Information Fusion xxx (2013) xxx–xxx

Contents lists available at SciVerse ScienceDirect

Information Fusion

journal homepage: www.elsevier.com/locate/inffus

An approach to implement data fusion techniques in wireless sensor networks using genetic machine learning algorithms

A.R. Pinto [a,*], C. Montez [b], G. Araújo [b], F. Vasques [c], P. Portugal [d]

[a] UNESP, Universidade Estadual Paulista, SP, Brazil
[b] PGEAS, UFSC, Universidade Federal de Santa Catarina, SC, Brazil
[c] IDMEC, FEUP, Universidade do Porto, Porto, Portugal
[d] INESC TEC, FEUP, University of Porto, Porto Portugal

ARTICLE INFO

Article history:
Received 25 April 2010
Received in revised form 5 March 2013
Accepted 9 May 2013
Available online xxxx

Keywords:
Partial data fusion
Autonomic computing
Classifier systems

ABSTRACT

Wireless Sensor Networks (WSNs) can be used to monitor hazardous and inaccessible areas. In these situations, the power supply (e.g. battery) of each node cannot be easily replaced. One solution to deal with the limited capacity of current power supplies is to deploy a large number of sensor nodes, since the lifetime and dependability of the network will increase through cooperation among nodes. Applications on WSN may also have other concerns, such as meeting temporal deadlines on message transmissions and maximizing the quality of information. Data fusion is a well-known technique that can be useful for the enhancement of data quality and for the maximization of WSN lifetime. In this paper, we propose an approach that allows the implementation of parallel data fusion techniques in IEEE 802.15.4 networks. One of the main advantages of the proposed approach is that it enables a trade-off between different user-defined metrics through the use of a genetic machine learning algorithm. Simulations and field experiments performed in different communication scenarios highlight significant improvements when compared with, for instance, the Gur Game approach or the implementation of conventional periodic communication techniques over IEEE 802.15.4 networks.

1. Introduction

Wireless Sensor Networks (WSNs) are composed of a large number of sensor nodes (also known as motes), usually with reduced size. Each node has a processor, memory, sensors, a wireless communication module and a power supply. In the most common situations, the power supply is provided by batteries. These networks can be deployed in hazardous areas, where maintenance is a difficult task. Therefore, energy consumption is a major concern because in these situations it is very difficult to replace batteries. Besides energy, applications for WSN may also have to deal with time-constrained messages [1].

The use of dense networks (i.e. with a high number of nodes per m^2), with hundreds or thousands of nodes, may be an advantage. A dense WSN can improve the accuracy of results, overpassing the unreliable nature of individual nodes. However, the usage of dense networks imposes more complex management approaches [1,2]. The use of autonomic approaches can be an interesting alternative for dense WSN, since self-management techniques [3] are adequate methodologies to manage large number of nodes.

In dense networks, the use of data fusion approaches may be an advantage due to the large number of messages exchanged, since a data fusion task may merge several messages in an useful and accurate information for the final user [4]. Additionally, some data fusion techniques proposed in the literature can deal with delays, message losses and discards, allowing greater flexibility in both the choice of the network technology and the communication approach used to disseminate data on the network [5].

Regarding the large number of nodes, the topology of network must be considered as being dynamic, as an unpredictable number of nodes can present hardware, software or communication failures. Thus, it is extremely difficult to obtain an accurate view of the network topology, which prevents the use of static scheduling of messages.

In WSN, applications may have multiple conflicting goals, such as timing constraints conflicting with energy consumption constraints. Another example may be the case of a data fusion application that requires a large number of messages to guarantee a minimum accuracy. Nevertheless, increasing the number of message transmissions can also increase the number of collisions. This leads both to an increase of the energy consumption due to message retransmissions and to a reduction of the number of messages that arrive to the final destination (e.g. message discards due to retransmission limits).

* Corresponding author.
E-mail addresses: arpinto@ibilce.unesp.br (A.R. Pinto), montez@das.ufsc.br (C. Montez).

1566-2535/$ - see front matter © 2013 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.inffus.2013.05.003

The IEEE 802.15.4 [6] has been used extensively as a communication infrastructure for WSN to support a large number of applications. These networks may be organized in peer-to-peer or star topologies. However, large coverage area and obstacles may prevent direct communication between nodes. Therefore, it may be impossible, in some circumstances, to implement star topologies. Nevertheless, star topologies constitute an interesting solution due to several reasons. First, star topologies are simpler to manage than other alternatives. Moreover, current RF transceivers compatible with IEEE 802.15.4 can operate in 868 MHz or 900 MHz bands reaching long distances in line-of-sight conditions, which makes feasible the establishment of large outdoor networks with direct communication with a base station. Finally, it is important to note that complex topologies based on cluster-tree approaches are ultimately formed by sets of clusters configured as star topologies.

In this paper, we present a generic approach that allows the implementation of parallel data fusion techniques in IEEE 802.15.4 networks based on a star topology. The main advantage of the proposal is the trade-off among different metrics through the use of a genetic machine learning algorithm – GMLA, which allows obtaining self-management properties in dense wireless sensor networks. The adopted communication scheme is based on local node decisions to define whether sensed information will be sent, or not, to the base station. The approach was developed considering requirements related with: dynamic topology changes, applications with conflicting goals and the impossibility of human intervention.

In order to assess the advantages of the proposed approach, a set of simulations was performed to compare it with both the Gur Game [7] and the use of a simple periodic application running over IEEE 802.15.4 standard [6]. Aiming to analyze the behavior and stability of the approach, different simulation scenarios were considered, varying the number of nodes (from few ones to thousands nodes), the deployment area (100×100 m and 1000×1000 m), and node mobility (nodes can enter or leave the radio range of the base station). Experiments with motes have also been performed with the purpose of to validate the simulation results in a real environment. They show that the envisaged approach is viable to be used on real scenarios.

The remainder of this paper is organized as follow. Section 2 presents the background related to this proposal. In Section 3 related works are discussed and the proposed model is introduced. The proposed approach is discussed in Section 4. In Section 5 the proposal is assessed and results are discussed. Finally, in Section 6 conclusions and final remarks are presented.

2. Background

2.1. IEEE 802.15.4

IEEE 802.15.4 [6] is becoming a *de facto* standard for low power and low rate WSNs. The physical layer can operate up to 250 Kbps of maximum transmission rate, working in different frequencies, such as the popular 2.4 GHz and 868/900 MHz bands for long range communication in Europe and North America, respectively. The MAC layer supports two operational modes that can be selected by the PAN coordinator: *beaconless mode*, using a non-slotted CSMA/CA mechanism; and *beacon mode*, where beacons are sent periodically by the PAN coordinator, and nodes are synchronized by a superframe structure. Since our proposal uses the beaconless mode, we restrict the following discussion to this operating mode.

Two variables are maintained by each device in beaconless mode: *NB*, the number of times CSMA/CA mechanism is required to backoff and *BE*, the backoff exponent, which is related to how many backoff periods a device shall wait before attempting to access a channel. There are also three main parameters: *macMinBE*, *macMaxBe* and *macMaxCSMABackoffs*, which corresponds, respectively, to the initial value of *BE*, the maximum value of *BE* and maximum number of backoffs before being declared a channel access failure.

The protocol works in the following manner. The *BE* is initially set to *macMinBE* value. When sending a message, the MAC sublayer shall delay for a random number of backoff periods in the range of $0–2 * BE - 1$, after which requests to the physical layer to perform a channel assessment. If the channel is assessed to be busy, the MAC sublayer will increment *NB* and *BE*. Once *BE* reaches to the *macMaxBE* value, it remains at this maximum value. If the value of NB is greater than *macMaxCSMABackoffs* (default value 4), the CSMA/CA shall end with a channel access failure status.

Therefore, *macMinBE*, *macMaxBe* and *macMaxCSMABackoffs* parameters can influence the beaconless CSMA/CA performance. These default values can decrease battery consumption (a device tries to retransmit only up to five times before aborting the transmission), however when the number of nodes increases, the communication efficiency drastically decreases [8,9]. Thus, the use of IEEE 802.15.4 protocol does not seem to be adequate for dense networks without addition of a mechanism such as the one proposed in this work.

2.2. Genetic algorithms and classifier systems

Classifier systems are machine-learning algorithms based on genetic algorithms [10]. These systems are able to learn simple rules, called classifiers. In this paper we call it Genetic Machine Learning Algorithm (GMLA), where "learning", in this context, means a *continuum* online adaptation process to a partially unknown and dynamic environment. Classifier systems are composed of three main components (Fig. 1): rules and message system; bank (usually called apportionment system); and genetic algorithm (GA).

The main idea behind a classifier system is to model the system, which represents the problem, by a set of condition-action rules. As a classifier system is a learning system based on Darwinian

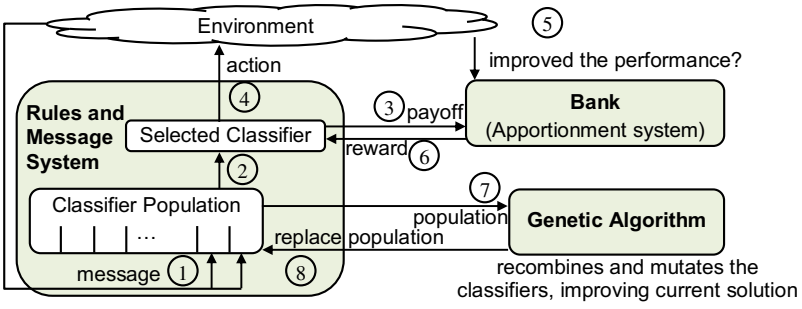

Fig. 1. Classifier system scheme.

Please cite this article in press as: A.R. Pinto et al., An approach to implement data fusion techniques in wireless sensor networks using genetic machine learning algorithms, Informat. Fusion (2013), http://dx.doi.org/10.1016/j.inffus.2013.05.003

principles, each rule is assumed to be a "creature" and the set of rules is a "population" of classifiers, which represents the current solution. A classifier system interacts with the environment receiving messages from it, selecting and performing actions.

Fig. 1 outlines the GMLA functioning. Initially, each classifier has associated a value called *budget* that represents its adaptation measure; that is, how good its action is for the system. A message sent from the environment is perceived by the system (①) and compared with the condition parts of all classifiers of the population. One classifier is selected, and becomes active (②). It has to pay a predetermined amount of its budget to the bank (③). The action part from selected classifier interacts with the environment (④). Depending on the efficiency of its actions in the environment (⑤), the bank component may eventually reward the classifier (⑥). The classifier system interacts with the genetic algorithm component by trying to maximize the amount of reward that receives from bank. At last, periodically new classifiers are inserted in the population while other are deleted keeping the population size constant (⑦ and ⑧).

Rules assume generally the following form: *if* <*condition*> *then* <*action*>. The meaning of this rule is that the *action* has to be imposed to the environment when the *condition* is satisfied. Classifiers are generally composed of three characters $\{0, 1, *\}$, where $*$ is a wildcard: it can mean 0 or 1. A message received by the system can activate one or more classifiers. Table 1 shows an example of a classifier population. When the system receives a message 101011 from the environment, then the first classifier will be activated and action 100 will be executed.

Algorithm 1 outlines the operation of the GMLA. At the startup, all classifiers receive the same budget (line 1). The system consults (receives message from) the environment (line 2) and matches with the condition part of the classifier population (line 3). When more than one classifier satisfies a predefined condition, the classifier that has the larger budget will be selected. The selected classifier pays a predetermined amount of its budget to the bank (line 4). The amount of accumulated budget by the bank will be paid to the last classifier that improved the system performance (lines 5–7). Therefore, the most adapted classifiers will increase their budgets. After some consults to the classifier system, in each time interval called *evolution_interval*, the genetic algorithm evolves the population of classifier in order to achieve better solutions to the problem (lines 8–10).

Algorithm 1. Classifier System Scheme – GMLA Algorithm

```
1:   createPopulationOfClassifiers (budget);
2:   receiveConditionFromEnvironment ();
3:   classifier = selectBestClassifier (input_condition);
4:   classifierPayToTheBank (classifier);
5:   if (System_Performance_was_improved) then
6:     classifierReceiveBankReward (classifier);
7:   end if
8:   if (consult==evolution_interval) then
9:     geneticAlgorithmEvolution ();
10:  end if
```

The genetic algorithm component in GMLA assumes a population of classifiers for its optimization problem, where individuals are represented by their genotypes, which are usually a set of bits or characters. This population is evolved by the GA after a predetermined number of consults. At each generation of answers, a new set of artificial creatures (set of characters) will be generated. These answers are based on fragments of the most adapted individuals. The main focus of GA is robustness. Once a system is more robust, it will require a smaller number of interventions or redefinitions. Moreover, it will achieve higher levels of adaptation and it will be able to execute better and longer.

The main difference between classical GA approaches and a classifier system is that a classifier system just evolves its population after some consults to the classifier set. Thus, it will perform the evolution while it is running.

2.3. Data fusion

In the literature different terms and definitions are used for data fusion techniques. Two of most often used nomenclatures, usually accepted as synonyms, are "data fusion" and "information fusion". In our work, we consider only correlated data (e.g. data obtained from sensors that monitors the same physical phenomenon in a spatio-temporal context) sent to a node that executes a data fusion operation. According to [4] this approach should be called "redundant information fusion" or just "redundant fusion". Nevertheless, in this paper we use the term "data fusion".

WSNs are often deployed in an environment to detect some physical phenomenon. This detection can be usually done in a collaborative way, i.e. combining data sensed by multiple sensors. The exchange of information between nodes is required for the reliable execution of the application due to the limited information gathered by each node, the variability on operating conditions, and also the unreliability of the nodes [4,5,11–13]. This way, the main target of a data fusion approach is to combine data sensed by multiple sensors in order to improve the quality of decisions that will be taken by a sink (i.e. base station).

In data fusion applications, sensors can be used in a variety of forms. According to [11], it is possible to distinguish three different types of sensor fusion: complementary, cooperative and competitive (also named redundant). Sensor fusion is complementary, when the configuration of sensors is such that they do not depend on each other, but their data are combined to obtain a more complete perspective of the phenomenon under observation and to solve the incompleteness of a single sensor data (e.g. multiple cameras examining separate parts of an object). Sensor fusion is cooperative, when the network uses data provided by independent sensors to gather information that would not be available from single sensors (e.g. two-dimensional images from cameras observing the same object at slightly different viewpoints in a 3D vision system [12]). On the other hand, sensor fusion is competitive if each sensor delivers independent and redundant measurements of the same phenomenon, providing fault tolerance and robustness to the system.

Collaborative detection can be done in a serial or parallel way, considering node communication. Parallel data fusion is executed when all nodes send the sensed data to the base station. On the other hand, serial data fusion uses routing techniques to collect data through the network [13]. Hybrid data fusion approaches organize the data fusion process in clusters or tree-based schemes. Therefore, parallel data fusion occurs inside a cluster, while cluster heads (nodes that perform data fusion) exchange data among them in a serial way.

Table 1
Example of classifier population.

Condition	10#01#	ll#l#0	0#1111	100,001
Action	100	111	001	110

Please cite this article in press as: A.R. Pinto et al., An approach to implement data fusion techniques in wireless sensor networks using genetic machine learning algorithms, Informat. Fusion (2013), http://dx.doi.org/10.1016/j.inffus.2013.05.003

As the communication consumes a significant part of the energy in wireless networks, ordinary parallel data fusion approaches may expend more energy than serial data fusion techniques, due to the fact that all sensed data is sent to a central node. On the other hand, serial data fusion imposes the utilization of routing algorithms. These routing algorithms are hard to implement in WSNs and can also incur in greater energy expenditure. This way, the development of approaches that autonomically adjust the data sending rate in a parallel data fusion can overcome the main problem of this technique.

2.4. Self-optimization in dense networks – Gur Game approach

Based on works of the Russian mathematician Tsetlin in the 1960s, Tung and Kleinrock [7] proposed the use of Gur Game: an approach aiming the self-optimization and self-control properties in distributed systems. The basic assumption of this work is that in distributed systems many of the tasks need to be managed in the absence of explicit coordination [7,14].

In the Gur Game proposal, each node uses a finite state automaton (Fig. 2) that runs by trial and error, seeking to maximize its probability of reward. This probability value (p) is known by all nodes and is recalculated by a central node after each period using a reward probability function. The central node (base station) has just the responsibility of evaluating for each period the benefit obtained by the system and of broadcasting this value to all nodes.

The underlying idea behind the Gur Game approach is to let each node to reward or to punish itself in a probabilistic manner, according to the benefit value obtained by the system, and evaluated from the reward probability function. Regardless of how it was the last action of the node, it is rewarded with a probability p or penalized with a probability $1 - p$. The finite state machine automaton in each node has 2 N states, with N states associated with a particular action X, and other N states associated with an action Y; being the value of N regarded as the automaton "memory" (Fig. 2).

Despite the apparent "selfish" behavior of each node, as discussed in [7], the final results usually converge to a collective gain. The intuition behind this behavior is as follows: a node that decides to reward itself remains in one of the states that maintain the previous actions (because this action presumably benefited the system). On the other hand, when a node decides to penalize itself, it changes its state towards a state that changes the previous action (probably this previous action was not good enough for the system).

In a WSN with redundant and homogeneous sensor nodes, the Gur Game can be used to control the total number of messages sent by nodes to the base station in a periodic application. An example of a Gur Game reward probability function is shown in Eq. (1), where *nnodes* represents the total number of nodes, *met* represents the number of messages that reach the base station in the last period and *target* means the objective.

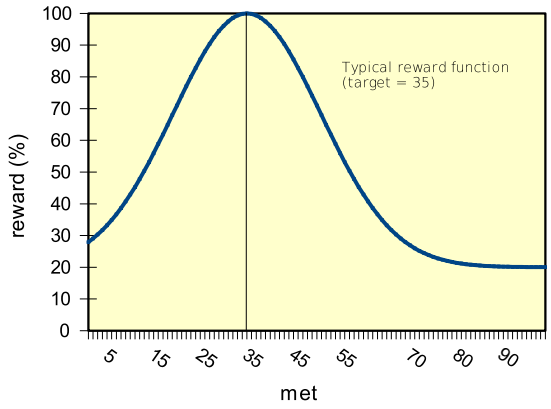

Fig. 3. A typical reward function in Gur Game approach, where the maximum of the reward function occurs when x axis assumes value 35, i.e. the target value for the function.

$$f(x) = 20 + 80 \times e^{\frac{-0.2}{(nnodes/10)} \times (met - target)^2} \quad (1)$$

Fig. 3 shows the plotting of this reward probability function when *nnodes* is equal to 100 and *target* value is equal to 35.

3. Adopted model

3.1. System architecture

The communication model used in the proposal considers one master node (base station) and multiple slave nodes (Fig. 4). The model is designed for periodic applications, thus it is assumed a interaction pattern with periodic measurements, where the slave nodes cyclically sense scalar data and send it to the master node that performs the data fusion operation. Slave nodes are organized to implement a competitive sensor fusion [12], i.e., they are redundant and sense the same phenomenon. All the slave nodes reach the master using just one hop communication. That is, a parallel data fusion is performed in the master node.

The sensor fusion process is organized in monitoring sessions, which are composed of several macrocycles (*MA*). A *MA* corresponds to a *checkpoint interval* in the GMLA, i.e. a time interval where master node receives messages, computes and sends new parameters to slave nodes. One *MA* is composed of several microcycles (*MI*). The *MI* concept is used to synchronize nodes, and also represents the periodicity of the data fusion task. On each *MI*, except for the first microcycle on each macrocycle, a slave node can send zero or one message containing the sensed data to the master node (Fig. 5).

Each message sent by a slave node has an absolute deadline D, which is the maximum time interval that it must be delivered to the master node; otherwise, it will be useless for the data fusion

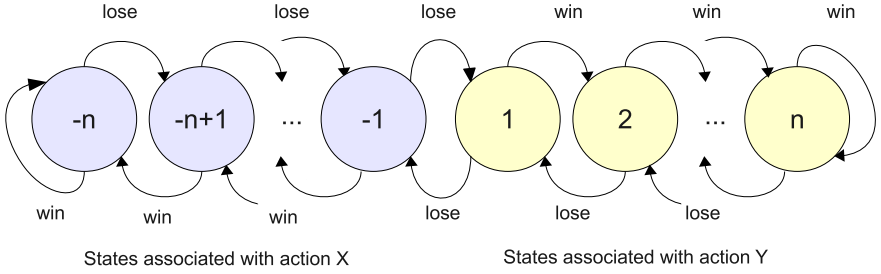

Fig. 2. Gur Game automaton with six states. When a node engages in a self-rewarding behavior (win event), its state moves away from the center; when it engages in a self-punishment (lose event), its states moves toward the center, shifting to the other side if it is already in the center.

Please cite this article in press as: A.R. Pinto et al., An approach to implement data fusion techniques in wireless sensor networks using genetic machine learning algorithms, Informat. Fusion (2013), http://dx.doi.org/10.1016/j.inffus.2013.05.003

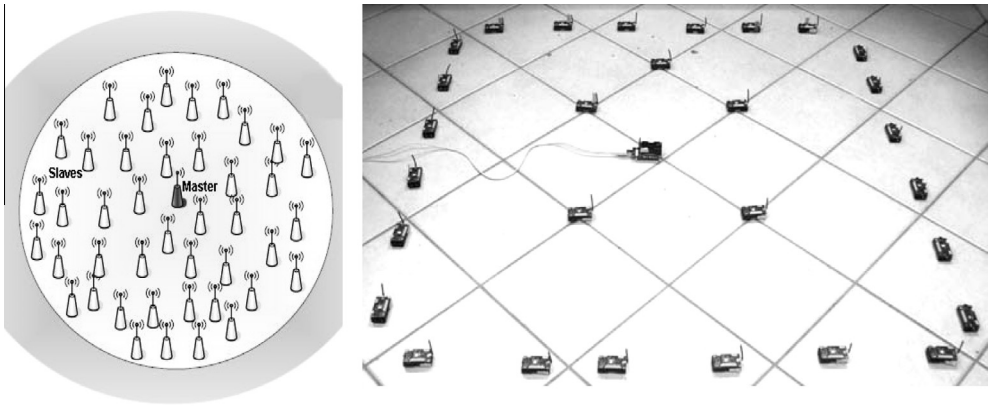

Fig. 4. System Architecture and used testbed.

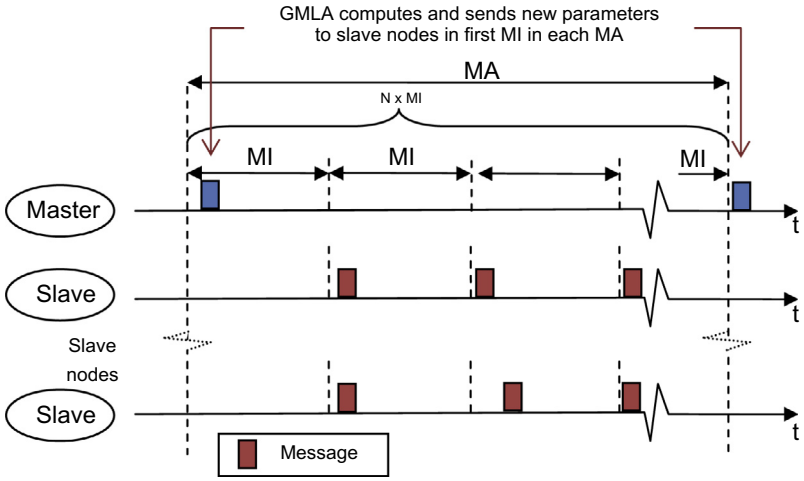

Fig. 5. A monitoring session composed of MA and MI.

task. This absolute deadline is computed based on a relative deadline d. We consider an homogeneous architecture where all slave nodes have the same relative deadline. The absolute deadline of a slave node at a microcycle n is $D = nR + d$, where R is the microcycle length.

The master node performs a data fusion operation considering just the messages that arrived on time. In this work, the master node just fuses data arrived within the same microcycle. Therefore, the relative deadline of a message sent in microcycle n is always $0 < d < R$, and consequently, the absolute deadline is $nR < D < (n + 1)R$.

A sending probability (*SP*) parameter is considered in the model. During a *MA* all slave nodes use the same *SP* value. This parameter is sent by master node to the slave nodes within each microcycle. For instance, if *SP* is configured to 0.1, in each microcycle, a slave will send its message with a 10% probability. As the physical phenomenon that is being sensed is considered to be homogeneous in the monitoring area, a well-configured *SP* parameter will save network energy, reducing the number of messages in the WSN.

The microcycle length and the relative deadline parameters are assumed to be known by all nodes in the network. The sending probability is sent by the master node in the first microcycle of each *MA*. This parameter may remain valid during the monitoring session, or can be changed at the beginning of each macrocycle based on the network behavior. Slave nodes do not send messages in this first microcycle; they just receive new parameter values.

During the macrocycle, the master node evaluates performance metrics, in order to tune the WSN. In the proposed model, two metrics are considered: Quality of Fusion (*QoF*) and Efficiency (*Ef*). *Ef* is the ratio between timely received messages (messages received by the master node within the deadline) and the number of messages sent. It is calculated according to Eq. (2):

$$Ef = \frac{\sum_{i=1}^{N} Mr_i}{E_{Ms}} \tag{2}$$

where N is the number of microcycles since previous macrocycle, Mr is the number of received messages and E_{Ms} is the estimated number of messages sent by slave nodes (3) in the current macrocycle. The latter estimates the number of messages that are supposed to be used by the data fusion task:

$$E_{Ms} = SP * S * N \tag{3}$$

where S is the number of slave nodes in the WSN deployment. Finally, *QoF* is the average number of messages received by the master node during a monitoring session, which is evaluated according to Eq. (4):

$$QoF = \frac{\sum_{i=1}^{N} Mr_i}{N} \tag{4}$$

The underlying idea of the *QoF* metric is to represent the quality of information presented to the data fusion task. A higher number of messages used by data fusion task results in more reliable information.

Please cite this article in press as: A.R. Pinto et al., An approach to implement data fusion techniques in wireless sensor networks using genetic machine learning algorithms, Informat. Fusion (2013), http://dx.doi.org/10.1016/j.inffus.2013.05.003

3.2. Related works

The proposed model has its roots in research works previously published by the authors, in other related works and in wireless network standards. The adopted star topology is part of IEEE 802.15.4 standard as one of two alternatives for network organization (the another one is peer-to-peer) [6]. Approaches proposed in [15–17], for example, also use star topologies, where sensor nodes are able to reach the base station in just one communication hop.

The use of dense WSN, composed of multiple sensor nodes and a base station in a star network topology, presents some difficult challenges with conflicting goals that must be solved (for instance, how to increase dependability and energy efficiency, while meeting time constraints). A typical scenario for such conflicting goals is increasing the number of nodes in a network to achieve better reliability, but fewer messages arrive on time to the base station due to WSN congestion. The difficulty in dealing with this problem was previously discussed in [2], which proposed a generic management architecture that take into account specific characteristics of WSNs. In addition, in a more recent work [9], the occurrence of this misbehavior has been highlighted, considering the same metrics used in this present paper: *Ef* and *QoF*. The results show that whenever the number of nodes in the network is increased, the *QoF* stops growing while the communication efficiency decreases rapidly.

A metric called *QoD* (Data Base Freshness), related to the proposed *QoF* concept, was presented in [18]. Real-time databases deal with periodic sensor updates and need to execute transactions within their deadlines using fresh data. However it is a challenge to respect simultaneously both parameters (timeliness and data freshness). To address this problem, the authors propose a new management architecture and introduced flexible *QoD* management schemes.

A similar microcycle concept was used in [19], where it was called round. The main goal of this concept is to discretize the time intervals at which decisions are made. In that work, the monitoring phase (that is equivalent to our macrocycle concept) was divided in equal duration rounds.

Serial data fusion based on a likelihood ratio test is presented in [13]. That paper uses routing techniques combined with collaborative detection, in order to decrease energy consumption and to increase detection dependability. Probability of False Alarm (PFA) and Probability of Detection (PDET) were assumed in order to achieve a desirable number of samples for fusion. The main idea is to use as fewer samples as possible, where a minimum level of quality is guaranteed by PDET and PFA.

A tree-based scheme is considered in [20]. The main objective is synchronizing multiple levels of data fusion. In that work, each node has to decide about when to begin the fusion process and how long to wait before the end of fusion. A redundant data transmission protocol was employed in [15]. The protocol is used to reduce the loss ratio of sensed data that is sent to an actuator node. A star topology is used, and sensor nodes make a decision whether to send or not data. Therefore, energy consumption and message losses are reduced. A parallel data fusion scenario was assumed in [17], where the master node is not aware about the number of sensor nodes. The data fusion rule (referred as counting rule) imposes that the number of packets must reach a pre-defined threshold in order to make a decision.

Chiuso and Schenato have proposed estimators for parallel data fusion strategies in wireless network environments subject to noise and packet losses [5]. Four strategies for data fusion were discussed, and two of them – *Measurement Fusion* (MF) and *Open-Loop Partial Estimate Fusion* (OPEF) – present a behavior suitable to be integrated with our proposed approach. Both estimators are based on the idea that the sensor nodes send their local estimates to a Fusion Center, which estimates the value of a given variable monitored in accordance with the measures that reach this Center. Within this context, it is not necessary to receive message transmissions from all nodes in each communication period. Fig. 6 illustrates the value in *n*th row and *t*th column that is a node information received in Fusion Center in the time instant *t*.

Q-DAP is a *QoS* data aggregation technique and processing approach that is executed at the intermediated nodes of a cluster-tree network [21]. Thus, the energy efficiency and network lifetime are increased while end-to-end latency and data loss are decreased. The main effort in Q-DAP is to determine when and where the data aggregation must be executed based in local information. The main drawback of this approach is that it considers a static cluster tree topology with predetermined routes. Therefore scalability and reliability issues may hinder its inherent quality.

There are also works whose primary goal is to achieve self-star properties in wireless networks [22–25]. In [22], for example, requirements to implement self-configuration and self-optimization are briefly discussed. Biologically-inspired techniques and algorithms [23,24] are often used to achieve autonomic properties. In [23] it is described a three-year project, called BISON, that deals with self-organization properties in highly dynamic networks. BISNET [24] is a middleware for WSN based on bee colonies organization. It implements several agent behaviors like: pheromone emission, replication, migration and death. For instance, when some agent detects an event (e.g. temperature change) it will emit a pheromone in order to alert its neighbors about the environmental change. When the pheromone level is above some threshold, the agent that monitors that scalar is replicated. This way, the scalar is monitored by other nodes. Migration is used to transmit data to base station and death occurs when the level of some scalar is very low. The similarity of these works with our proposed approach is the use of biological principles to deal with highly dynamic networks and achieve autonomic features. For example, while BISNET is based on the idea of pheromone and colony of ants to the development of software agents, our approach adopts genetic algorithm techniques.

4. GMLA approach

The triggering idea for the GMLA proposal is that the use of dense networks has conflicting goals that must be solved. For example, increasing the number of nodes in the network in order to improve reliability can significantly increase the network load, incurring in fewer messages that reach the destination. To highlight this characteristic we reproduced some previous experiments reported in [9]. We used a conventional periodic model in a beaconless IEEE 802.15.4 network configured as a star topology with period equal to 1 s and default CSMA/CA parameters. In this communication pattern (that hereafter is referred as "IEEE 802.15.4"),

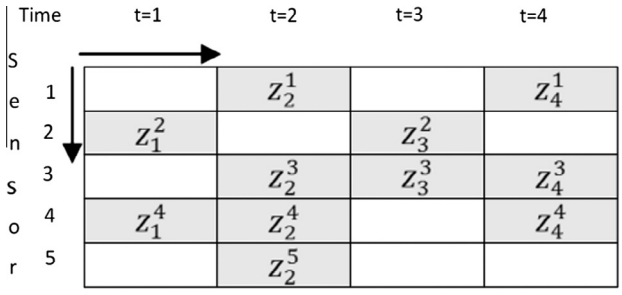

Fig. 6. Data fusion operation in networks subject to packet losses e discards. At time $t = 4$ only messages from nodes 1, 3 and 4 reached the Fusion center. *Source*: [5].

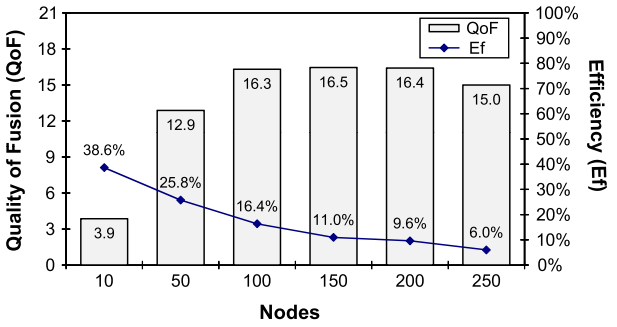

Fig. 7. IEEE 802.15.4 Efficiency and QoF behavior.

all nodes wait for the beginning of the period and then try to start their transmissions.

Fig. 7 illustrates the results from this scenario. It is possible to observe that increasing the number of nodes (from 100 to 200 nodes), the number of messages that reach the destination (*QoF*) is almost constant, due to the sharp drop in efficiency (*Ef*). This is an expected result because IEEE 802.15.4 is not an appropriate protocol for dense network [9].

The main target of the GMLA is to perform a trade-off between *QoF* and *Ef* by adjusting dynamically the sending probability (*SP*). The classifier algorithm presented in Algorithm 1 was configured as shown in Table 2. The proposed classifier is composed of four parts: C1, C2, A1 and A2. The form of a classifier is <*C*1 + *C*2>:<*A*1 + *A*2>, where C1 indicates if efficiency has increased or decreased since the last checkpoint, and C2 indicates the efficiency gain level. A1 value indicates if *SP* will be increased or decreased, and A2 indicates the level of change in *SP*.

Algorithm 2. GMLA Algorithm.

```
1:   In: Evolution_interval, LastEf, LastClassifier,
     CurrentSendProbability, EstabilizationPhase;
2:   Out: NewSP;
3:   Ef = calculateEfficiency ();
4:   if (Ef > BestEf) then
5:     BestSP = LastSP;
6:   end if
7:   deltaEf = ((LastEf/Ef) − 1) ∗ 100;
8:   if(deltaEf > 0) then
9:     receiveBankReward (LastClassifier);
10:  end if
11:  CurrentSP = consultClassifierSystem (deltaEf);
12:  payTaxToBank (currentClassifier);
13:  if (consult == Evolution_interval) then
14:    repeat
15:      chooseParents (between_the_most_adapted);
16:      crossover ();   mutate ();
17:      generateNewClassifiers (budget);
18:      Reposition_Rate = Reposition_Rate − 1;
19:    until (Reposition_Rate > 0)
20:  end if
21:  if (ExpertPhase) then
22:    NewSP = BestSP;
23:  else
24:    NewSP = CurrentSP;
25:  end if
```

The variation in the efficiency (efficiency delta), which is used to obtain the C1 and C2 values, is evaluated as follows:

$$\Delta Ef_i = \left(\frac{Ef_{i-1}}{Ef_i} - 1\right) * 100 \quad (5)$$

The GMLA is shown in Algorithm 2. The classifier initialization generates 64 classifiers, and each condition is represented by four classifiers (each condition has four different actions in the population classifier). The action part (A1 + A2) is randomly generated. The evolution function (selection, crossover and mutation) is executed just considering the action part (lines 15–16). In this way the condition part of the classifiers is preserved which ensures that all the conditions of the system will be satisfied in every classifier consult (line 11). During the evolution, new classifiers are generated and the new classifiers with the lower budget will be replaced in the population of classifier (line 17). The parent selection is performed in an elitist way (the best classifiers are selected) and mutation probability was set to 0.01. When a classifier is selected, it will pay 10% of its budget to the bank (line 12). Each classifier that improves the system performance will receive a reward from the bank (lines 3–10).

The overhead imposed by GMLA is much smaller than the overhead of traditional GAs. However, the evolution requires a longer system execution time. The key aspect of GMLA is that the evolution is done during the execution time, after a predetermined number of consults that were performed (lines 13–20 in Algorithm 2), whereas a traditional GA evolutes the candidate solutions before executing them. This is one of the reasons why we consider GMLA-based solutions to be more suitable for dynamic systems, such as WSN applications.

As GMLA is based on a classical classifier system, it could maximize *Ef* but the level of *QoF* would not be an optimization target. Thus, in the proposed GMLA, the Genetic Machine Learning Algorithm acts in two consecutive phases (lines 21–25 in Algorithm 2):

1. **First Phase (Learning Phase)**: in the first phase the main goal is to optimize *Ef* based on classifier systems. Algorithm 2 is used and the best results are stored (*SPs* that could achieve a certain *QoF* threshold and that presented the best *Ef* levels). This phase is considered to be finished after a pre-defined period of time and a specific number of best results is discovered. Each *SP* level receives a score when it is stored, that will be used in the second phase.
2. **Second Phase (Expert Phase)**: The second phase is dictated by the *SP* values stored during the Learning Phase. The *SP* values that were discovered during the first phase will continue to be used until they get useless. *SP* values receive a score whenever they are selected. This score is increased by one, each time they achieve the *QoF* target, otherwise it is decreased. When a certain *SP* score is zero that *SP* value is discarded, and the next one will be used. When all the *SP* are discarded, GMLA returns again to the Learning Phase.

5. Experimental assessment and simulation results

In this work, the assessment of the GMLA approach was done both by simulation and experimental assessment with real motes.

Simulations present the advantage of greater flexibility when dealing with complex scenarios such as hundreds or thousands of nodes with mobility features. On the other hand, simulation might not take into account all factors that may occur in a real environment. Therefore, we also performed a set of measurements on a real WSN. The purpose of the experimental assessment was twofold: (i) to validate simulation results in a real environment, and (ii) to show that the envisaged approach is viable to be used on real scenarios.

Please cite this article in press as: A.R. Pinto et al., An approach to implement data fusion techniques in wireless sensor networks using genetic machine learning algorithms, Informat. Fusion (2013), http://dx.doi.org/10.1016/j.inffus.2013.05.003

Table 2
Classifiers configuration.

Field	Size	Description	Codification			
Cl	1 bit	Indicates variation of `Ef`	0 = decreased		1 = increased	
C2	3 bits	Indicates variation of `Ef` level in last Macrocycle	000 = [0%; 12%] 100 = (48%; 64%]	001 = (12%; 24%] 101 = (64%; 72%]	010 = (24%; 36%] 110 = (72%; 36%]	011 = (36%; 48%] 111 = (84%; 100%]
Al	l bit	Indicates the increase or decrease of `Sp` on next Macrocycle	0 = to decrease		1 = to increase	
A2	3 bits	Indicates the level of change in `Sp` on next Macrocycle	000 = 12% 100 = 64%	001 = 24% 101 = 72%	010 = 36% 110 = 84%	011 = 48% 111 = 100%

5.1. Experimental assessment

The GMLA behavior was initially assessed using MicaZ nodes. The experimental setup was composed of 30 nodes, featuring an Atmel ATmega128L 8-bit microcontroller with 128 kB of in-system programmable memory and IEEE 802.15.4 support. TinyOS was used as the development platform. The deployment area was a 1.3 × 1.3 m square (Fig. 4).

The message sent by master node in each *MA* has 19 bytes, and messages sent by slave nodes have 18 bytes (Fig. 5). The memory footprint is the following: for the Master Algorithm 572 bytes (RAM memory) and 15,902 bytes (ROM memory); for the Slave Algorithm 333 bytes (RAM memory) and 12,088 bytes (ROM memory). IEEE 802.15.4 was used in beaconless mode, channel 16 (2.48 GHz), with a macrocycle of 1s and microcycle of 0.1 s.

The need for two phases when running GMLA is highlighted in Fig. 8, for an experimental assessment of 30 min, where the *Expert Phase* was turned off. In this experiment, the network started with 14 slaves (nodes) and after 15 min of operation, 15 new slaves were turned on. It is possible to notice three learning phases in the algorithm. During the 3–15 min period GMLA achieved 80% of *Ef*. When the new 15 nodes were turned on, a significative decrease in the *Ef* was noticed. This effect is emphasized during the period 16–21 min period, when the average of *Ef* was just 59%. During the 25–29 min period, and after another learning phase, the GMLA actuated and achieved 87% of average *Ef*.

The results emphasize the self-adaptation capability of GMLA, and it also highlights the need of an *Expert Phase*, as the efficiency is never settled due to the behavior of the GA, which always tries new *SP* values that can increase *Ef*.

The same experimental assessment was made using a complete GMLA (with two phases). The network started with 14 nodes and, after 7.5 min, 15 nodes were turned on (Fig. 9). After 5 min of learning phase, the network was already stabilized. Moreover, even when the density is increased, the network was already stabilized and it remains in this state until the end of the experiment. Therefore, it is possible to conclude that GMLA is able to trade-off *QoF* and *Ef* even when the environment has an evolving behavior.

Another experimental assessment of GMLA was made using as target *QoF* = 7. The results are presented in Fig. 10 for different number of slaves. The test duration was set to 15 min and each point in the *x* axis represents the average of 1 min of tests. During 1 min of tests, there are 60 MA (1 s each MA) and 600 MI (0.1 s each MI).

Regarding the GMLA assessment, higher levels of *Ef* were achieved when compared with the IEEE 802.15.4 case. The only exception occurs when the number of nodes is below or equal to 19. This behavior was due to the *QoF* target. In this case GMLA decreases *Ef* trying to achieve the desired *QoF*. The gain in *Ef* obtained by GMLA in comparison with IEEE 802.15.4 means that more messages can be of useful interest for the data fusion task, meaning that less energy is spent.

5.2. Simulation results

The GMLA behavior was also assessed by simulation using the OMNeT++ tool [26]. Several experiments were carried out to compare the results with both IEEE 802.15.4 and the Gur Game. For all simulated approaches, the IEEE 802.15.4 was assumed as the physical and MAC underlying protocol.

The parameters for the network, GMLA and Gur Game are shown in Table 3. Regarding the network parameters, this configuration will incur in a line-of-sight range of approximately 100 m. Regarding the GMLA:

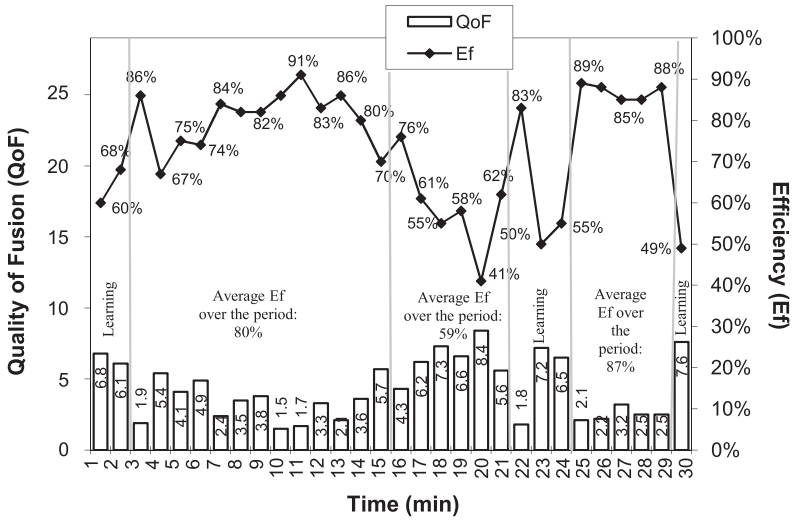

Fig. 8. GMLA behavior without Expert Phase.

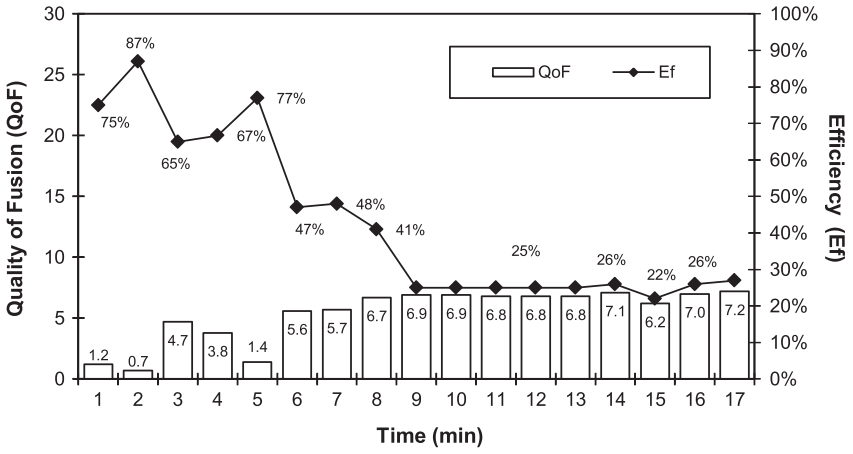

Fig. 9. GMLA started with 14 nodes and doubled the number during execution.

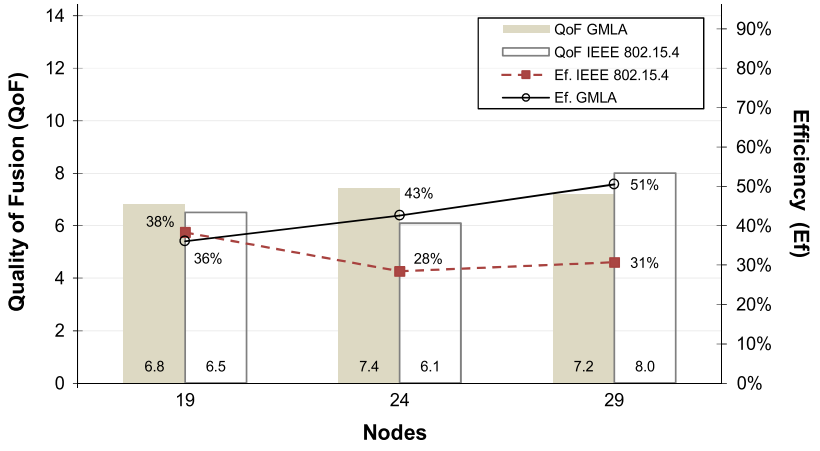

Fig. 10. GMLA efficiency and QoF with different densities.

Table 3
Network, GMLA and Gur Game parameters for OMNeT++.

Frequency	Data rate	Transmit power	Receiver signal threshold	Pathloss exponent	MAC retry
Network parameters					
2.4 GHz	250 Kbps	0dbm (1 m W)	−85 dbm	2	No
Checkpoint interval	Evolution interval	Replacement rate	Budget amount	Population size	Probability of mutation
GMLA parameters					
5 MI	15 MI	2	100 units	64	0.01
Number of states	Number of masters		Target	Reward function	
Gur game parameters					
6	1		10 messages	$f(x) = 20 + 80 \times e^{\frac{-0.2}{(nnodes/10)} \times (met-10)^2}$	

- *checkpoint interval* corresponds to the time interval between two classifier consults;
- *evolution interval* corresponds to the time interval between evolutions;
- *replacement rate* corresponds to the number of classifiers that will be replaced after the evolution;
- *probability of mutation* is a well-known parameter in the GA area that means the probability of mutation in the new classifier.

Regarding the Gur Game, the adopted reward function was the same represented by Eq. (1) assuming *target* = 10. There are six states in automaton (Fig. 2): three states associated with the action "send message" and other three states associated with the action "do not send message".

Two simulation scenarios were defined: one considering static nodes (Fig. 11a) and another with mobile nodes (Fig. 11b). In both scenarios the master node was placed at the center of the area and it sends and receives messages directly from slave nodes (single-hop assumption). The static scenario represents a square area of 100 × 100 m, which corresponds approximately to the network diameter formed by the radio range of master node with the adopted network parameters. Slave nodes are placed uniformly inside this region in fixed positions (i.e. no mobility). The number of

Please cite this article in press as: A.R. Pinto et al., An approach to implement data fusion techniques in wireless sensor networks using genetic machine learning algorithms, Informat. Fusion (2013), http://dx.doi.org/10.1016/j.inffus.2013.05.003

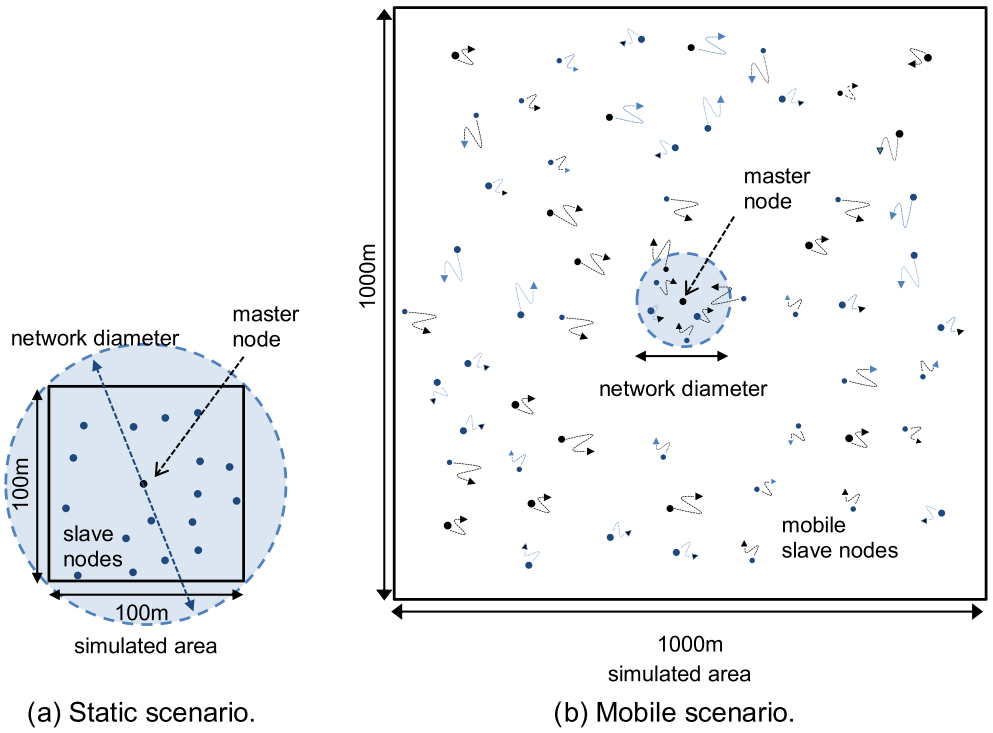

Fig. 11. The simulation scenarios with static and mobile nodes.

slave nodes was modified offline, assuming networks with 65, 125, 250, 500, 750, 1000, 1250 and 1500 nodes.

The mobile scenario represents a square region of 1000 × 1000 m. It is assumed that slave nodes can move within this region. We used the mobility model provided by OMNeT++ with the following parameters: *mobility update interval* = 100 ms, *speed* = 5 mps and *update interval* = 100 ms. The area used in this scenario is larger than the used in other static one because we wanted to simulate behaviors where the node density changes dynamically, with nodes leaving and entering the radio coverage area of the master node.

5.2.1. Static scenario

Fig. 12 highlights the behavior of GMLA, Gur Game and IEEE 802.15.4 with different number of nodes in static scenarios. A full assessment of these approaches cannot be done just taking into account the *Ef* metric, being also necessary to assess the achieved *QoF*. It can be noticed that GMLA have tried to maximize *Ef*, while *QoF* values were maintained at an acceptable level. However, GMLA does not provide good *QoF* values when the number of nodes in the network is lower than 250. In these situations, the Gur Game approach achieves higher levels of *QoF* than GMLA maintaining an acceptable trade-off between *QoF* and *Ef* (Gur Game had always obtained *Ef* values above or equal to the IEEE 802.15.4).

GMLA presents better results in highly dense networks. The timeline of GMLA efficiency is showed in Fig. 13 for a network with 1500 nodes. In GMLA approach, *SP* has a great influence on *Ef*. If the *SP* value is reduced, there will be a smaller number of messages being sent, and *Ef* would be maximized (smaller network load and therefore the packets collision would be reduced). However, if *SP* becomes too small, GMLA detects it and tries to increase it. It is possible to notice that during the first 30 MA (classifier

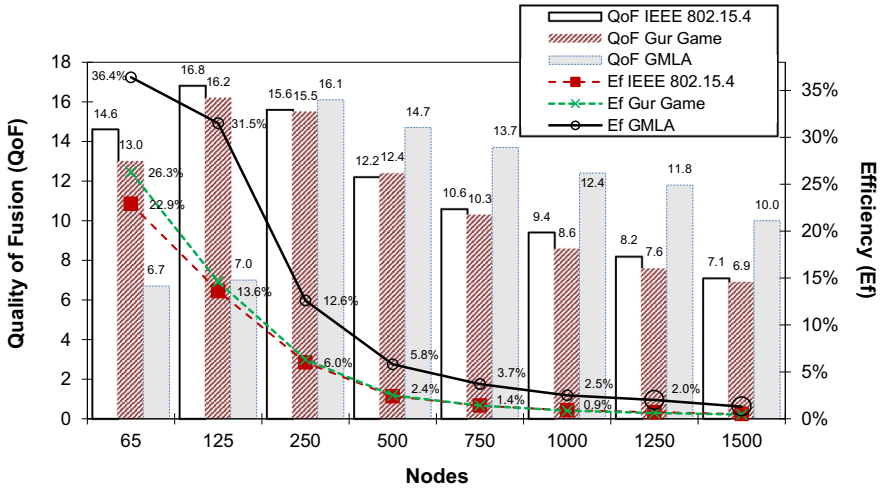

Fig. 12. GMLA, Gur Game and IEEE 802.15.4 efficiency and QoF in dense networks with static nodes.

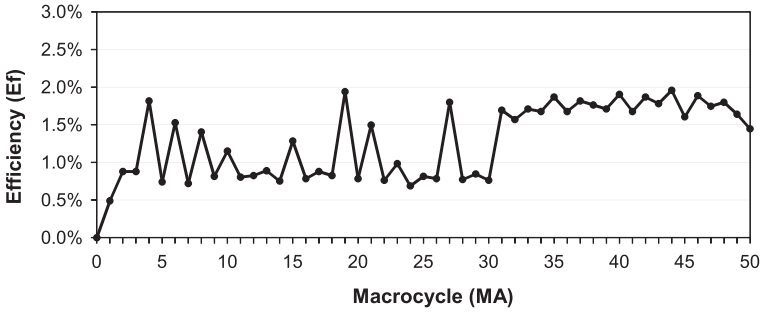

Fig. 13. GMLA efficiency along time in a network with 1500 nodes.

consults points 1–30) the efficiency shows a variation from 1% to 2%. This behavior is due to the *Learning Phase* of GMLA. After the 30 *MA* the *Expert Phase* starts and the efficiency was stabilized around 2%. This behavior indicates its adaptability when a dense WSN is considered.

5.2.2. Mobile scenario

The comparison between the efficiency of GMLA, Gur Game and IEEE 802.15.4 approaches is highlighted in Fig. 14. Each point in the graph indicates the average efficiency after 600 s of simulation. It is possible to notice that GMLA always achieves a higher efficiency than Gur Game and IEEE 802.15.4. The gain in efficiency is more than 100% in a network with 250 nodes. As GMLA tries to maximize *Ef*, it significantly increases the efficiency. This behavior is sustained even for high density scenarios when the efficiency is very low. As an example, with 1500 nodes GMLA achieves $Ef = 1.1\%$, while IEEE 802.15.4 achieves just $Ef = 0.5\%$. This behavior indicates that GMLA is clearly more suitable for dense and dynamic networks than Gur Game and IEEE 802.15.4.

6. Conclusion

WSN applications may require a large number of nodes to monitor an area due to the fact that when more nodes participate in the sensing task, the obtained information becomes more accurate. Besides, when a high number of nodes is sensing the same phenomenon, the low reliability of individual node is overcome through a cooperative effort. However, the IEEE 802.15.4 standard without addition of new mechanisms is not suitable for dense wireless networks because its backoff scheme incurs in a high number of message collisions and channel access failures, inducing a significant degradation of communication efficiency.

There are optimization techniques in the literature that improve the IEEE 802.15.4 MAC backoff algorithm and that can enhance the communication performance of dense wireless networks. However, the use of this kind of approach can just minimize the communication problem presented above, because when the density is increased, the same communication performance degradation will necessarily occur again.

In this context, approaches that autonomically adjust the percentage of sent messages are an interesting alternative to increase the network throughput and decrease the energy consumption. The main idea is to tune the nodes in a way that just the necessary number of messages is sent by them. In this way, this kind of technique can be considered very important for data fusion applications.

As aforementioned, there are currently research efforts in data fusion that consider WSN applications with message loss and packet drops, and which can give rise to new opportunities for the development of communication approaches that supports this innovative data fusion paradigm. These data fusion schemes evoke dynamic communication approaches where nodes can locally decide if they will send or not the collected data. The main advantage of this scheme is to achieve a trade off between several conflicting network metrics like: number of messages delivered to the base station, energy economy, data fusion quality and other.

In this paper we presented a genetic machine learning approach (GMLA) to optimize the communication efficiency in dense wireless sensor networks. The proposed approach is able to

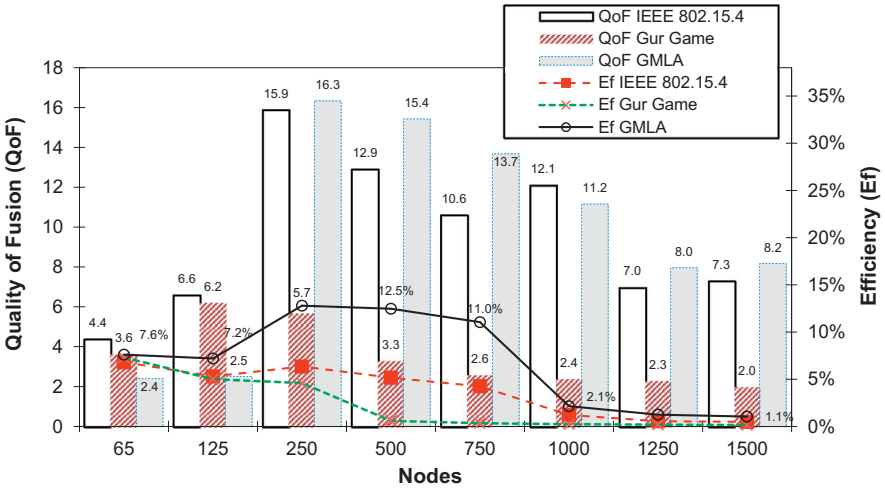

Fig. 14. GMLA, Gur Game and IEEE 802.15.4 efficiency in dense network with mobile nodes.

Please cite this article in press as: A.R. Pinto et al., An approach to implement data fusion techniques in wireless sensor networks using genetic machine learning algorithms, Informat. Fusion (2013), http://dx.doi.org/10.1016/j.inffus.2013.05.003

autonomically adjust the sending rate of a WSN with dynamic topologies (composed of static or mobile nodes), and presents a significant performance improvement when compared with a pure periodic approach based on IEEE 802.15.4 standard and also with the Gur Game approach. Moreover, the implementation feasibility of the GMLA in a real data fusion application was proved by the experimental testbed based on mote prototype.

As a final remark, we observe that the proposed approach was developed at the application layer; therefore it can be easily used in COTS (Component of the Shelf) devices, as a framework to support other well-known data fusion techniques. Further research directions include the utilization of different types of data fusion models (serial or hybrid) and the optimization of other GMLA parameters such as *checkpoint interval* and *evolution interval*.

Acknowledgements

The authors acknowledge the support received from national funding agencies (CAPES, CNPq, FAPESP and FCT) and also from their institutions (INCT-SEC 573963/2008-9 and 08/57870-9, Project PTOC/EEA-TEL/104185/2008 and IDMEC).

References

[1] J.A. Stankovic, C. Lu, L. Sha, T. Abdelzaher, J. Hou, Real-time communication and coordination in embedded sensor networks, Proceedings of the IEEE 91 (2003) 1002–1022.
[2] L.B. Ruiz, J.M. Nogueira, A.A.F. Loureiro, Manna: a management architecture for wireless sensor networks, IEEE Communications Magazine 41 (2003) 116–125.
[3] M.C. Huebscher, J.A. McCann, A survey of autonomic computing–degrees, models, and applications, ACM Computing Surveys 40 (2008) 1–28.
[4] E.F. Nakamura, A.A.F. Loureiro, A.C. Frery, Information fusion for wireless sensor networks: methods, models, and classifications, ACM Computing Surveys 39 (2007) 1–55.
[5] A. Chuso, L. Schenato, Information fusion strategies and performance bounds in packet-drop networks, Automatica 47 (2011) i1304–1316.
[6] Ieee standard for information technology – telecommunications and information exchange between systems-local and metropolitan area networks specific requirements part 15.4: Wireless medium access control (mac) and physical layer (phy) specifications for low-rate wireless personal area networks (wpans), IEEE Std 802.15.4-2011 (Revision of IEEE Std 802.15.4-2006), 2011, pp. 1–314.
[7] B. Tung, L. Kleinrock, Using finite state automata to produce self-optimization and self-control, IEEE Transactions on Parallel and Distributed Systems 7 (1996) 439–448.
[8] G. Anastasi, M. Conti, M. Di Francesco, A comprehensive analysis of the mac unreliability problem in ieee 802.15.4 wireless sensor networks, IEEE Transactions on Industrial Informatics 7 (2011) 52–65.
[9] A.R. Pinto, M. Camada, M. Dantas, C. Montez, P. Portugal, F. Vasques, Genetic machine learning algorithms in the optimization of communication efficiency in wireless sensor networks, in: 35th Annual Conference of IEEE Industrial Electronics, IECON '09, 2009, pp. 2448–2453.
[10] D.E. Goldberg, Genetic Algorithms in Search, Optimization and Machine Learning, 1st ed., Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1989.
[11] R.R. Brooks, S.S. Iyengar, Multi-sensor Fusion: Fundamentals and Applications with Software, Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1998.
[12] W. Elmenreich, Sensor Fusion in Time-Triggered Systems, 2002.
[13] S. Patil, S. Das, A. Nasipuri, Serial data fusion using space-filling curves in wireless sensor networks, in: 2004 First Annual IEEE Communications Society Conference on Sensor and Ad Hoc Communications and Networks, IEEE SECON 2004, 2004, pp. 182–190.
[14] R. Iyer, L. Kleinrock, QoS control for sensor networks, in: Proceedings of ICC'03, vol. 1, Anchorage, Alaska, pp. 517–521.
[15] K. Morita, K. Watanabe, N. Hayashibara, T. Enokido, M. Takanizawa, Efficient data transmission in a lossy and resource limited wireless sensor-actuator network, in: 10th IEEE International Symposium on Object and Component-Oriented Real-Time Distributed Computing, ISORC '07, 2007, pp. 156–163.
[16] A. Somasundara, A. Rammorthy, M. Srivastava, Mobile element scheduling with dynamic deadlines, IEEE Transactions on Mobile Computing 6 (2007) 395–410.
[17] R. Niu, P.K. Varshney, Q. Cheng, Distributed detection in a large wireless sensor network, Information Fusion 7 (2006) 380–394. Special Issue on the Seventh International Conference on Information Fusion–Part I, Seventh International Conference on Information Fusion.
[18] J.S.K.-D. Kang, S.H. Son, Managing deadline miss ratio and sensor data freshness in real-time databases, IEEE Transactions on Knowledge and Data Engineering 16 (2004) 1200–1216.
[19] J.S.T. Yan, T. He, Differentiated surveillance for sensor networks, in: Proceedings of the 1st international conference on Embedded networked sensor systems, SenSys'03, ACM, New York, NY, USA, 2003, pp. 51–62.
[20] W. Yuan, S. Krishnamurthy, S. Tripathi, Synchronization of multiple levels of data fusion in wireless sensor networks, in: Global Telecommunications Conference, GLOBECOM '03, IEEE, vol. 1, 2003, pp. 221–225.
[21] J. Zhu, S. Papavassiliou, J. Yang, Adaptative localized qos-constrained data aggregation and processing in distributed sensor networks, IEEE Transactions on Parallel and Distributed Systems 17 (2006) 923–933.
[22] C. Shen, D. Pesch, J. Irvine, A framework for self-management of hybrid wireless networks using autonomic computing principles, in: Proceedings of the 3rd Annual Communication Networks and Services Research Conference, 2005, pp. 261–266.
[23] A. Montresor, O. Babaoglu, The BISON project, IEEE Computational Intelligence Bulletin 1 (2002) 6–9.
[24] P. Boonma, J. Suzuki, Bisnet: a biologically-inspired middleware architecture for self-managing wireless sensor networks, Computer Networks 51 (2007) 4599–4616.
[25] A.R. da Rocha, F.C. Delicato, J.N. de Souza, D.G. Gomes, L. Pirmez, A semantic middleware for autonomic wireless sensor networks, in: Proceedings of the 2009 Workshop on Middleware for Ubiquitous and Pervasive Systems, WMUPS '09, pp. 19–25.
[26] A. Varga, Using the omnet++ discrete event simulation system in education, IEEE Transactions on Education 42 (1999) 11.

Please cite this article in press as: A.R. Pinto et al., An approach to implement data fusion techniques in wireless sensor networks using genetic machine learning algorithms, Informat. Fusion (2013), http://dx.doi.org/10.1016/j.inffus.2013.05.003