Comparison-Based System-Level Fault Diagnosis in Ad-Hoc Networks

REGULAR PAPER

Stefano Chessa[1] and Paolo Santi[2]

[1] Dipartimento di Informatica, Univ. of Pisa, Italy
[2] Dept. of Elec. and Comp. Eng., Georgia Institute of Technology, Atlanta, GA 30332-0280

Abstract

The problem of identifying faulty mobiles in ad-hoc networks is considered. Current diagnostic models were designed for wired networks, thus they do not take advantage of the shared nature of communication typical of ad-hoc networks. In this paper we introduce a new comparison-based diagnostic model based on the one-to-many communication paradigm. Two implementations of the model are presented. In the first implementation, we assume that the network topology does not change during diagnosis, and we show that both hard and soft faults can be easily detected. Based on this implementation, a diagnosis protocol is presented. The evaluation of the communication and time complexity of the protocol indicates that efficient diagnosis protocols for ad-hoc networks based on our model can be designed. In the second implementation we allow the system topology to change during diagnosis. We show that the ability of diagnosing faults under this scenario decreases, meaning that mobility significantly reduces the "quality" of the diagnosis returned by a diagnosis protocol.

Keywords. Ad-hoc networks, fault-tolerance, system-level diagnosis, diagnostic model, comparison-based diagnosis

Contact author:

Stefano Chessa
Dipartimento di Informatica, Corso Italia 40, 56125, Pisa, Italy
Phone #: +39 050 221 2736
E-mail: **ste@di.unipi.it**

1 INTRODUCTION

Recent advances in hardware design, resulting in affordable, portable, wireless communication and computation devices, and the rapid growth in the communication infrastructure, let mobile wireless networks become more and more popular. Among them, *ad-hoc networks* [8], i.e. networks of mobile, untethered units communicating via radio transmitters/receivers, are gathering growing interest in the scientific community. Ad-hoc networks, also called *multi-hop packet radio networks*, can be used whenever a wired backbone is not viable, e.g. to provide communications during emergencies or within a team of cooperating robots [16].

When designing protocols for ad-hoc networks, the different nature of the communication medium has to be taken into account. Although selective communication could be implemented, the most natural communication paradigm is the *one-to-many*: when a unit transmits, all the units within its transmitting range receive the message. Given the shared nature of communication, many protocols designed for wired networks turn out to be hardly adaptable to ad-hoc networks. For this reason, routing and broadcast protocols explicitly designed for these networks have been recently proposed in the literature [3][14][17][19].

As the popularity of ad-hoc networks grows and the computational power of mobile units increases, the need for dependability becomes an issue. However, so far few have been done in the design of fault-tolerant protocols for wireless networks. Recently, Pagani and Rossi [15] proposed a reliable broadcast protocol for ad-hoc networks, which is able to tolerate crash faults. In this paper, we consider the problem of fault diagnosis in the framework of multi-hop packet radio networks. Fault diagnosis is one of the main building blocks of many dependable protocols. In fact, once faulty units have been identified, the remaining units can isolate them from the rest of the system, and the computation can proceed, although with reduced performance. As far as the authors know, this is the first paper addressing the problem of system-level fault diagnosis in ad-hoc networks.

System-level fault diagnosis was introduced by Preparata, Metze and Chien in 1967 [18], as a technique aimed at diagnosing faults in system composed by a number of units interconnected by a wired point-to-point network based on the outcomes of reciprocal tests performed by the units themselves. In the original model by Preparata, Metze and Chien, a test involves a pair of adjacent (i.e., directly connected) units: the *testing* and the *tested* unit. The testing unit sends a test task to the tested unit, which, in turn, computes the result and returns it to the testing unit. The testing unit generates the test outcome by comparing the returned result with the expected one: if they agree the outcome is 0, otherwise it is 1. Since the seminal paper of Preparata, Metze and Chien, many variants of their diagnostic model have been proposed [2][5][6][7][11][12][13][20]. In particular, models based on comparisons rather than explicit tests have been proposed [5][7][12][20]. However, all the models proposed so far assume that units communicate according to the one-to-one paradigm typical of wired networks. This means that, if applied to ad-hoc networks, these models do not take advantage of the shared nature of communication.

In this paper we present a new diagnostic model based on the one-to-many communication paradigm. Both hard (e.g., crash, fail-stop and fail-silence) and soft faults are considered. The model is based on

comparisons of the outcomes returned by different units executing the same task and uses the invalidation rule of the generalized Maeng and Malek (gMM) model [12][20]. Two implementations of the model are presented. First, we assume that the network topology does not change during diagnosis, and we show that both hard and soft faults can be easily detected. Based on this implementation, we introduce a diagnosis protocol, we prove its correctness, and we evaluate its communication and time complexity. Then, we allow the network topology to change during diagnosis, thus taking into account a relevant feature of ad-hoc networks. We show that the "diagnostic efficiency" of the model (i.e., the number of diagnostic decisions that can be taken given the same set of task results) under this implementation is notably reduced. This means that, as it can be expected, mobility significantly reduces the "quality" of the diagnosis returned by a diagnosis protocol.

2 THE SYSTEM MODEL

The system is composed by n mobile hosts, henceforth called *mobiles*, which communicate via a packet radio network. The topology of the system at time t can be described as a directed graph $G(t)=(V,L(t))$, called the *communication graph*, where V is the set of nodes, denoting mobiles[1], and $L(t)$ is the set of *logical links* at time t. Given any $u,v\in V$, edge $(u,v)\in L(t)$ if and only if v is in the transmitting range of u at time t. For the sake of simplicity, in the following we assume that $G(t)$ is undirected, that is, $(u,v)\in L(t)$ if and only if $(v,u)\in L(t)$. Mobiles u and v are said to be *adjacent at time t* if $(u,v)\in L(t)$. When it is clear from the context, mobiles u and v adjacent at time t will be simply referred to as *adjacent*. The set of units adjacent to a given node u at time t is called the *neighbor set of u at time t*, denoted $N(u,t)$.

In this paper we assume the following:

A1. each mobile has a unique identifier; for the sake of simplicity, we assume that mobiles are ordered and that each mobile is identified by its ordinal.

A2. there exists a link-level protocol providing the following services:

- **A2.1.** a MAC protocol is executed to solve contentions over logical links.
- **A2.2.** the protocol provides a *1-hop reliable broadcast* primitive, called 1_*rb*(·), to the upper level.
- **A2.3.** the receiver of a message knows the identity of the sender.

Assumption A2.2 deserves particular attention. The primitive 1_*rb*(·) is such that, when invoked by mobile u with parameter m at time t, message m is correctly delivered to all mobiles in $N(u,t)$. Given the nature of the communication network, assumption A2.2 can be easily accomplished by the link level protocol. In fact, in a packet radio network all communications are 1-hop broadcasts, and reliability can be obtained using error detecting/correcting codes and/or by retransmission of corrupted messages.

[1] In the following, words *mobile*, *node* and *unit* will be used indifferently.

3 THE DIAGNOSTIC MODEL

3.1 THE FAULT MODEL

Each mobile in the system can be in one of two states: faulty or fault-free. Faults are *permanent*, i.e. a faulty mobile remains faulty until it is repaired and/or replaced. Faults can be either *hard* or *soft*. When a unit is hard-faulted, it is unable to communicate with the rest of the system. In a wireless network, a unit can be hard-faulted either because it is crashed or due to battery depletion. Soft faults are subtle, since a soft-faulted mobile continues to operate and to communicate with the others mobiles in the system, although with altered specifications. However, in order for a diagnosis to be possible, the behavior of soft-faulted unit is somewhat constrained. This leads to the definition of the invalidation rule, which is used to diagnose the state of the units in the system. In this paper we utilize the invalidation rule of the gMM model [12][20], which is summarized in Table 1.

u	***v***	***w***	**comparison outcome of *v* and *w* generated by *u***
fault-free	fault-free	fault-free	0
fault-free	faulty	fault-free	1
fault-free	fault-free	faulty	1
fault-free	faulty	faulty	1
faulty	any	any	x

Table 1. The invalidation rule of the gMM model

In the gMM model, diagnosis is based upon comparison of the results generated by test tasks assigned to pairs of units with a common neighbor. Let u be a unit adjacent to both unit v and w. If units u, v and w are fault-free, then the results agree and the comparison outcome is 0 (the comparison *passes*). If unit u is fault-free and any unit between v and w is faulty, then the results disagree and the comparison outcome is 1 (the comparison *fails*). If unit u is faulty, then the comparison outcome may be arbitrary, regardless of the state of v and w. Observe that in the gMM model one of the compared units may be u itself.

Under the hypotheses of the gMM model, units v and w can be diagnosed as fault-free by the fault-free unit u if their comparison passes. If it fails, unit u can only conclude that at least one unit between v and w is faulty. However, if one of the compared units is u itself, then the other can be diagnosed as faulty.

The gMM model constraints the behavior of soft-faulted nodes, assuming that the test task result generated by a soft-faulted unit differs both from the expected one (i.e., the test task has perfect fault coverage) and from the result of any other soft-faulted unit. This assumption is realistic if the test is designed such that the space of possible results is large and soft-faulted mobiles produce random and independent results, as it is the case in many applications.

3.2 A DIAGNOSTIC MODEL FOR AD-HOC NETWORKS

In this subsection we introduce a comparison-based diagnostic model for ad-hoc networks. Comparisons between units exploit the shared nature of the communication. Essentially, a fault-free unit u (the *testing unit*) tests its neighbors sending them a test request and waiting for their responses. As the responses are

received, units are diagnosed based on the invalidation rule of Table 1. Note that, given the shared nature of communication, the test results generated by the neighbors of u are received by many other units. In the following we discuss how and to which extent these results could be used to diagnose the neighbors of u. Observe that hard-faulted nodes are unable to respond to the test request. Furthermore, test responses sent by nodes that migrated out of the u's transmitting range can not be received by the testing unit. For these reasons, a timeout is needed to avoid starvation. The timeout time T_{out} is chosen in such a way that all the fault-free units in the neighborhood of u which do not migrate out of its transmitting range are guaranteed to respond to the test request within that time.

Depending on the assumptions regarding the topology of the network, different decisions on the (faulty or fault-free) state of the units that did not reply to the test request can be taken. In the following we present two implementations of the model under the hypothesis of fixed and time-varying topology. We show that, given the same set of diagnostic information (i.e. test task responses and "time-outed" units), in the former case more diagnostic decisions can be taken, thus leading to the design of more "efficient" diagnosis protocols.

In both implementations we assume that faults are *static*, i.e. that no new faults occur during the test execution. Observe that, if dynamic faults are allowed, units can only be diagnosed as faulty; in fact, a unit may fail soon after have been diagnosed as fault-free by other units. The implementation of the model under the hypothesis of dynamic faults is beyond the scope of this paper and is matter of ongoing research.

Assume that the network topology does not change during test execution, i.e. that if unit u sends its test request at time t, and T_{out} is the timeout time, then $N(u,t')=N(u,t)=N(u)$ for any $t<t'\leq t+T_{out}$. Observe that this assumption does not mean that the network is static, rather that its topology does not change during diagnosis: mobiles are allowed to migrate, but they cannot migrate out of their neighbors' transmitting ranges. Comparisons are performed based on the following *fixed topology comparison protocol*:

Test request generation

At time t, unit u (the testing unit) generates a test sequence number i, a test task T_i, the expected result $R_{u,i}$ and sends the message $m=(u,i,T_i)$ to $N(u)$ (i.e., it invokes the primitive $1\_rb(m)$). Message m is called a *test request*, and (u,i) is the *header* of the test request. Then, unit u sends a message to the timer.

Test request reception

Any unit v in $N(u)$, upon receiving m, generates the result $R_{v,i}$ for T_i and invokes $1\_rb(m')$ at time t', with $t<t'<t+T_{out}$, where $m'=(u,i,R_{v,i})$. Message m' is called a *test response*, and (u,i) is the header of the test response.

Test response reception

Any unit w in $N(v)$, upon receiving m', does the following:

- If $w=u$, i.e. if w is the testing unit itself, it compares $R_{v,i}$ with the expected result $R_{u,i}$ and generates the comparison outcome. Unit v is diagnosed as fault-free if the outcome is 0, as faulty otherwise.
- If $w\neq u$, the following cases arise:

a) $w \in N(u)$. In this case (Figure 1.*a*), unit w received the test request m from u, hence it can compare $R_{v,i}$ with $R_{w,i}$. Unit v is diagnosed as fault-free if the comparison outcome is 0, as faulty otherwise.

b) $w \notin N(u)$. The test response is compared to test responses received for the same task (Figure 1.*b*) (if any). If there exists some $z \in N(u)$ such that $R_{z,i}=R_{v,i}$ then both mobiles are diagnosed as fault-free; otherwise, if unit z has been diagnosed as fault-free, then mobile v is diagnosed as faulty. Otherwise, the test result $R_{v,i}$ is stored.

Timeout reception

At time $t+T_{out}$ the testing unit u receives the message from the timer and diagnoses as faulty all the units that did not reply to the test request.

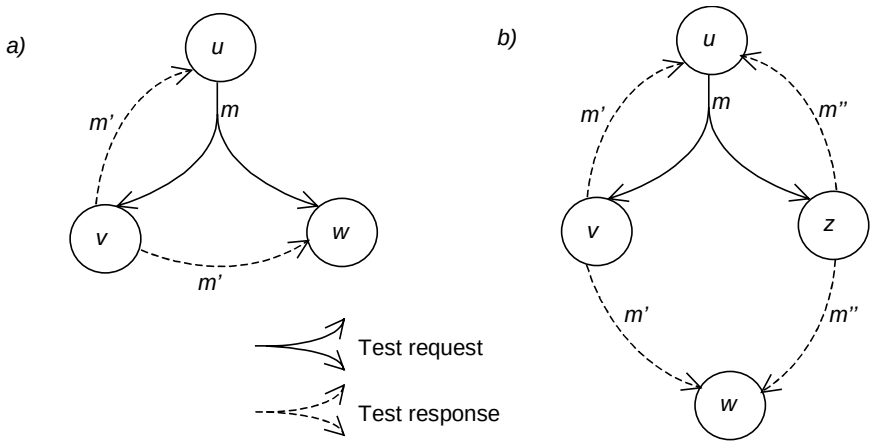

Figure 1. Case *a)* Unit w received the test request m from u. Case *b)* Unit w received test responses m' and m'' concerning the test request m.

A further restriction on the behavior of soft-faulted nodes must be imposed in order to ensure the correctness of the above comparison protocol. Consider the situation depicted in Figure 2, where unit v is soft-faulted. Fault-free unit z sends the test request (z,i,T_i) at time t, which is received by mobile v. Unit v generates the (incorrect) result $R_{v,i}$ for T_i and set up the test response message. However, it alters the header of the test response[2]. Let $(s,j,R_{v,i})$ be the test response sent to $N(v,t'')$ at time $t''>t$. Suppose that s corresponds to the identifier of a mobile in $N(u,t')$, for some $t<t'<t''$, and suppose that the unit identified by s sent the test request (s, j, T_j) to u at time t'. This way, unit u can compare $R_{v,i}$ with $R_{u,j}$, i.e. with the result of test task T_j received by s. Observe that the assumption that result $R_{v,i}$ differs from those generated by any other (faulty or fault-free) unit to the same task T_i, does not prevent $R_{v,i}$ from being equal to the correct result of a different test task T_j. This means that unit u might erroneously diagnose unit v as fault-free. This behavior of soft-faulted mobile is explicitly forbidden in our diagnostic model, which thus slightly strengthens the assumptions of the gMM model.

The likelihood of occurrence of this undesired behavior of soft-faulted mobiles can be significantly reduced by performing a consistency check on the header of the message before processing it. For example,

[2] According to the gMM model, a soft-faulted unit can alter arbitrarily the header of any message.

letting {0,..,n-1} be the set of mobile identifiers, unit i might be constrained to generate test sequence numbers congruent to i modulo n. This way, upon receiving a message with header (s,j), any fault-free node u may perform a consistency check based on s and j, diagnosing the sender of the message[3] as faulty if j is not congruent to s modulo n.

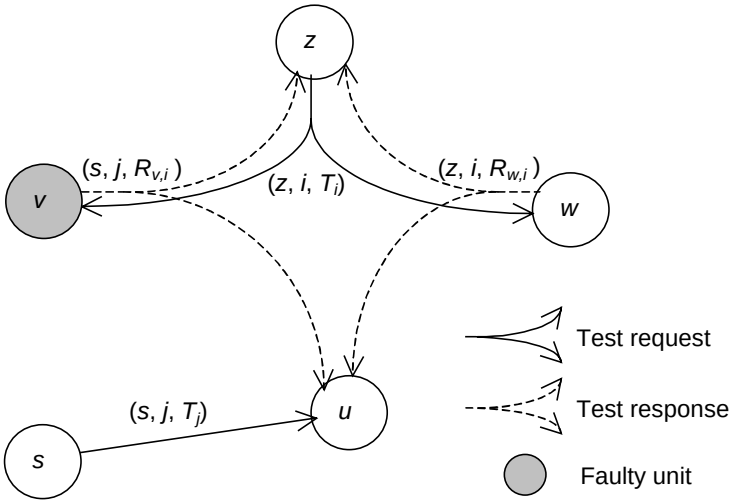

Figure 2. The soft-faulted unit v may impair the diagnosis of fault-free unit u.

Given the assumption of fixed network topology, any fault-free unit u which generates a test request at time t is guaranteed to correctly diagnose the state of all the units in $N(u,t)$ within time $t+T_{out}$. In fact, consider any fault-free unit $v \in N(u,t)$. Unit v replies to the test request at some time $t' < t+T_{out}$. Since the topology does not change, $N(u,t')=N(u,t)$, hence $v \in N(u,t')$. It follows that, given that the communication graph is undirected, $u \in N(v,t')$, i.e. the testing unit u receives the test response from unit v. Hence, all the units which did not reply to the test request within time T_{out} can be safely diagnosed as faulty.

Given units $u,v \in V$, define $N_u(v)=N(v) \cap N(u)$. Assume unit u issues a test request. The fault-free and the soft-faulted units in $N(u)$ respond to that request within time T_{out}. Hence, any fault-free unit v in $N(u)$ correctly diagnoses the state of all the fault-free and sof-faulted units in $N_u(v)$ within time T_{out}.

Finally, consider the set $N_2(u)=\{z \in V\text{-}N(u): |N_u(z)| \geq 2\}$. Let z be any fault-free unit in $N_2(u)$, and assume that at least two of the units in $N_u(z)$ are fault-free. By applying a similar argument, it can be seen that z correctly diagnoses the state of all the fault-free and soft-faulted units in $N_u(z)$ within time T_{out}.

Based on the preceding discussion, we can state the following theorem:

Theorem 1. *Assume the network topology is fixed, and assume that the fault-free node u generates a test request at time t. Then, at time $t+T_{out}$:*

- *unit u has correctly diagnosed the state of all the units in N(u);*
- *any fault-free unit v in N(u) has correctly diagnosed the state of the fault-free and soft-faulted units in $N_u(v)$;*

[3] By property A2.2, the sender of any message can be correctly identified.

- *any fault-free unit z in $N_2(u)$ has correctly diagnosed the state of the fault-free and soft-faulted units in $N_u(z)$ if at least two units in $N_u(z)$ are fault-free.*

In Section 5 we show how the fixed topology comparison protocol can be used to design an efficient diagnosis protocol for ad-hoc networks, i.e. a distributed diagnosis algorithm that allows every fault-free unit in the system to correctly diagnose the state of every other unit.

Assume now that units are allowed to migrate during the test execution time. Comparisons are performed according to the following *time-varying topology comparison protocol*:

Test request generation

At time t, unit u (the testing unit) generates a test sequence number i, a test task T_i, the expected result $R_{u,i}$ and sends the message $m=(u,i,T_i)$ to $N(u,t)$ (i.e., it invokes the primitive $1\_rb(m)$). Then, unit u sends a message to the timer.

Test request reception

Any unit v in $N(u,t)$, upon receiving m, generates the result $R_{v,i}$ for T_i and invokes $1\_rb(m')$ at time t', with $t<t'<t+T_{out}$, where $m'=(u,i,R_{v,i})$.

Test response reception

Any unit w in $N(v,t')$, upon receiving m', does the following:

- If $w=u$, i.e. if w is the testing unit itself, it compares $R_{v,i}$ with the expected result $R_{u,i}$ and generates the comparison outcome. Unit v is diagnosed as fault-free if the outcome is 0, as faulty otherwise.
- If $w\neq u$, the following cases arise:
  - *a)* $w\in N(u,t)$. In this case, unit w received the test request m from u, hence it can compare $R_{v,i}$ with $R_{w,i}$. Unit v is diagnosed as fault-free if the comparison outcome is 0, as faulty otherwise.
  - *b)* $w\notin N(u,t)$. Unit v is classified as *alive*. Furthermore, its test response is compared to test responses received for the same task (if any). If there exists some $z\in N(u)$ such that $R_{z,i}=R_{v,i}$ then both mobiles are diagnosed as fault-free; otherwise, if unit z has been diagnosed as fault-free, then mobile v is diagnosed as faulty. Otherwise, the test result $R_{v,i}$ is stored.

Timeout reception

At time $t+T_{out}$ the unit u receives the message from the timer and classifies as *time-outed* all the units that did not reply to the test request.

Since the topology of the network varies with time, in general we have $N(u,t)\neq N(u,t+T_{out})$. As a consequence, hard-faulted units cannot be distinguished from fault-free units that migrated out of the testing unit's transmitting range. For this reason, when the message from the timer is received, the testing unit can

only classify the units that did not reply to its test request as *time-outed*. This information could be useful in the design of a diagnosis protocol to identify hard-faulted mobiles.

Consider now the set $N_S(u)=N(u,t)\cap N(u,t+T_{out})$, and let $N_S^r(u)$ be the set of fault-free or soft-faulted units in $N_S(u)$. Units in $N_S^r(u)$ reply to the test request issued by u at some time $t'<t+T_{out}$. Observe that the fact that a unit v is in $N_S^r(u)$ does not imply that its test response is received by u, since v could be out of the transmitting range of u at time t'. For this reason, we further assume that u be in the v's transmitting range at time t', for any $v\in N_S^r(u)$. This way, the testing unit is guaranteed to correctly diagnose all the units in $N_S^r(u)$ within time T_{out}. This is the only diagnostic information the time-varying topology comparison protocol is guaranteed to achieve.

Further diagnostic decisions can be taken by any fault-free unit $z\in V$ which receives the test responses from at least two fault-free units in $N(u,t)$ (at least one if z itself is in $N(u,t)$). Observe that, contrary to the case of fixed topology, unit z might not be a distance two neighbor of u. However, due to the time-varying topology of the network, the exact number of such diagnostic decisions is hard to quantify, unless some restrictions on the mobility of the units are imposed.

The time-varying topology comparison protocol also classifies as *alive* any unit from which a test response is received. This information could be useful in the design of a diagnosis protocol to distinguish hard-faulted from fault-free or soft-faulted migrating units. The design of a diagnosis protocol for time-varying topology systems is beyond the scope of this paper and is matter of ongoing research.

Based on the discussion above, we can state the following theorem:

Theorem 2. *Assume that the fault-free node u generates a test request at time t, and that the network topology can vary. Then, at time $t+T_{out}$ unit u has correctly diagnosed the state of all the units in $N_S^r(u)$.*

4 COMPARISON WITH RELATED MODELS

The existing distributed diagnostic models [1][4][10], which were designed for the diagnosis of multicomputer systems, exploit tests involving a tester and a tested unit, along a dedicated, wired link. These models require that all the units in the system perform tests on their neighbors to produce a "local" diagnosis, which is then disseminated throughout the system. Since the test is executed along one-to-one dedicated links, the execution of a test affects mainly the performance of the units involved in the test, and it has little influence on the other units in the system, which can continue to operate and communicate.

Due to the shared nature of communication, the application of the existing diagnostic models to ad-hoc networks would reduce the available bandwidth significantly. In fact, a test involving only a pair of units u and v would affect seriously the performances of the neighbohood of u and v, since the neighbors of u and v compete with u and v to access the communication medium, and they can not communicate while u and v are

exchanging test information. Moreover, a test should be generated separately for each neighbor, thus affecting the latency of the diagnosis and increasing the test overhead.

The recently introduced broadcast comparison model [5] exploits broadcast to diagnose nodes based on the comparison of the outcomes returned by different units executing the same task. Contrary to the models in [1][4][10], the broadcast comparison model might exploit the shared nature of the communication which characterizes ad-hoc networks, thus reducing the test and diagnosis overhead. However, this model relies on the existence of a weak reliable broadcast protocol, which ensures that any message sent by a fault-free unit be correctly received by any other fault-free unit in the system. When considering wireless networks, the existence of a broadcast protocol with such strong properties appears to be unrealistic.

The model proposed in this paper exploits 1-hop reliable broadcast, which, given the shared nature of communication, can be easily and efficiently implemented in ad-hoc networks. Our model allows subsets of the neighbohood of a unit to share tests and test responses, thus reducing the bandwidth needed to execute tests. The extent to which this model allows to share tests and test responses depends on the network topology, and it is matter of ongoing studies.

5 A DIAGNOSIS PROTOCOL FOR FIXED TOPOLOGY NETWORKS

In this section we introduce a diagnosis protocol for fixed topology ad-hoc networks and static, permanent faults, which is based on the fixed topology comparison protocol presented in Section 3. In order for the fixed network topology assumption to be realistic, the diagnosis session should terminate as soon as possible. For this reason, our design goal was to minimize *diagnosis latency*, i.e. the elapsed time between the inception and the end of the diagnosis session.

The *diagnosis session* starts when a fault-free unit initiates its diagnosis protocol, and ends when the diagnosis protocol execution is terminated by every fault-free unit. A fault-free unit initiates its diagnosis protocol either spontaneously (e.g. periodically, or when abnormal operating conditions are detected) or when the first *diagnostic message* is received. A diagnostic message can be a test request, a test response, a timeout message or a *dissemination message*. Dissemination messages are messages generated by fault-free units to propagate the diagnosis of their neighbors throughout the network.

The diagnosis protocol, whose formal specification is reported in Table 2, can be informally described as follows:

- Eventually, a fault-free unit u generates a test request (u,i,T_i), sends it to $N(u)$, and computes the expected results $R_{u,i}$. Then it sends a message to the timer.
- Unit u waits for the responses of units in $N(u)$ (either concerning its request or other requests) and diagnoses their state according to the comparison protocol. When u has diagnosed the state of all the units in $N(u)$ (in time at most T_{out}), it generates a dissemination message containing its local diagnosis.
- Then, unit u waits for dissemination messages generated by other fault-free mobiles in order to complete its diagnosis. The diagnosis protocol for u terminates when the state of all the units in the system has been identified.

Data structures for mobile u:
- $N(u)$: neighbors set at the time of diagnosis;
- V : set of all mobiles in the system;
- $F(u)$: set of mobiles diagnosed as faulty, initialized to $\emptyset$;
- $FF(u)$: set of mobiles diagnosed as fault-free, initialized to $\emptyset$;
- *TestGenerated*: boolean variable initialized to **FALSE**. It is **TRUE** if unit u generated the test request, **FALSE** otherwise;
- *WaitDiagnosis*: boolean variable initialized to **FALSE**. It is **TRUE** if unit u disseminated its local diagnosis but it does not have a complete diagnosis of the system, **FALSE** otherwise;
- *Disseminated*[]: vector of Boolean variables initialized to **FALSE**. *Disseminated*[v] is **TRUE** if the dissemination message originated by mobile v has been propagated.

```
Procedure Test_Generation
    begin
        generate the test sequence number i and the correspondent test task T_i;
        1_rb((u, i, T_i));
        generate the expected result R_{u,i} for T_i;
        set the timer to T_out;      {set the timeout for hard faults detection}
        TestGenerated:= TRUE;
    end;

begin
repeat
Wait for diagnostic message msg;
case msg of
    Timeout: {mobiles in N(u) not diagnosed as fault-free must be faulty}
        F(u):= F(u) ∪ (N(u)-FF(u));

    Test Request (w, j, T_j) from v∈N(u):
        performs the consistency check on (w, j);  {it also checks that w=v)
        if the consistency check fails then
            F(u):= F(u)∪{v}; {the consistency check fails; unit v is diagnosed as faulty}
        else begin
            generate the result R_{u,j} for T_j;
            1_rb((v, j, R_u))  {send the test response to its neighbors}
        end;

    Test Response (w, j, R_{v,j}) from v∈N(u):
        if not (WaitDiagnosis) then
            if v∈V-(FF(u)∪F(u)) then {if the unit is diagnosed, the message is discarded}
            begin
                  performs the consistency check on (w, j);
                  if the consistency check fails then
                        {the consistency check fails; unit v is diagnosed as faulty}
                        F(u):= F(u)∪{v};
                  else if w=u then {response to the test T_j generated by u}
                              if R_{v,j} = R_{u,j}  then FF(u):= FF(u) ∪ {v};  {the comparison passes}
                                          else F(u):= F(u) ∪ {v}; {the comparison fails}
                  else  if w∈N(u) then
                        {response to the test T_j generated by w≠u, w∈N(u): u have already
                         received the test request T_j}
                              if R_{v,j} = R_{u,j}  then FF(u):= FF(u) ∪ {v};  {the comparison passes}
                                          else F(u):= F(u) ∪ {v}; {the comparison fails}
                  else {w does not belong to the neighborhood of u, then u did not receive  the test
                         request T_j}
                        if (u already received a test response (w, j, R_{z,j}) from z∈N(u))
                         and R_{v,j} = R_{z,j}
                              then FF(u):= FF(u) ∪ {v, z}; {the comparison passes}
                              else
                                if u already received a test response (w, j, R_z) from z∈N(u)
                                   and R_{v,j} ≠ R_{z,j} and z∈FF(u)
                                      then
                                         {if the results disagree adn z is known to be fault-free,
                                         then v must be faulty}
                                         F(u):= F(u) ∪ {v};
                                      else store the test response;
            end;

    Dissemination Message (w, FF(w), F(w)) from mobile v∈N(u):
        if v∈FF(u) and not(Disseminated[w]) then begin
            1_rb((w, FF(w), F(w))); {propagate the diagnosis produced by w}
            Disseminated[w]:= TRUE;
```

```
            {the diagnosis of u is extended}
            F(u):= F(u) ∪ F(w);
            FF(u):= FF(u) ∪ FF(w);
        end
        else
            if v∈V-(FF(u) ∪ F(u)) then
                  {if v is still undiagnosed the dissemination message is stored}
                  store the dissemination message;

end case;

if not TestGenerated then
    {after receiving any diagnostic message, u generates its test, if not already generated}
    Test_Generation;

for each v∈FF(u) such that:
    v has been diagnosed in the last step and
    u received a dissemination message from v
do {propagates the dissemination message received by v if v has been diagnosed as
     fault-free}
    begin
        1_rb((w, FF(w), F(w)));     {(w, FF(w), F(w)) is the dissemination message previously
                                    stored}
        Disseminated[w]:=TRUE;
        {the diagnosis of u is extended}
        F(u):= F(u) ∪ F(w);
        FF(u):= FF(u) ∪ FF(w)
    end;

if N(u)⊆F(u)∪FF(u) and not WaitDiagnosis then
    begin
        {as soon as the state of all neighbors has been diagnosed, u disseminates its local
        diagnosis}
        1_rb((u, FF(u), F(u)));
        WaitDiagnosis:=TRUE;
    end;

until FF(u)∪F(u)=V; {the diagnosis protocol terminates when the state of all the
                       mobiles in the system has been diagnosed}
```

Table 2. The diagnosis protocol for fixed topology networks.

The dissemination of diagnostic information throughout the network is essential in order to ensure that all the mobiles in the system be correctly diagnosed by every fault-free unit. However, the propagation of the dissemination messages must be implemented carefully in order to avoid that soft-faulted mobiles corrupt them, thus leading to incorrect diagnosis. For this reason, any fault-free unit, upon receiving a dissemination message from a neighbor v, does not propagate it until mobile v has been diagnosed as fault-free. If this is not the case, the dissemination message is discarded. It should be noted that every fault-free unit either propagates or discards any dissemination message in time at most T_{out}.

Observe that, due to the shared nature of communication, the dissemination message generated by a fault-free unit u can be received more than once by other units. In order to avoid message implosion, every unit uses an array of boolean variables to keep track of the units whose dissemination message has been propagated.

Once unit u terminates its diagnosis protocol, it knows the state of all the other units in the system. However, if the diagnosis session is not terminated, some diagnostic messages may still be received by u. If this is the case, messages are processed as follows:

- test requests are served; however, the test task result is not stored.

- test responses are discarded;
- dissemination messages are either discarded or propagated depending on the state of the sending unit. In order to reduce diagnosis latency, the diagnostic information contained in the dissemination message is completed using the diagnosis of *u*.

Observe that, when a mobile *u* generates its test request, all of its neighbors, in turn, generate their test requests, and this could result in a burst of diagnostic messages. The introduction of a random delay before test generation would reduce this phenomenon. On the other hand, the use of random delays could increase the diagnosis latency.

6 PROTOCOL CORRECTNESS

A distributed diagnosis protocol is correct if, at the end of the diagnosis session, every fault-free unit correctly diagnoses the state of all the mobiles in the system.

The correctness proof is based on the following properties:

- *Local correctness*: every fault-free unit correctly diagnoses the state of its neighbors;
- *Dissemination correctness*: the dissemination message generated by any fault-free unit is correctly received by any other fault-free mobile in the system.

Local correctness is a consequence of Theorem 1, while dissemination correctness is proved in the following lemma:

Lemma 1. (Dissemination correctness)

Let G=(V,L) be the graph representing the system at the time of diagnosis. If G is connected and the total number of faulty mobiles in the system is at most k(G)-1, where k(G) is the connectivity[4] of G, then the dissemination message generated by a fault-free unit is correctly received by any other fault-free unit in the system in a finite time.

Proof.

Let *G'* be the subgraph induced on *G* by the set of fault-free units. Since the total number of faulty mobiles is less than $k(G)$, *G'* is connected. It follows that, given any pair *u*,*v* of fault-free units, there exists a path *P* from *u* to *v* that traverses only fault-free nodes. Suppose that unit *u* generates a dissemination message *m*. We have to show that *m* is correctly received by *v* in a finite time. The proof proceeds by induction on the length $l(P)$ of *P*. If $l(P)=1$, then *v* is a neighbor of *u*, hence, by property A2.2, *m* is correctly delivered to *v* by the link-level protocol. Message *m* is received only if *u* is diagnosed as fault-free by *v*. If this is the case, *m* is correctly received by *v* and the thesis follows. Otherwise, due to local correctness, *u* must be undiagnosed; therefore, message *m* is stored. Observe that an incoming dissemination message forces unit *v* to generate its test request (if the test request was not already generated). This means that eventually unit *u* will be diagnosed as fault-free; at that time, the message previously stored will be correctly received by *v* and the thesis follows. If $l(P)=h$, let *w* be the (h-1)-th

[4] The connectivity $k(G)$ of a graph *G* is the minimum number of vertices whose removal results in a disconnected or trivial graph [9].

unit traversed in P. Unit w is a neighbor of v and, by the inductive hypothesis, correctly received message m. Unit w updates its diagnosis according to m and sends m to its neighbors, possibly combining the original message with its own local diagnosis. The thesis follows by the same argument as in the case $l(P)=1$.

❑

We are now ready to prove that the diagnosis protocol is correct.

Theorem 3.

Let $G=(V,L)$ be the graph representing the system at the time of diagnosis. If G is connected and the total number of faulty mobiles in the system is at most $k(G)-1$, then every fault-free unit correctly diagnoses the state of all the mobiles in the system in finite time.

Proof.

By Theorem 1, any fault-free unit u correctly diagnoses the state of the units in $N(u)$. Once all the units in $N(u)$ have been diagnosed, a dissemination message m is generated and, by Lemma 1, it is correctly received by all the fault-free units in the system in finite time. Hence, all the fault-free units correctly receive the local diagnosis of any other fault-free mobile in finite time. The thesis follows by observing that, since G is connected and the total number of faulty mobiles is less than $k(G) \leq d_{min}$, where d_{min} is the minimum of node degrees in G, then every unit is adjacent to at least one fault-free mobile; hence, it is correctly diagnosed by at least one fault-free unit.

❑

7 PROTOCOL ANALYSIS

In this section we evaluate the *communication* and *time* complexities of the protocol. Due to the nature of ad-hoc networks, the communication complexity is defined as the total number of 1-hop reliable broadcasts performed during the diagnosis session. The time complexity is defined in terms of the diagnosis latency, i.e. the elapsed time between the inception and the end of the diagnosis session.

Theorem 4.

Let $G=(V,L)$ be the graph representing the system at the time of diagnosis. The communication complexity of the diagnosis protocol is $O(n(n+1+d_{max}))$, where $n=|V|$ and d_{max} is the maximum of the node degrees.

Proof.

Given any fault-free node u, it can be easily seen that u generates at most one test request m and at most one dissemination message m'. In turn, the test request m generates at most $|N(u)| \leq d_{max}$ test responses (one for every fault-free neighbors). Message m' is propagated at most once by every fault-free node, hence it generates at most n-1 further 1-hop reliable broadcast primitives. It follows that the total number of 1-hop reliable broadcast primitives performed during the diagnosis session is $O(n(n+1+d_{max}))$.

❑

Observe that using a more sophisticated broadcast protocol to disseminate diagnostic information could reduce the communication complexity of the protocol. However, the yield in the communication complexity is counterbalanced by the increased time needed to process dissemination messages. In turn, this could increase the time complexity of the diagnosis protocol. Since our design goal was to minimize diagnosis latency, we chose the simplest broadcast protocol, i.e. flooding.

Theorem 5.

Let $G=(V,L)$ be the graph representing the system at the time of diagnosis. Let T_{gen} be an upper bound to the elapsed time between the reception of the first diagnostic message and the generation of the test request, and let T_f be an upper bound to the time needed to propagate a dissemination message. The time complexity of the diagnosis protocol is $O(\Delta(T_{gen}+T_f)+T_{out})$, where Δ is the diameter of G.

Proof.

Any fault-free mobile generates its test request at most in time T_{gen} since the first diagnostic message is received. This means that at most in time $\Delta \cdot T_{gen}$ all the fault-free mobiles generate their test request. Once the test request is issued, any fault-free unit diagnose its neighbors at most in time T_{out}. Hence, the last dissemination message is generated at most in time $\Delta \cdot T_{gen}+T_{out}$. The thesis follows by observing that any dissemination message is received by all the fault-free units at most in time $\Delta \cdot T_f$. ❑

8 CONCLUSIONS

In this paper we addressed the problem of fault identification in ad-hoc networks. We presented a new comparison-based diagnostic model based on the one-to-many communication paradigm which takes advantage of the shared nature of communication typical of multi-hop packet radio networks. We presented two implementations of the model. The first implementation assumes that the network topology is fixed. Under this scenario, hard faults can be detected using a timeout, and efficient diagnosis protocols can be easily designed. If the fixed topology assumption is released, thus taking into account a relevant feature of ad-hoc networks, the "diagnostic efficiency" of the model decreases notably: hard-faults cannot be detected and fault-free nodes are no longer guaranteed to correctly diagnose all their neighbors within a certain time. This indicates that the design of diagnosis protocols ensuring correct diagnosis within a limited time could turn out to be very difficult under this scenario. It is our opinion that achieving correct diagnosis in the traditional sense (i.e., all the fault-free units of the system correctly diagnose the state of any other unit in finite time) in mobile systems be extremely hard, unless some restrictions on the mobility of the units are imposed. The identification of a "minimal" set of restrictions ensuring a somewhat weaker notion of correct diagnosis is matter of ongoing research.

REFERENCES

[1] A. Bagchi, and S. L. Hakimi, "An Optimal Algorithm for Distributed System Level Diagnosis", *Proc. FTCS-21*, pp. 214- 221, 1991.

[2] Barsi, F., Grandoni, F., and Maestrini, P., "A theory of diagnosability of digital systems". *IEEE Transactions on Computers,* Vol. C-25, No. 6, pp. 585-593, June 1976.

[3] S. Basagni, D. Bruschi, and I. Chlamtac, "A Mobility-Transparent Deterministic Broadcast Mechanism for Ad Hoc Networks", *IEEE Transactions on Networking*, vol. 7, No. 6, pp. 799 - 807, Dec. 1999.

[4] R. Bianchini, and R. Buskens, "Implementation of On-Line Distributed System-Level Diagnosis Theory", *IEEE Transactions on Computers*, vol. 41, No. 5, pp. 616 - 626, May 1992.

[5] D. M. Blough, and H. W. Wang, "The Broadcast Comparison Model for On-Line Fault Diagnosis in Multicomputer Systems: Theory and Implementation", *IEEE Transactions on Computers*, vol. 48, No. 5, pp. 470 - 493, May 1999.

[6] Blount, M., L., "Probabilistic Treatment of Diagnosis in Digital Systems", *Proceedings of the 7th Fault Tolerant Computing Symposium*, 1977, pp. 72-77.

[7] Chwa, K., Y. and Hakimi, S., L., "Schemes for Fault-Tolerant Computing: A Comparison of Modulary Redundant and t-Diagnosable Systems", *Information and Controls*, Vol. 45 No. 3, pp. 212-238, 1981.

[8] W. Diepstraten, G. Ennis, and P. Berlanger, "DFWMAC: Distributed Foundation Wireless Medium Access Control", IEEE Document P802.11-93/190 (November 1993).

[9] F. Harary, *Graph Theory*, Reading, MA, Addison-Wesley, 1972.

[10] S. Hosseini, J. Kuhl, and S. Reddy, "A Diagnosis Algorithm for Distributed Computing Systems with Dynamic Failure and Repair", *IEEE Transactions on Computers*, Vol. 33, No. 3, pp. 223 - 233, Mar. 1984.

[11] Karunanithi, S. and Friedman, A., "System Diagnosis with *t/s* Diagnosability", *Proceedings of the 7th Fault Tolerant Computing Symposium*, 1977, pp. 65-71.

[12] J. Maeng, and M. Malek, "A Comparison Connection Assignment for Self-Diagnosis of Multiprocessor Systems", *Proc. FTCS-11*, pp. 173 - 175, 1981.

[13] Maheshwari, S., N. and Hakimi, S., L., "On Models for Diagnosable Systems and Probabilistic Fault Diagnosis", *IEEE Transactions on Computers*, Vol. C-25, No. 3, pp. 228-236, March 1976.

[14] S. Murthy, and J. J. Garcia-Luna-Aceves, "An Efficient Routing Protocol for Wireless Network", *Mobile Networks and Applications*, vol. 1, No. 2, pp. 183 - 197, 1996.

[15] E. Pagani, and G. P. Rossi, "Reliable Broadcast in Mobile Multihop Packet Networks", *Proc. of MOBICOM'97*, Budapest, pp. 34 - 42, 1997.

[16] M. Pauly, M. Finke, L. Peters, and K. Beck, "Control and Service Structure of a Robot Team", *Proc. of IEEE/RSJ International Conference on Intelligent Robots and Systems*, Korea, pp. 1069 - 1074, 1999.

[17] C. E. Perkins, and P. Bhagwat, "Highly Dynamic Destination-Sequenced Distance-Vector Routing (DSVD) for Mobile Computers", *Proc. of SIGCOMM'94*, London, pp. 234 - 244, 1994.

[18] F.P. Preparata, G. Metze, and R.T. Chien, "On the Connection Assignment Problem of Diagnosable Systems", *IEEE Transactions on Computers*, vol. EC-16, pp. 848 - 854, December 1967.

[19] S. Ramanathan, and M. Steenstrup, "A Survey of Routing Techniques for Mobile Communication Networks", *Mobile Networks and Applications*, vol. 1, No. 2, pp. 89 - 104, 1996.

[20] A. Sengupta, and A. T. Dahbura, "On Self-Diagnosable Multiprocessor Systems: Diagnosis by the Comparison Approach", *IEEE Transactions on Computers*, Vol. 41, No. 11, pp. 1386 - 1396, Nov. 1992.