Fault Tolerance in Collaborative Sensor Networks for Target Detection

Thomas Clouqueur, *Student Member*, *IEEE*, Kewal K. Saluja, *Fellow*, *IEEE*, and Parameswaran Ramanathan, *Senior Member*, *IEEE*

Abstract—Collaboration in sensor networks must be fault-tolerant due to the harsh environmental conditions in which such networks can be deployed. This paper focuses on finding algorithms for collaborative target detection that are efficient in terms of communication cost, precision, accuracy, and number of faulty sensors tolerable in the network. Two algorithms, namely, value fusion and decision fusion, are identified first. When comparing their performance and communication overhead, decision fusion is found to become superior to value fusion as the ratio of faulty sensors to fault free sensors increases. As robust data fusion requires agreement among nodes in the network, an analysis of fully distributed and hierarchical agreement is also presented. The impact of hierarchical agreement on communication cost and system failure probability is evaluated and a method for determining the number of tolerable faults is identified.

Index Terms—Collaborative target detection, decision fusion, fault tolerance, sensor networks, value fusion.

1 Introduction

Recent advances in computing hardware and software are responsible for the emergence of sensor networks capable of observing the environment, processing the data, and making decisions based on the observations. In particular, the development of technologies such as Bluetooth [3] or the IEEE 802.11 standard [16] enables us to connect the nodes together wirelessly. This allows for deployment of ad hoc networks that do not require backbone infrastructure. This, together with progress in sensing and computing technology, give rise to many new applications. Such sensor networks can be used to monitor the environment, detect, classify, and locate specific events, and track targets over a specific region. Examples of such systems are in surveillance, monitoring of pollution, traffic, agriculture, or civil infrastructures [23].

The essence of sensor networks is to have nodes within a region make local observations of the environment and collaborate to produce a global result that reflects the status of the region covered [5]. This collaboration requires local processing of the observed data, communication between different nodes, and information fusion. For many applications, the network is deployed in a harsh environment and some of the nodes may be faulty or may fail during the network's lifetime, thus requiring collaboration to be robust to node failures. Two other constraints in wireless networks of autonomous nodes come from the limited bandwidth and power source of these elements, requiring collaboration to be communication and power efficient.

• *The authors are with the Department of Electrical and Computer Engineering, 1415 Engineering Dr., University of Wisconsin-Madison, Madison, WI 53706-1691.*
E-mail: clouqueu@ece.wisc.edu, {saluja, parmesh}@engr.wisc.edu.

Manuscript received 17 Apr. 2002; revised 30 Jan. 2003; accepted 6 June 2003. For information on obtaining reprints of this article, please send e-mail to: tc@computer.org, and reference IEEECS Log Number 116370.

Thus, the challenges of sensor networks include distributed signal processing that makes use of the processing power of all the nodes, ad hoc routing, and communication protocols that enable information sharing among nodes and fault tolerance that accounts for the possible misbehaviors of a subset of the nodes. All these challenges need to cope with the power constraint of the network. This paper focuses on finding and analyzing algorithms for robust collaborative target detection. Therefore, it addresses both the distributed signal processing and fault tolerance challenges. This work completes a preliminary study presented in [6]. The basic premise of target detection is that sensors are deployed over a region of interest and are required to determine if a target is present in that region. In general, targets emit signals characterizing their presence in the region that can be measured by the sensors. For example, a vehicle produces a sound when traveling on a road or when at rest if the engine is on. The strength of this signal usually decreases with distance and the presence of noise makes the target more difficult to sense as the distance between target and sensor increases. Since sensors are, in general, spread over the region, they measure signals of different strength and, were they merely to use their own measurement, they may conclude differently upon the presence or absence of a target in the region. Therefore, sensor nodes need to collaborate by exchanging and fusing their local information to produce a result global to the region. The presence of faulty sensor nodes affects this fusion process and can potentially corrupt the detection result. Algorithms for target detection need to specify a way to fuse the signals measured at each sensor to produce one consistent and useful result characterizing the whole region. These algorithms can be evaluated for their performance in terms of accuracy, communication overhead, and robustness to sensor node failure.

This paper first presents the sensor model considered and formulates the target detection problem in Section 2. Section 3

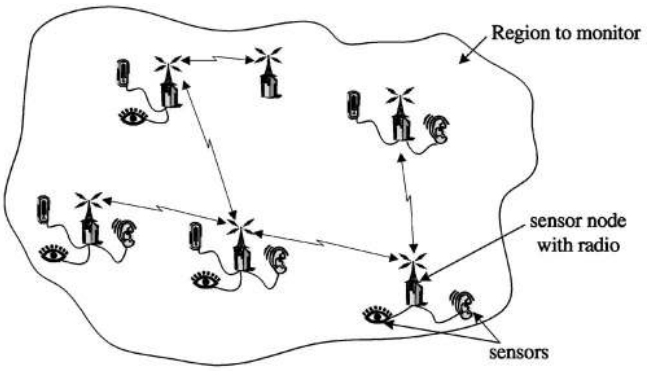

Fig. 1. Wireless sensor network.

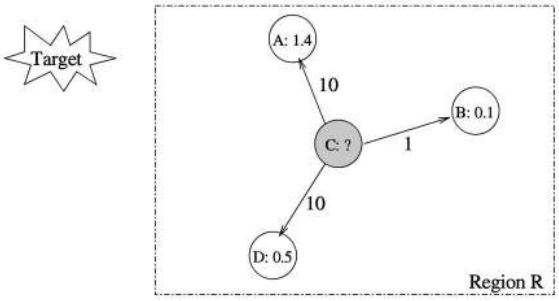

Fig. 2. Byzantine faulty behavior.

presents previous work conducted in signal processing and distributed systems fault tolerance that is relevant to this study. Section 4 presents two detection schemes and develops an analytical model of the system. Section 5 gives performance results of the algorithms presented in Section 4. Section 6 presents an approach for reducing the communication overhead of the fusion algorithms developed and the paper concludes with Section 7.

2 Model and Problem Formulation

This section presents the model for the sensor network and formulates the target detection problem being investigated. The sensor network is assumed to be composed of a set of nodes connected to sensors as presented in Fig. 1, called sensor nodes or simply nodes. A model is developed for each sensor node in fault-free and faulty mode and for the collaboration among nodes. The target is modeled by the signal it emits.

2.1 Sensors for Target Detection

Sensor nodes, with possibly different sensing modalities, are deployed over a region R to perform target detection. Sensors measure signals at a given sampling rate to produce time series that are processed by the nodes. The nodes fuse the information obtained from every sensor according to the sensor type and location to provide an answer to a detection query. The nodes are assumed to have the ability to communicate with each other. However, this work is not concerned with communication issues and, therefore, the node peer-to-peer communication is assumed to be reliable through the use of appropriate communication techniques [17], [25].

This work assumes that the sensor nodes obtain a target energy measurement after T seconds while a target was at a given position inside or outside the region R. Obtaining that energy requires preprocessing of the time series measured during period T and, possibly, fusion of data from different sensors by each node. The detection algorithm consists of exchanging and fusing the energy values produced by the sensor nodes to obtain a detection query answer. Note that a more accurate answer can be obtained in general if the sensors exchange their time series rather than energies; however, that would require high communication bandwidth that may not be available in a sensor network. The performance of fusion is partly defined by the accuracy that measures how well sensor decisions represent the environment or "ground truth."

Finally, this work assumes that detection results need to be available at each node. The reason for such a requirement is that the results can be needed for triggering other actions such as localization of the target detected. This requirement can be fulfilled by having a central node make a decision and disseminate that decision to all the nodes in the network. The correctness of such a scheme relies on the central node's correctness, therefore, central node-based schemes have low robustness to sensor failure. Distributing the decision making over several or all the nodes improves reliability at the cost of communication overhead. That trade off is analyzed in detail in Section 6.

2.2 Fault Model

The network considered is likely to contain faulty sensor nodes due to harsh environmental conditions. Faults include misbehaviors ranging from simple crash faults, where a node becomes inactive, to Byzantine faults, where the node behaves arbitrarily or maliciously. In this work, faulty nodes are assumed to send inconsistent and arbitrary values to other nodes during information sharing. Fig. 2 gives an example of such behavior where four nodes, A, B, C, and D, measure energy values to determine if a target is in region R.

As the target is outside region R, sensor A measures an energy level of 1.4 (including noise), whereas sensors B and D measure an energy level of 0.1 and 0.5, respectively. Sensor C is assumed to be faulty and sends different measurements to the other sensors (10, 1, and 10 to A, B, and D, respectively). As a result, nonfaulty sensors obtain different global information about the region and may conclude differently on the presence of the target (e.g., sensors A and D may conclude that a target is present while sensor B concludes that no target is present).

The algorithm for target detection needs to be robust to such inconsistent behavior that can jeopardize the collaboration in the sensor network. For example, if the detection results trigger subsequent actions at each node, then inconsistent detection results can lead each node to operate in a different mode, resulting in the sensor network going out of service. The performance of fusion is therefore also defined by precision [4], [18]. Precision measures the closeness of decisions from each other, the goal being that all nodes obtain the same decision.

2.3 Target Energy

A target at location u emits a signal which is measured by the sensors deployed at locations s_i, $i = 1, \ldots, n$. The strength of the signal emitted by the target decays as a polynomial of the distance. If the decay factor is k, the signal energy of a target at location u measured by a sensor at location s_i is given by:

$$S_i(u) = \begin{cases} K.T, & \text{if } d < d_0; \\ \frac{K.T}{(d/d_0)^k}, & \text{otherwise,} \end{cases} \quad (1)$$

where K is the power emitted by the target during time T and $d = ||u - s_i||$ is the geometric distance between the target and the sensor and d_0 is a constant that accounts for the physical size of the sensor and the target. Depending on the environment, e.g., atmospheric conditions, the value k typically ranges from 2.0 to 5.0 [15]. The energy $S_i(u)$ also depends on the possible presence of obstacles lying between the target and the sensors. But, this study assumes no such obstacles to be present in the region considered.

Energy measurements at a sensor are usually corrupted by noise. If N_i denotes the noise energy at sensor i during a particular measurement, then the total energy measured at sensor i when the target is at location u is

$$E_i(u) = S_i(u) + N_i. \quad (2)$$

3 Previous Work

Detecting targets' signals in a noisy environment can make use of decision theory, a well-developed branch of engineering and mathematics. For the distributed detection problem, results from data fusion theory are also of primary interest. Further, robustness to inconsistent sensor nodes behavior is also related to studies carried out on the consensus problem and general fault tolerance theory. Below, these concepts are described along with their relation to the work presented in this paper.

3.1 Distributed Detection

Classical multisensor detection assumes that all local sensors communicate their data to a central processor performing optimal or near optimal detection using conventional statistical techniques [14]. Later studies, however, focused on decentralized processing in which some preliminary processing of data is performed at each sensor node so that compressed information is gathered at the fusion center [27]. Decentralizing the detection results in a loss of performance compared to the performance of centralized systems since the fusion center of a decentralized system has only part of the information collected by the sensor nodes. However, decentralized schemes require reduced communication bandwidth and it will be argued in this paper that they may achieve increased reliability. Further, the performance loss of decentralized schemes may be reduced by optimally processing the information at each sensor node.

Fig. 3 illustrates the sensor network parallel topology where N sensors measure a signal y_i produced by a phenomenon H [28]. Each sensor S_i processes its signal y_i to generate a quantized information u_i and all the u_is are then fused into u_0 at the fusion center. In the binary hypothesis testing problem, the observations at all the sensors either correspond to the presence of a target (hypothesis H_1) or to the absence of a target (hypothesis H_0). The performance of detection is measured by the false alarm probability P_F and the probability of miss P_M. The false alarm probability is the probability of concluding that a target is present when the target is absent, i.e., $P_F = P(u_0 = 1 \mid H_0)$. The miss probability is the probability of concluding that a target is absent when a target is actually present, i.e., $P_M = P(u_0 = 0 \mid H_1)$.

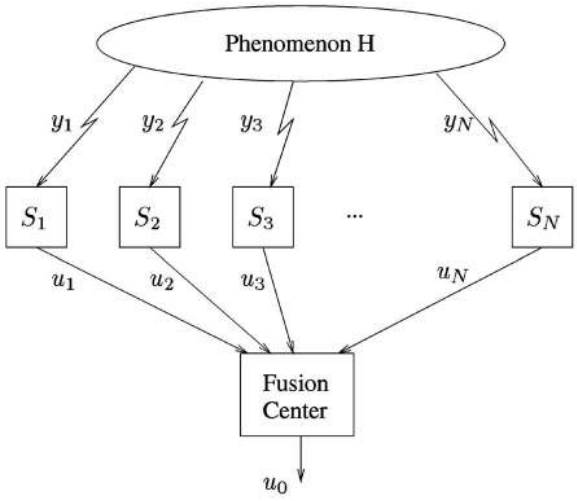

Fig. 3. Parallel topology for distributed detection.

The Neyman-Pearson criterion can be used to find optimum local and global decision rules that minimize the global probability of miss P_M assuming that the global probability of false alarm P_F is below a given bound α [27]. For this criterion, the mapping rules used at the nodes to derive u_i and the decision rule at the fusion center are threshold rules based on likelihood ratios [22].

The thresholds used at each sensor and at the fusion center need to be determined simultaneously to minimize the miss probability P_M under the constraint $P_F \leq \alpha$. This is a difficult optimization problem since the number of fusion rules to be considered, i.e., the number of choices for thresholds, is large. The problem becomes somewhat tractable when assuming conditional independence of sensor observations and when limiting the number of quantization levels used for values of u_is. Although many studies assume u_is to be binary values, design of multilevel quantizers for distributed hypothesis testing has also been considered [27]. Increasing the number of levels improves the system performance and coding quantized values into 3 bits was shown to give a near-optimum solution, i.e., performance close to the one of the centralized system [19].

The Neyman-Pearson criterion is applicable when observation statistics under each hypothesis are completely known a priori. This is often not the case and probability distribution may be known approximatively or very coarsely. When detecting a target in a region, the probability distribution depends on the noise, the target emitted energy, and its position, which are unknown a priori. If only very coarse information about the observations is available, detection performance can be guaranteed only by nonparametric

techniques. Such techniques usually make some general assumptions about observation statistics, such as symmetry of the probability density functions or continuity of the cumulative distribution functions. Most nonparametric detectors employ the sign or the rank of the observed samples, two common examples being the sign detector and the Wilcoxon detector [2], [12].

3.2 Agreement Problem and Fault Tolerance

Building a robust sensor network for target detection requires an understanding of the agreement problem in unreliable distributed systems. As processors in such a system cooperate to achieve a specified task, they often have to agree on a piece of data that is critical to subsequent computation. Although this can be easily achieved in the absence of faulty processors, for example, by simple message exchange and voting, special protocols need to be used to reach agreement in the presence of inconsistent faults, as presented in Section 2. Three problems have drawn much attention in trying to develop these protocols, namely, the *consensus problem*, the *interactive consistency problem*, and the *generals problem* [1], [11]. The consensus problem considers n processes with initial values x_i and these processes need to agree on a value $y = f(x_1, \ldots, x_n)$, with the goal that each nonfaulty process terminates with a value $y_i = y$. Further, the protocols for consensus need to be nontrivial, i.e., the consensus value y must depend on the initial values x_i and should not be just a constant. They also need to meet the unanimity requirement, i.e., the consensus value is $y = x$ if all nonfaulty processes have the same initial value x. The interactive consistency problem is like the consensus problem with the goal that the nonfaulty processes agree on a vector $y = (y_1, \ldots, y_n)$ with $y_i = x_i$ if process i is nonfaulty. Finally, the generals problem considers one specific processor, named "general," trying to broadcast its initial value x to other processors with the requirement that all nonfaulty processes terminate with identical values y and $y = x$ if the general is nonfaulty.

The consensus problem can be solved using a protocol for interactive consistency by having each processor apply the function f to its result vector y_i. Also, the interactive consistency problem can be solved using n copies of a protocol for the generals problem. Finally, the generals problem can be solved using a protocol for the consensus problem by having the general broadcast its value to all processes and having the processes agree on the value sent. This shows that the three problems are equivalent when considering their solvability, although optimal protocol for each problem may not make use of the protocols for the other problems.

The solvability of the problems and the complexity of solutions depend on the models for computation and the kind of faults considered. Further, distinction needs to be made between synchronous and asynchronous computations. A synchronous system proceeds in a sequence of rounds of fixed duration where each process sends messages to other processes and then receives messages from other processes. Only synchronous systems are considered in this paper. A distinction is also made between authenticated and unauthenticated message exchange. Authenticated protocols assume that messages exchanged by the processes are signed so that the receiver of a message can reliably know the sender and/or originator of the message. Unauthenticated protocols assume that messages exchanged by the processors are not signed. Regarding process failure, two types of faults are often considered, namely, crash and Byzantine faults. The crash fault model assumes that, when a processor fails, it stops all activity. The Byzantine fault model assumes that a faulty processor can behave in an arbitrary manner, in particular acting maliciously against the protocol. A protocol is said to be t-resilient if it runs correctly when no more than t out of N processes fail before or during operation. The following results were derived regarding the generals problem under different assumptions.

Theorem 1. *There is a t-Byzantine resilient authentication synchronous protocol which solves the generals problem [18].*

Theorem 2. *There is a t-Byzantine resilient synchronous protocol without authentication which solves the generals problem if and only if $t/N < 1/3$ [21].*

A commonly used protocol for the generals problem, namely, the oral message algorithm (OM), was developed in [18]. Whenever t out of N nodes are faulty, the $\mathrm{OM}(t)$ algorithm is guaranteed to provide agreement among the N nodes if and only if $N \geq 3t + 1$. Below is an example where a protocol for the generals problem is used to reach interactive consistency among the four nodes of Fig. 2. After the OM algorithm is performed for each node acting as general, inconsistent values sent by node C are replaced by a common value (i.e., 10.0) also, node C is not able to corrupt the values broadcasted by nodes A, B, and D. Note that, in this example, the final decisions of the nonfaulty sensors are incorrect since the target is outside the region of interest; however, the decisions are consistent.

The overhead of agreement protocols can be measured by the number of rounds and the number of messages required. The t-Byzantine resilient $\mathrm{OM}(t)$ algorithm presented above requires $t + 1$ rounds and a number of messages that is exponential in t. Protocols using fewer messages but requiring more rounds were developed; however, it was shown that the number of rounds required in the presence of t faults is at least $t + 1$. Stopping criteria can be used to reduce the number of rounds necessary whenever fewer than t processors are faulty. A large variety of results have been derived to address the complexity issue that depend on the assumptions on the computation model and fault model [9], [8], [10].

In applications where processors hold an estimate of some global value, it may be sufficient to guarantee that the nodes agree on values that are not exactly identical but are relatively close (by some predetermined measure) to one another. This was formulated as the approximate or inexact agreement problem. Protocols for approximate agreement require, in general, fewer messages exchanged among nodes at the cost of degraded precision [7], [20], [4]. However, it is not the focus of this paper to study the trade off between precision and communication overhead. In this study, protocols that achieve perfect precision are mainly used, although less precise protocols could be applicable. Finally, some agreement protocols attempt to diagnose the identity of faulty nodes so that diagnosis results can be used in subsequent agreement procedures

TABLE 1
Example of Interactive Consistency, Sensor C Being Faulty

	A	B	C	D
Value measured	1.4	0.1	?	0.5
Value received from C	10.0	1.0	-	10.0
Set S after running 4 OM(1)	$\{1.4, 0.1, 10.0, 0.5\}$	$\{1.4, 0.1, 10.0, 0.5\}$	?	$\{1.4, 0.1, 10.0, 0.5\}$
Final decision	1	1	?	1

[26]. Again, such protocols are not considered in this study although they could be applicable and improve the reliability of the system.

4 Algorithms and Analytical Mmodel

This section presents two algorithms for target detection: value fusion and decision fusion. Then, it develops analytical expressions for the false alarm and detection probabilities for both algorithms in the presence and in the absence of faults.

4.1 Algorithms

The problem of target detection differs from previously studied problems in distributed signal detection because of the presence of faults that require special processing of the data. The problem also differs from previously studied problems in agreement in the sense that nodes sharing information may contain local information that can be totally different from one node to another. In target detection, nodes close to the target report high energy measurement, while nodes far from the target report low energy measurements. *Thus, in fusion, there is a lack of common truth in the measured values. Yet, it is desirable to arrive at a common value or common values and determine the impact of faults in the methods developed to arrive at consensus.*

The algorithms considered are nonparametric detection algorithms that let the nodes share their information and use a fusion rule to arrive at a decision. They use exact agreement to reach consensus, although other agreement types such as inexact agreement might be appropriate. Exact agreement guarantees that all the nonfaulty nodes obtain the same set S of data and the data sent by the nonfaulty nodes are part of this set. However, consistent outlying data can remain in the set, as shown in Table 1. To prevent corruption of the decision by these outliers, the largest and smallest data values are dropped from the set S and the average data is computed over the remaining data. Different fusion algorithms can be derived by varying the size of the information shared between sensor nodes. Two extreme cases are explored: 1) *value fusion* where the nodes exchange their raw energy measurements and 2) *decision fusion* where the nodes exchange local detection decisions based on their energy measurement [6]. Both algorithms are described below.

Value Fusion (with faults)
at each node:

1. obtain energy from every node
2. drop largest n and smallest n values
3. compute average of remaining values
4. compare average to threshold for final decision

Decision Fusion (with faults)
at each node:

1. obtain local decision from every node
2. drop largest n and smallest n decisions
3. compute average of remaining local decisions
4. compare average to threshold for final decision

In the case where all the nodes are known to be nonfaulty, there is no need for dropping data and the algorithms reduce to:

Value Fusion (fault-free)
at each node:

1. obtain energy from every node
2. compute average of values
3. compare average to threshold for final decision

Decision Fusion (fault-free)
at each node:

1. obtain local decision from every node
2. compute average of local decisions
3. compare average to threshold for final decision

4.2 Evaluation Metrics

The algorithms presented in the previous subsection can be evaluated with the following metrics: precision, accuracy, communication overhead, and robustness. Precision, as introduced in Section 2, measures the closeness of the final decisions obtained by all sensors, the goal being that all nonfaulty nodes obtain the same decision. Since the algorithms developed use exact agreement during fusion and then redundancy during dissemination, consistency among nonfaulty nodes is guaranteed as long as the number of faulty nodes doesn't exceed the necessary bound. If this bound is exceeded, no level of precision is guaranteed and the system is considered failing. Accuracy, also introduced in Section 2, measures how well the node decisions represent the environment, the goal being that the decision of nonfaulty nodes is "object detected" if and only if a target is present. Again, if the bound on the number of faulty nodes tolerable is exceeded, no level of accuracy is guaranteed and the system is considered failing. However, when this bound is not exceeded, only relative accuracy is attainable due to background noise. The accuracy is measured by the false alarm probability and the detection probability introduced in Section 3.

False alarm and detection probability are determined by the threshold defined in the algorithms of Section 4.1, the noise level, the target position, and the node layout. Analytical models for false alarm and detection probability are developed in the next section. The last two metrics are communication overhead and robustness. The communication overhead can be evaluated by counting the number and the size of messages exchanged. This is not the focus of this paper and only qualitative evaluations are made. Also, system failure occurs when the bound on the number of faulty nodes acceptable is violated and robustness is measured by the system failure probability, as discussed in Section 6 of this paper.

4.3 Comparison of Algorithms

To compare the fusion algorithms identified in this section, these different metrics need to be evaluated in the absence and in the presence of faulty nodes. As mentioned, the communication overhead will not be evaluated here and will only be used qualitatively. Due to the reduced amount of information shared in decision fusion, the communication cost is lower in decision fusion than in value fusion. The system failure probability is identical for value and decision fusion since failures depend on the number of faulty nodes present and not on the algorithm used. However, the performance measured in terms of precision and accuracy differs from one option to the other. The exact agreement guarantees final consistency among nonfaulty nodes and, therefore, the faulty nodes can only degrade the accuracy of the system. Let us now identify those faulty behaviors that have highest impact on accuracy to identify the worst-case scenario.

Assume that a faulty node reports inconsistent values $v_1 \geq v_2 \geq \ldots \geq v_N$ to the other nodes. During exact agreement, the nodes will remove the inconsistency and agree on a value $\hat{v}$ between v_1 and v_N. However, if the faulty node reports the value v consistently, then $\hat{v} = v$. Furthermore, in the absence of a target, the accuracy is lower if the value obtained for the faulty node after exact agreement is high (i.e., $\hat{v} = v_N$) and, in the presence of a target, the accuracy is lower if the value obtained for the faulty node after exact agreement is low (i.e., $\hat{v} = v_1$). From these remarks, we conclude that, when using exact agreement, a faulty node acting consistently can degrade the system accuracy more than a faulty node acting inconsistently. Therefore, the performance evaluation was restricted to the case where all the nodes act consistently as this applies for the worse-case scenario in the faulty case.

4.4 Performance

This subsection derives equations for false alarm and detection probability in the absence and in the presence of faulty nodes. It first gives a set of notations used in the equations.

4.4.1 Notations and Assumptions

Let N be the total number of sensors. n is the number of maximum and minimum values dropped in fault tolerant fusion. t is the number of faulty sensors in the network. η_v and η_d are the thresholds for value and decision fusion. α is the second threshold for decision fusion. P_{fa} and P_d are the false alarm and detection probabilities. $f_X(x)$ is the probability density function (pdf) of noise energy that is assume to be χ_1^2: chi-square with one degree of freedom [13]. Since the sensors hold energy values that are squares of the signal time series, the chi-square assumption for the noise corresponds to the common Gaussian assumption for the signal. The pdf is given by

$$f_X(x) = \frac{1}{\sqrt{2\pi x}} \exp\left[-\frac{x}{2}\right] I_{(0,\infty)}(x). \tag{3}$$

$F_X(x)$ is the cumulative distribution function (cdf) of noise and is given by The noise has χ_1^2 distribution, therefore, the cdf $F_X(x)$ can be expressed as:

$$F_X(x) = \operatorname{erf}\left(\frac{\sqrt{x}}{\sqrt{2}}\right), \tag{4}$$

where erf is the error function given by

$$\operatorname{erf}(x) = \frac{2}{\sqrt{\pi}} \int_0^x e^{-u^2} du.$$

Furthermore, the noise at different sensors is assumed to be independent.

4.4.2 Value Fusion

Expressions for the false alarm probability and the detection probability in the absence of faults are first derived.

False alarm probability. For non-fault-tolerant value fusion, false alarms occur when the average of the N values measured by the sensors is above the threshold η_v in the absence of target. The measured values contain noise and the false alarm probability is given by:

$$\begin{aligned} P_{fa} &= P\left[\frac{1}{N}\sum_{i=1}^{N} N_i > \eta_v\right] \\ P_{fa} &= 1 - P\left[\sum_{i=1}^{N} N_i \leq N\eta_v\right]. \end{aligned} \tag{5}$$

$\sum_{i=1}^{N} N_i$ is Chi-square noise with N degrees of freedom with a cumulative distribution function $F_{\chi_N^2}(x)$. Therefore, (5) becomes:

$$P_{fa} = 1 - F_{\chi_N^2}(N\eta_v). \tag{6}$$

Detection probability. For non-fault-tolerant value fusion, detections occur when the average of the N values measured by the sensors is above the threshold η_v in the presence of target. The values measured consist of energy (function of the distance of the target from the sensor) plus noise and the false alarm probability for a given position of target u is given by:

$$\begin{aligned} P_d(u) &= P\left[\frac{1}{N}\sum_{i=1}^{N}(E_i(u) + N_i) > \eta_v\right] \\ P_d(u) &= P\left[\sum_{i=1}^{N} N_i > N\eta_v - \sum_{i=1}^{N} E_i(u)\right] \\ P_d(u) &= 1 - F_{\chi_N^2}\left(N\eta_v - \sum_{i=1}^{N} E_i(u)\right). \end{aligned} \tag{7}$$

For varying position of target, the detection probability is given by:

$$P_d = \left\langle 1 - F_{\chi_N^2}\left(N\eta_v - \sum_{i=1}^{N} E_i(u)\right)\right\rangle, \tag{8}$$

where $\langle f(u)\rangle$ denotes the average of f over different positions u of the target in the region considered.

Expressions for the false alarm probability and the detection probability in the presence of faults are now derived.

False alarm probability. The false alarm probability given that t faults are present is determined in the worst-case scenario, i.e., t sensors report the maximum allowed value. In fault-tolerant value fusion, the n highest and

n lowest values are dropped so that the decision is based on the $N-2n$ middle-range values. Let $w_i, 1 \le i \le N-2n$ be the $N-2n$ values that are not dropped (with $w_1 \le w_2 \le \cdots \le w_{N-2n}$ and $-\infty \le w_i \le +\infty$). False alarms occur when the average of w_i is above the threshold η_v. There are $\binom{N-t}{n}$ ways of choosing the n sensors that have lowest values (i.e., value less than w_1) and the probability for each of these sensors to have a value lower than w_1 is $F_X(w_1)$. There are $\binom{N-n-t}{n-t}$ ways of choosing the $m-t$ nonfaulty sensors that have highest values (i.e., values greater than w_{N-2m}) and the probability for each of these sensors to have a value greater than w_{N-2m} is $1-F_X(w_{N-2m})$. The probability for the remaining $N-2n$ sensors to have value $w_1, w_2, \ldots, w_{N-2n}$ is $f_X(w_1), f_X(w_2), \ldots, f_X(w_{N-2n})$ and there are $(N-2n)!$ possible permutations of these sensors (these permutations need to be considered since the values w_i are ordered). Therefore, the false alarm probability is given by:

$$P_{fa} = \underbrace{\int_{w_1=0}^{+\infty}\int_{w_2=w_1}^{+\infty}\cdots\int_{w_{N-2n}=w_{N-2n-1}}^{+\infty}}_{\frac{1}{N-2n}\sum_{i=1}^{N-2n} w_i \ge \eta_v} \binom{N-t}{n} F_X(w_1)^n \binom{N-n-t}{n-t} (1-F_X(w_{N-2n}))^{n-t}(N-2n)! \prod_{j=1}^{N-2n} f_X(w_j)dw_j. \tag{9}$$

Detection probability. The following terms are now defined before deriving the expressions for detection probability.

Definition. *Let P_n be the set of one-to-one functions from $\{1 \le i \le n\}$ to $\{1 \le i \le n\}$. P_n is the set of permutations of n elements and* $\mathrm{card}(P_n) = n!$.

Definition. *Elements p_1, p_2 of the set P_n are said to be R_k $(k, 0 \le k \le n)$ related provided*

$$\{p_1(i), 1 \le i \le k\} = \{p_2(i), 1 \le i \le k\}.$$

It can be easily shown that R_k is an equivalence relation.

Let $C_{k,n}$ be the set P_n under the relation R_k. $C_{k,n}$ is the set of combinations of k objects among n. For example, to choose a combination of $N-t$ nonfaulty sensors among N sensors, let $f \in C_{N-t,N}$ so that $\{f(i), 1 \le i \le N-t\}$ is the set of indices of nonfaulty sensors. Note that $\mathrm{card}(C_{k,n}) = \binom{n}{k}$.

The detection probability given that t faults are present is determined in the worse case scenario, i.e., t sensors report the minimum allowed value. In fault-tolerant value fusion, after dropping the n highest and n lowest values $N-2n$ values are left, $w_i, 1 \le i \le N-2n$ (with $w_1 \le w_2 \le \cdots \le w_{N-2n}$ and $-\infty \le w_i \le +\infty$). Detections occur when the average of values $(w_i + E_i(u))$ is above the threshold η_v. Since the energy measured is a function of the position of the sensor, the detection probability depends on which sensors are faulty and which nonfaulty sensor values are dropped. The N sensors are divided into t faulty sensors with low values, $n-t$ nonfaulty sensors with low values that are dropped, n nonfaulty sensors with high values that are dropped, and $N-2n$ nonfaulty sensors with middle values that are not dropped. Let $f \in C_{N-t,N}$ be the combination which represents the nonfaulty sensors (i.e., $\{f(i), 1 \le i \le N-t\}$ is the set of indices of nonfaulty sensors and $\{f(i), N-t+1 \le i \le N\}$ is the set of indices of faulty sensors). Similarly, let $h \in C_{N-t-n,N-t}$ be the combination which represents the $N-t-n$ nonfaulty sensors that do not have highest values. Also, let $l \in C_{N-2n,N-t-n}$ be the combination which represents the $N-2n$ remaining nonfaulty sensors that do not have lowest values. And, let $p \in P_{N-2n}$ be a permutation ordering the remaining $N-2n$ nonfaulty sensors. The probabilty that a given set of values $\{w_i\}$ is obtained for given f, h, l, and p is:

$$\begin{aligned} &\prod_{i=1}^{n}\left(1 - F_X(max_{f,h,l,p}(u,w) - E_{f(h(i+N-n-t))}(u))\right) \\ &\prod_{j=1}^{n-t} F_X(min_{f,h,l,p}(u,w) - E_{f(h(l(j+N-2n)))}(u)) \\ &\prod_{k=1}^{N-2n} f_X(w_k), \end{aligned} \tag{10}$$

where:

$$max_{f,h,l,p}(u,w) = \max_{1\le i\le N-2n}\left(E_{f(h(l(p(i))))}(u) + w_i\right) \tag{11}$$

$$min_{f,h,l,p}(u,w) = \min_{1\le i\le N-2n}\left(E_{f(h(l(p(i))))}(u) + w_i\right). \tag{12}$$

Integrating (10) over the sets $\{w_i\}$ that trigger a detection and over the possible permutations p and combinations l and h, and averaging over the different combinations of faulty sensors f and the different target positions u, the detection probability is given by:

$$\begin{aligned} P_d = \Bigg\langle \frac{1}{\binom{N}{n}} &\sum_{f\in C_{N-t,N}} \sum_{h\in C_{N-n-t,N-t}} \sum_{l\in C_{N-2n,N-n-t}} \\ &\underbrace{\int_{w_1=-\infty}^{+\infty}\int_{w_2=w_1}^{+\infty}\cdots\int_{w_{N-2n}=w_{N-2n-1}}^{+\infty}}_{\frac{1}{N-2n}\sum_{i=1}^{N-2n} w_i \ge \eta_v - \frac{1}{N-2n}\sum_{i=1}^{N-2n} E_{f(h(l(i)))}(u)} \sum_{p\in P_{N-2n}} \\ &\prod_{i=1}^{n}\left(1 - F_X(max_{f,h,l,p}(u,w) - E_{f(h(i+N-n-t))}(u))\right) \\ &\prod_{j=1}^{n-t} F_X(min_{f,h,l,p}(u,w) - E_{f(h(l(j+N-2n)))}(u)) \\ &\prod_{k=1}^{N-2n} f_X(w_k)dw_k \Bigg\rangle. \end{aligned} \tag{13}$$

4.4.3 Decision Fusion

Expressions for the false alarm probability and the detection probability in the absence of faults are first derived.

False alarm probability. For decision fusion, false alarms occur when more than αN sensors have a value above the threshold η_d in the absence of target, where α is the threshold used in Step 3 of the non-fault-tolerant decision fusion algorithm presented in Section 3.2. The probability that i sensors have a value above η_d is $(1-F_X(\eta_d))^i$ and the probability that the remaining $N-i$ sensors have a value below η_d is $F_X(\eta_d)^{N-i}$. Since there are $\binom{N}{i}$ ways of choosing the i sensors among N sensors and i can vary from $\lceil \alpha N \rceil$ to

N for a false alarm to occur, the false alarm probability is given by the following equation:

$$P_{fa} = \sum_{i=\lceil \alpha N \rceil}^{N} \binom{N}{i} F_X(\eta_d)^{N-i}(1 - F_X(\eta_d))^i. \quad (14)$$

Detection probability. Detections occur when $i \geq \lceil \alpha N \rceil$ sensors have a value above the threshold η_d in the presence of a target. For a given set of detecting sensors defined by the permutation h (such that the set $\{h(j), 1 \leq j \leq i\}$ are the indices of detecting sensors), the probability of detection is $\prod_{j=1}^{i}(1 - F_X(\eta_d - E_{h(j)}(u)))\prod_{j=i+1}^{N} F_X(\eta_d - E_{h(j)}(u))$. The detection probability for a given position of target is the sum of these terms for different combinations h and different number of detecting sensors (from $\lceil \alpha N \rceil$ to N). The detection probability is the average of this expression over different position u of the target in the region:

$$P_d = \left\langle \sum_{i=\lceil \alpha N \rceil}^{N} \sum_{h \in C_{i,N}} \left[\prod_{j=1}^{i}(1 - F_X(\eta_d - E_{h(j)}(u))) \prod_{j=i+1}^{N} F_X(\eta_d - E_{h(j)}(u)) \right] \right\rangle. \quad (15)$$

Expressions for the false alarm probability and the detection probability in the presence of faults are now derived.

False alarm probability. In the presence of t faults reporting a detection, only $\lceil \alpha N \rceil - t$ out of $N - t$ sensors need to detect for a false alarm to occur. Therefore, the false alarm probability is given by the following equation:

$$P_{fa} = \sum_{i=\lceil \alpha N \rceil - t}^{N-t} \binom{N-t}{i} F_X(\eta_d)^{N-t-i}(1 - F_X(\eta_d))^i. \quad (16)$$

Detection probability. In the presence of t faulty sensors reporting a nondetection, $\lceil \alpha N \rceil$ out of $N - t$ sensors need to locally detect for a global detection to occur. The detection probability is averaged over the different possible sets of faulty sensors (defined by the combination f) and is given by the following equation:

$$P_d = \left\langle \frac{1}{\binom{N}{t}} \sum_{f \in C_{N-t,N}} \sum_{i=\lceil \alpha N \rceil}^{N-t} \sum_{h \in C_{i,N-t}} \left[\prod_{j=1}^{i}(1 - F_X(\eta_d - E_{f(h(j))}(u))) \prod_{j=i+1}^{N-t} F_X(\eta_d - E_{f(h(j))}(u)) \right] \right\rangle. \quad (17)$$

5 Simulation Results

Although equations to evaluate the performance of value and decision fusion were derived and validated for a small number of nodes, they were found computationally impractical when the number of nodes exceed 20. Simulation was therefore used to compare the performance of the algorithms.

5.1 Design of Experiment

The performance of the algorithms was evaluated using simulations in which sensor nodes were placed randomly in a region of size 20×20 unit length. To measure false alarm, no target is placed in the region. To measure detection probability, the target is placed in a random position. The results presented here are averages of many random target placements. The number of placements simulated is determined so as to obtain a 80 percent confidence that the mean found is within 10 percent of the actual mean, using the central limit theorem [24]. The target energy and noise models are the same as the analytical model.

Simulations were performed for a variable number of sensors, variable target maximum power, and a variable number of faulty sensors. The number of values dropped (i.e., n) is chosen using Table 3 in Section 6 satisfying the bound $N \geq 3n + 1$ so that the number of values dropped does not exceed the number of faults the system can tolerate. The false alarm and detection probability in the presence of faults is evaluated, given that the system does not fail. Algorithms are compared for their detection probability at constant false alarm probability, which depends on the values of the thresholds used in the fusion algorithms.

5.2 Results

5.2.1 Without Faulty Nodes

First, value and decision fusion are compared assuming no sensor nodes are faulty. In such a case, no values/decisions need to be dropped and n is set to 0. The detection probability of both algorithms was measured for different false alarm probability, number of nodes, maximum power, and decay factor as defined in (1). Fig. 4 shows the average detection probability for value and decision fusion as a function of the false alarm probability for $N = 9$ and 100 nodes and decay factors of 2, 3, and 4. Results for different N and different maximum power were found similar to those presented here. The graphs show that, for a decay factor of 2, value fusion is superior to decision fusion for all false alarms and all number of nodes. Also, the superiority of value fusion over decision fusion decreases as the decay factor increases. At a decay factor of 4, decision fusion is better than value fusion for a small number of nodes and low false alarm probability. However, in this case, the detection probability of both approaches is small.

The observations can be explained as follows: Value fusion lets the nodes exchange more information about the target since all the nodes obtain all the energy measurements. Furthermore, for low decay factors, the target is detectable by all or most nodes over a large region and, therefore, most of the nodes collect meaningful information about the presence of the object. Sharing this information benefits the overall performance and, therefore, value fusion is superior to decision fusion. As the energy decay factor increases, only the nodes close to the target collect meaningful information and there is no benefit for other nodes to share their information. Decision fusion becomes superior since it gives more weight to the nodes closest to the target, indeed, for the threshold parameter used, the

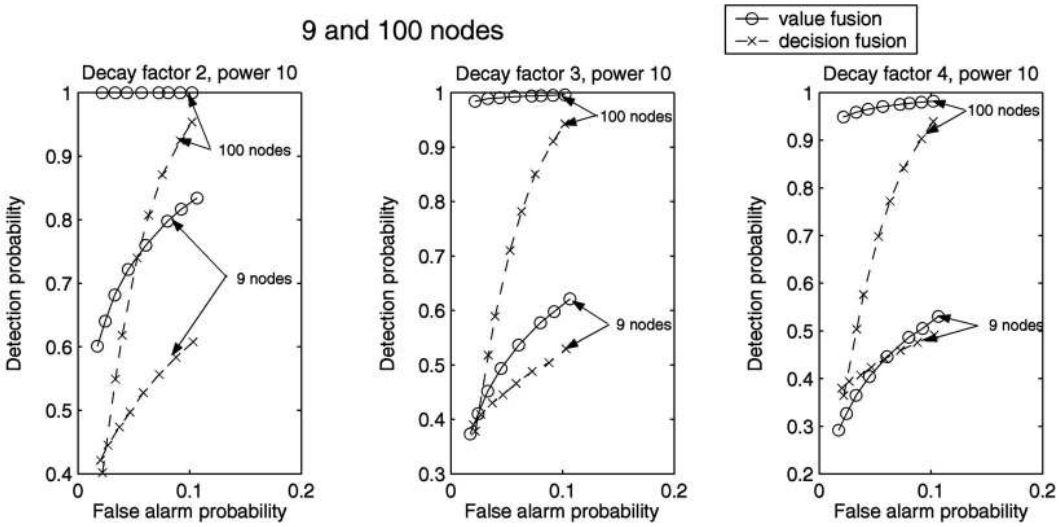

Fig. 4. Performance of value and decision fusion without faulty nodes.

final decision is "target present" as soon as one node locally decides that the target is present.

To complete the performance evaluation, Table 2 presents the standard deviation of the detection probability measured with the simulator. The standard deviation of value and decision fusion are comparable and are functions of the average detection probability. The standard deviation becomes very large when the detection probability mean is below 95 percent and, considering the statistical distribution of the difference between the two approaches, one approach is not found to be consistently better than the other. Therefore, one approach is not superior to the other when the mean detection probability is below 95 percent. Limiting the comparison to the region where the detection probability is above 95 percent, value fusion is superior to decision fusion in the absence of faults.

5.2.2 With Faulty Nodes

Value and decision fusion are now compared in the presence of faulty nodes. As mentioned in Section 5.1, all the nodes are assumed to act consistently and the faulty nodes are consistent outliers defined as follows: In the absence of a target, a faulty node reports the highest permissible value in value fusion and a "local detection" in decision fusion. In the presence of a target, a faulty node reports the lowest permissible value in value fusion and a "local no detection" in decision fusion. Again, the detection probability of both algorithms was measured for varying false alarm probability, number of nodes, maximum power, decay factor, number of values dropped, and number of faults. Only results for constant false alarm probability of 5 percent are presented here, but, throughout the simulations, the conclusions were similar for different false alarm probabilities. The graph of Fig. 5 represents the average detection probability for value and decision fusion as a function of the maximum power for 9, 36, and 99 nodes and decay factors 2, 3, and 4. For these simulations, the number of values dropped is taken from Table 3 and the number of faults injected is $t = n$. Note that the maximum power was increased substantially in the presence of faults to obtain comparable performance than in the absence of faults.

The graphs show that value fusion is superior to decision fusion for a small number of nodes, but decision fusion becomes superior as the number of nodes increases. For increasing decay factor, the superiority of decision fusion occurs for a larger number of nodes. The difference in performance of the two algorithms decreases as the decay factor increases. Overall, faults have more impact on value fusion than on decision fusion. Unlike in the fault-free case, decision fusion performs better than value fusion when the detection probability is 0.8 or above. The reason for this switch is that the value fusion algorithm is often forced to discard meaningful readings from the nonfaulty sensor nodes since it does not know the identity of the faulty nodes. Although decision fusion may also discard decisions by nonfaulty sensor nodes, decisions contain less information than energy readings and, therefore, dropping them does not adversely impact decision fusion as much as value fusion.

Limiting again the comparison to the region where the detection probability is above 95 percent, we conclude that

TABLE 2
Performance Standard Deviation

Mean of detection probability	.25	.40	.50	.60	.75	.85	.95	.98	.995
Standard deviation of detection probability	.44	.49	.50	.49	.43	.35	.22	.15	.07

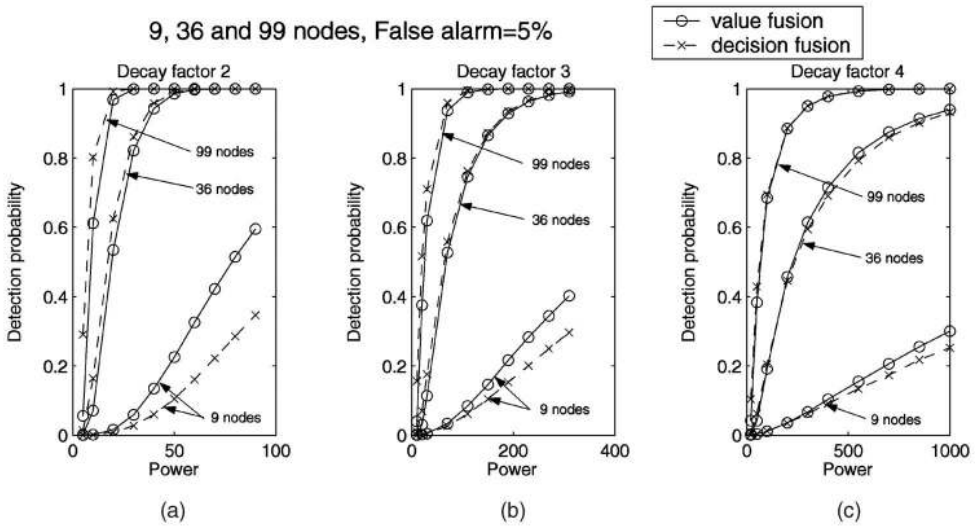

Fig. 5. Performance of value and decision fusion with faulty nodes.

the performance of value fusion and decision fusion are comparable in the presence of faults.

The performance of value and decision fusion is now presented when the number of faulty nodes t varies from 0 to n. Fig. 6 shows the average detection probability of value and decision fusion as a function of t for a fixed maximum power of 50, decay factor of 3, and for 15, 48, and 99 nodes. The number of values dropped is as specified in Table 3. The graph shows that the performance of both algorithms decreases as the number of faulty nodes increases. However, for the region of interest (i.e., when the mean detection probability is high), the mean performance of decision fusion remains about equal to the one of value fusion for any number of faulty nodes t. When $t = n$, the results become identical to the one in the graph of Fig. 5 for a power of 50.

We conclude that, for a given number of values/decisions dropped n, the relative superiority of one approach over the other doesn't vary with the number of faults t present in the system.

Fig. 7 shows the average detection probability as a function of n, the number of values/decisions dropped, when the number of faults present in the system is $t = n$. We observe that, when $n = 0$, value fusion is superior to decision fusion, but, as n increases, decision fusion becomes comparable to value fusion (due to large standard deviation, we cannot conclude that one algorithm is better than the other for large n).

Since the two algorithms have comparable performance, it can be argued that, if one were to take into account the cost of communication, decision fusion would be considered superior to value fusion.

6 Hierarchical Approach for Information Sharing

This section presents and analyzes an approach for reducing the communication cost of the fault-tolerant target detection algorithms.

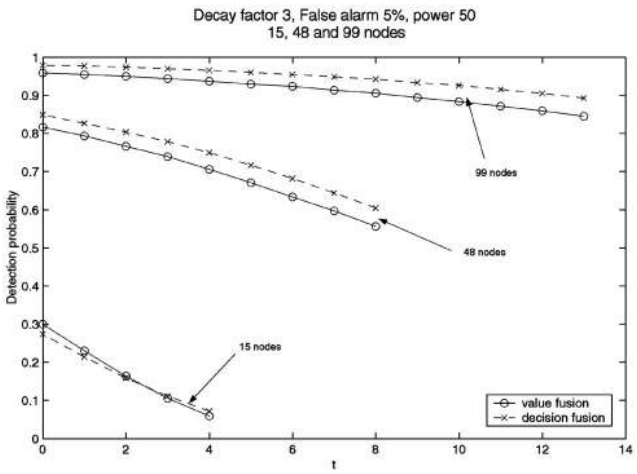

Fig. 6. Performance of value and decision fusion for variable number of values dropped.

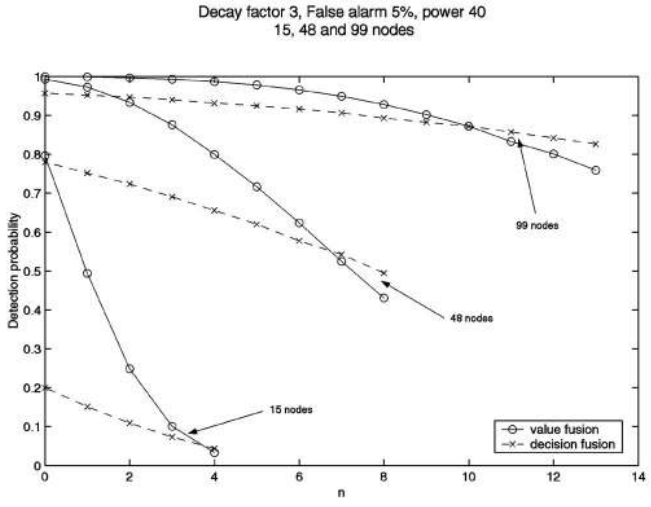

Fig. 7. Performance of value and decision fusion for variable number of values dropped.

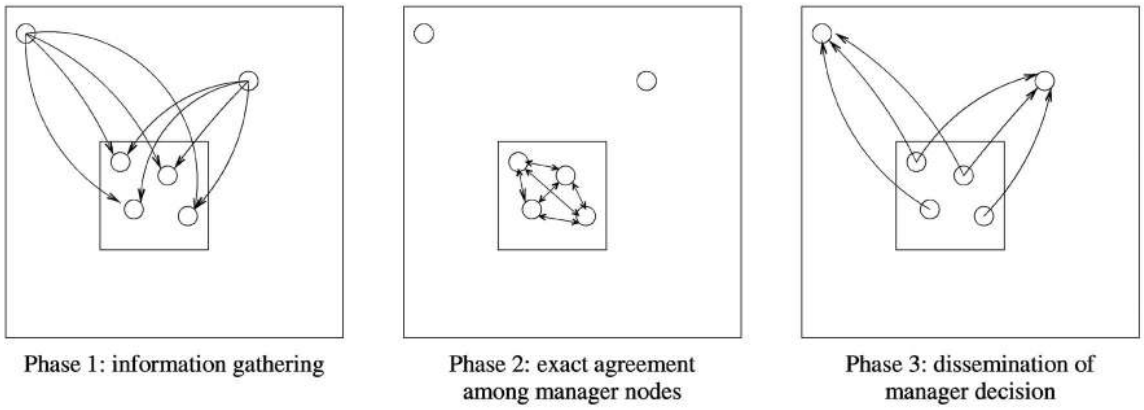

Fig. 8. Hierarchical agreement.

6.1 Approach

Agreement algorithms used in value and decision fusion offer robustness to node failure at the cost of extra communication overhead. For exact agreement, in the presence of t faults for a total of N nodes in the network, $t+1$ rounds of message exchanges and a total of $O(Nt)$ messages of size $O(t^2)$ are necessary to obtain consistency, provided $N \geq 3t+1$ [1], [18]. This communication overhead becomes critical as the number of nodes in the network increases provided that a fixed proportion of the nodes are faulty. Because of power constraints and limited bandwidth, communication efficient agreement needs to be developed. One approach is to perform agreement in a centralized manner in a subset of all the nodes in the region, called manager nodes. Fig. 8 illustrates this hierarchical approach for a set of six nodes, four of them being managers.

In the first phase, every node in the region monitored sends its information to all the M manager nodes. Faulty nodes may send inconsistent information to different manager nodes and, therefore, all manager nodes may not obtain the same global information. In the second phase, manager nodes perform agreement in order to guarantee that nonfaulty manager nodes obtain the same global information. In the third phase, manager nodes disseminate their global information to all the nonmanager nodes in the region. Note that several manager nodes need to send the global information since some manager nodes can be faulty and can disseminate wrong or inconsistent information. Assuming that m or fewer manager nodes are faulty, only $2m+1$ manager nodes need to send their global information to all the nonmanager nodes in the region for them to obtain the global information the manager nodes agreed upon using simple voting.

Overall, the number of message exchanges reduces from $O(Nn)$ in fully distributed agreement to $O(N+Mm)$ in hierarchical agreement. Note that this gain in communication overhead comes along with a loss in system reliability since the hierarchical approach requires that less than one third of the manager nodes be faulty. This is a stronger requirement than in fully distributed agreement, where it is required that less than one third of all the nodes be faulty.

6.2 Algorithms

The basic structure of the hierarchical algorithms is: *gathering*, *manager agreement and fusion*, and *dissemination*, described as phases 1, 2, and 3, respectively, in Fig. 8. In this study, managers use exact agreement to reach consensus, although other agreement types such as inexact agreement might be appropriate. Exact agreement guarantees that all the nonfaulty managers obtain the same set S of data and the data sent by the nonfaulty nodes are part of this set. However, consistent outlying data can remain in the set, as shown in Table 1. To prevent corruption of the decision by these outliers, the largest and smallest data are dropped from the set S and the average data is computed over the remaining data.

Fig. 9 represents different fusion algorithms using the hierarchical approach. For gathering, sensor nodes can either report their raw energy measurement or report a local decision based on their energy measurement. We call the first approach value fusion and the second approach decision fusion, as shown in the first-level branch of Fig. 9. Value fusion lets the sensors exchange more information at the cost of higher communication overhead. Once the managers obtain a vector of values or decisions from every node, they can perform agreement on these vectors or fuse the vectors into a single value to perform agreement on the fused values. We call the first approach raw agreement and the second approach fused agreement, as shown in the second-level branch of Fig. 9. Again, raw agreement lets the sensors exchange more information at the cost of higher communication overhead. In the case of fused agreement, the manager nodes can then perform exact agreement on the fused values or convert these into decisions and perform exact agreement on decisions. This is shown with a third-level branch leading to options 2 and 3 on the value fusion side and options 5 and 6 on the decision fusion side. Note that, for option 1, transforming the raw value vector into a decision vector is equivalent to option 4, but is more expensive in terms of communication. Therefore, option 1 is not subdivided with a third-level branch.

These six options can be reduced to two options in the worst-case scenario identified in Section 4.3, i.e., when the nonfaulty nodes send consistent outliers. Looking at value fusion in Fig. 9, if all the vectors obtained by the manager nodes during gathering are identical, then exact agreement

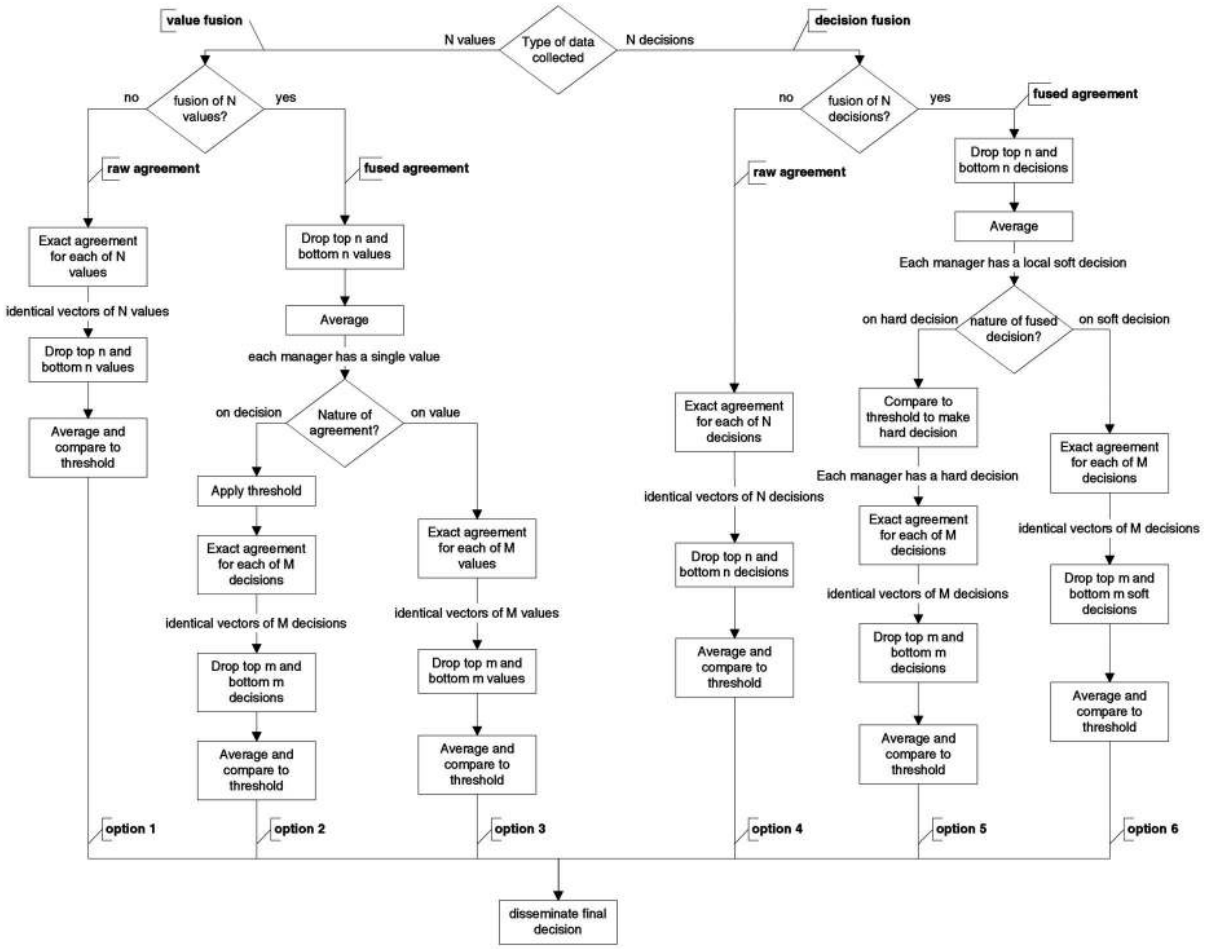

Fig. 9. Algorithms for hierarchical fusion.

doesn't modify these vectors and option 1 functionally reduces to: Drop maximum n and minimum n values, average and compare to threshold. For options 2 and 3, all the manager nodes obtain an identical fused value when performing fused agreement; therefore, the fused agreement doesn't modify this value and options 2 and 3 functionally reduce to: Drop top n and bottom n values, average, and compare to threshold. This shows that all the value fusion options are equivalent in the worst-case scenario when all the nodes act consistently. Similarly, all the decision fusion options are equivalent in the worst-case scenario and they functionally reduce to: Drop top n and bottom n decisions, average, and compare to threshold.

The performance evaluation was restricted to the case where all the nodes act consistently, which applies for the worst-case scenario and the nonfaulty case. Under such an assumption, the six options identified above reduce to value and decision fusion and their precision and accuracy performance have been studied. As we observed that the two algorithms have comparable performance, the best option from Fig. 9 is chosen by comparing communication overhead. We conclude that decision fusion algorithms using fused agreement on hard decision (option 5) is superior.

6.3 Impact on Performance

6.3.1 System Failure Probability

The system failure probability is the probability that the system cannot make a precise and accurate decision. It is a function of the number of values/decisions dropped, n and m, defined in Fig. 9, as well as the number of manager nodes M. Here, the system failure probability is studied and a method of determining the parameters n, m and M is developed. Note that the six options in Fig. 9 have identical system failure probability as long as they use the same parameters n, m and M.

Reducing the number of nodes performing agreement affects the probability of system failure. The system is considered to have failed if the nodes cannot make a precise and accurate decision. For the system to be precise, consistency needs to be achieved by exact agreement among M manager nodes. This requires $M \geq 3s + 1$, where s is the number of faulty manager nodes [18]. For the system to be accurate, outlying values/decisions need to be dropped. Assuming that as many as t nodes can be faulty, the t highest and lowest values/decisions (i.e., a total of $2t$) need to be dropped when collecting values/decisions to remove outliers. Therefore, it is required that $n \geq t$. Also, outliers introduced by the s faulty manager nodes during exact agreement need to be removed and, therefore, $m \geq s$.

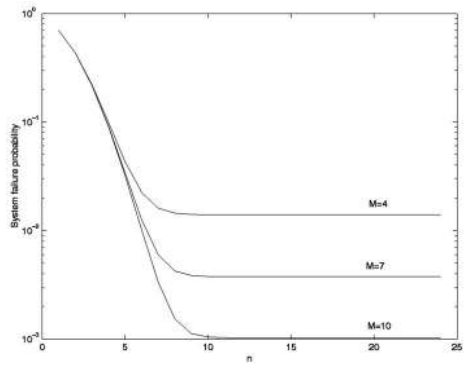

Fig. 10. System failure probability for $p = 0.05$ and $N = 48$.

In this study, $m = \lfloor \frac{M-1}{3} \rfloor$ so that $M \geq 3s + 1$ guarantees $m \geq s$ ($\lfloor x \rfloor$ denotes the floor of x).

To evaluate the probability of system failure, every node is assumed to have the same probability of failure p and the node failures are assumed independent. For a system with a total of N nodes, M manager nodes, and algorithms dropping highest n and lowest n values/decisions, the probability of system failure is given by the following equation:

$$P_{sys\ failure} = P\left(s \geq \left\lceil \frac{M}{3} \right\rceil \text{ and } s \leq t \leq n\right) + P(t > n) \quad (18)$$

$$P_{sys\ failure} = P\left(s \geq \left\lceil \frac{M}{3} \right\rceil \text{ and } 0 \leq r \leq n - s\right) + P(t > n), \quad (19)$$

where $\lceil x \rceil$ denotes the ceiling of x and $r = t - s$ is the number of faulty nonmanagers.

$$P_{sys\ failure} = \sum_{s=\lceil \frac{M}{3} \rceil}^{M} \binom{M}{s} p^s (1-p)^{M-s} \left(1 - \sum_{r=max(n-s,0)}^{N-M} \binom{N-M}{r} p^r (1-p)^{N-M-r}\right) + \sum_{t=n+1}^{N} \binom{N}{t} p^t (1-p)^{N-t}. \quad (20)$$

The graph of Fig. 10 shows the probability of system failure as a function of n for different values of M. Values considered are $M = 4, 7, 10$ and larger values of M are not considered, primarily to limit the communication overhead.

The graph shows that the system failure probability decreases as M and n increase. It saturates as n becomes large. This is because, for small n, the probability is that $t > n$ is predominant in the system failure probability, but, as n increases, the probability of failure among managers becomes predominant in the system failure probability. Since the system accuracy decreases as more values are dropped (i.e., when n increases), the best accuracy is obtained for small n provided the system doesn't fail. The graph can be used to first determine a value for M and then a value of n such that n is as small as possible and a given system failure is met. Setting $M = 7$, optimum values of n for various total number of nodes N are shown in Table 3.

TABLE 3
Optimum n for Various N, Given $M = 7$, $p = 0.05$, and $P_{sys\ failure} \leq 0.005$

N	9	15	24	36	48	63	81	99
n	3	4	6	7	8	9	11	13

6.3.2 Communication Overhead

Among the six algorithms options identified, it was argued that value fusion or fused agreement-based approaches are more costly than decision fusion or raw agreement-based approaches. We conclude that option five is the less costly algorithm and option one the more costly. The communication overhead is also determined by the number of manager nodes and it was argued that centralizing the fusion process reduces the communication cost. There is a clear trade off between the system failure probability and the communication overhead when choosing the number of manager nodes for a given total number of nodes.

7 Conclusion

This paper studied the problem of target detection by a sensor network deployed in a region to be monitored. A model of sensor network for target detection was developed, specifying the signal energy measured by the sensors function of the target characteristics and the faulty sensor behavior. Previous work in the field of signal detection and distributed systems was presented and their relevance to this study was highlighted. Two algorithms using exact agreement were identified to solve the problem: value fusion and decision fusion. They were analyzed for their robustness to sensor nodes failure and compared for their performance and communication overhead. Also, a hierarchical approach for reducing the communication overhead induced by exact agreement was proposed and analyzed. Several hierarchical algorithms were identified and they were shown to reduced to value fusion and decision fusion in the worse case scenario.

The performance comparison was performed both in the presence and in the absence of faulty nodes. The scope of the comparison was limited to the worst-case scenario, where all faulty nodes act consistently. The performances of value and decision fusion were first analyzed analytically and then evaluated through simulation. Value fusion-based algorithms were found to perform better than decision fusion-based algorithms in the absence of faults. However, value fusion-based and decision fusion-based algorithms performance become comparable as faults are introduced in the system and decision fusion-based algorithms are then preferred for their lower communication overhead.

The performance of localization and tracking algorithms, as opposed to detection algorithms, and the replacement of exact agreement by other agreement algorithms such that inexact or approximate agreement need to be investigated. Also, methods for determining how to deploy the sensors in

the region of interest need to be developed to optimize the detection performance.

ACKNOWLEDGMENTS

This work was supported in part by the US Defense Advanced Research Projects Agency (DARPA) and the US Air Force research Laboratory, Air Force Material Command, USAF, under agreement number F30602-00-2-055. The US Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright annotation thereon.

REFERENCES

[1] M. Barborak, M. Malek, and A. Dahbura, "The Consensus Problem in Fault-Tolerant Computing," *ACM Computing Surveys,* vol. 25, no. 2, pp. 171-220, June 1993.
[2] R. Blum, S. Kassam, and H.V. Poor, "Distributed Detection with Multiple Sensors: Part II-Advanced Topics," *Proc. IEEE,* pp. 64-79, Jan. 1997.
[3] J. Bray and C.F. Sturman, *Bluetooth: Connect without Cables.* Prentice Hall PTR, 2000.
[4] R. Brooks and S. Iyengar, "Robust Distributed Computing and Sensing Algorithms," *Computer,* vol. 29, no. 6, pp. 53-60, June 1996.
[5] R.R. Brooks and S.S. Iyengar, *Multi-Sensor Fusion: Fundamentals and Applications with Software.* Prentice Hall, 1998.
[6] T. Clouqueur, P. Ramanathan, K.K. Saluja, and K.-C. Wang, "Value-Fusion versus Decision-Fusion for Fault-Tolerance in Collaborative Target Detection in Sensor Networks," *Proc. Fourth Ann. Conf. Information Fusion,* pp. TuC2/25-TuC2/30, Aug. 2001.
[7] D. Dolev et al., "Reaching Approximate Agreement in the Presence of Faults," *J. ACM,* pp. 499-516, July 1986.
[8] D. Dolev, M.J. Fischer, R. Fower, N.A. Lynch, and H.R. Strong, "An Efficient Algorithm for Byzantine Agreement without Authentication," *Information and Control,* vol. 52, pp. 257-274, 1982.
[9] D. Dolev and H. Strong, "Polynomial Algorithms for Multiple Processor Agreement," *Proc. 14th ACM Symp. Theory of Computing,* pp. 401-407, 1982.
[10] D. Dolev, R. Reischuk, and H.R. Strong, "'Eventual' Is Earlier than 'Immediate'," *Proc. 23rd Ann. Symp. Foundations of Computer Science,* pp. 196-203, 1982.
[11] M.J. Fischer, "The Consensus Problem in Unreliable Distributed Systems (a Brief Survey)," *Fundamentals of Computation Theory,* pp. 127-140, 1983.
[12] J.D. Gibsin and J.L. Melsa, *Introduction to Nonparametric Detection with Application.* Academic Press, 1975.
[13] G. Grimmett and D. Stirzaker, *Probability and Random Processes.* Oxford Science Publications, 1992.
[14] D.L. Hall, *Mathematical Techniques in Multisensor Data Fusion.* Artech house, Inc., 1992.
[15] M. Hata, "Empirical Formula for Propagation Loss in Land Mobile Radio Services," *IEEE Trans. Vehicular Technology,* vol. 29, pp. 317-325, Aug. 198.0
[16] IEEE Std 802.11-1997, Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications, 1997.
[17] I.F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "A Survey on Sensor Networks," *IEEE Comm. Magazine,* vol. 40, no. 8, pp. 102-114, Aug. 2002.
[18] L. Lamport, R. Shostak, and M. Pease, "The Byzantine Generals Problem," *ACM Trans. Programming Languages and Systems,* vol. 4, no. 3, pp. 382-401, July 1982.
[19] C. Lee and J. Chao, "Optimum Local Decision Space Partitioning for Distributed Detection," *IEEE Trans. Aerospace and Electronic Systems,* vol. 25, no. 4, pp. 536-544, July 1989.
[20] S. Mahaney and F. Schneider, "Inexact Agreement: Accuracy, Precision and Graceful Degradation," *Proc. Fourth ACM Symp. Principles of Distributed Computing,* pp. 237-249, 1985.
[21] M. Pease, R. Shostak, and L. Lamport, "Reaching Agreement in the Presence of Faults," *J. ACM,* vol. 27, no. 2, pp. 228-234, Apr. 1980.
[22] L.L. Scharf, *Statistical Signal Processing: Detection, Estimation, and Time Series Analysis.* Addison-Wesley, 1991.
[23] Sensor Information Technology Website, http://www.darpa.mil/ito/research/sensit/index.html, year?
[24] J. Shao, *Mathematical Statistics.* Springer, 1999.
[25] P.R.U. Shashi, "Survey on Sensor Networks," year?
[26] K.G. Shin and P. Ramanathan, "Diagnosis of Processors with Byzantine Faults in a Distributed Computing System," *Proc. Fault-Tolerant Computing Symp.,* pp. 55-60, July 1987.
[27] P. Varshney, *Distributed Detection and Data Fusion.* New York: Springer-Verlag, 1996.
[28] R. Viswanathan and P. Varshney, "Distributed Detection with Multiple Sensors: Part I-Fundamentals," *Proc. IEEE,* pp. 54-63, Jan. 1997.

Thomas Clouqueur (S'01) obtained the engineering degree in electrical engineering from the Ecole Superieure d'Electricite, France, in 1999, and the MS degree in electrical and computer engineering from the University of Wisconsin-Madison in 1999. He is currently a PhD candidate and research assistant in the Department of Electrical and Computer Engineering at the University of Wisconsin-Madison. His research activity focuses on problems related to the deployment of sensors and fault-tolerant collaborative signal processing in sensor networks. His research interests include VLSI design and testing and fault-tolerant computing. He is a student member of the IEEE.

Kewal K. Saluja (S'70-M'73-SM'89-F'95) obtained the Bachelor of Engineering (BE) degree in electrical engineering from the University of Roorkee, India, in 1967, and the MS and PhD degrees in electrical and computer engineering from the University of Iowa, Iowa City, in 1972 and 1973, respectively. He is currently with the Department of Electrical and Computer Engineering at the University of Wisconsin-Madison as a professor, where he teaches courses in logic design, computer architecture, microprocessor-based systems, VLSI design and testing, and fault-tolerant computing. Prior to this, he was at the University of Newcastle, Australia. His current research interests include testing, sequential circuit test generation, built-in self-test, Iddq testing, test and diagnosis of crosstalk faults, testing RAMs and flash memories, fault-tolerant computing, and sensor networks. He has published nearly 200 refereed journal and conference papers in these areas. He served as an editor of the *IEEE Transactions on Computers* (1997-2001) and he is an associate editor for the letters section of the *Journal of Electronic Testing: Theory and Applications* (JETTA) published by Kluwer. He has served on the program committees of numerous national and international conferences. He is a member of Eta Kappa Nu, Tau Beta Pi, a member of the IEEE Computer Society, a fellow of the JSPS, and a fellow of the IEEE.

Parameswaran Ramanathan (S'84-M'89-SM'02) received the BTech degree from the Indian Institute of Technology, Bombay, India, in 1984, and the MSE and PhD degrees from the University of Michigan, Ann Arbor, in 1986 and 1989, respectively. Since 1989, he has been a faculty member in the Department of Electrical and Computer Engineering, University of Wisconsin-Madison, where he is presently a full professor. He leads research projects in the areas of sensor networks and next generation cellular technology. His research interests include wireless and wireline networking, real-time systems, fault-tolerant computing, and distributed systems. He is presently an associate editor for the *IEEE Transactions on Mobile Computing* and *Elsevier AdHoc Networks Journal.* He served as an associate editor for the *IEEE Transactions on Parallel and Distributed Systems* from 1996-1999. He has also served on program committees of conferences. He is a member of the ACM and the IEEE Computer Society and a senior member of the IEEE.