Adaptive Distributed Outlier Detection for WSNs

Alessandra De Paola, Salvatore Gaglio, *Member, IEEE*, Giuseppe Lo Re, *Senior Member, IEEE*, Fabrizio Milazzo, and Marco Ortolani, *Member, IEEE*

***Abstract*—The paradigm of pervasive computing is gaining more and more attention nowadays, thanks to the possibility of obtaining precise and continuous monitoring. Ease of deployment and adaptivity are typically implemented by adopting autonomous and cooperative sensory devices; however, for such systems to be of any practical use, reliability and fault tolerance must be guaranteed, for instance by detecting corrupted readings amidst the huge amount of gathered sensory data. This paper proposes an adaptive distributed Bayesian approach for detecting outliers in data collected by a wireless sensor network; our algorithm aims at optimizing classification accuracy, time complexity and communication complexity, and also considering externally imposed constraints on such conflicting goals. The performed experimental evaluation showed that our approach is able to improve the considered metrics for latency and energy consumption, with limited impact on classification accuracy.**

***Index Terms*—Bayesian networks (BNs), outlier detection, WSN.**

I. Introduction

GIVEN their pervasiveness, ease of use, and reduced costs, wireless sensor networks (WSNs) are nowadays increasingly used in many industrial and research applications; they are composed of small devices with limited power supply, named sensor nodes, able to sense the environment, perform small on-board computations and communicate with each other in order to collaboratively detect possible events of interest.

Despite their many advantages, such as flexibility, one of the main drawbacks of WSNs is the possibility of faults occurring during sensing [1], which could limit their effectiveness as pervasive monitoring tool. Such faults may be due to several causes, such as harsh environmental conditions that can damage hardware, lack of energy that can affect the values of the sensory readings and the quality of communications, or sensor miscalibration that may affect the ADC transducer.

In our work, we focus on the task of detecting outliers in the flow of sensory data; they negatively affect the performance of the WSN since transmission and processing of corrupted data inevitably result into a waste of energy and time. Early detection of outliers might constitute a prefiltering phase, necessary for reducing the amount of data to be processed by high-level systems; for such reason, in-network outlier detection becomes a crucial functionality for many WSN-based applications [2], [3].

Manuscript received May 15, 2013; revised April 28, 2014; accepted June 29, 2014. This work was supported by the PO FESR Sicilia 2007/2013 through the SmartBuildings Project under Grant G73F11000130004. This paper was recommended by Associate Editor R. Selmic.

The authors are with the Department of DICGIM, University of Palermo, Palermo 90128, Italy (e-mail: alessandra.depaola@unipa.it).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TCYB.2014.2338611

The main contribution of this paper is the proposal of the adaptive distributed outlier detection (ADOD) algorithm to detect data faults in a WSN. Unlike other approaches presented in literature, ADOD allows designers to fine tune classification accuracy, time complexity and communication complexity, according to application-specific requirements. Each sensor node might choose to cooperate with other nodes to perform outlier detection; higher classification accuracy is achieved as the set of cooperating nodes grows larger, whereas limited cooperation leads to lower time and communication complexity. We address the issue of combining several conflicting goals, namely maximizing classification accuracy and minimizing both time complexity and communication complexity by exploiting constrained Pareto optimization.

Outlier detection is carried out by a set of Bayesian networks (BNs) scattered over the WSN. The structure of each BN is distributed over a set of cooperating sensor nodes, and the novelty of our approach consists in the dynamic construction of the cooperating set according to superimposed constraints and to variations in the observed physical phenomenon. As a consequence, a single WSN may contain areas where outlier detection is performed by simpler BNs, due to the presence of few outliers, and other ones where the BNs present a more complex structure.

The remainder of the paper is organized as follows. Section II discusses approaches in literature for outlier detection in WSNs. Section III states some assumptions that guarantee the correct behavior of our algorithm. Section IV describes the proposed approach while Section V shows the results of performed experimental evaluation. Finally, Section VI reports the conclusions of our work.

II. Related work

According to the definition proposed in [4], an outlier is a pattern that does not match the expected trend in analyzed data. The correct detection of outliers in data acquired by a WSN may provide useful information about the state of the network and about the surrounding environment, e.g., residual WSN lifetime, malfunctioning nodes and unexpected environmental events.

It is possible to identify different types of data outliers in WSNs [5]–[7], such as follows.

1) *Spike:* Characterized by one or more out-of-bound readings in a very limited amount of time.
2) *Noise:* A sequence of data characterized by a greater variance as compared to the environmental one.
3) *Stuck-at:* A pattern characterized by quasi zero variance.

Most outlier detection algorithms for WSNs exploit correlation among sensed data. The temporal correlation within

readings from a single node allows to compare the current behavior of the device with the past one. The spatial correlation allows to analyze the trend of the physical phenomenon monitored by the WSN in a wider area, and requires the comparison of sensory readings gathered by different nodes.

According to Zhang *et al.* [8], outlier detection in WSNs can be categorized into five main classes, namely: statistical, nearest neighbor, clustering, classification, and spectral decomposition.

Statistical approaches [9], [10] build mathematical models for the data and compute the probability that a sensory reading is generated by that model; if such probability falls below a given threshold, the reading is classified as an outlier. Statistical methods are characterized by low computational complexity, but generally require supervised learning and an *ad-hoc* fixed threshold.

Nearest-neighbor algorithms [11], [12] use a distance metric to express similarity in data. An outlier is a reading which appears dissimilar from the rest of the data. Such approaches exhibit low computational complexity and do not require supervised learning, but they are characterized by variable and unpredictable detection accuracy.

Clustering approaches [13] rely on similarity metrics, but are additionally able to provide identification for the outlier class. Inter/intracluster distance thresholds are, however, not easy to determine, and their values can significantly affect performance.

All the mentioned approaches are heavily dependent on the choice of specific thresholds, so classification methods [14], [15] have been proposed to overcome such drawback. They learn mathematical models both for regular data and for outliers during a training phase, and classify previously unseen data according to those models. This class includes, for instance, Bayesian networks (BN) [16]–[18] and neural networks [19]–[21]. The main advantage of such approaches is that classification accuracy is predictable during the learning phase; however, in general, they require high computational effort.

Methods based on spectral decomposition [22] exploit principal component analysis (PCA) to reduce data dimensionality and to build a structure that represents normal data trend. Each reading which does not respect the structure expressed by the significant components is classified as outlier. Such methods are characterized by a very high computational effort for performing the PCA reduction.

The outlier detection algorithm proposed here belongs to the category of classification approaches and, in particular is based on BNs. We chose to adopt a distributed implementation where the nodes of a WSN cooperate with each other in order to classify their readings and discard outliers before they are transmitted toward the base station. This choice allows to reduce the energy waste resulting from the unnecessary transmission of outliers.

The collaborative distributed approach adopted for the development of several WSN applications [23]–[25] has proven effective to deal with the intrinsic limitations of such networks, and has been successfully used in many applications such as target detection and tracking, node localization and outlier detection. Cooperating sensor nodes are able to exploit the correlation among their respective sensory readings, and become aware of the conditions of the surrounding environment.

To the best of our knowledge, other works in the literature adopting BNs for classification [16]–[18], [26], [27] consider a static structure of the BN; as a consequence, they are not able to tune classification performance based on the scenario conditions or to application constraints. On the contrary, our algorithm periodically adapts the structure of its BNs in order to find the best trade-off with respect to classification accuracy, time complexity and communication complexity.

Other works in the literature propose dynamically tunable algorithms for WSNs based on the evaluation of some quality metrics. For instance, Hoes *et al.* [28] propose a clustering algorithm which evaluates the reliability, lifetime and coverage of the WSN, in order to select the optimal set of nodes which should be active for their tracking application. In [29], the adoption of such quality metrics as reliability, cost, scalability, communication load is proposed to compute the optimal placement of WSN nodes. However, part of the novelty of our proposal consists in the adoption of a dynamic structure for the cooperating sets of sensor nodes, over which dynamic BNs are scattered in order to perform outlier detection.

III. Basic Assumptions

In this section, we discuss the necessary assumptions to ensure the correctness of our algorithm.

We assume that the WSN is constituted by N nodes capable of sensing the same physical phenomenon, which is characterized by a spatio-temporal dependency among readings. This is not a strong assumption, as plenty of evidence shows that it holds for quantities commonly monitored by WSNs (e.g., temperature, air humidity, atmospheric pressure) [30], [31].

A more constraining assumption concerning the adopted routing protocol is made to guarantee the computational feasibility of our approach. ADOD is based on a set of distributed BNs built on the WSN communication graph. Inference on BNs generally requires solving a NP-hard problem; however, if the BN is structured as a polytree then the computational complexity is reduced and the problem becomes tractable, as shown in [32] and [33]. In order to guarantee this acyclic structure, we assume that the underlying routing algorithm produces a connected loop-free network. To this end, it is sufficient to adopt a hierarchical routing algorithm [34]. Hierarchical routing is a common solution for WSNs, with a view to minimizing the communication overhead and, consequently the overall energy consumption; moreover, it is often exploited in order to support data aggregation and to increase the resilience to faults [35].

Finally, for the sake of simplicity, we will assume that all sensor nodes adopt the same sampling rate, Δt, that is used as a basic time unit in the rest of the paper. It is worth noting that such assumption does not imply a strong synchronization among sensor nodes. As each node performs outlier detection on its latest sensory reading, in the worst case the timestamps of the readings from two sensor nodes performing the same round of the algorithm could differ by at most Δt.

IV. Adaptive Distributed Outlier Detection

The main goal of the ADOD algorithm is in-network detection of data outliers in a WSN, in order to avoid wasting

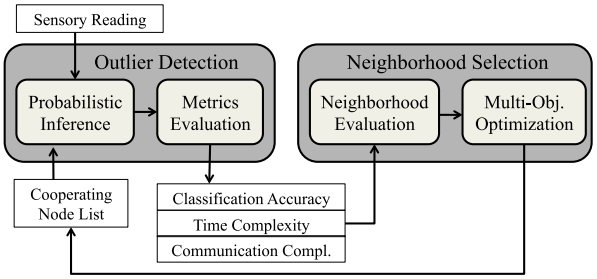

Fig. 1. Node-centric view of the ADOD algorithm.

energy for useless transmissions toward the base station. Outlier detection is performed by means of a set of BNs distributed over the WSN. Each BN relies on a group of collaborative sensor nodes to perform distributed probabilistic inference; moreover, the subset of collaborative nodes is chosen dynamically and in a fully decentralized way, since each node makes its decision autonomously.

From the point of view of an individual node, ADOD consists of two main phases, namely outlier detection and neighborhood selection, as shown in Fig. 1. The first phase, which occurs after each sensing event, detects a possible outlier by collaborating with neighboring nodes and results into the evaluation of three metrics: classification accuracy, time complexity and communication complexity. The second phase, which is performed periodically, aims at identifying the best set of neighbors to cooperate with, and thus corresponds to a reconfiguration of the BN structure.

The higher the number of cooperating nodes, the higher the classification accuracy; such higher accuracy comes at the cost of an increase in time and communication complexity and, consequently, in detection delay and energy consumption. The dynamic reconfiguration of the BNs allows to adapt the system performance to the features of the specific application and of the current scenario.

If a node chooses not to cooperate, its outlier detection phase only needs to exploit time dependency among local measurements; otherwise, spatial dependency among measurements gathered by different nodes is used by the BN for classifying sensory readings.

The effect of the BN structure on time complexity is due to the delay for sharing both the set of measurements and the belief about the correctness of sensory readings within a cooperating set of nodes. Its influence on communication complexity arises from the higher number of messages to be sent to as many nodes during the outlier detection phase.

Fig. 2 shows the evolution of the cooperation network used by ADOD. From the viewpoint of node 1, the example shows how decisions taken by a single node affect the composition of the cooperating set. Initially ($t = 0$), the cooperation network consists of a single group including all sensor nodes. Such configuration corresponds to the best case for classification accuracy, and to the worst case for time and communication complexity. When optimization is performed ($t = T$), node 1 computes the values of the quality metrics for all of its possible decisions about cooperation. Multiobjective optimization results into disconnection from node 3, as this decision allows to reduce both time and communication complexity while leaving classification accuracy unmodified, as shown by the table on top of the figure. The notation adopted in the following for describing ADOD is summarized in Table II at the end of the paper.

Node 1 Quality Indices			
Decision	**Accuracy**	**Comm. Compl.**	**Time Compl.**
Disconnect node 2	0.97	4	2
Disconnect node 3	0.97	4	1
Disconnect node 4	0.97	4	2
No changes	0.97	6	2

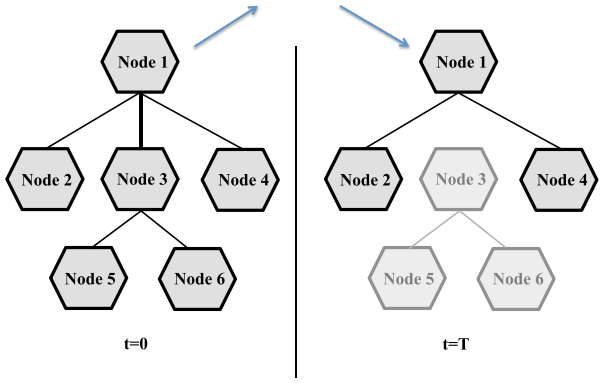

Fig. 2. Dynamic evolution of the cooperation network at two different time instants, for a network composed of six sensor nodes. Hexagons represent sensor nodes and links represent cooperation relationships. The upper part of the figure highlights which considerations drive node 1 not to cooperate with node 3.

A. Outlier Detection

Using Bayesian networks for outlier detection allows to take into account the probabilistic dependency between random variables. Formally, BNs are represented as a direct acyclic graph whose nodes correspond to such variables, and are connected by directed links representing causal relations among them.

In ADOD, each node of the WSN implements only a portion of a BN, and whenever a sensor node cooperates with other nodes, its portion of the BN is connected to those residing elsewhere. A single BN portion consists in a hidden variable and a group of local evidence variables. The hidden variable is associated with the class of the current sensory reading, while the local evidence variables depend on temporal correlation within readings gathered by a single node. The connection between two BN portions occurs via the insertion of shared observable variables, expressing spatial correlation within readings gathered by different nodes.

This approach can be instantiated in different ways by varying the set of classes to be detected, as well as local and shared variables. In our proposal, the hidden variable c in each node can take upon one of the following values: {`spike`, `noise`, `stuck-at`, `correct`}. The set of local observed variables is $L = \{l_1, l_2, l_3\} = \{$`inner-gradient`, `repetitions`, `variance`$\}$, where l_1 is computed as the difference between the last two readings; assuming that r_t indicates the last reading, l_2 is computed as the number of consecutive repetitions of r_t; l_3 is computed as the variance of the last K readings. Finally, the set of shared observed variables contains just one component which depends only on the last readings of nodes i and j, defined as $S_{i,j} = \{s_{i,j,1}\} = r_t^i - r_t^j$.

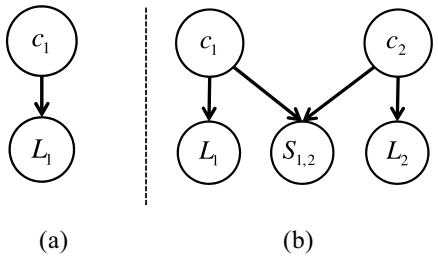

Fig. 3. Bayesian network structure for (a) single node, where the evidence consists only of local features and (b) two cooperating nodes, where the evidence consists of two sets of local features, one for each node, and a set of shared features.

It is worth pointing out that the structure of BNs in ADOD strictly relies on the active cooperations between sensor nodes. Let us represent the communication network as a graph $G = (V, E)$, where V indicates the set of sensor nodes, and E is the set of communication links; the cooperation network is a graph $G' = (V, E')$, where $E' \subseteq E$, and the presence of a link $e_{i,j}$ in E' indicates the active cooperation between nodes i and j. In order to build the BNs, as a first step, each sensor node i is mapped onto a BN composed by a hidden variable c_i, a set of local observed variables L_i, and a set of causal links from the hidden variable to the observed ones, as shown in Fig. 3(a), where the set of local variables is represented as a single node. As a second step, each link $e_{i,j}$ in the cooperation network is mapped onto a new node $S_{i,j}$ representing a set of shared variables, and the corresponding causal links from c_i and c_j toward variables in $S_{i,j}$. Fig. 3(b) shows the Bayesian network resulting from the cooperation of two sensor nodes, composed by two naive Bayes classifiers connected through a set of shared variables.

More formally, if the BN is represented as $B = (X, R)$, where X is the set of random variables and R is the set of links representing causal relations, the mapping from the cooperation network to the Bayesian Network is defined according to the following rules.

1) $i \in V \rightarrow c_i \in X$.
2) $i \in V \rightarrow L_i \in X$.
3) $i \in V \rightarrow \langle c_i, L_i \rangle \in R$.
4) $e_{i,j} \in E' \rightarrow S_{i,j} \in X$.
5) $e_{i,j} \in E' \rightarrow \langle c_i, S_{i,j} \rangle \in R$.
6) $e_{i,j} \in E' \rightarrow \langle c_j, S_{i,j} \rangle \in R$.

Fig. 4 shows how the decisions taken by a single node affect the whole set of BNs. When optimization occurs ($t = T$), the shared variables $S_{1,3}$ are deleted to reflect the choice by node 1 about disconnecting from node 3. As a result, two independent BNs are created.

Given the BNs built as described, outlier detection amounts to finding the most probable classes for sensory readings of participating nodes. With regards to a single BN, it means determining the set of optimal classes $c^* = (c_1^*, \ldots, c_N^*)$ characterized by the maximum value of the *a posteriori* probability conditioned by the considered evidence. This may be defined as a maximum *a posteriori* (MAP) problem as follows:

$$p(c_1, \ldots, c_N | L_1, \ldots, L_N, S_{1,2}, \ldots, S_{N-1,N}) = \prod_{\substack{i \\ j \in CN(i)}} p(L_i|c_i) p(S_{i,j}|c_i, c_j) p(c_i) \quad (1)$$

where $CN(i)$ represents the set of neighbors of node i in the cooperation network.

The conditional probabilities $p(L_i|c_i)$, $p(S_{i,j}|c_i, c_j)$, and $p(c_i)$ are computed for each node through off-line supervised learning, via a frequentist approach based on a set of previously collected and classified readings.

It is worth noting that in practice no sensor node is explicitly aware of the structure of the whole BN, which is rather distributed over the sensor network. Its definition as a whole has the sole purpose of defining the distributed inference process and the rules that each sensor node applies in order to compute its own belief. Whenever a node gathers a reading, it initially computes local observed variables, and then it exchanges its reading with cooperating nodes to compute shared variables; finally they all run the distributed inference procedure to compute the solution of the MAP problem.

In our implementation, we chose to use the well-known "max-product" inference algorithm [32], [36], that is a convergecast-broadcast message-passing procedure based on a flow of belief within the cooperating cluster. We chose to adopt the "max-product" algorithm, which has been defined for probabilistic graphical models, such as Bayesian networks, Markov random fields, chain graphs, because it allows to exactly compute the joint probability over a BN, and because of its formulation as a message passing procedure, which naturally prompts for an implementation as as a distributed algorithm. Moreover, if the BN is structured as a poly-tree, the algorithm is characterized by a polynomial computational complexity, which is particularly suitable to the strict requirements of a WSN.

In the max-product algorithm, each sensor node plays one of the following three roles, according to the information provided by the hierarchical routing algorithm, and to the structure of the cooperation network.

1) *Leaf:* A node that has no child.
2) *Intermediate:* A node that has a parent and at least one child.
3) *Root:* A node that has no parent.

Each leaf starts the inference algorithm by setting its initial belief about the class of its last reading to $p(c_i|L_i)$, thus exploiting only its local evidence; then it starts the convergecast message-passing procedure. At each step of the convergecast procedure, each node i, after receiving all convergecast messages from its children, sends the following convergecast message $\mu_{i \rightarrow j}(c_j)$ to its parent j:

$$\mu_{i \rightarrow j}(c_j) = \max_{c_i} \phi(c_i, c_j) \quad (2)$$

where $\mu_{i \rightarrow j}(c_j)$ is a vector dimensioned as the set of possible values of c_j, and the matrix $\phi(c_i, c_j)$ represents the joint belief about the class of the last reading sensed by i and j, given the local evidence of i, the shared evidence of i and j, and all the local and shared evidences of the subtree rooted at i.

The $\phi(c_i, c_j)$ matrix is computed as follows:

$$\phi(c_i, c_j) = p(c_i) p(L_i|c_i) p(S_{i,j}|c_i, c_j) \prod_{z \in CN(i)/j} \mu_{z \rightarrow i}(c_i) \quad (3)$$

where the product is replaced by 1 if node i is a leaf.

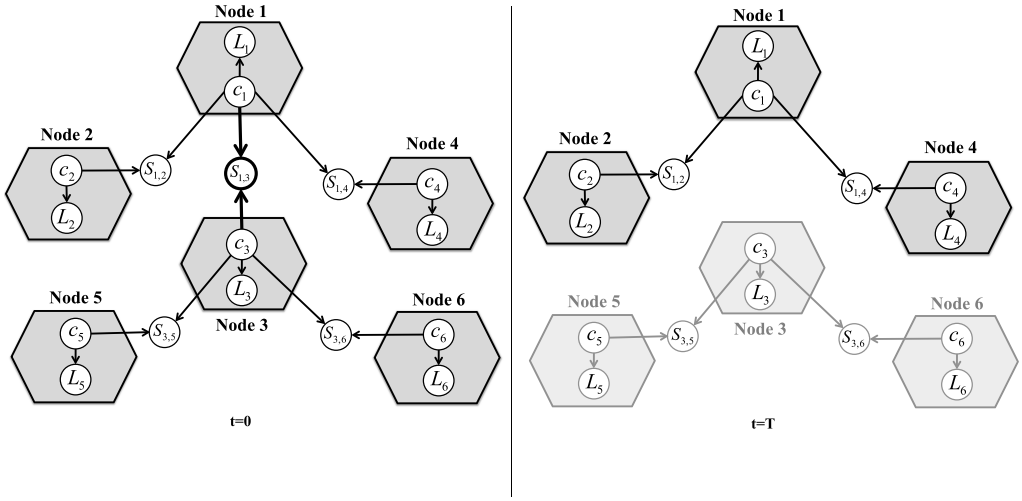

Fig. 4. Evolution of the BNs for a simple WSN consisting of six nodes, according to the decision of node 1 disconnecting from node 3, as described in Fig. 2.

At the end of the convergecast phase, the root node r computes its optimal class label assignment as

$$c_r^* = \underset{c_r}{\text{argmax}}\ p(c_r)p(L_r|c_r) \prod_{z \in CN(r)} \mu_{z \to r}(c_r). \tag{4}$$

Afterwards, the root node starts the broadcast procedure that allows intermediate and leaf nodes to compute their optimal class given the one of their respective parent. Each child node i computes its optimal label, after receiving the optimal class of its parent, c_j^*, as follows:

$$c_i^* = \underset{c_i}{\text{argmax}}\ \phi(c_i, c_j^*). \tag{5}$$

Moreover, each node is able to compute the probability of classification error p_{err}^i as follows:

$$\begin{aligned} p_{\text{corr}}^i &= p(c_i^*|L_1, \ldots, L_N, S_{1,2}, \ldots, S_{N-1,N}) = \phi(c_i^*, c_j^*) \\ p_{\text{err}}^i &= 1 - p_{\text{corr}}^i \end{aligned} \tag{6}$$

where p_{corr}^i is the probability that the chosen class label c_i^* is correct, given all the evidence within the cluster. The previous formula represents the marginalization of (1), with respect to the other class labels, and is equal to the belief $\phi(c_i^*, c_j^*)$.

For the sake of simplicity, the above description assumes that computation in nodes is triggered after each reading; however, especially when considering high-rate phenomena, it is more efficient to buffer a number of consecutive readings before transmission; in our practical implementation we used a slightly modified version which runs the inference algorithm on data tuples, thus minimizing the overall number of exchanged messages.

B. Neighborhood Selection

In order to find the optimal structure for the cooperation network, we propose a dynamic and distributed algorithm, where each sensor node chooses the neighbors to cooperate with, on the basis of the values of some quality metrics associated with the different configurations. Our algorithm aims to identify the network configuration which corresponds to the optimal trade-off among classification accuracy, time complexity and communication complexity. Neighborhood selection is performed locally by each sensor node, through Pareto optimization that allows to consider more than one conflicting objective function. Pareto optimization is suited for problems involving multiple objective functions that require to be simultaneously optimized, when these functions are characterized by non-comparable measurement units, and thus it is not possible to combine them into a single objective function [37], [38]. In a multiple-objective problem, a single solution which simultaneously optimizes all the considered functions may not exist. In such case, Pareto optimization allows to find a set of optimal solutions, named Pareto optimal front, which can be detected through a polynomial algorithm [39], [40]. The choice of a single solution inside the Pareto front has to be made according to specific application criteria.

In our system each node can select its neighborhood by making a decision d about connecting to or disconnecting from some nearby node. The node evaluates the fitness of each available decision by means of a quality vector Q_d. In order to determine the best decision to be taken, we adopt the Pareto dominance as an order relation.

A decision d_1 Pareto dominates another decision d_2, if d_1 outperforms d_2, with respect to all the considered quality metrics. If Q_{d_1} and Q_{d_2} are the quality vectors of the considered decisions, and a minimization problem is considered, then the Pareto dominance of d_1 with respect to d_2 is expressed by the following equation:

$$d_1 \preceq d_2 \Leftrightarrow \left\{\forall k = 0, \ldots, n \Rightarrow Q_{d_1}(k) \leqslant Q_{d_2}(k)\right\}. \tag{7}$$

A decision can be considered Pareto optimal if it is not worse than (i.e., dominated by) any other decision

$$d^* = \{d_i \in D{:}\forall d_j \in D, d_j \neq d_i \Rightarrow d_i \preceq d_j\}. \tag{8}$$

Furthermore, we propose to consider a constrained optimization problem, where it is possible to impose some application-specific constraints about the quality metrics. If v is the vector of such constraints, then a decision d is said to

TABLE I
PREDICTED CHANGES FOR q_{time} FOR DIFFERENT RECONFIGURATION ACTIONS

Action	Description	q_{time}
action = (connect, j) and j will become the new parent	i is the root of its cooperation network, and has to evaluate how its subtree affects the structure of the new cooperation network	$q^i_{time} = max\{q^i_{time} + 2 + 2dist^j, q^j_{time} + 1\}$
action = (connect, j), and j will become a new child	i has to evaluate how the subtree of the new child affects the structure of the new cooperation network	$q^i_{time} = max\{q^i_{time}, q^j_{time} + 1 + 2dist^i\}$
action = (disconnect, j), and j is the current parent	i will became the root of its cooperation network; it has to evaluate the depth of the subtrees rooted in its children k	$q^i_{time}(action) = \max_k \{ST(k)\} + 1$
action = (disconnect, j), and j is a child	(1) if the subtree rooted in j is the main cause of the current q_{time}, then in the new configuration another child k will affect such value; (2) otherwise, the current q_{time} is due to another area of the cooperation network, and thus such change does not affect the q_{time} metric.	$q^i_{time} = \max_{k \neq j} \{ST(k)\} + 1 + 2dist^i$ $q^i_{time} = q^i_{time}$

TABLE II
ADOPTED NOTATION

Name	Description	Name	Description
N	Number of network nodes	c_i	Hidden variable of node i in X
Δt	Sampling rate of sensor nodes	C	Set of possible values for the hidden variables c_i
$G(V, E)$	Communication network (V is the set of network nodes and E the set of communication links)	d	A decision about the reconfiguration of the cooperating neighborhood
$G'(V, E')$	Cooperation network made up of a set of network nodes and a set of cooperation links	D	Set of the possible decisions
$e_{i,j}$	Edge of the cooperation network	Q_d	Quality vector for decision d
$B(X, R)$	a BN of a set random variables (X) and a set of causal links (R)	q_{err}	Heuristic for predicting the classification error
$CN(i)$	Neighborhood set of node i in the cooperation network	q_{time}	Heuristic for predicting the time complexity
$N(i)$	Neighborhood set of node i in the communication network	q_{comm}	Heuristic for predicting the communication complexity
L_i	Set of local observed variables of node i in X	v	Vector of constraints on the metrics
$l_{i,j}$	The j-th local observed variables of node i	p_{err}	Probability of classification error
k_l	Number of local variables $\|L_i\|$	M	Number of objective functions to be optimized
$S_{i,j}$	Set of shared observed variables of node i and j in X	T	Number of steps before a new optimization occurs
$s_{i,j,k}$	The k-th shared observed variables of node i and j	ST	Vector storing the depth of the subtrees rooted at the children of the evaluating node

be admissible if and only if $d \preceq v$, that is

$$\forall k = 0, \ldots, n \Rightarrow Q_d(k) \leqslant v(k). \tag{9}$$

Neighborhood selection aims to find the best decision d^* that steers the cooperating node list $CN(i)$ toward the configuration characterized by an optimal quality vector Q_{d^*} without violating the constraints v.

Each decision by node i corresponds to a single atomic action; in particular i can choose to connect to or disconnect from one of its neighbors, or to leave the cooperating node list unchanged (this decision is named do-nothing).

The method proposed here can be generalized with respect to any set of quality metrics. We propose to consider classification error, time complexity and communication complexity, so that the quality vector is defined as

$$Q_d = (q_{\text{err}}, q_{\text{time}}, q_{\text{comm}}). \tag{10}$$

Considering a specific time t, the first metric, q_{err}, represents the average classification error of cooperating neighbors, over a time window of size T, and is defined for node i as

$$q^i_{\text{err}} = \frac{\sum_{j \in CN(i)} \sum_{\tilde{t}=t-T+1}^{t} p^j_{\text{err}}(\tilde{t})}{|CN(i)| \cdot T}. \tag{11}$$

The second metric, q_{time}, is the time complexity, measured as the number of rounds necessary for a node to classify its current readings [41]; such metric is related to the delay of the outlier detection algorithm. As shown in [33], the time complexity of max-product algorithm for node i is

$$q^i_{\text{time}} = 1 + depth + dist^i \tag{12}$$

where *depth* is the depth of the whole tree underlying the cooperation network and $dist^i$ is the distance of node i from its root node, as will be detailed later. Such value, for the current configuration of the cooperation network, can be simply evaluated by piggybacking some counters during the max-product algorithm; during the convergecast phase, leafs start individual counters which are incremented at each hop toward the root, and let nodes store the depth of the subtrees rooted at their children in a vector named *ST*. Each node informs its parent about the maximum depth of its subtrees in order to allow the root node to compute the depth of its cluster within the cooperation network (as the maximum depth of its subtrees). In order to assess possible modifications of the cooperation network, each node needs to predict how a change in its cooperation list may affect q_{time}, as described in Table I.

The third metric, q_{comm}, corresponds to the communication complexity of the outlier detection algorithm, that is the number of messages required for the algorithm to converge [41].

Such metric is directly related to the energy consumption of the WSN, and thus to its lifetime. Indeed, since radio is the most energy-hungry component for WSN nodes, the greater the number of messages exchanged by the algorithm, the greater the energy consumption for a single sensor node, as stated by Puccinelli and Giordano [42]. In the following, we will prove that the number of messages required by ADOD (per sensor node) is proportional to the number of cooperating neighbors; thus the heuristic adopted by node *i* for estimating the third metric is:

$$q^i_{\text{comm}} = |CN(i)|. \tag{13}$$

The computation of these metrics requires that each node collects *ST* and the mean values of p^j_{err}, $dist^j$ and q^j_{time} from its neighbors in the communication network, before performing the neighborhood selection algorithm.

In summary, node *i* evaluates the impact of changing its cooperating node list by predicting the future values of the quality metrics for the decisions of connecting to or disconnecting from each neighbor in the communication network; all decisions violating the specified constraints are filtered out. Finally, the surviving decisions are ordered by using the Pareto sorting algorithm proposed in [39] that computes the Pareto optimal front. If several solutions belong to the Pareto optimal front, one among them is selected randomly.

C. Complexity Analysis of ADOD

ADOD is a fully distributed algorithm and each sensor node runs the same code, as described by the high-level description reported in Fig. 5.

In order to perform outlier detection, the module takes as input the *K* most recent sensory readings and computes the most probable class to which the last reading belongs, the probability of a classification error and the estimated quality metrics for the current configuration of the cooperation network. Those *K* readings are used to compute the set of local observable variables L_i. Moreover, node *i* shares its last reading with its cooperating neighbors and receives their readings in order to compute the set of shared observable variables $S_{i,j}$. Finally, node *i* performs the convergecast and the broadcast phases of the max-product in order to update its belief about the classification of its last reading, and to acquire some structural information about the current cooperation network, namely cluster depth, the depth of its subtrees, and its distance from the cluster root.

Every *T* time steps, a reconfiguration of the cooperating node list *CN* occurs; the neighborhood selection module uses the current cooperating node list *CN* to compute the convenience of the possible actions (connecting/disconnecting/do-nothing) by means of Pareto optimization. Non satisfactory actions are rejected by comparison against the constraint vector *v*. Neighborhood selection eventually produces a new cooperating node list, which will be used by outlier detection for the subsequent *T* time steps. If a node decides to connect to or disconnect from another node, it sends a message in order to allow that node to properly update its own cooperating node list.

Data: i the node id; t, the current time step;
Z the vector of the last K readings;
CN, the current cooperating node list;
v, the vector of constraints.
Msg from i **to** j: $M_{meas} : [r^i_t]$;
$M_{conv} : [\mu_{i\to j}(c_j), ST_i]$;
$M_{broad} : [c^*_i, depth, dist_i]$;
$M_{notif} : [decision]$.

begin block Outlier Detection:
Compute local observable variables L_i;
Exchange the last reading with nodes in CN (send and receive M_{meas} msgs);
Compute shared observable variables $S_{i,j}$ for all $j \in CN$;
//— convergecast phase —
if *i* is a leaf
Compute the initial belief bel using Equation 3;
Set $ST \leftarrow [0]$;
Send M_{conv} to parent node in CN;
else
Wait for M_{conv} from children in CN;
Updated bel and ST according to received messages;
Send M_{conv} to parent node in CN;
//— broadcast phase —
if *i* is the cluster root
Compute $depth$ according to received messages;
Set $dist^i = 0$;
Compute c^* using Equation 4;
Send M_{broad} to children in CN;
else
Wait for M_{broad} from parent node in CN;
Compute c^* using Equation 5;
Update $dist^i$ and $depth$ according to received messages;
Send M_{broad} to children in CN;
Compute p^i_{err};
Compute current quality metrics Q.
return $\rightarrow c^*$, p^i_{err}, Q, $dist^i$, $depth$

begin block Neighborhood Selection:
if received M_{notif} messages
Update CN;
do (every T time steps)
Compute q^i_{time};
Exchange $dist^j$, q^j_{time}, and p^j_{err}, averaged over last K readings, with neighbors in the communication network;
for all decision $d \in D$
Compute quality vector Q_d;
Filter d if not $d \preceq v$;
Pareto sort $(d_1, d_2, ..., d_n) \rightarrow d^*$;
Update CN according to the decision d^*;
if $d^*.action$==disconnect || $d^*.action$==connect
Send M_{notif} to $d^*.id\text{-}neighbor$;
return $\rightarrow CN$

Fig. 5. ADOD algorithm.

In order to prove the suitability of ADOD for a WSN we performed a complete complexity analysis of ADOD.

1) Computational Complexity: The computational complexity of ADOD is assessed by individually considering the computational complexity of its functional blocks.

The outlier detection module performs probabilistic inference over a distributed Bayesian network. Such inference has been proved to be *NP*-hard; however, if the BN is loop-free, it becomes polynomial. The following theorem proves that

starting from a loop-free communication network also the resulting Bayesian network is a loop-free graph.

Theorem 1: If the routing algorithm builds a connected loop-free graph $G = (V, E)$, then the resulting Bayesian network $B = (X, R)$ is also loop-free.

Proof: For the sake of simplicity, and without loss of generality, let us consider the particular case $G' = G$, i.e., $E' = E$. Proving that, when $E' = E$, B is loop-free, implies that this stays true also when $E' \subseteq E$. This theorem is proved by below induction.

a) *Base case 1:* If $G = (\{1\}, \varnothing)$ then the mapping $G \rightarrow B$, imposed by the construction rules, produces a loop-free graph (see rules 1, 2, 3 for the definition of a BN on page 4), as shown in Fig. 3(a).
b) *Base case 2:* Let us consider $G = (\{1, 2\}, \{e_{1,2}\})$. For the base case 1, node 1 and node 2 correspond to a pair of loop-free structures. The mapping of $e_{1,2}$ adds the shared variables set $S_{1,2}$ and two links from c_1 and c_2 directed toward $S_{1,2}$ (rules 4, 5, 6). This construction produces a loop-free structure [see Fig. 3(b)].
c) *Inductive case:* Let us assume that for $|V| = N$ the corresponding B has a loop-free structure. When a new node j is added to the communication network, its new representation is $G^{new} = (V \cup j, E \cup e_{i,j})$. This is due to our assumption that the communication network is a connected and loop-free graph, the new node has to be connected exactly to one node i belonging to V. Adding node j, according to the construction rules, again produces a loop-free structure: the mapping of node j corresponds to a loop-free structure (rules 1, 2, 3) and the mapping of $e_{i,j}$ (rules 4, 5, 6) produces a loop-free structure according to the base case 2. Thus B^{new} built over G^{new} has a loop-free structure.

We demonstrated in a previous work [33] that, in the worst-case, the computational complexity of the max-product algorithm running over a loop-free Bayesian network is equal to $O(|C|^2(k_f + |N(i)|))$ for sensor node i, where $|C|$ is the number of outlier classes, k_f is the number of observed variables and $|N(i)|$ is the number of neighboring nodes of i in the communication network.

The neighborhood selection block relies on the Pareto sort algorithm proposed in [39], whose complexity is $O(M|D|^2)$, where M is the number of objective functions and $|D|$ is the number of decisions to be analyzed. Since, in our approach, the possible actions for a node only include either toggling its connection to a neighbor, or do nothing, then the number of possible actions is equal to the size of the neighborhood in the communication network incremented by one. The computational complexity therefore is $O(M(|N(i)| + 1)^2)$.

The average number of operations of ADOD, for any sensor node at each time step, is

$$\begin{aligned} &O\left(|C|^2\left(k_f + |N(i)|\right) + \frac{1}{T}M\left(|N(i)| + 1)^2\right)\right) \\ &\quad = O\left(4^2\left(4 + |N(i)|\right) + \frac{3}{T}\left(|N(i)| + 1)^2\right)\right) \\ &\quad = O\left(|N(i)| + \frac{1}{T}|N(i)|^2\right) \end{aligned} \tag{14}$$

since neighborhood selection occurs only every T time steps. The number of operations per sensor node is polynomial, which makes ADOD suitable for execution also on devices with limited resources.

2) Communication Complexity: The communication complexity per single sensor node corresponds to the number of messages required by ADOD. Such messages are due to the initial exchange of readings for the computation of shared variables, to the convergecast and the broadcast phase of the max-product, and finally to the communication of reconfiguration actions after the neighborhood selection.

The initial exchange of readings involves all the cooperative neighbors for a total amount of $|CN(i)|$ messages. The convergecast phase only requires each node to send one message to its parent, while during the broadcast phase, each node needs to communicate the chosen class label for its sensory reading to all of its children, which requires $|CN(i)| - 1$ messages. The number of messages due to the neighborhood selection (one message every T time steps in the worst case) can be considered negligible.

The total number of messages sent by each node running ADOD amounts to $2|CN(i)|$, and its communication complexity is $O(|CN(i)|)$.

3) Time Complexity: ADOD does not require explicit synchronization of the nodes' clocks, nevertheless the interaction between nodes requires their coordination during the broadcast and covergecast phases. More specifically, each step in the flow of belief within a cooperation network can be regarded as round of a synchronous algorithm. As a consequence, the time complexity of ADOD can be evaluated as the number of rounds between sensory measurement and the relative classification, according to the definition provided in [41].

One round is required for the evaluation of shared variables, a number of rounds equal to the depth of the cooperation network is due to the convergecast phase, and a number of rounds equal to the distance of the considered node from the root is due to the broadcast phase. For a generic node i the required number of rounds is thus expressed by $1 + depth + dist^i$, whose values vary between $1 + depth$ for the root and $1 + 2depth$ for the deepest leaf. Thus, for a given cooperation network, the time complexity is $O(depth)$.

V. Experiment Results

The experimental evaluation of the ADOD algorithm aims to prove its adaptivity with respect to different constraints on the quality metrics, and with respect to different amounts of corruption on the sensed data. We built a communication network arranged as a tree, with depth 8 and maximum branching factor 3, and composed by $N = 100$ sensor nodes measuring temperature for five days at $\Delta t = 120s$ intervals.

The dataset used for the experiments was built upon a public repository available from [43] (Mica2Dot sensor nodes at Berkeley Lab), through the simulator proposed in [44]. In particular, ten sensor nodes were selected as real traces. Nine additional artificial traces per each real one were obtained by adding a random Gaussian noise signal $\mathcal{N}(0, \sigma_E^2)$ ($\sigma_E^2 = 0.02$) to the original trace, for a total amount of 90 artificial traces, and 10 unmodified ones.

Outliers were injected by corrupting the obtained dataset, according to the method proposed in [6] which assumes the following models of faults:

$$\begin{aligned} \text{Spike:}&\tilde{r}(t) = g \times r(t) \\ \text{Noise:}&\tilde{r}(t) = r(t) + \mathcal{N}(0, \sigma_N^2) \\ \text{Stuck-at:}&\tilde{r}(t+i) = r(t) + \mathcal{N}(0, \sigma_\theta^2),\ i \in \{1, \ldots, k\} \end{aligned} \quad (15)$$

where $\tilde{r}(t)$ is a corrupted reading at time t, g is a gain constant, $\mathcal{N}(0, \sigma_\theta^2)$ is a Gaussian noise with zero mean and σ_θ^2 variance, and k is the duration of the stuck-at outlier. We set $g = 1$, $\sigma_N^2 = 2.5$, $\sigma_\theta^2 = 10^{-9}$, and $k = 10$.

Performance of ADOD has been evaluated for two different types of scenarios, named standard and critical. A standard scenario assumes a uniform distribution of outliers over time and space; different standard scenarios have been considered by varying the percentage of outliers from 10% to 50% of the total number of readings. In a critical scenario, only a small portion of the sensor network, and for a limited time, is corrupted by a high amount of outliers, namely 70% of the total number of readings. For each scenario, the performance of ADOD, in terms of classification accuracy, time complexity and communication complexity, has been evaluated. Learning the parameters required by the outlier detection algorithm, namely conditional and *a priori* probability tables for the BN, has been performed by using the readings of the first two days as training set; the remaining readings were used as test set. Both the training and the test sets contain corrupted readings.

Finally, in our experiments, Pareto optimization is performed every $T = 30$ time steps; the results shown are averaged over time windows of one hour and over all the considered sensor nodes.

A. Standard Scenario

The performance of ADOD was assessed for three different amounts of corruption, i.e., 10%, 25%, and 50% of the total readings, for different constraints on the considered metrics, and was compared against two static benchmark topologies. The first benchmark consists in a connected network where $G = G'$, i.e., the cooperation network and the communication network coincide; in this case the average time complexity amounts to 14 rounds, while the average communication complexity amounts to six sent messages, i.e., twice the number of neighbors in the cooperative list of a single node. This first benchmark (named "100% active links") marks the upper bound for the three considered metrics. In the second benchmark, nodes do not cooperate with each other, i.e., $E' = \varnothing$ and the number of clusters is equal to the number of nodes; in this case both the average time complexity, and the average communication complexity fall to zero. This second benchmark (named "0% active links") identifies the lower bound for the considered metrics, as sensor nodes exploit only temporal correlation among their own readings to detect outliers.

The aim of the experimental evaluation performed in standard scenarios is to prove that ADOD is able to adapt the structure of the cooperation network, and thus of the corresponding BN, according to different imposed constraints, at the cost of penalizing unconstrained objective functions. The adopted constraints are the following:

$$\begin{aligned} v_1 &= (v_{\text{err}} = 5, v_{\text{time}} = \infty, v_{\text{comm}} = \infty) \\ v_2 &= (v_{\text{err}} = \infty, v_{\text{time}} = 2, v_{\text{comm}} = 2). \end{aligned} \quad (16)$$

Constraint v_1 imposes a bound of 5% on the classification error while time and communication complexity are both left unbounded; in this case ADOD will increase the dimension of collaborative clusters until the bound on the classification error is satisfied. On the contrary, constraint v_2 imposes a maximum of 2 rounds for time complexity and of 2 messages for communication complexity, while leaving the classification error unbounded; this means that ADOD will break enough links in the cooperation network, so as to obtain clusters composed by two nodes at most.

Fig. 6 evaluates the performance of ADOD executed over the static benchmarks and over the dynamic ones, for three rates of data corruption; the vertical axis in the first-row diagrams represents classification accuracy, computed as the percentage of correctly classified sensory readings; in the second-row diagrams, it indicates time complexity, evaluated as the average number of rounds required to complete the outlier detection; in the third-row diagrams, it represents communication complexity, evaluated as the average number of messages that each node has to send to its cooperative neighbors.

The "100% active links" configuration fully exploits spatio-temporal dependencies among readings and thus it obtains the best classification accuracy; on the other hand, it requires the highest time complexity and the highest communication complexity.

The "0% active links" configuration achieves the best result in terms of time complexity and communication complexity, since no communication is required to classify readings, and nodes can quickly detect outliers by exploiting only temporal dependencies. This configuration allows ADOD to adequately perform outlier detection when the amount of corruption is limited: in the scenario characterized by 10% corruption, it correctly classifies 96% of the readings, while when the amount of corruption is 50%, the classification accuracy drops to 82%. Obviously, in all considered scenarios, the "0% active links" configuration achieves the worst result for classification accuracy.

Analyzing the performance of ADOD with a constraint over classification accuracy (v_1), it is possible to note that it approaches the constraint when the amounts of corruption is 10% or 25%, while it falls slightly below the threshold when corruption amounts to 50%. The drawback is an increase in time and communication complexity as the corruption percentage increases. In particular, when the corruption amounts to 10%, the mean value for time complexity is two rounds, while it grows to 12 rounds when the corruption rate rises to 50%; at the same time, communication complexity rises from 2 to 5 messages. In other words, the higher the corruption rate, the greater the size of the cooperating clusters necessary to achieve an adequate classification accuracy. Moreover, while classification accuracy remains close to the "100% active links" benchmark, the average time complexity and communication complexity obtain a significant reduction.

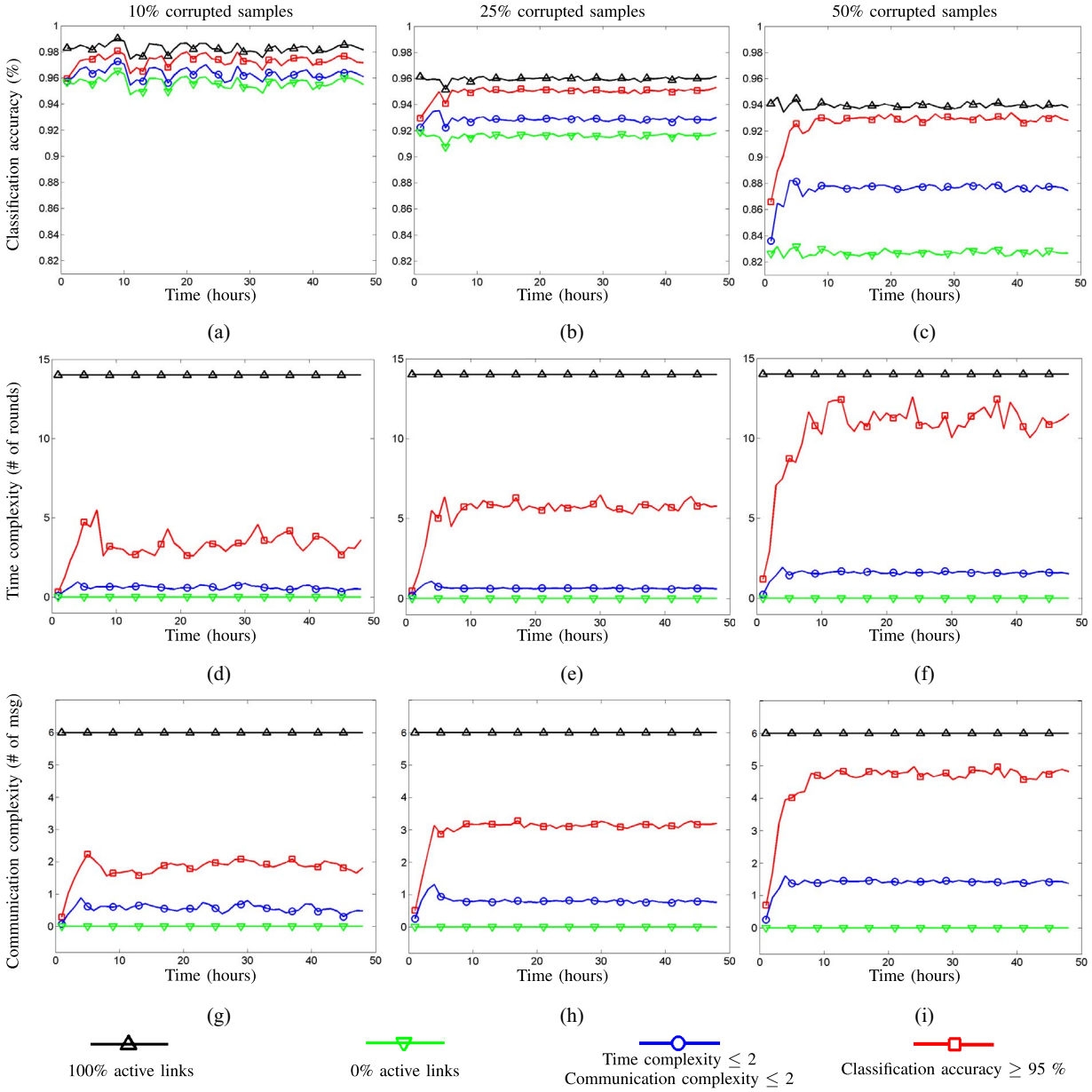

Fig. 6. Performance of ADOD compared with performance of static benchmarks: classification accuracy when the amount of corruption is (a) 10%, (b) 25%, (c) 50%; time complexity when the amount of corruption is (d) 10%, (e) 25%, (f) 50%; communication complexity when the amount of corruption is (g) 10%, (h) 25%, (i) 50%

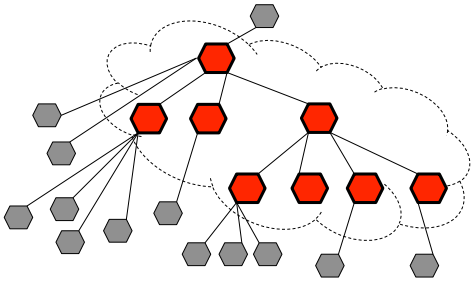

Fig. 7. Sensor nodes affected by an anomalous amount of corruption with respect to the rest of the WSN.

Finally, when ADOD has a constraint on time and communication complexity (v_2), it always satisfies such requirement at the cost of a small decrease in classification accuracy as the corruption rate increases. The benefit of constructing small clusters is evident for the scenario characterized by 50% corruption rate; in this case, the average time complexity and communication complexity are quite close to the "0% active links" result, but ADOD achieves an increment in the classification accuracy of about 88% as opposite to 82% of the benchmark.

B. Critical Scenario

We aim to prove that ADOD is able to automatically tune its behavior with respect to different percentages of corruption in sensory data, even if this occurs in different areas of the WSN. This adaptability corresponds to trading the unconstrained objective for the constrained one, only in the areas where this compromise is required.

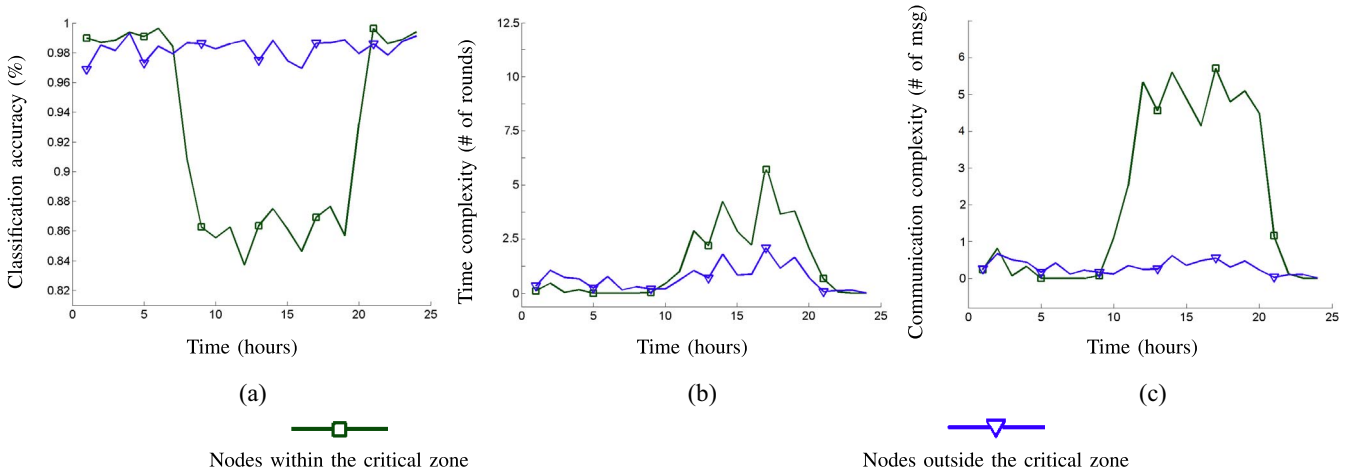

Fig. 8. Performance of ADOD for a critical scenario where a limited region of the network is affected by faults. (a) Classification accuracy. (b) Time complexity. (c) Communication complexity

In such scenario, the whole network was affected by a corruption rate of 4%, and an additional 66% rate of stuck-at and short outliers was added for 12 h within a limited area consisting of eight nodes, highlighted in Fig. 7. In such scenario there are no constraints over the considered metrics, and ADOD selects, from the Pareto front, the solution corresponding to the highest classification accuracy.

Fig. 8 compares the performance of ADOD for the nodes within and outside the critical area. As expected, before the anomalous event occurs, the performance of ADOD does not relevantly change in the two considered areas; in particular, the classification accuracy approaches 98%, while time and communication complexity are both close to zero. When the unexpected event occurs, the performance within the critical area dramatically drops, while the rest of the network maintains a better behavior, with a small increase in time complexity and communication complexity, due to the cooperation with nodes on the boundary of the critical area.

VI. Conclusion

This paper proposed a distributed outlier detection algorithm for WSNs, able to find the optimal trade-off among three conflicting goals, namely we intended to maximize the classification accuracy while minimizing time complexity and communication complexity. Our proposal is based on a probabilistic inference on Bayesian networks distributed over the wireless sensor nodes; a Pareto optimization algorithm is used to adapt the Bayesian network structure by allowing sensor nodes to choose the best neighbors to cooperate with.

Experimental results proved the capability of our system to adapt its behavior depending on different dynamics of the deployment scenario, thus achieving significant reduction in time complexity and communication complexity at the cost of a small decrease in classification accuracy if compared to non-adaptive approaches.

References

[1] A. Farruggia, G. Lo Re, and M. Ortolani, "Detecting faulty wireless sensor nodes through stochastic classification," in *Proc. 2011 IEEE Int. Conf. Pervasive Comput. Commun. Workshops*, Seattle, WA, USA, pp. 148–153.

[2] A. De Paola, S. Gaglio, G. Lo Re, and M. Ortolani, "Sensor9k: A testbed for designing and experimenting with WSN-based ambient intelligence applications," *Pervasive Mobile Comput.*, vol. 8, no. 3, pp. 448–466, 2012.

[3] A. K. Bashir, S.-J. Lim, C. S. Hussain, and M.-S. Park, "Energy efficient in-network RFID data filtering scheme in wireless sensor networks," *Sensors*, vol. 11, no. 7, pp. 7004–7021, 2011.

[4] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection: A survey," *ACM Comput. Surv.*, vol. 41, no. 3, pp. 15.1–15.58, 2009.

[5] R. Jurdak, X. Wang, O. Obst, and P. Valencia, "Wireless sensor network anomalies: Diagnosis and detection strategies," *Intell.-Based Syst. Eng.*, vol. 10, pp. 309–325, 2011.

[6] A. B. Sharma, L. Golubchik, and R. Govindan, "Sensor faults: Detection methods and prevalence in real-world datasets," *ACM Trans. Sensor Netw.*, vol. 6, no. 3, pp. 23.1–23.39, 2010.

[7] K. Ni *et al.*, "Sensor network data fault types," *ACM Trans. Sensor Netw.*, vol. 5, no. 3, pp. 25.1–25.29, 2009.

[8] Y. Zhang, N. Meratnia, and P. Havinga, "Outlier detection techniques for wireless sensor networks: A survey," *IEEE Commun. Surv. Tuts.*, vol. 12, no. 2, pp. 159–170, Apr. 2010.

[9] Y. Zhang *et al.*, "Statistics-based outlier detection for wireless sensor networks," *Int. J. Geograph. Inf. Sci.*, vol. 26, no. 8, pp. 1373–1392, 2012.

[10] W. Wu *et al.*, "Localized outlying and boundary data detection in sensor networks," *IEEE Trans. Knowl. Data Eng.*, vol. 19, no. 8, pp. 1145–1157, Aug. 2007.

[11] J. W. Branch, C. Giannella, B. Szymanski, R. Wolff, and H. Kargupta, "In-network outlier detection in wireless sensor networks," *Knowl. Inf. Syst.*, vol. 34, no. 1, pp. 23–54, 2013.

[12] S. Rajasegarar, C. Leckie, M. Palaniswami, and J. Bezdek, "Distributed anomaly detection in wireless sensor networks," in *Proc. 10th IEEE Singapore Int. Conf. Commun. Syst.*, Singapore, 2006, pp. 1–5.

[13] S. Bandyopadhyay *et al.*, "Clustering distributed data streams in peer-to-peer environments," *Inf. Sci.*, vol. 176, no. 14, pp. 1952–1985, 2006.

[14] X. Wang, J. Lizier, O. Obst, M. Prokopenko, and P. Wang, "Spatiotemporal anomaly detection in gas monitoring sensor networks," in *Proc. 5th Eur. Conf. Wireless Sensor Netw.*, Bologna, Italy, 2008, pp. 90–105.

[15] D. J. Hill, B. S. Minsker, and E. Amir, "Real-time Bayesian anomaly detection for environmental sensor data," in *Proc. Congr. Int. Assoc. Hydraulic Res.*, vol. 32. 2007, p. 503.

[16] K. Ni and G. Pottie, "Sensor network data fault detection with maximum *a posteriori* selection and Bayesian modeling," *ACM Trans. Sensor Netw.*, vol. 8, no. 3, pp. 23.1–23.21, 2012.

[17] M. Takruri, S. Challa, and R. Chakravorty, "Recursive Bayesian approaches for auto calibration in drift aware wireless sensor networks," *J. Netw.*, vol. 5, no. 7, pp. 823–832, 2010.

[18] A. Farruggia, G. Lo Re, and M. Ortolani, "Probabilistic anomaly detection for wireless sensor networks," in *AI*IA 2011: Artificial Intelligence Around Man Beyond*, vol. 6934. Berlin, Germany: Springer, pp. 438–444.

[19] M. Markou and S. Singh, "Novelty detection: A review—Part 2: Neural network based approaches," *Signal Process.*, vol. 83, no. 12, pp. 2499–2521, 2003.
[20] P. Yang, Q. Zhu, and X. Zhong, "Subtractive clustering based RBF neural network model for outlier detection," *J. Comput.*, vol. 4, no. 8, pp. 755–762, 2009.
[21] W. Ren and Y. Cui, "A parallel rough set tracking algorithm for wireless sensor networks," *J. Netw.*, vol. 7, no. 6, pp. 972–979, 2012.
[22] V. Chatzigiannakis, S. Papavassiliou, M. Grammatikou, and B. Maglaris, "Hierarchical anomaly detection in distributed large-scale sensor networks," in *Proc. 11th IEEE Symp. Comput. Commun. (ISCC)*, 2006, pp. 761–767.
[23] M. Bal, W. Shen, and H. Ghenniwa, "Collaborative signal and information processing in wireless sensor networks: A review," in *Proc. IEEE Int. Conf. Syst., Man, Cybern. (SMC)*, San Antonio, TX, USA, 2009, pp. 3151–3156.
[24] X. Su, L. Wu, and P. Shi, "Sensor networks with random link failures: Distributed filtering for T-S fuzzy systems," *IEEE Trans. Ind. Inf.*, vol. 9, no. 3, pp. 1739–1750, Aug. 2013.
[25] A. Savvides, C.-C. Han, and M. B. Strivastava, "Dynamic fine-grained localization in *ad-hoc* networks of sensors," in *Proc. 7th Annu. Int. Conf. Mobile Comput. Netw.*, 2001, pp. 166–179.
[26] D. Janakiram, V. Adi Mallikarjuna Reddy, and A. V. U. Phani Kumar, "Outlier detection in wireless sensor networks using Bayesian belief networks," in *Proc. 1st Int. Conf. Commun. Syst. Softw. Middleware (Comsware)*, New Delhi, India, 2006, pp. 1–6.
[27] K. Ni and G. Pottie, "Bayesian selection of non-faulty sensors," in *Proc. IEEE Int. Symp. Inf. Theory*, Nice, France, 2007, pp. 616–620.
[28] R. Hoes, T. Basten, C.-K. Tham, M. Geilen, and H. Corporaal, "Quality-of-service trade-off analysis for wireless sensor networks," *Perform. Eval.*, vol. 66, no. 3, pp. 191–208, 2009.
[29] L. Wang *et al.*, "The node placement of large-scale industrial wireless sensor networks based on binary differential evolution harmony search algorithm," *Int. J. Innov. Comput. Inf. Control*, vol. 9, no. 3, pp. 955–970, 2013.
[30] M. C. Vuran, Ö. B. Akan, and I. F. Akyildiz, "Spatio-temporal correlation: Theory and applications for wireless sensor networks," *Comput. Netw.*, vol. 45, no. 3, pp. 245–259, 2004.
[31] M. C. Vuran and Ö. B. Akan, "Spatio-temporal characteristics of point and field sources in wireless sensor networks," in *Proc. IEEE Int. Conf. Commun.*, vol. 1. Istanbul, Turkey, 2006, pp. 234–239.
[32] Y. Weiss and W. Freeman, "On the optimality of solutions of the max-product belief-propagation algorithm in arbitrary graphs," *IEEE Trans. Inf. Theory*, vol. 47, no. 2, pp. 736–744, Feb. 2001.
[33] G. Lo Re, F. Milazzo, and M. Ortolani, "A distributed Bayesian approach to fault detection in sensor networks," in *Proc. IEEE Global Telecommun. Conf. (GlobeCom)*, Anaheim, CA, USA, 2012, pp. 634–639.
[34] S. Singh, M. Singh, and D. Singh, "Routing protocols in wireless sensor networks—A survey," *Int. J. Comput. Sci. Eng. Surv.*, vol. 1, no. 2, pp. 63–83, 2010.
[35] A. M. Islam, K. Wada, and W. Chen, "Dynamic cluster-based architecture and data congregation protocols for wireless sensor network," *Int. J. Innov. Comput. Inf. Control*, vol. 9, no. 10, pp. 4085–4099, 2013.
[36] T. Le and C. Hadjicostis, "Max-product algorithms for the generalized multiple-fault diagnosis problem," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 37, no. 6, pp. 1607–1621, Dec. 2007.
[37] D. Jourdan and O. de Weck, "Layout optimization for a wireless sensor network using a multi-objective genetic algorithm," in *Proc. IEEE 59th Veh. Technol. Conf. (VTC 2004-Spring)*, vol. 5. pp. 2466–2470.
[38] D. Niyato, E. Hossain, M. Rashid, and V. Bhargava, "Wireless sensor networks with energy harvesting technologies: A game-theoretic approach to optimal energy management," *IEEE Wireless Commun.*, vol. 14, no. 4, pp. 90–96, Aug. 2007.
[39] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Trans. Evol. Comput.*, vol. 6, no. 2, pp. 182–197, Apr. 2002.
[40] X.-B. Hu, M. Wang, and E. Di Paolo, "Calculating complete and exact Pareto front for multiobjective optimization: A new deterministic approach for discrete problems," *IEEE Trans. Cybern.*, vol. 43, no. 3, pp. 1088–1101, Jun. 2013.
[41] N. Lynch, *Distributed Algorithms*. San Francisco, CA, USA: Morgan Kaufmann, 1996.
[42] D. Puccinelli and S. Giordano, "Connectivity and energy usage in low-power wireless: An experimental study," in *Proc. 2013 IEEE Int. Conf. Pervasive Comput. Commun. Workshops (PERCOM Workshops)*, San Diego, CA, USA, pp. 590–595.
[43] S. Madden. (2004). *Intel Lab Data*. [Online]. Available: http://db.csail.mit.edu/labdata/labdata.html
[44] A. Lalomia, G. Lo Re, and M. Ortolani, "A hybrid framework for soft real-time WSN simulation," in *Proc. 13th IEEE/ACM Int. Symp. Distrib. Simul. Real Time Appl. (DS-RT)*, 2009, pp. 201–207.

Alessandra De Paola received the bachelor's, master's, and Ph.D. degrees in computer engineering from the University of Palermo, Palermo, Italy, in 2004, 2007, and 2011, respectively.

She is an Assistant Professor of Computer Engineering at the University of Palermo, since 2012. Her current research interests include artificial intelligence applied to distributed systems, wireless sensor networks, ambient intelligence, and network security.

Salvatore Gaglio is Full Professor of Computer Science and Artificial Intelligence at the University of Palermo, Palermo, Italy, from 1986. From 1998 to 2002, he was the Director of the Study Center on Computer Networks of the CNR (National Research Council), from 2002, he is the Director of the Branch of Palermo of High Performance Computing and Computation of CNR. His current research interests include artificial intelligence and robotics.

Prof. Gaglio was member of the Scientific Council of the ICT Department of CNR, from 2005 to 2012. He is also a member of ACM and AAAI.

Giuseppe Lo Re received the Laurea degree in computer science from the University of Pisa, Pisa, Italy, in 1990, and the Ph.D. degree in computer engineering from the University of Palermo, Palermo, Italy, in 1999.

He is an Associate Professor of Computer Engineering at the University of Palermo, since 2004. In the 1991, he joined the Italian National Research Council (CNR), where he achieved the Senior Researcher position. His current research interests include computer networks, distributed systems, wireless sensor networks, ambient intelligence, and internet of things.

Prof. Lo Re is Senior Member of the IEEE Communication Society and the Association for Computer Machinery.

Fabrizio Milazzo received the B.S., M.Sc., and Ph.D. degrees in computer science from the University of Palermo, Palermo, Italy, in 2009, 2006 and 2014, respectively.

From 2013 to 2014, he was Research Fellow with the Power Systems Laboratory at the University of Palermo. His current research interests include network security, fault detection, data prediction and machine learning applications in wireless sensor networks, discrete optimization techniques, and applications in power systems.

Marco Ortolani received the M.Sc. and Ph.D. degree in computer engineering from the University of Palermo, Palermo, Italy, in 2000 and 2004 respectively.

Since 2008, he has been an Assistant Professor at the University of Palermo. His current research interests include networking, distributed systems, and intelligent data analysis techniques applied to ambient intelligence and the internet of things. He has authored over 50 peer-reviewed papers.

Dr. Ortolani has served as a a reviewer for several international journals and a TPC Member of international conferences sponsored by IEEE and ACM. He is member of ACM.