Now, the route established between two roots of the neighboring trees is the shortest. We omit the proof due to page constraints; it can be found in [19].

Inter-tree Overlay Event Routing. *Advertisement* messages are widely used in P/S systems. On joining a P/S tree, a node sends both its subscription and publication interests to the root nodes. From the top layer view, root nodes can be treated as super publishers/subscribers, with the *effective subscription* and the *effective publication* capabilities. Event routing at the inter-tree level is based on periodic advertisement messages flooding across root nodes such that the root nodes' effective P/S interests are propagated across the top layer. Though the flooding among root nodes incurs message overhead, it is worth it when node mobility is high. (We show in [19] that the overhead is reasonable.) When a root node r_0 receives an advertisement message forwarded by a neighbor root r_1 originally from r_2, r_0 updates r_2's P/S interests and the inter-tree route entry for r_2, and updates the corresponding next-hop root-address with r_1's address. In this way, the inter-tree routing uses the shortest reverse path to effectively propagate event notifications.

5 Self-reconfiguring Architectural Facilitation for Event Causal Dependencies

Self-Construction of Causal Graph. As the example in Section 1 shows, the causal graph is a graph comprised of causally-related nodes of the P/S system. The causal relationship between two nodes are established by the causal dependency of their publication interests. The two necessary conditions for a node a to be a "causal child" of node b are: (1) a subscribes to b's publication; and (2) a's receipt of b's event message is the sufficient condition for a to publish. In turn, b is a's "causal parent". To simplify the problem, we assume that there is at most one "causal parent" for each causally-related node, and none of a node's "causal descendants" can be its "causal parent". Hence, the causal graph

boils down to a causal tree [1] or a causal forest with disjoint causal trees. Now, advertising the subscription/publication interests through the system enables the root nodes to build the causal graph for the whole system.

Self-reconfiguration with Awareness of Event Causal Dependencies. The original physical trees stop their natural growth when each node has joined a tree. The tree merging and new root selection strategies confine the tree sizes with $Merging\_Thres$ and New_Root_Thres. However, this may not be the optimal solution for a system with event causal dependencies. We observe that, a physical tree of a comparatively larger size typically has more neighboring trees, implying increasing possibilities to establish direct inter-tree connection with more other trees. Consequently, the probability increases for a causally-involved node to get to its causal children through fewer intermediate forwarding nodes.

One tree's expansion means other neighboring trees' contraction, as if trees are "fighting" for nodes which can only belong to one tree at any given time. The metric for trees' "fighting" must be carefully designed. The goal is to shorten the event notification time along the P/S causal chains that originate at the "causal root" until the "causal leaves".

For a given physical tree, if it contains more "causal nodes" at lower levels in the causal tree, or a larger number of causal nodes, it should be endowed with more credits for physical expansion. That is because, lower level causal nodes usually influence more subsequent event notification deliveries that directly or indirectly depend on them in the causal chain. For instance, in Figure 1, any delay that occurs for the event notification delivery between the commander and the agents will have an impact on every subsequent causal chain, whereas the soldier-relay link will only have an influence on one causal chain.

Furthermore, for the causal nodes covered by a physical tree, if a large portion of their causal descendants are already covered by the same tree, then it is quite possible that the tree's expansion may not facilitate timely and reliable the event delivery in the causal tree.

Here, we define a tree's *expansion potential* (EP) as the metric to evaluate the potential benefit out of a tree's expansion for event delivery along the causal tree. For a physical tree T, let $N_{Causal}(T)$ be the number of causal nodes that T covers, $Avg\_lev_{Causal}(T)$ be the average level value in the causal tree of the causal nodes covered by T, and γ be the percentage of causal descendants that is covered by T for all the causal nodes in T. Now, $EP(T)$ is given by:

$$EP(T) = \frac{(N_{Causal}(T(v)) + \eta)}{(Avg\_lev_{Causal}(T) + \eta) \cdot (\gamma + \eta)}. \quad (8)$$

[1] To avoid confusion, we always say "causal tree" when we refer to one.

Here, η is a parameter with a constant positive value to offset the variables that could be zero.

It is the leaf nodes that actually perform the expansion (i.e., fighting) through periodically requesting neighbor foreign nodes for a comparison of each other's *local expansion potential* (LEP). Because we need to also constrain the tree's branches from abnormal growth, a node v's LEP is:

$$\begin{aligned} LEP(v) &= \frac{EP(T(v))}{(level(v)+\eta)} \\ &= \frac{(N_{Causal}(T(v))+\eta)}{(Avg\_lev_{Causal}(T(v))+\eta)\cdot(\gamma+\eta)\cdot(level(v)+\eta)} \end{aligned} \quad (9)$$

For two nodes u and v involved in such a fighting, the one with the higher LEP will win, and the loser will join the winner tree as a child of the winner. It is possible that a node may constantly change its affiliation. To avoid that situation, we stop two nodes from fighting when:

$$1 - \delta \leq \frac{LEP(v)}{LEP(u)} \leq 1 + \delta \quad (10)$$

Here, δ is a parameter to tune the fighting severity.

The result of the fighting may change the size of a tree. The fighting is allowed to take place only when the merging threshold or new root selection threshold is not violated.

6 Experimental Evaluation

We conducted sets of simulation experiments using NS-2 with the Random Trip Mobility Model package [2] to generate sets of random MANET topologies.

Our simulation environment is built with each node having its own subscription interest. A randomly selected set of nodes act as publishers. We used the model of Number Intervals [13] for P/S pattern generation. We used a large number interval as the interest pool, and a node's subscription interest is represented by a random subset of the interest pool. We call it a match when the number associated with a published event falls into the range of a node's interest. Detailed simulation settings can be found in [19].

6.1 Influence of D_T

As mentioned in Section 4, D_T is a parameter that can affect SOMER's performance.

We evaluated this through experiments. We deactivated the tree merging and new root selection strategies to ensure that the number of roots does not change during the experiments. Using *delivery time* as the performance metric, we measured the time cost in 250-node networks for both single-publish-hop and complete causal graph event delivery. We normalized the delivery time with respect to the minimum value in each case, so that the trend can be clearly observed.

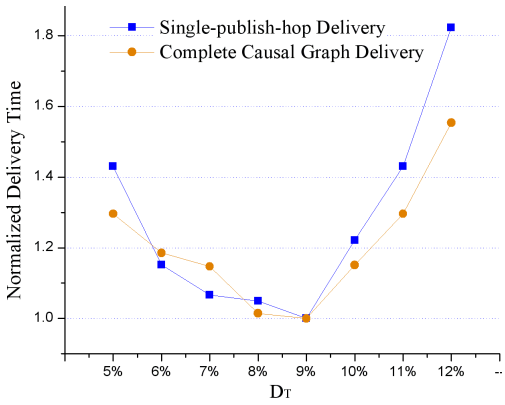


Figure 6. Timeliness Performance vs. D_T

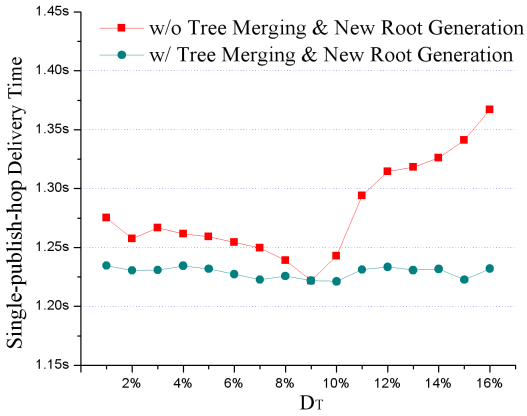


Figure 7. Effectiveness of Tree Merging and New Root Selection

As expected, the curves in Figure 6 exhibit identical trend. Further, at $D_T = 9\%$, both curves reach the minimum value, indicating that 9% is the best value for D_T under current simulation modularity. When D_T decreases from 9%, the time cost increases because there would be fewer root nodes which act as brokers in the top-layer overlay, degrading the efficiency of the inter-tree routing. On the other hand, as the number of trees increases, a publisher and one of its subscribers in the same tree could be separated to be in two different trees. Consequently, The event messages from the publisher would have to make a detour through at least two root nodes to reach the subscriber, in contrast with the fact that the messages only need to go through one root node if they were in the same tree. For a given network topology, we can find the best D_T. However, doing so requires a' priori knowledge of the system. We will show in the following results that we can avoid calculating the best D_T.

6.2 Tree Merging/New Root Selection

Recall that through the tree merging and new root selection strategies, the system interconnection is self-reorganized, thereby optimizing the tree distribution toward timely event delivery. From Figure 7, we observe that after these strategies are used, the average time cost for single-publish-hop event delivery is effectively lowered and the system gives a stable performance as the initial density of root nodes (D_T) varies all the way from 1% to 16%. (We used 200-node networks for these experiments.) This implies that our strategies are stable. The decrease in the event delivery time is due to the new root selection strategy when the initial D_T is small. Furthermore, it is the tree merging strategy that reduces the delivery time while the initial D_T is comparatively large.

6.3 Event Delivery with Awareness of Causal Dependencies

To evaluate the effectiveness of SOMER's causal awareness design, we performed two sets of experiments via: (1) deactivating the tree merging and new root selection mechanisms to observe the "pure" performance gain from system self-reconfiguration for causal event delivery; and (2) reactivating those two mechanisms, and evaluating the performance gain with every part of SOMER working together. Here, by performance, we mean the timeliness for the event delivery across a causal tree. We omit the results for reliability performance due to space constraints; these can be found in [19].

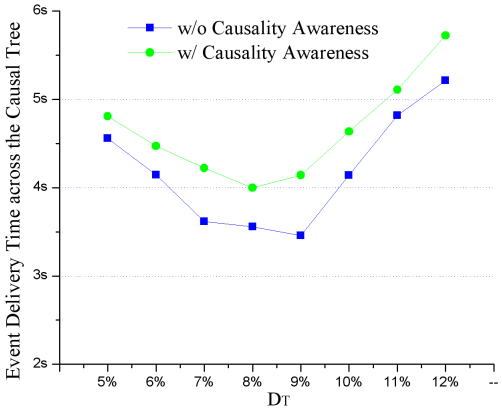
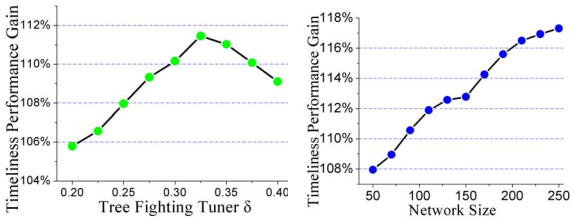


Figure 8. Improvement from Causal Event Delivery Architectural Support

Figure 8 is produced by the first set of experiments with 250-node network scenarios. We observe that our design yields as much as over 15% improvement at around the best D_T (9%). In the second set of experiments, we evaluate the timeliness gain by varying the network size and the tree

fighting severity tuning parameter δ.

We first explored the space of δ by obtaining the average performance gain with network sizes increasing from 50 to 250 for a given value of δ. From Figure 9(a), we observe a peak performance gain around $\delta = 0.325$. This gain reduces when δ increases due to the degradation of tree fighting severity such that the system has less self-reconfigurability for causal event delivery. The gain also reduces as δ decreases because the overhead incurred by severe tree fighting adds to the system instability and thus partially negates the performance gain.



(a) Timeliness Gain vs. δ (b) Timeliness Gain vs. Network Size

Figure 9. Timeliness Gain

Figure 9(b) illustrates that SOMER achieves around 10% performance gain, and higher when the network size scales up. This trend corresponds to Equation 6, which states that the upper bound of the path overhead ratio usually decreases for larger size networks. Thus, our experiments illustrate that SOMER's design is scalable and effective toward reducing the total event delivery time across a causal tree.

6.4 Timely and Reliable Event Delivery

We compared SOMER's timeliness and reliability against significant past MANET P/S works including: (1) SP-COMBO [13], a PST protocol with a combination of shortest path and publication-overhead-aware metrics; (2) DSAPST [3], a distributed subscription-aware PST protocol; and (3) PS-MAODV [17], a MAODV-based PST protocol. The network size was varied with a constant D_T in the experiments. We omitted SensTrac from our comparison due to the significant differences between SensTrack and SOMER as described in Section 2. Note that none from the past work addresses event causal dependencies.

Timeliness. We measured the timeliness performance by reciprocating the measured time cost for event delivery. Then we normalized each protocol's performance to the worst performance. In this way, we can also observe the trend of relative performance as the network size changes.

Figure 10 illustrates SOMER's advantage in timely delivery over others. DSAPST performs the worst because, the way it constructs the PST is mainly based on a publication-overhead-aware metric as discussed in Section 4.

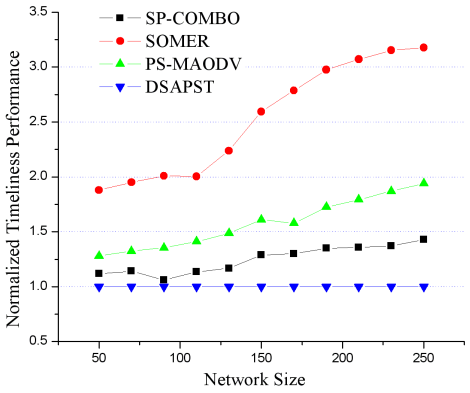


Figure 10. Comparison for Timely Delivery

By normalizing with respect to DSAPST's performance, we clearly observe SOMER's performance gain over others. SOMER outperforms PS-MAODV, which is the second best, with at least a 30% improvement. As the network size scales up, SOMER's improvement over PS-MAODV increases up to around 100%, illustrating SOMER's superior scalability. This is due to increasing number of trees that can be used as top-layer brokers to make the route closer to the straight line between two nodes.

Reliability. We measured reliability through *delivery ratio*, which is calculated as the number of copies of event messages successfully received by their subscribers over the number of copies of event messages that should arrive at all their subscribers, given no failures or no topology changes. We used node failure rate as a variable to assess system reliability, with node roaming speed randomly ranging from 0 to 10m/s. Figure 11 shows that SOMER can survive a 12% node failure rate with over 75% event delivery ratio. The highest performance improvement of SOMER is over 10% higher delivery ratio than that of PS-MAODV.

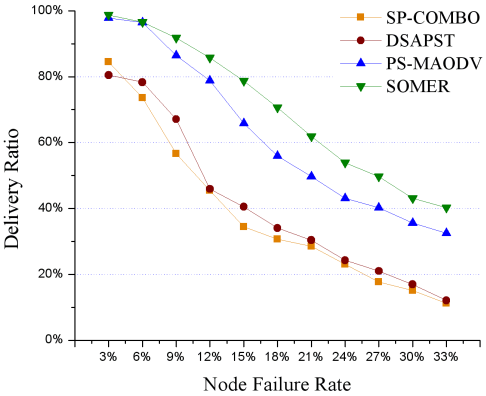


Figure 11. Delivery Ratio vs. Node Failure Rate

Note that SOMER also provides 30% higher delivery ra-

tio over the other two rivals. SOMER's improvements are due to its inherent distributed multi-tree structure, and also due to its tree merging and new root selection strategies that effectively counter network unreliability.

Network Traffic Load. Among PS-MAODV, DSAPST and SP-COMBO, only PS-MAODV had relatively acceptable timeliness and reliability. Thus, we compared SOMER and PS-MAODV with respect to the network traffic load incurred under the same simulation scenarios and publication patterns. We observe that SOMER outperforms PS-MAODV under a comparably high P/S traffic load, while SOMER only spends slightly more P/S messages (e.g., about at most 10%) than PS-MAODV. Detailed experiment results and analysis can be found in [19].

7 Conclusions and Future Work

In this paper, we presented SOMER, a self-organizing MANET event routing architecture with awareness of event causal dependencies. Our design facilitates the event delivery with timeliness and reliability properties. We elaborate the issues of event causal dependencies in event-based systems, and develop architectural support for the causally related event deliveries across the causal trees. Effective strategies and mechanisms for P/S system self-organization and self-reconfiguration are proposed motivated by an analytical model. Our simulation experiments illustrate significant improvement over previous work.

Future work includes extending the architectural support to allow arbitrary causal graphs (i.e., allowing multiple causal parents), and integrating real-time scheduling strategies and multi-path schemes into SOMER to further boost timing and reliability properties of causally-related event delivery. Further study on structure-less topology architectures as alternatives to SOMER with experiments and understanding their tradeoffs are also interesting directions.

References

[1] R. Baldoni, R. Beraldi, G. Cugola, M. Migliavacca, and L. Querzoni. Structure-less content-based routing in mobile ad hoc networks. In *IEEE International Conference on Pervasive Services*, pages 37–46, 2005.

[2] J.-Y. L. Boudec and M. Vojnovic. Perfect simulation and stationarity of a class of mobility models. In *IEEE INFOCOM*, pages 2743–2754, March 2005.

[3] X. Cao and C.-C. Shen. Subscription-aware publish/subscribe tree construction in mobile ad hoc networks. In *IEEE ICPADS*, pages 1–9, November 2007.

[4] A. Carzaniga and C. P. Hall. Content-based communication: a research agenda. In *International Workshop on Software Engineering and Middleware*, pages 2–8, November 2006.

[5] S. Corson and J. Macker. *Routing Protocol Performance Issues and Evaluation Considerations (RFC 2501)*. Network Working Group, 1999.

[6] P. Costa and D. Frey. Publish-subscribe tree maintenance over a dht. In *DEBS*, pages 414–420, 2005.

[7] P. Costa, M. Migliavacca, G. P. Picco, and G. Cugola. Epidemic algorithms for reliable content-based publish-subscribe: An evaluation. In *ICDCS*, pages 552–561, 2004.

[8] A. Datta, S. Quarteroni, and K. Aberer. Autonomous gossiping: A self-organizing epidemic algorithm for selective information dissemination in wireless mobile ad-hoc networks. In *International Conference on Semantics of a Networked World*, pages 126–143, 2004.

[9] L. Fiege, G. Muhl, and P. R. Pietzuch. *Distributed Event-based Systems*. Springer-Verlag B&H, 2006.

[10] http://www.inf.unisi.ch/carzaniga/siena/.

[11] http://www.research.ibm.com/gryphon/.

[12] http://www.rti.com/products/data_distribution/RTIDDS.html.

[13] Y. Huang and H. Garcia-Molina. Publish/subscribe tree construction in wireless ad-hoc networks. In *IEEE MDM*, pages 122–140, 2003.

[14] M. Junginger and Y. Lee. A self-organizing publish/subscribe middleware for dynamic peer-to-peer networks. *IEEE Network*, 18(1):38–43, 2004.

[15] E. Lupu, N. Dulay, M. Sloman, J. Sventek, S. Heeps, S. Strowes, S. Strowes, K. Twidle, K. Twidle, S.-L. Keoh, and A. Schaeffer-Filho. Amuse: autonomic management of ubiquitous e-health systems. *Concurr. Comput.: Pract. Exper.*, 20(3):277–295, 2008.

[16] E. R. B. Marques, G. M. Goncalves, and J. B. Sousa. The use of real-time publish-subscribe middleware in networked vehicle systems. In *1st IFAC Workshop on Multivehicle Systems*, 2006.

[17] L. Mottola, G. Cugola, and G. P. Picco. A self repairing tree topology enabling content-based routing in mobile ad hoc networks. *IEEE Transactions on Mobile Computing*, February 2008.

[18] B. Oki, M. Pfluegel, A. Siegel, and D. Skeen. The information bus - an architecture for extensive distributed systems. In *ACM SOSP*, 1993.

[19] G. Pei, B. Ravindran, and E. D. Jensen. On a self-organizing manet event routing architecture with causal dependency awareness. Technical report, ECE Dept., Virginia Tech, 2008. Available at: `http://www.real-time.ece.vt.edu/saso08-tr.pdf`.

[20] G. P. Picco, G. Cugola, and A. L. Murphy. Efficient content-based event dispatching in the presence of topological reconfiguration. In *IEEE ICDCS*, pages 234–243, May 2003.

[21] S. Pleisch and K. Birman. Senstrac: Scalable querying of sensor networks from mobile platforms using tracking-style queries. In *IEEE MASS*, pages 306–315, October 2006.

[22] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Schenker. A scalable content-addressable network. In *SIGCOMM*, pages 161–172, 2001.

[23] E. M. Royer and C. E. Perkins. *Multicast Ad hoc On-Demand Distance Vector (MAODV) Routing (INTERNET DRAFT)*. Mobile Ad Hoc Network Working Group, 2000.

[24] S. K. Sarkar, T. Basavaraju, and C. Puttamadappa. *Ad Hoc Mobile Wireless Networks: Principles, Protocols and Applications*. Auerbach, 2007.

[25] I. Stoica, R. Morris, et al. Chord: a scalable peer-to-peer lookup protocol for internet applications. *IEEE/ACM Transactions on Networking*, 11:17–32, Februrary 2003.