Cluster-Based Failure Detection Service for Large-Scale Ad Hoc Wireless Network Applications

Ann T. Tai Kam S. Tso
IA Tech, Inc.
10501 Kinnard Avenue
Los Angeles, CA 90024

William H. Sanders
Coordinated Science Laboratory
University of Illinois
Urbana, IL 61801

Abstract

The growing interest in ad hoc wireless network applications that are made of large and dense populations of lightweight system resources calls for scalable approaches to fault tolerance. Moreover, the nature of these systems creates significant challenges for the development of failure detection services (FDSs), because their quality often depends heavily on reliable communication. In particular, ad hoc wireless networks are notoriously vulnerable to message loss, which precludes deterministic guarantees for the completeness and accuracy properties of FDSs. To meet the challenges, we propose an FDS based on the notion of clustering. Specifically, we use a cluster-based communication architecture to permit the FDS to be implemented in a distributed manner via intra-cluster heartbeat diffusion and to allow a failure report to be forwarded across clusters through the upper layer of the communication hierarchy. In doing so, we extensively exploit the message redundancy that is inherent in ad hoc wireless settings to mitigate the effects of message loss on the accuracy and completeness properties of failure detection. As shown by our mathematical analysis, the resulting FDS is able to provide satisfactory probabilistic guarantees for the desired properties.

1 Introduction

Recent advances in MEMS (micro-electro-mechanical systems) and wireless networking technologies have led to a growing interest in applications that are made of large and dense populations of lightweight, inexpensive system resources. Examples of such applications include air-dropped sensor networks, smart-dust devices, and micro-UAV or nano-satellite swarms. Not surprisingly, those applications will be built over ad hoc wireless networks, since it is impossible for them to have any fixed communication infrastructure. In addition, such applications are often mission-critical. For example, sensor networks are deployed to detect potential threats to homeland security, to support crisis management, or to help natural disaster relief and recovery [1, 2]. However, due to their large sizes, lightweight components, and inhospitable operational environments, such systems are particularly vulnerable to failures. Accordingly, given the unattended nature of such systems, it is crucial that the operation team be kept updated on the network's health. Such information could offer early warnings of system failure (e.g., a significant number of lost resources may suggest an imminent system capacity exhaustion) and would aid in maintenance scheduling for the deployment of additional resources to the field to preserve system capacity.

Failure detection services (FDSs) are thus important for ad hoc wireless network applications that are built on large and dense populations of lightweight resources. Nonetheless, the development of such services leads to greater challenges than those involved in the development of failure detectors for traditional distributed systems. Specifically, the large size and high population density of such networks often lead to scalability problems [1, 3]. Moreover, ad hoc wireless networks are notoriously vulnerable to message loss, which makes it difficult to let every operational host in the system be aware of detected failures and may cause frequent false detections. Consequently, it is impossible for an FDS to provide deterministic guarantees for completeness and accuracy (see Section 4.1 for the definitions) in the context of ad hoc network applications. Recently, researchers devised methods for wireless sensor network monitoring (see [4], for example), proposed the use of heartbeat mechanisms for connectivity probing in large sensor networks (see [5], for example), and investigated the communication strategies for heartbeat-style failure detectors in ad hoc wireless network settings [6]. Nonetheless, efforts that directly tackle failure detection problems themselves for ad hoc network applications are still largely lacking.

With the above motivation and based on the observation that *clustering* approaches have become an emerging technology for building scalable, robust, and energy-balanced ad hoc network applications [1, 7], we derive a cohesive solution for FDSs that exploits a cluster-based communication hierarchy to achieve scalability, completeness, and accuracy simultaneously.

More specifically, we use a cluster-based communication architecture to allow an FDS to execute in parallel in clusters and to detect failures via intra-cluster heartbeat diffusion. Further, if a failure is detected in a local cluster, the detection result will be forwarded across the clusters, in a way that is resilient to message loss and node failure, through the upper layer of the communication hierarchy. More importantly, this cluster-based architecture enables us to extensively exploit the message redundancy that is inherent in ad hoc wireless networks to mitigate the effects of message loss on the completeness and accuracy properties of failure detection. Consequently, the resulting FDS is able to provide satisfactory probabilistic guarantees for the desired properties.

The remainder of the paper is organized as follows. Section 2 describes the application model, assumptions, terminologies, and the problem we aim to solve. Section 3 presents a cluster-based communication architecture. Section 4 discusses the implementation of the failure detection service, followed by Section 5 which provides a probabilistic analysis. Section 6 concludes the paper.

2 Fundamentals

2.1 Application Model

As described in Section 1, the type of system we consider consists of a large, dense population of lightweight resources and is built over ad hoc wireless networks. Applications of such systems often use localized algorithms to enable hosts to interact with each other in clusters but collectively achieve a global objective [1]. A system will normally have hundreds or thousands of hosts. Among those hosts, some will be so-called "base-stations," which are able to transmit periodically, or on demand, the collected observations, measurements, and system conditions to an aircraft, an LEO satellite, or another network [2].

While those large-scale systems emphasize localized host interaction, local systems must be aware of condition changes in the global system or changes of the global objective. In addition, the information collecting points (base-stations) may be scattered in the field. Accordingly, it is important to let the summary information regarding locally detected failures be propagated to all the clusters to make the failure information accessible anywhere in the system. Furthermore, due to the redundancy in large-size high-density systems, completeness and accuracy of failure detection are more important than time to failure detection.

Nodes could be the hosts that become stationary after deployment (as is assumed/postulated in most of the recent investigations of large sensor networks) or mobile hosts that have localization capability and may migrate in the field autonomously (e.g., nano-sat swarms or networks of smart sensors). For simplicity, we do not address resource migration problems in this paper. Nonetheless, as sound clustering algorithms will support cluster and routing stability in mobile ad hoc wireless settings [8, 9], our failure detection framework can be extended accordingly to accommodate host migration.

Hosts are equipped with solar cells for "energy harvest" [10]; therefore, intra-cluster heartbeat diffusion with a reasonably low frequency will be feasible. Finally, hosts may fail over time. When the number of operational hosts drops below a threshold, additional resources will be deployed to replenish the system to maintain its population density. On the other hand, excessive false detections will increase maintenance cost significantly and unnecessarily.

2.2 Assumptions

Our proposed FDS assumes a fail-stop model. In addition, we assume that a node will not fail during an FDS execution. More precisely, if a node sends its heartbeat at the epoch of a heartbeat interval ϕ (the duration between two consecutive FDS executions), the node will not crash within the FDS execution duration (a small fraction of ϕ). Further, we assume that there will be no creations or alterations of messages over the transmission links. However, we assume that it is always possible for a message to be lost during transmission with a non-negligible probability.

In addition, we assume that 1) the clock rate on each host is close to accurate, and 2) in most cases the delay of message delivery *within* the transmission range is smaller than a reasonable time T_{hop}, as typically assumed in the literature concerning failure detectors for asynchronous systems (see [6, 11], for example). Finally, since 1) radio links in the type of application we consider are typically symmetric [12], and 2) clustering can help circumvent asymmetric link problems via open- or closed-loop power control [8], we assume that all the hosts have the same transmission range.

2.3 Terminologies

An ad hoc network will be modeled by a graph $G = (V, E)$, where V is a set of nodes (hosts) and E is a set of edges (links). A link between nodes v and v' means that v is within the transmission range of v', and vice versa. If we let R denote the transmission range of v, all the nodes that are at a distance from v less than or equal to R are called *one-hop neighbors* of v. For simplicity of illustration, unless stated otherwise, we use the term "neighbor" or "immediate neighbor" to refer to a "one-hop neighbor" in the rest of the text. In addition, the terms "node" and "host" are used interchangeably in the remainder of this paper.

In ad hoc wireless network applications, hosts normally operate under the *promiscuous receiving mode* [12]. More succinctly, when a node sends a message, all its immediate neighbors may hear the message, regardless of whether or not they are the intended recipients of that message. Therefore, the effect of sending a message to a neighbor is indeed equivalent to that of a broadcast to all the neighbors, which suggests that the two terms "send" and "broadcast" could be used interchangeably. Nonetheless, for clarity we do not use the word “broadcast” unless all the neighbors of the sender are the intended recipients.

To distinguish unwanted redundant message-forwarding from the phenomenon caused by the promiscuous receiving mode (in which a message may always be heard or overheard by all the neighbors of the sender), we use the terms “redundant message forwarding” and “inherent message redundancy” to refer to the former and the latter, respectively. In addition, we use “across-cluster forwarding” to refer to a process that forwards a message from cluster C to cluster C' which is an immediate neighbor of C, whereas by “inter-cluster forwarding,” we mean the general cluster-level message-forwarding process in which the source and destination are not necessarily neighboring clusters.

2.4 Problem Description and Scope

A major challenge for FDSs in ad hoc wireless networks is their vulnerability to message loss. For example, a healthy node may be wrongly identified as failed if the node’s heartbeat is lost during transmission, which leads to the violation of the accuracy property (see the definitions of accuracy and completeness in Section 4.1). As another example, if a message that announces a detected failed node v cannot be forwarded successfully to all the operational nodes, some nodes will be unaware of v’s failure, and the completeness property will thus be compromised. Those factors have collectively motivated us to seek a solution that can simultaneously address the issues concerning scalability, completeness, and accuracy. Accordingly, our effort is not intended to develop a new abstract failure detector; instead, the central purpose of this paper is to show how a cluster-based communication architecture coupled with the algorithms exploiting the characteristics of ad hoc networks will lead to probabilistic guarantees of the properties suggested by the established abstract failure detectors (see [13, 11, 14], for example).

Although the proposed failure detection service is also intended to support group membership management, we do not address group membership issues such as membership subscription and unsubscription. While energy issues are beyond the scope of this paper, we suggest an intra-cluster message-loss-recovery mechanism that considers energy balancing. As a cluster-based communication architecture gives us the flexibility to adopt or adapt various existing routing algorithms, we assume the presence of a routing protocol at the inter-cluster communication layer. Meanwhile, we implement some mechanisms that take advantage of the inherent message redundancy to make across-cluster forwarding robust and efficient; when implemented in middleware, those mechanisms will be able to coordinate with a routing protocol. Finally, as the FDS is designed for a middleware implementation, we do not address issues at or below the MAC layer.

3 Communication Architecture

It is well-understood that the quality of failure detection services often depends heavily on efficient and reliable communication. However, to make communication in ad hoc wireless networks both scalable and robust is difficult. As clustering approaches are particularly appealing to large-scale high-density ad hoc network applications, we seek a solution from the notion of clustering for ad hoc wireless communications [15, 7, 9].

A cluster can be viewed as a unit disk with a radius equal to the center node’s transmission range. As a result, any non-center nodes in a cluster are one-hop neighbors of the center node. With a clustering algorithm, the center node is called the *clusterhead* (CH), while a node that is a one-hop neighbor of the CHs of two different clusters can become the *gateway* (GW) node (see Figure 1). (A node that is located outside two clusters but has at least one non-CH neighbor in each of the clusters can also become a gateway, but we do not adopt that option under normal situations because it may reduce robustness of communication and failure detection.) After the autonomous cluster formation, only clusterheads and gateway nodes, which are elected in a fully distributed fashion, participate in the inter-cluster communication (see Figure 1(b)), while ordinary members (OMs) in each cluster talk only to their CHs (and to other members when necessary). As shown in Figure 1(a), the distance between any two non-CH nodes in a cluster will be at most two hops, as the clustering algorithm ensures that all the OMs are the immediate neighbors of the CH.

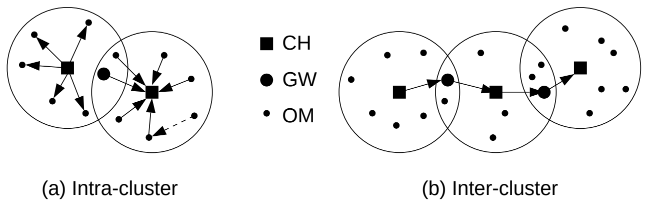


Figure 1: Intra-Cluster and Inter-Cluster Communication

The main purposes of clustering algorithms are 1) to permit the communication in ad hoc wireless networks to be scalable to node populations, and 2) to deal with mobility [7]. With a cluster-based ad hoc wireless network, scalability stems from a two-tiered communication architecture, namely, intra-cluster and inter-cluster communications, as shown in Figures 1(a) and 1(b), respectively. Since a cluster can always be mapped to a unit disk with a radius equal to the transmission range of the CH, the organization of a cluster is inherently a *clique*, meaning that within a cluster, anyone can talk to anyone else either directly or via the CH (a trivial route). Hence, the notion of clustering inherently facilitates localized algorithms. In addition, as only the clusterheads and gateways participate in the inter-cluster communication, system-wide information dissemination can be done far more efficiently than with flat flooding.

A typical clustering algorithm begins with a distributed clusterhead selection procedure. Specifically, through a heartbeat diffusion that functions as one-hop neighborhood probing, a node v identifies itself as a clusterhead when it recognizes that it meets a qualifying policy. The policy could be that a node having the lowest node ID (NID) within its one-hop neighborhood will become a CH[1]. A CH and all its neighbors thus form a cluster. In addition, the nodes that can hear from the CHs of two or more clusters are qualified to become the gateways. Multiple clusters can be formed simultaneously, and the clustering procedure may continue iteratively. Eventually, all the nodes except the isolated ones (i.e., those outside the transmission range of any networked nodes) will become affiliated with clusters.

Our cluster-formation algorithm [16] is a variant of the algorithms developed in [15, 7], and offers a number of unique features as follows:

F1) The algorithm ensures 1) that the resulting clusters partially overlap one another, which makes the GWs directly connect to two or more CHs, and 2) that with high probability, multiple gateway candidates will be available for robust cluster connectivity.

F2) The algorithm takes advantage of high population density to create deputy clusterheads (DCHs) and backup gateways (BGWs), which allow the FDS to become more resilient to both node and link failures.

F3) While previously developed clustering algorithms often leave gateway nodes with no particular cluster affiliations, our algorithm ensures that each gateway is affiliated with one and only one cluster, which prevents ambiguity in cluster-based failure detection.

F4) The algorithm intentionally leaves an "open end" by not providing an explicit termination rule to stop cluster-formation iterations. This enables the algorithm to treat newly arriving hosts (which are unmarked nodes[2]) in the same way as hosts that enter the system prior to the algorithm execution but have not yet been admitted to clusters.

F5) After the initial iteration of cluster formation, we allow the first round of the cluster-formation algorithm to "merge" with the first round in the failure detection service (see Section 4.2). More specifically, at the epoch of each heartbeat interval, both marked and unmarked nodes will transmit their heartbeats. That serves multiple purposes concurrently:

   – For the marked nodes, their heartbeats enable them to participate in the failure detection service that executes in their local clusters.
   – For the unmarked nodes that are located inside the established clusters but are not yet recognized by the CHs (due to the events of message loss, resource migration, or replenishment), their heartbeats can be treated as membership subscriptions by a group membership service.
   – For the unmarked nodes that are located outside the clusters, their heartbeats will lead the clustering algorithm to form additional clusters to cover as many of the nodes as possible.

[1]Although message loss during cluster formation may result in concurrent and conflicting CH declarations, strategies such as the RCC (random-competition-based clustering) scheme proposed in [9] can be adopted (or adapted) for resolution.

[2]Initially, all the nodes are unmarked, and a node marks itself after it is admitted to a cluster.

Note that after the first round, the clustering and FDS algorithms (see Section 4.2) will execute separately. On the other hand, if during the kth iteration ($k > 1$), none of the heartbeats diffused in the first round (which is shared by the cluster-formation algorithm and the failure detection service, as described in F5) show the presence of unmarked nodes, then the kth iteration of the cluster-formation algorithm will become degenerate, implying that the non-stopping iterative execution of the algorithm will incur no additional costs unless there is a need for cluster formation.

4 Failure Detection Service

4.1 Completeness and Accuracy

Extensive research efforts on the subject have yielded a rich body of literature concerning the abstract failure detector models characterized by formally defined completeness and accuracy properties (see [13, 14], for example). Since our FDS is aimed at serving ad hoc network applications by providing probabilistic guarantees for completeness and accuracy, rather than attempt to provide deterministic guarantees for a weaker version of the properties, we choose to informally define those two properties with respect to a perfectly reliable FDS in the context of our application:

Completeness: Every node failure will be reported to every operational node.

Accuracy: No operational node will be suspected by other operational nodes.

By an "operational node," we mean a node that is not crashed or partitioned from the network in question. Due to the vulnerability of message loss, a deterministic guarantee of the accuracy property will not be possible in the context of ad hoc wireless networking. On the other hand, the clustering approach facilitates a probabilistic accuracy guarantee. In particular, the cluster-based communication architecture permits an FDS to execute locally in each cluster and allows the result to be forwarded to other clusters *only if* a node failure is detected. Then, based on the assumption that there will be no creations or alterations of messages on transmission links, the problem of providing a probabilistic accuracy guarantee can be reduced to a simpler problem. More succinctly, for achieving a satisfactory probabilistic accuracy guarantee at the system level, it will be sufficient if we can effectively reduce, at the local-cluster level, the likelihood of false detection.

Similarly, in order to achieve a satisfactory probabilistic guarantee of the completeness property, it will be sufficient to ensure that with high probability, a failure report will 1) reach every member of the cluster in which the corresponding failure is detected, 2) go through the backbone of the communication architecture to reach every CH, and 3) reach all the cluster members via their CHs that receive the report through the inter-cluster communication.

As described in Sections 4.2 and 4.3, robust intra- and inter-cluster communications are an integral part of our FDS, which enables the probabilistic guarantees of completeness and accuracy.

4.2 Localized Failure Detection

During the distributed cluster formation, a self-elected CH identifies cluster members and broadcasts a cluster organization announcement. Hence, every cluster member will have an initial view of the cluster-based local membership, and the CH will know from which nodes it should expect to hear during an FDS execution. This service starts to execute at the epoch of each heartbeat interval and is described below.

Heartbeat-Style Failure Detection Service.

fds.R-1: *Heartbeat exchange*

Every node in the local cluster C sends to the CH a heartbeat message which contains the sender's NID and a one-bit mark indicator. Meanwhile, the CH broadcasts such a heartbeat message. Because of the cluster-formation algorithm and the promiscuous receiving mode, the heartbeat of the CH can be heard by all the nodes in C, while the heartbeat of a non-CH node v can be heard or overheard by the CH and a subset of the cluster members (i.e., the in-cluster one-hop neighbors of v). Thus, this round can be viewed as a *heartbeat diffusion*, during which the CH and non-CH nodes perform heartbeat broadcasting and "multicasting," respectively.

fds.R-2: *Digest exchange*

Every node in cluster C sends to the CH a digest which enumerates the nodes in C from which the sender node hears or overhears their heartbeats during fds.R-1. Meanwhile the CH broadcasts its own digest to every member in C. The actual result of this round is a *digest diffusion*.

fds.R-3: *Health-status-update broadcast*

By analyzing the heartbeat information collected through fds.R-1 and fds.R-2, the CH identifies failed nodes according to the failure detection rule (see below) and then broadcasts an update on the cluster health status, which indicates the newly detected failed nodes in C (if any).

Failure detection rule: A node v is determined to have failed if and only if 1) the CH receives neither v's heartbeat in fds.R-1 nor the digest from v in fds.R-2, and 2) none of the digests that the CH receives reflect a member's awareness of the heartbeat of v.

From the fail-stop model and the assumption that no message creation or alteration will happen on communication links, it follows that the above rule is sufficient to guarantee that no failed cluster members will go undetected by the FDS. Since every node in C will send at most one message in each of the rounds, the duration of each round is set to a fixed value T_{hop} (see Section 2.2), which serves as a timeout. Furthermore, the above rule simultaneously exploits time, spatial, and message redundancies, which significantly reduces the likelihood of false detection. More specifically,

1) By treating both the heartbeat in fds.R-1 and the digest in fds.R-2 as indicators of a node's health, the rule exploits time redundancy;

2) By summarizing the digests from all the cluster members, the rule takes advantage of the inherent spatial redundancy in high-density systems and the inherent message redundancy in ad hoc network settings (i.e., the heartbeats that a non-CH node *overhears* during fds.R-1 will be reflected in its digest sent to the CH).

The logic of the failure detection rule is also applied to the detection of CH failures, except that for CH failures the highest-ranked DCH (see Section 3 and [16]) is the authority that makes the decision. If the DCH detects a CH failure, it will broadcast a health status update and take over from the CH at the end of fds.R-3.

CH-failure detection rule: A CH will be judged to have failed if and only if 1) the DCH receives neither the CH's heartbeat in fds.R-1 nor the digest from the CH in fds.R-2, 2) none of the digests that the DCH receives reflect a member's awareness of the heartbeat of the CH, and 3) the DCH does not receive the health status update from the CH in fds.R-3.

Due to possible losses of messages, the DCH may mistakenly judge an operational CH to be a failed host. Such false detection could result in severe damage to the quality of the FDS. Specifically, this mistake may cause the CH and DCH to generate two conflicting failure reports and to broadcast them simultaneously. In such a situation, perhaps exacerbated by further message losses, the GWs may not notice the discrepancy and thus may forward the conflicting reports to neighboring clusters, resulting in inconsistent views on failures. Nonetheless, due to the exploitation of time, spatial, and message redundancies, the likelihood of such a scenario will be extremely low. Section 5 provides a probabilistic analysis that validates this assertion.

After the detection of a failed CH, the DCH takes over from the CH. Although it is unlikely to happen due to the high-density nature of the systems we consider, we must take into account the case in which the DCH's transmission range cannot reach some of the cluster members. As illustrated in Figure 2(a), the nodes in the area A_v are out of the transmission range of the DCH because of the distance d between the CH and DCH. While this situation might cause accuracy problems in failure detection, the digest collection in fds.R-2 will significantly reduce their probability. Consider the scenario illustrated in Figure 2(a), in which the dashed circle marked A_u represents the area that is within the transmission range of node v. This node is located in area A_v (the darkest area), and area A_g (marked by horizontal lines) is the region that can be reached by both the DCH and v. Clearly, unless A_g is empty, the DCH may learn that v is still alive based on the digest from v', a node in A_g. We have conducted a model-based analysis of a DCH's reachability. The evaluation results show that unless the node population density is low and the DCH's distance from the original CH is big, with high probability a DCH will be able to hear from an "out-of-range" cluster member through the round of digest diffusion. (The study is not included in this paper due to space limitations.)

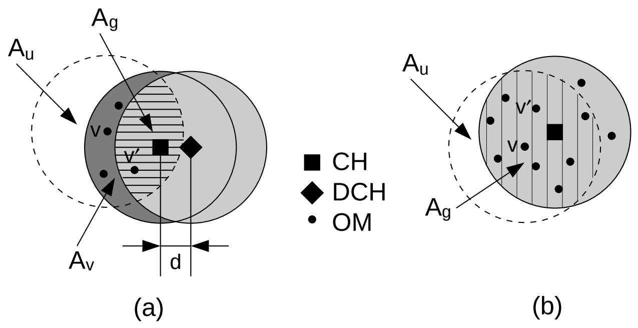


Figure 2: Exploitation of Inherent Message Redundancy

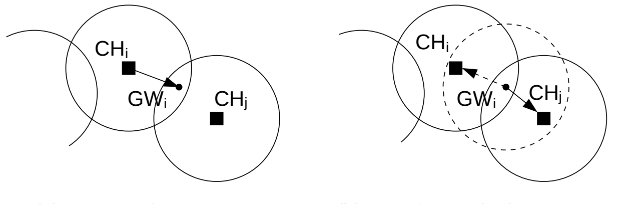
Intra-Cluster Completeness Enhancement. Nonetheless, fds.R-3 is not equipped with redundancy, which implies a potential completeness problem (i.e., some nodes may not be able to receive the update that indicates a detected failure from the CH). In addition, as discussed above, after a CH failure, the DCH may not be able to deliver a failure report directly to every member in the cluster. Our solution to those potential completeness problems is to allow *intra-cluster peer forwarding*. Consider the node v and its immediate neighbor v' in Figure 2(b). After receiving a health status update from the CH, v' will forward the report to v upon the request from v (which is made at the end of fds.R-3, a timeout for report receiving). Peer forwarding would also allow node v to receive messages from the DCH of which v is not an immediate neighbor (see Figure 2(a)). In that case, v' may perform forwarding based on its knowledge about the DCH's inability to reach v (learned from the digest received from the DCH).

Energy Considerations. We prefer peer forwarding over the retransmission by the DCH (or CH) because of energy-balancing considerations. However, a potential problem with intra-cluster peer forwarding is that multiple neighbors may respond to a forwarding request simultaneously. That may result in serious wastage of energy if a cluster has a dense population. Thus we use the following strategy: When a node v broadcasts a forwarding request, each of v's in-cluster neighbors will set a waiting period for the requested forwarding. The waiting period could be a function of the node's NID (which is globally unique in the network) and be inversely proportional to the node's remaining energy, which would allow each of v's neighbors to have a unique waiting period and would balance energy. Accordingly, the neighbors would not perform forwarding simultaneously. Rather, after one of them successfully forwards the message, the other neighbors will quit upon overhearing an acknowledgment from v.

4.3 Inter-Cluster Failure Report Forwarding

If at the end of fds.R-3 a GW in a cluster C receives an update from the CH that indicates a newly detected failure, the GW will forward the update as a failure report to the CH(s) of the neighboring cluster(s) it connects to. In addition to the NID of the newly detected failed node, a failure report may also include the NIDs of the previously detected failed nodes. That enables the detection service to provide a better probabilistic guarantee of the completeness property (if an earlier failure report could not successfully reach some cluster, the cluster may eventually be able to learn the missed failure information). On the other hand, an update that does not indicate any newly detected failures will not yield any inter-cluster forwarding. Accordingly, the clusters in the system will interpret the absence of such a report from cluster C as "no news is good news." However, if a cluster sends a failure report to its neighboring clusters, we must ensure that with high probability, the report will be successfully forwarded to all other clusters in the system.

A major challenge in inter-cluster forwarding is that we must mitigate the effect of message loss during the failure report forwarding and meanwhile keep the costs of transmission and retransmission low. Specifically, retransmission, the means for tolerating link failure along the inter-clustering forwarding, needs to be activated by the absence of an acknowledgment from the intended recipient. This suggests that each across-cluster forwarding would always involve two acknowledgments, which is not acceptable due to energy limitations. To meet this challenge, we again take advantage of the nodes' promiscuous receiving mode in ad hoc wireless settings to reduce the cost of retransmission. The first mechanism is illustrated in Figure 3. Specifically, we make use of the message that is forwarded by the successor forwarding node (GW$_i$) and is overheard by the sender (CH$_i$) as an *implicit acknowledgment*[3], to reduce the number of messages involved in inter-cluster forwarding. Accordingly, CH$_i$ will set its timer to $2\,T_{\text{hop}}$ right after forwarding a message to GW$_i$; then, if CH$_i$ does not overhear GW$_i$'s forwarding by timer expiration, CH$_i$ will assume that the first transmission is unsuccessful and thus will retransmit the message.



(a) CH$_i$ sends a message (b) GW$_i$ forwards the message

Figure 3: Implicit Acknowledgment

We note that in addition to its ability to take over from a failed GW, a BGW can be employed to enable energy-balanced message-loss recovery. Accordingly, we devise a mechanism to enable BGW-assisted forwarding as follows.

Suppose that between two neighboring clusters C and C$'$ there are n BGWs, each of which has a unique rank k ($k \in \{1, \ldots, n\}$). Upon overhearing a message m forwarded by the CH of C and realizing further forwarding to C$'$ is necessary, a BGW ranked k will set a timeout period $k\,2T_{\text{hop}}$ that it uses to see if m is successfully forwarded to C$'$ by the GW or a BGW with a rank j ($j < k$). If the BGW does not overhear an implicit acknowledgment from the CH of C$'$ by timer expiration, it will forward m by itself and then wait for $(n+1)\,2T_{\text{hop}}$ to see if an implicit acknowledgment can be heard from the CH of C$'$. This BGW will release itself from standby if it hears such an acknowledgment prior to its timer expiration.

As to the GW (between C and C$'$), it will forward m immediately after receiving the message and learning of the need to forward. In order to allow the BGWs to assist in re-forwarding m if the GW's forwarding is unsuccessful, the GW will set its timer to $(n+1)\,2T_{\text{hop}}$ and will not re-forward m unless the implicit acknowledgment from the CH of C$'$ has not been heard by timer expiration.

[3]A similar mechanism to eliminate the need for explicit acknowledgments in order to reduce the cost of retransmission was independently proposed by [17].

5 Probabilistic Analysis

The quantitative study presented in this section evaluates a number of probabilistic measures concerning the completeness and accuracy properties of the FDS. In the analysis, we assume that all the hosts have a transmission range of 100 meters and that each cluster will have 50 to 100 operational hosts whose locations are *statistically* uniformly distributed. Furthermore, we let p denote the message loss probability[4]. More precisely, we assume that if a node v transmits a message, the message may fail to reach a neighbor of v with probability p (as assumed by others in similar settings; see [6, 3], for example). In addition, based on our assumption of appreciable message loss probability, we let $[0.05, 0.5]$ be the range of p. Finally, as shown in the following sections, we chose to define probabilistic measures on a local cluster basis. The rationales for this choice are as follows: 1) global-level measures will require the assumptions of an inter-cluster routing algorithm and a network topology, and 2) the proposed FDS emphasizes localized algorithms and thus the per-cluster level measures will suffice for the purpose of an initial demonstration.

5.1 Measures Concerning Accuracy

Recall that as defined in Section 4.1, "accuracy" means that "no operational node will be suspected by other operational nodes." As the FDS executes locally in the clusters and there is no message creation or alteration over the network, the accuracy property will be affected only if a CH (or DCH) reports a false detection. Hence, we define a measure $P(\text{False detection})$, which is the probability that an operational host will be mistakenly judged to be a failed host in an FDS execution. To evaluate $P(\text{False detection})$, we identify the conditions under which a false detection will happen:

C1) The CH receives neither the heartbeat from an operational node v nor the digest sent by v by the end of fds.R-2; and

C2) None of the digests that the CH receives during fds.R-2 show that v is alive.

We note that the number of in-cluster neighbors that a (non-CH) host v can have depends on its position in the cluster. In particular, as illustrated in Figures 4(a) and 4(b), the farther v is from the center, the smaller the overlapping region (the area with vertical lines) between cluster C (the

[4]Collision-caused message losses at the sender side are not considered here, as they will be masked by a CSMA scheme at the MAC layer.

shaded circle) and the neighborhood area of v (the dashed circle) is, and thus the lower the chance that v will have many in-cluster neighbors that can hear its heartbeat and report it via a digest to the CH.

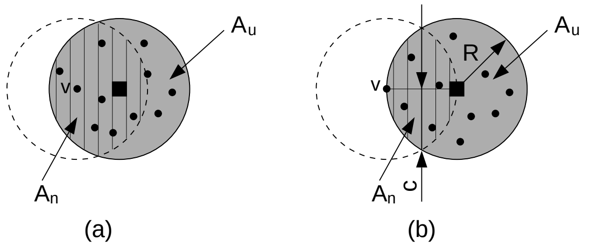


Figure 4: Transmission Range Overlapping

It follows that a host v that locates on the circumference of a cluster will represent the worst case (see Figure 4(b)). Accordingly, we evaluate the upper bound of $P(\text{False detection})$ based on that case, and let this measure be denoted by $\hat{P}(\text{False detection})$. From the false detection conditions discussed above (namely C1 and C2) and the assumption that hosts are approximately uniformly distributed, we formulate $\hat{P}(\text{False detection})$ as follows:

$$\hat{P}(\text{False detection}) = p^2 \sum_{k=0}^{N-2} \binom{N-2}{k} \times \left(1 - \frac{A_n}{A_u}\right)^{(N-2)-k} \left(\frac{A_n}{A_u}\right)^k \sum_{j=0}^{k} \binom{k}{j} (1-p)^j p^{k-j} p^j$$

where N is the number of nodes in the cluster, A_u is the total area of the cluster (the shaded area in Figure 4(b)), and A_n is v's neighborhood area within C (the shaded area with vertical lines in Figure 4(b)). Thus $A_n = 4\int_0^c \left(\sqrt{R^2 - x^2} - 0.5R\right) dx$, where $c = \sqrt{R^2 - (0.5R)^2}$ (see Figure 4(b)). Note that in the above equation, the coefficient p^2 computes the probability that C1 is true, while each summand of the inner summation evaluates the conditional probability that C2 is true given that v has k in-cluster neighbors. More precisely, the inner summation enumerates 1) the probability that v's heartbeat is not overheard by any of its k in-cluster neighbors during fds.R-1 (i.e., $j = 0$), and 2) the probabilities that v's heartbeat is overheard by j in-cluster neighbors during fds.R-1 ($0 < j \leq k$), but none of their digests are received by the CH during fds.R-2. Finally, the term $\binom{N-2}{k}(1 - \frac{A_n}{A_u})^{(N-2)-k}(\frac{A_n}{A_u})^k$ evaluates the probability that v has k in-cluster neighbors.

We then evaluate $\hat{P}(\text{False detection})$ as a function of p. The results are displayed in Figure 5, where the top, middle, and bottom curves correspond to the cases in which the node population sizes of C are 50, 75, and 100, respectively. As illustrated by the curves, the FDS behaves well in terms of being resilient to message loss. More specifically, if the cluster is densely or moderately densely populated (i.e., $N = 100$ or $N = 75$, respectively), the values of $\hat{P}(\text{False detection})$ are very small, even when p equals 0.5. Even with a reduced node population density, namely $N = 50$, the results of the measure are still very reasonable.

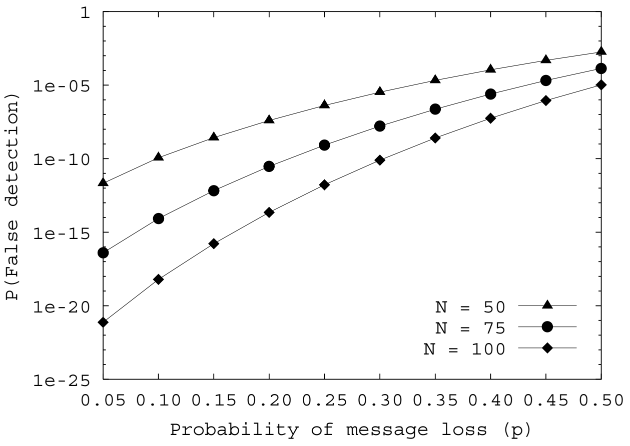


Figure 5: $\hat{P}(\text{False detection})$

As discussed in Section 4, the consequences would be severe if a DCH makes a false detection on the CH. Therefore, we next evaluate the probability of such an event, denoted by $P(\text{False detection on CH})$. The evaluation results are shown in Figure 6. (Due to space limitations, we omit the description of the measure's formulation.) The curves reveal that for the range of node population size considered, the likelihood of such a false detection is practically negligible or extremely low when p is below 0.25. When p reaches 0.5, $P(\text{False detection on CH})$ is still very low for $N = 100$ and $N = 75$, and the value of this measure is still below 10^{-6} even when N drops to 50.

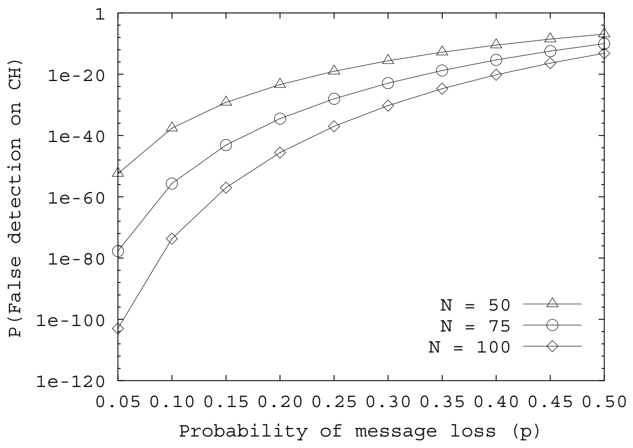


Figure 6: $P(\text{False detection on CH})$

To contrast Figure 5 with Figure 6, it seems a bit surprising that the CH is more likely than the DCH to make a false detection. A careful examination reveals that those are indeed reasonable results. Specifically, during an FDS execution, the responsibility of the CH is to examine the health of everyone else in a cluster C, while the duty of the DCH is to check whether the CH itself is healthy. Recall that the CH is selected by the clustering algorithm in such a way that all the members in C are at most one hop away from the CH, and thus the CH's heartbeat may be heard by everyone else in the cluster; however, a non-CH node is away from the center and thus can be heard or overheard by only a subset of the cluster members. This factor implies that it is more likely that the CH's heartbeat will be reflected in at least one digest received by the DCH, than that the heartbeat of a non-CH node will be reflected in at least one digest received by the CH. As explained in Section 4.2, the accuracy of a DCH's decision will be more critical than that of a CH's decision, hence the results discussed above are indeed highly desirable.

5.2 Measure Concerning Completeness

According to its definition, the completeness property of the FDS will ultimately also be determined by the robustness of intra-cluster communication. Specifically, after a CH detects a failure or receives a failure report from another cluster, the failure report should be received by every member in the cluster. This means that system-wide completeness will be a function of the probability that a cluster member will receive a failure report, given that such a report is broadcast by the CH. Hence, it will be meaningful for us to solve this constituent probabilistic measure. For clarity of illustration, we compute its complement, which is denoted as $P(\text{Incompleteness})$.

As described in Section 4.2, intra-cluster completeness is enhanced by progressive peer forwarding. Note that the peers must be the in-cluster, immediate neighbors of v, a node that needs help to recover a lost message. Furthermore, as revealed by Figures 4(a) and 4(b), the number of v's in-cluster neighbors is a decreasing function of v's distance from the CH. Thus, much as in the evaluation of $\hat{P}(\text{False detection})$, we can compute the upper bound of $P(\text{Incompleteness})$, denoted by $\hat{P}(\text{Incompleteness})$, if the evaluation is based on the case in which v is located on the circumference of the cluster (see Figure 4(b)).

While space limitations do not allow us to describe the measure formulation, the results are displayed in Figure 7. The curves demonstrate that the completeness property of the FDS is robust against message loss.

In addition, it can be observed in Figure 7 that when N increases from 50 to 100, $\hat{P}(\text{Incompleteness})$ decreases significantly. On the other hand, $\hat{P}(\text{Incompleteness})$ becomes more sensitive to p when N becomes larger. Moreover, a similar phenomenon can be observed in Figures 5 and 6. The phenomena are indeed due to some interesting interactions among N, p, and the measures we choose. First, when R is fixed, an increase of N actually means an increase of node population density in a cluster. Then, as N increases, spatial redundancy and inherent message redundancy will increase accordingly. In turn, this implies a decreased likelihood of false detection and an increased probability of a successful peer-assisted message-loss recovery, for a given value of p. On the other hand, a larger N means more messaging activities in a cluster; that, in turn, makes the system behavior more sensitive to the variations of p.

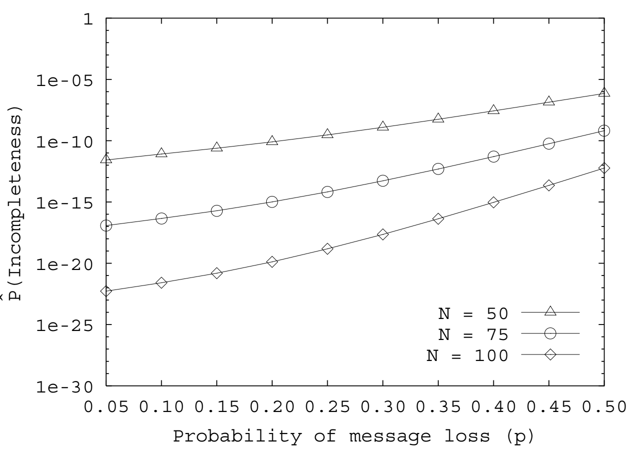


Figure 7: $\hat{P}(\text{Incompleteness})$

6 Concluding Remarks

We have developed an algorithm for failure detection in distributed systems that are constituted over ad hoc wireless networks. By exploiting a cluster-based communication architecture, we are able to make the failure detection service scalable and resilient to link failure. Moreover, by taking advantage of the inherent message redundancy in ad hoc wireless networks, we allow intra- and inter-cluster communication to be resilient to message loss and node failure.

Conceptually, the impossibility of providing deterministic guarantees of failure detection completeness and accuracy in ad hoc wireless networks is closely related to the impossibility of reliably distinguishing a process that is crashed from one that is just very slow in general asynchronous systems [13]. Nonetheless, while ad hoc network settings exacerbate the impossibility of developing a reliable failure detector, their characteristics can be utilized for implementation of failure detection services that provide probabilistic guarantees for the desired properties. Accordingly, the contribution of this investigation is twofold. First, this effort demonstrates the feasibility of developing failure detection services that can provide probabilistic guarantees of completeness and accuracy for large-scale distributed applications that are built over ad hoc wireless networks. Second, our novel application of the notion of clustering enables an FDS to exploit inherent message redundancy to achieve both robustness and efficiency. As clustering approaches are becoming increasingly popular for large-scale ad hoc network applications, this investigation fosters a fusion between the technologies of fault tolerance and wireless networking.

It is worth noting that cluster-based communication architectures can also be utilized for scalable, robust aggregation (e.g., coordinated in-network computation for average, maximum, or minimum of sensor measurements) [12] in large sensor networks. This indeed suggests that a cluster-based FDS may become an integral part of application-level host coordination activities. Specifically, by exploiting a cluster-based communication architecture and developing sensible aggregation query and data routing algorithms, it will be possible to embed an FDS in the aggregation query and data routing activities. The anticipated benefits include 1) energy efficiency induced by the "message sharing" between failure detection and data aggregation, and 2) further improvement of failure detection accuracy resulting from the sharing of the algorithms for reliable aggregation, such as the "streaming aggregates" proposed in [12].

We also note that a cluster-based architecture may support sleep/wakeup power management strategies for ad hoc wireless networks [18], since clustering may naturally help circumvent connectivity problems caused by node sleeping. On the other hand, sleep mode may cause false detections. Accordingly, we plan to investigate the feasibility of embedding FDS in application-level host coordination activities and of deriving algorithms to reduce the likelihood of sleep-mode-caused false detection.

References

[1] D. Estrin, R. Govindan, J. Heidemann, and S. Kumar, "Next century challenges: Scalable coordination in sensor networks," in *Proceedings of the Fifth Annual International Conference on Mobile Computing and Networks (MobiCOM)*, (Seattle, WA), pp. 263–270, Aug. 1999.

[2] K. Delin, S. Jackson, and R. Some, "Sensor Webs," *NASA Tech Briefs Journal*, vol. 23, p. 80, Oct. 1999.

[3] I. Gupta, R. van Renesse, and K. P. Birman, "Scalable fault-tolerant aggregation in large process groups," in *Proceedings of the International Conference on Dependable Systems and Networks (DSN 2001)*, (Göteborg, Sweden), pp. 433–442, July 2001.

[4] J. Zhao, R. Govindan, and D. Estrin, "Computing aggregates for monitoring wireless sensor networks," in *Proceedings of the First IEEE International Workshop on Sensor Network Protocols and Applications (SNPA 2003)*, (Anchorage, AK), pp. 139–148, May 2003.

[5] J. Elson *et al.*, "EmStar: An environment for developing wireless embedded systems software," CENS Technical Report 0009, Center for Embedded Networked Sensing, University of California, Los Angeles, CA, Mar. 2003.

[6] S.-C. Wang and S.-Y. Kuo, "Communication strategies for heartbeat-style failure detectors in ad hoc networks," in *Proceedings of the International Conference on Dependable Systems and Networks (DSN-2003)*, (San Francisco, CA), pp. 361–370, June 2003.

[7] M. Gerla and J. Tsai, "Multicluster, mobile, multimedia radio network," *Wireless Networks*, vol. 1, no. 3, pp. 255–265, 1995.

[8] T. J. Kwon and M. Gerla, "Clustering with power control," in *Proceedings of the IEEE Military Communications Conference (MILCOM 1999)*, (Atlantic City, NJ), pp. 1424–1428, Nov. 1999.

[9] K. Xu and M. Gerla, "A heterogeneous routing protocol based on a new stable clustering scheme," in *Proceedings of the IEEE Military Communications Conference (MILCOM 2002)*, (Anaheim, CA), pp. 838–843, Oct. 2002.

[10] J. Rabaey *et al.*, "PicoRadios for wireless sensor networks — the next challenge in ultra-low power design," in *Proc. IEEE Int. Solid-State Circuits Conference (ISSCC)*, (San Francisco, CA), Feb. 2002.

[11] R. van Renesse, Y. Minsky, and M. Hayden, "A gossip-style failure detection service," in *Proceedings of Middleware '98: IFIP International Conference on Distributed Systems Platforms and Open Distributed Processing*, (The Lake District, England), pp. 55–70, Sept. 1998.

[12] S. Madden, R. Szewczyk, M. J. Franklin, and D. Culler, "Supporting aggregate queries over ad-hoc wireless sensor networks," in *Proceedings of the 4th IEEE Workshop on Mobile Computing Systems and Applications*, (Callicoon, NY), pp. 49–58, June 2002.

[13] T. Chandra and S. Toueg, "Unreliable failure detectors for reliable distributed systems," *J. ACM*, vol. 43, no. 2, pp. 225–267, 1996.

[14] M. Hurfin, A. Mostéfaoui, and M. Raynal, "A versatile family of consensus protocols based on Chandra-Toueg's unreliable failure detectors," *IEEE Trans. Computers*, vol. 51, pp. 395–408, Apr. 2002.

[15] D. J. Baker, A. Ephremides, and J. A. Flynn, "The design and simulation of a mobile radio network with distributed control," *IEEE Journal on Selected Areas in Communications*, vol. 2, pp. 226–237, Jan. 1984.

[16] A. T. Tai and K. S. Tso, "Failure detection service for ad hoc wireless network applications: A cluster-based approach," Technical Report IAT-302184, IA Tech, Inc., Los Angeles, CA, Feb. 2004.

[17] B. Hull and H. Balakrishnan, "Resource management in sensor networks," Research Abstract, MIT Computer Science and Artificial Intelligence Laboratory (CSAIL), 2003.

[18] W. Ye, J. Heidemann, and D. Estrin, "An energy-efficient MAC protocol for wireless sensor networks," in *Proceedings of the 21st Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM 2002)*, (New York, NY), pp. 1567–1576, June 2002.