UNIVERSITÀ
DEGLI STUDI
DI PALERMO

NDS
NETWORKING AND DISTRIBUTED SYSTEMS
RESEARCH GROUP

A Symbolic Distributed Event Detection Scheme for Wireless Sensor Networks.

Article

Accepted version

S. Gaglio, G. Lo Re, G. Martorella, D. Peri

In Proceedings of the International Conference on Emerging Technologies and Factory Automation (ETFA'16)

A Symbolic Distributed Event Detection Scheme for Wireless Sensor Networks

Salvatore Gaglio*†, Giuseppe Lo Re*, Gloria Martorella* and Daniele Peri*
*DICGIM - Università degli Studi di Palermo, Viale delle Scienze, Ed. 6, 90128 Palermo, Italy
{salvatore.gaglio, giuseppe.lore, gloria.martorella, daniele.peri}@unipa.it
†ICAR-CNR, Viale delle Scienze, Ed. 11, 90128 Palermo, Italy

***Abstract*—Due to the possibility of extensive and pervasive deployment of many tiny sensor devices in the area of interest, Wireless Sensor Networks (WSNs) result particularly suitable to detect significant events and to react accordingly in industrial and home scenarios. In this context, fuzzy inference systems for event detection in WSNs have proved to be accurate enough in treating imprecise sensory readings to decrease the number of false alarms. Besides reacting to event occurrences, the whole network may infer more information to enrich the event semantics resulting from reasoning processes carried out on the individual nodes. Contextual knowledge, including spatial and temporal relationships, as well as neighborhood confidence levels, can be used to improve the detection accuracy, but requires to extend the number of variables involved in the reasoning process.**

In this paper, we propose a distributed scheme for event detection in WSNs that combines fuzzy reasoning and executable code exchange to infer contextual information. The proposed scheme lets resource constrained interconnected devices perform on-board reasoning on numerical and symbolic data, and exchange additional symbolic knowledge –in the form of data, rules and executable code– to enrich the event semantics.

In the presented case study the nodes can determine the number of neighbor nodes, their spatial proximity to the position where an event occurred, or classify themselves into clusters according to their final decisions. Differently from mainstream approaches, node cooperation is carried out through an effective and inexpensive executable code exchange approach.

I. Introduction

Event detection is of paramount importance in Wireless Sensor Network (WSN) applications ranging from structural health monitoring [1], [2] and fault detection in industrial scenarios [3] to environmental disaster detection [4], [5]. A simple event detector system can be realized with a very simple scheme that sees nodes report data once sensory readings exceed a fixed threshold. Although it takes just little effort, this approach is fairly inaccurate due to sensory reading fluctuations and imprecisions. In more demanding applications, accurate event detectors running on WSNs are required to embed advanced skills that goes beyond mere thresholding, such as symbolic reasoning and prediction. Moreover, even resource-constrained sensor nodes, if endowed with such abilities, may become an enabling technology for the Internet of Things (IoT) [6]. In order to overcome the inaccuracy of crisp logic, machine learning techniques have been proposed that perform event detection on-board individual nodes [7], [8]. However, to exploit the inner nature of a network, distributed approaches for event detection in WSNs have been also proposed that include collaboration and data exchange into the decision process, adopting, for instance, machine learning techniques [9] and distributed fuzzy inference systems [10]. However, many effective machine learning algorithms require considerable amount of resources and also need prior acquisition of massive training data for the classifier initialization [11]. Fuzzy inference systems, instead, can be implemented on nodes with limited resource consumption through a natural description of the domain of interest but without requiring any prior observations. Furthermore, they have proved to achieve good results in event detection once the local decision undertaken by a given node depends both on local observations and on fuzzy linguistic labels provided by the neighborhood [12]. Context information can be included to enrich the event semantics at the price of an increase of memory footprint due to the rule base extension, thus processing methods to reduce the complexity of fuzzy rule sets are highly required [13].

In this paper we present a distributed scheme for event detection running on our software platform for WSN nodes [14] based on an extension of a distributed fuzzy inference system [12]. The proposed scheme lets resource constrained interconnected devices perform on-board reasoning on numerical and symbolic data, and exchange additional symbolic knowledge –in the form of data, rules and executable code– to enrich the event semantics. For instance, the nodes detecting an event may start exchanging symbolic code to infer contextual information in order to build a shared representation of their spatial proximity to the place the event occurred. Nodes may also infer their spatial distribution on the basis of the similarity of their own sensed data to readings of their neighbors. Finally, nodes may exploit the distributed processing approach to build a representation about how events evolve in space and time.

The rest of the paper is organized as follows: Section II introduces a brief overview about the key features of our target hardware and software platform, while Section III discusses the proposed scheme for event detection in WSNs and provides implementation details. Finally, Section IV concludes the paper.

II. A Distributed Symbolic Processing Environment

In order to provide WSNs with distributed computing and dynamic reprogramming many efforts have been put into

978-1-5090-1314-2/16/$31.00 © 2016 IEEE

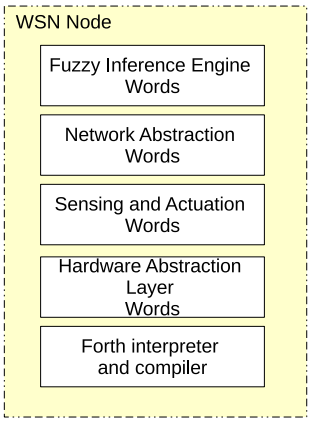

Fig. 1: Layered overview of our word-based operating environment

implementing application specific virtual machines (ASVM), which provides an interpreter on top of an available general purpose operating system. In the proposed approach, instead, a software platform integrates a stack based interactive interpreter and compiler implementing the Forth methodology at its lowest level [15].

In our programming environment, coding consists in implementing symbols, the so-called *words*. Abstract and expressive applications are then implemented as sequences of high-level words.

We have so far developed a set of words to provide support for typical WSAN functionalities [15], as well as for hardware and networking abstractions. A layered overview of our word-based operating environment is shown in Figure 1. Proceeding from the bottom upwards, words are more abstract and hardware independent than those belonging to lower levels.

However, such a word set is extensible as user-defined symbols can be added easily due to the presence of the low-level compiler. Interactions among already deployed nodes is based on the transmission of symbolic executable code implementing the distributed symbolic applications. Communication is implemented at level 2 of the ISO-OSI model. A node executes the words `tell:` and `:tell` to create and send messages compliant to the 802.15.4-2003 standard. The executable code to be remotely sent is enclosed among these two words as such: `tell:` <code> `:tell`, or `[tell:]` <code> `[:tell]` when the construct is inside a word definition. The message payload is the sequence of words that are to be interpreted by the remote destination node.

To incorporate reasoning abilities on-board, e.g. for event detection purposes, the topmost layer consists in the symbolic abstraction to implement fuzzy inference on individual nodes [16].

As an example, provided that all the nodes are equally equipped with temperature and smoke sensors, Listing 1 shows the sequence of words that can be used to define fuzzy variables and membership functions for a very common WSN event detection application, i.e. indoor distributed fire detection, as in [12].

Listing 1: Sequence of words to define input fuzzy variables and membership functions for indoor distributed fire detection

```
1  -50 100 fvar temp
2  -50 -50 0 50 member temp.low
3  -50 0 50 100 member temp.high
4
5  -50 100 fvar smoke
6  -50 -50 0 50 member smoke.low
7  -50 0 50 100 member smoke.high
8
9  -20 20 fvar dtemp
10 -20 -20 -10 10 dtemp.low
11 -20 -10 10 20 dtemp.high
12
13 -20 20 fvar dsmoke
14 -20 -20 -10 10 dsmoke.low
15 -20 -10 10 20 dsmoke.high
16
17 -50 100 fvar temp-neighbor
18 -50 -50 0 50 member temp-neighbor.low
19 -50 0 50 100 member temp-neighbor.
       high
20
21 -50 100 fvar smoke-neighbor
22 -50 -50 0 50 member smoke-neighbor.
       low
23 -50 0 50 100 member smoke-neighbor.
       high
```

The word `fvar` defines a new fuzzy variable according to the syntax:

<*min_val*> <*max_val*> `fvar` <*name*>

where *min_val* and *max_val* compose the domain of the crisp input variable. Similarly, the creation of a new membership function requires the execution of the word `member`. The word is preceeded by four control points that determine its shape.

III. Case Study: A Symbolic Distributed Scheme for Event Detection

Event detection systems are often designed to support other tasks or for running concurrently with other applications. Therefore a proper tradeoff between lightweight deployments and complexity is desirable.

In this regard, to enrich event descriptions with additional information, which can be intended for the event detection application but also for other purposes, it is often required to increase the number of variables involved in the fuzzy inference process.

Obviously, this could lead to significant memory consumption. Therefore, the inferential process either undertaken by the nodes using a restricted set of rules, or it takes place on a more powerful node, e.g. a gateway, which has no constraints in terms of memory and computational resources.

To enrich the event semantics with context information, the scheme we propose exploits the communication among nodes and especially the distributed computing paradigm through the exchange of executable code, instead. Existing event detection schemes confine node interactions to include the

neighborhood opinion in the decision making process, which is performed locally. The basis of the proposed distributed scheme is to exploit communication to make the nodes infer further knowledge and reach a shared representation about the target event in a distributed manner. The proposed scheme consists of two phases. In the first phase, named the *decisional stage*, the node assesses the event occurrence on the base of the sensed physical quantities and the network opinion, and possibly reports the event to the networked actuator or to a gateway.

Let us suppose that the node identified by ID 1 is executing the *decisional stage*, as shown in Figure 2. To this end, it fuzzifies its temperature and smoke readings and broadcasts the code to start a timer with random timeout on remote neighbor nodes along with the word to be executed on remote timer expiration. Then, it waits for neighbor opinions until a predefined amount of time has elapsed. Once the random timer on receiver nodes expires, neighbors execute the word `opinion-reply`, which makes them reply with the symbolic code to let the sender increment the number of nodes and a partial counter, `fcount`, by the truth value associated to each memebership function. Finally, once the maximum amount of time needed to collect neighbor opinions elapses, the node executes the word `decision` whose definition is provided in Listing 2. As a node executes this word, it stops the timer associated to nodes reply and assesses the opinion received by neighbors. This implies to compute the cardinality of each fuzzy set and to apply the fuzzy *most* operator [12]. Then `rules` are executed and finally, defuzzification is performed by the word `conclude`, which is applied to the fuzzy output variable `fire`.

Listing 2: Code for the decisional stage, which is executed by each node after having collected all neighbor opinions

```
1 : decision
2         waiting timer stop
3         opinions assess  rules
4         fire conclude ;
```

Actually, this is a pseudo-distributed process in the sense that each node is influenced by the neighbor opinions but there is no actual distributed processing. The *context inference stage* complements the previous phase and allows for the extraction of further context knowledge about the event. Such a phase is triggered by specific nodes, e.g. nodes placed strategically, detecting an event as a result of the *decisional stage*.

As shown in Figure 3, the initiator node broadcasts the executable code consisting in a simple *if-then* rule that is interpreted by all receivers in its radio range. Nodes that have not detected the event discard the message, while the others execute the *if* part of the rule. The initiator sends the raw measured values along with the code to store these values in a neighboring table. In addition, the initiator activates a random timer on remote nodes to start the distributed computing scheme. When this timer expires, the receiver nodes in turn broadcast their data along with the code for storing them in

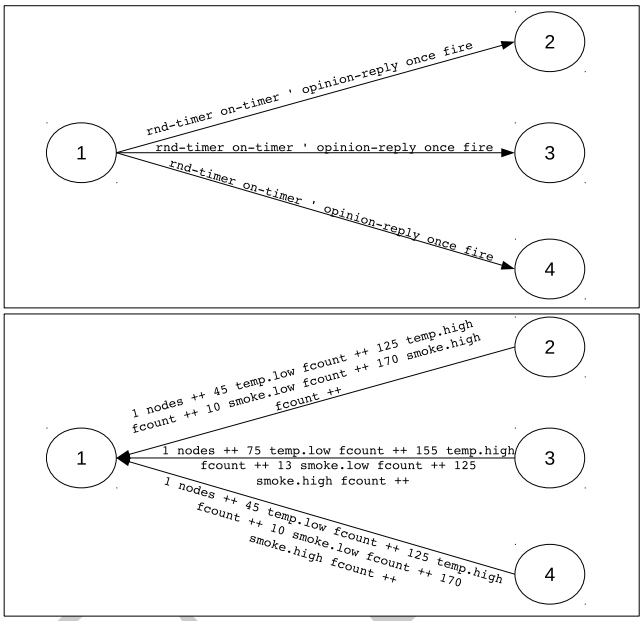

Fig. 2: *Decisional stage*. The topmost figure shows the code to start collecting neighbor opinions, while the figure on the bottom shows the code sent by remote nodes.

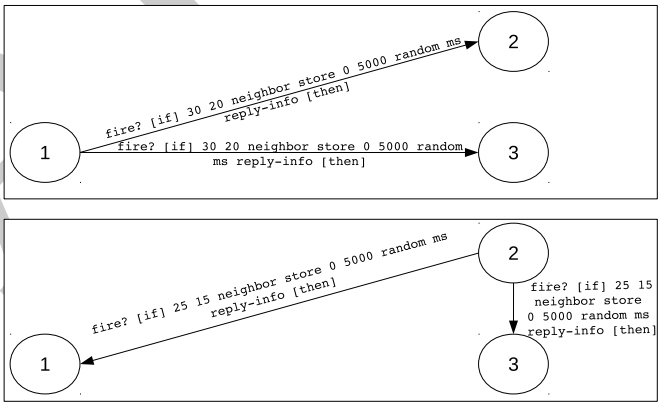

Fig. 3: *Context inference stage*. Each node, in turn, sends the code to store the neighbor and its sensed value in the neighbor table. For the sake of simplicity, the process is shown for nodes 1 and 2.

the table. A table entry stores the address and the data received from each neighbor.

At the end of this process, when all nodes have transmitted once, they hold additional shared information concerning the number of nodes that lay in the same area –i.e. nodes belonging to the same cluster determined on the basis of the output decision.

This knowledge can be used for a variety of purposes that depend both on the type of the event that the network must recognize and on the context information to be inferred to augment the event description.

For instance, in the case of fire detection, the nodes may sort the table with decreasing values of temperature. This allows to extract context information such as the node distribution as

well as the relative position of the neighbors from the fire, and consequently infer also their own.

Other context information that could be useful when an event occurs is the spatial proximity between nodes. To this purpose, each node sorts the table according to the similarity degree of the sensory readings received by neighbors to its own, thus building and holding its own representation of the current spatial distribution of all the nodes. This mechanism is suitable for diagnostic inference. In fact, if a node reports that the event occurred but its readings are very different from those of the neighbors, then it could infer that its decision is wrong or that its own sensors are defective.

In the proposed method, the degree of confidence of the neighbors can be also determined to make the *decisional stage* dynamic by modifying the weight associated to each neighbor opinion.

At the beginning, the opinion of the neighbors are equally weighted. However, at the end of the *context inference stage*, a node may assign a higher confidence level to those neighbors that are in proximity or to those with a high degree of similarity in the transmitted readings. The initiator node may then sends to all nodes the code to execute depending on the information to be extracted.

Our approach makes event detection applications developable on real devices through an environment running directly on the target hardware without cross-compilation, and reflashing phases. Furthermore, distributed computation is inexpensive due to the fact that executable code exchange consists in a low level implementation.

The fuzzy inference implementation requires only 6 bytes of RAM and 863 bytes of Flash memory and consists of 31 words. Finally, the word set implementing the distributed scheme for the inclusion of context inference abilities consists of 17 words and occupies less than 300 bytes of RAM and less than 800 bytes of Flash.

IV. Conclusions

In this paper we presented an approach to implement distributed schemes for event detection on WSNs.

While existing schemes for distributed event detection partially exploit interactions among nodes, as context information requires to extend the fuzzy rule set, we showed how it is possible to infer shared context information, without providing it in advance, by exploiting the exchange of symbolic executable code.

The proposed scheme runs on our software platform for WSNs –currently targeting IRIS motes– that has been designed to support distributed symbolic computation on resource-constrained devices.

Further work will extend the experimental evaluation and will incorporate prediction functionalities in order to reduce network traffic.

References

[1] X. Liu, J. Cao, S. Tang, and P. Guo, "Fault Tolerant Complex Event Detection in WSNs: A Case Study in Structural Health Monitoring," *IEEE Transactions on Mobile Computing*, vol. 14, no. 12, pp. 2502–2515, Dec 2015.

[2] S. K. Ghosh, M. Suman, R. Datta, and P. K. Biswas, "Power Efficient Event Detection Scheme in Wireless Sensor Networks for Railway Bridge Health Monitoring System," in *2014 IEEE International Conference on Advanced Networks and Telecommuncations Systems (ANTS)*, Dec 2014, pp. 1–6.

[3] L. D. Xu, W. He, and S. Li, "Internet of things in industries: A survey," *IEEE Transactions on Industrial Informatics*, vol. 10, no. 4, pp. 2233–2243, Nov 2014.

[4] Y. Singh, S. Saha, U. Chugh, and C. Gupta, "Distributed Event Detection in Wireless Sensor Networks for Forest Fires," in *Computer Modelling and Simulation (UKSim), 2013 UKSim 15th International Conference on*, April 2013, pp. 634–639.

[5] M. Bahrepour, N. Meratnia, M. Poel, Z. Taghikhaki, and P. J. M. Havinga, "Distributed Event Detection in Wireless Sensor Networks for Disaster Management," in *Intelligent Networking and Collaborative Systems (INCOS), 2010 2nd International Conference on*, Nov 2010, pp. 507–512.

[6] J. Gubbi, R. Buyya, S. Marusic, and M. Palaniswami, "Internet of things (iot): A vision, architectural elements, and future directions," *Future Generation Computer Systems*, vol. 29, no. 7, pp. 1645 – 1660, 2013. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0167739X13000241

[7] M. A. Alsheikh, S. Lin, D. Niyato, and H. P. Tan, "Machine learning in wireless sensor networks: Algorithms, strategies, and applications," *IEEE Communications Surveys Tutorials*, vol. 16, no. 4, pp. 1996–2018, Fourthquarter 2014.

[8] N. J. Patel and R. H. Jhaveri, "Detecting packet dropping nodes using machine learning techniques in mobile ad-hoc network: A survey," in *Signal Processing And Communication Engineering Systems (SPACES), 2015 International Conference on*, Jan 2015, pp. 468–472.

[9] S. Rashid, U. Akram, S. Qaisar, S. A. Khan, and E. Felemban, "Wireless sensor network for distributed event detection based on machine learning," in *Internet of Things (iThings), 2014 IEEE International Conference on, and Green Computing and Communications (GreenCom), IEEE and Cyber, Physical and Social Computing(CPSCom), IEEE*, Sept 2014, pp. 540–545.

[10] S. M. Dima, C. Panagiotou, D. Tsitsipis, C. Antonopoulos, J. Gialelis, and S. Koubias, "Performance evaluation of a {WSN} system for distributed event detection using fuzzy logic," *Ad Hoc Networks*, vol. 23, pp. 87 – 108, 2014. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1570870514001218

[11] G. Wittenburg, N. Dziengel, S. Adler, Z. Kasmi, M. Ziegert, and J. Schiller, "Cooperative event detection in wireless sensor networks," *IEEE Communications Magazine*, vol. 50, no. 12, pp. 124–131, December 2012.

[12] M. Marin-Perianu and P. Havinga, *Ubiquitous Computing Systems: 4th International Symposium, UCS 2007, Tokyo, Japan, November 25-28, 2007. Proceedings*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007, ch. D-FLER – A Distributed Fuzzy Logic Engine for Rule-Based Wireless Sensor Networks, pp. 86–101. [Online]. Available: http://dx.doi.org/10.1007/978-3-540-76772-5_7

[13] K. Kapitanova, S. H. Son, and K.-D. Kang, "Using fuzzy logic for robust event detection in wireless sensor networks," *Ad Hoc Networks*, vol. 10, no. 4, pp. 709 – 722, 2012, advances in Ad Hoc Networks (II). [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1570870511001326

[14] S. Gaglio, G. L. Re, G. Martorella, and D. Peri, "A Lightweight Middleware Platform for Distributed Computing on Wireless Sensor Networks," *Procedia Computer Science*, vol. 32, no. 0, pp. 908 – 913, 2014, the 5th International Conference on Ambient Systems, Networks and Technologies (ANT-2014), the 4th International Conference on Sustainable Energy Information Technology (SEIT-2014).

[15] S. Gaglio, G. Lo Re, G. Martorella, and D. Peri, "A Fast and Interactive Approach to Application Development on Wireless Sensor and Actuator Networks," in *Emerging Technology and Factory Automation (ETFA), 2014 IEEE*, Sept 2014, pp. 1–8.

[16] S. Gaglio, G. L. Re, G. Martorella, and D. Peri, "High-level Programming and Symbolic Reasoning on IoT Resource Constrained Devices," *EAI Endorsed Transactions on Cognitive Communications*, vol. 15, no. 2, 5 2015.