Distributed Localized Contextual Event Reasoning under Uncertainty

Kostas Kolomvatsos *, Christos Anagnostopoulos†, Stathes Hadjiefthymiades‡

*Dept. of Informatics and Telecommunications, University of Athens, Greece &
Dept. of Computer Science, University of Thessaly, Greece
e-mail: kolomvatsos@cs.uth.gr
†School of Computing Science, University of Glasgow, UK
e-mail: Christos.Anagnostopoulos@glasgow.ac.uk
‡Dept. of Informatics and Telecommunications, University of Athens, Greece
e-mail: shadj@di.uoa.gr

***Abstract*—We focus on *Internet of Things* (IoT) environments where sensing and computing devices (nodes) are responsible to observe, reason, report and react to a specific phenomenon. Each node (e.g., an unmanned vehicle or an autonomous device) captures *context* from data streams and reasons on the presence of an event. We propose a distributed predictive analytics scheme for *localized context reasoning under uncertainty*. Such reasoning is achieved through a contextualized, knowledge-driven clustering process, where the clusters of nodes are formed according to their belief on the presence of the phenomenon. Each cluster enhances its localized opinion about the presence of an event through *consensus* realized under the principles of *Fuzzy Logic* (FL). The proposed FL-driven consensus process is further enhanced with semantics adopting Type-2 Fuzzy Sets to handle the uncertainty related to the identification of an event. We provide a comprehensive experimental evaluation and comparison assessment with other schemes over real data and report on the benefits stemmed from its adoption in IoT environments.**

***Index Terms*—Distributed predictive event analytics, data streams, knowledge-centric clustering, Type-2 Fuzzy Logic.**

1. Introduction

In IoT environments, *objects* transfer contextual knowledge to the network without human intervention. Such environments involve a *Wireless Sensor Network* (WSN), which consists of distributed wireless nodes (sensors) capable of sensing (observing) and reasoning about the occurrence of phenomena. Each node has sensing and computation capabilities for inferring localized knowledge. The fundamental requirement is the autonomous nature of nodes to perform sensing & inference, and disseminate *localized inferred knowledge* to their neighbors, or to a centralized information processing system, e.g., a *concentrator*.

Many critical IoT applications have been implemented on top of contextual data streams captured by (WSN) nodes. There are two main orientations: (i) nodes apply a distributed decision making mechanism to, locally, infer events; (ii) nodes send their measurements to a concentrator that, consequently, performs the inference process. Such events are related to e.g., security issues or violations of pre-defined constraints. As an example, in security applications, a monitoring infrastructure is imperative by adopting a centralized architecture which applies an efficient mechanism to derive alerts when specific criteria are satisfied [18], [30]. Another application domain is environmental monitoring [11], [15].

We propose a mechanism for distributed, localized contextual event reasoning involving (i) *context prediction* (adopting time-series forecasting), (ii) *context inference* (adopting *Fuzzy Logic* (FL)-based inference), and (iii) *knowledge-centric clustering* for disseminating pieces of knowledge among nodes and concentrators. Our mechanism builds on top of streaming contextual data and provides immediate inference to any identified abnormality, hereinafter referred to as *event*. A distributed clustering scheme groups nodes according to their *contextualized view* on the presence of an event. These groups are formed to further monitor and reason about the event and the concentrator forms a *contextual map* of the presence of inferred events.

Our mechanism combines both: *localized context inference* and *nodes clustering*. We introduce distributed event reasoning over the *compactness* of clusters concerning their recent opinion about an event. With the term *compactness*, we characterize the unanimity of the members of each cluster about the event. The rationale is to infer an event by minimizing the false alarms that could affect decision making, i.e., unsuitable decisions of handling a hazardous phenomenon. The technical contributions of our paper are:

- *localized contextual event inference* in light of minimizing the rate of false alerts;
- *distributed event inference* based on *knowledge-centric nodes clustering*;
- combination of (i) *localized context prediction* and (ii) localized context inference through FL for handling the uncertainty in describing events;
- sensitivity analysis of our mechanism and comparative assessment with: the *Single Sensor Alerting*

(SSA), the *Average Measurements Alerting* (AMA), the *Simple Prediction Model* (SPM), the *Moving Average Model* (MAM) and the centralized model discussed in [19], [20].

The paper is organized as follows: Section 2 reports on prior work, while Section 3 presents our rationale. Sections 4 and 5 introduce the localized context inference and the knowledge-centric clustering, respectively. Section 6 presents experimental evaluation and comprehensive comparison assessment. Section 7 concludes our paper by giving future research directions.

2. Related Work

Context monitoring and inference support the development of IoT applications [11]. An analysis on architectural solutions and a case study on distributed context inference for environmental monitoring is discussed in [5]. IoT devices are utilized to (i) monitor a specific area and (ii) deliver the captured contextual data to a central system [15]. The authors in [24] present a context monitoring framework involving computing aggregate methodologies. In [23], a WSN application for air quality monitoring in indoor environments is proposed. A concentrator collects contextual data from nodes equipped with a set of sensors. In [9], the authors present a framework for inference of forest fire events based on vision-enabled nodes. The authors in [18] present a mechanism for home monitoring based on the received signal strength of nodes. In [30], a WSN application for the surveillance of critical areas is proposed. Such a system focuses on ensuring integrity and authenticity of the generated alerts. In [4], a WSN-based framework is proposed for inferring hazardous underground materials. The focus of [16] is on large scale WSN deployments oriented to provide a monitoring and inference mechanism that aims at localized decisions. In [22], the authors present another distributed monitoring algorithm that aims to minimize false alerts. An hierarchical architecture that allows distributed context monitoring is proposed in [7]. DRAGON [17] is another inference model which is able to handle all types of events. It employs two physics metaphors: event center of mass, and node momentum. In [25], a model for the virtual representation of event sources is proposed. Events are modelled as internet resources accessible by any application, following an IoT approach. The goal of [26] is to propose a machine learning model for the identification of rule patterns adopted for inference. In [2], the authors present an event identification algorithm that takes into account the correlation among the sensed attributes and the spatio-temporal correlations with similar attributes measured by neighboring nodes. A general framework is discussed in [36]. The framework enables a flexible number of sensor nodes to dynamically collaborate in detecting and delivering any specified event. The authors in [38] propose an event detection model defined by a data fusion model. Events can be detected by computing data fusion probabilities on top of a genetic algorithm. In [1], a distributed algorithm to detect dynamic phenomena is proposed. Sensors self-organize into disjoint groups by first electing a few of them to be group leaders. Each group of sensors detect phenomena locally.

In context inference, reasoning over the principles of FL is a useful technique for delivering high quality results. The model in [10] predicts the peak particle velocity of ground vibration levels adopting FL-based inference. The FL-based context reasoning model in [29] estimates the radiation levels in the air. The adoption of the FL aims to handle missing values. The FL-based context fusion model presented in [32] reduces the uncertainty and false-positives within the process of fault detection. In [13], a FL-based inference system (Type-2 FL system) is proposed for ambient intelligence environments. Such a system learns the users' behavior in light of being adapted on users' profiles. In [6], the authors discuss an advanced multimodal approach for complex event detection through the use of FL. The proposed inference system evaluates the probability of fire detection while aiming at power conservation. In [34], the authors propose an improved spatial-based FL event detection algorithm to decrease the probability of false positives. A cluster-based data fusion algorithm for event detection is described in [14]. The K-means algorithm is adopted to form clusters while a FL method is adopted by cluster heads for local decision making. In [19], the authors propose a centralized mechanism that derives the appropriate decisions for the immediate identification of events. The system adopts data fusion and prediction for aggregating sensor measurements. The mechanism in [19] adopts FL for handling the uncertainty on the event reasoning. In [20], the authors propose a centralized mechanism that adopts multivariate data fusion, time-series prediction, and consensus theory for aggregating measurements using FL-based inference.

The major difference of our mechanism compared to the aforementioned efforts is that *we propose a distributed monitoring and localized context inference scheme instead of a centralized system*. In the aforementioned efforts, the concentrator centrally undertakes the responsibility of event reasoning. Our mechanism adopts *Type-2 Fuzzy Sets* instead of *Type-1 Fuzzy Sets*, as in [19] and [20], to deal with the induced uncertainty of the event knowledge representation. In that case, Type-2 FL-based inference achieves better performance compared with Type-1 FL-based inference, as it is substantiated through our comparative assessment. In combination with the proposed knowledge-centric clustering scheme, our mechanism is robust and reduces the communication overhead between nodes and concentrators.

3. Overview & Rationale

Consider a finite set of $|\mathcal{N}|$ nodes, $\mathcal{N} = \{n_1, n_2, \ldots, n_{|\mathcal{N}|}\}$, that monitor a specific area and perform localized reasoning to infer the presence of events. Nodes observe the same phenomenon with the help of sensors. A node receives incoming contextual data (sensors measurements) and, then, infers whether an event occurs or not. The degree of occurrence of an event, as locally

inferred by a node n_i, is associated with a *degree of danger* (DoD_i). DoD_i is *only* disseminated by n_i to its spatial neighboring nodes to further enhance the contextual knowledge of its neighborhood. The dissemination of the localized contextual knowledge represented by the DoD_i leads an automated clustering of nodes according to their view on the phenomenon. The clustering is achieved by the election of a node, referred to as *cluster head*, based only on DoDs. Hence, clusters are formed involving nodes (members) that have *similar* opinion about the presence of an event. Each cluster head, then, communicates with the concentrator. In this context, no centralized processes are adopted for clustering or inference. Through this approach, cluster heads convey aggregated contextual knowledge (i.e., aggregation of DoDs) to the concentrator, thus, minimizing the number of messages circulated in the network. Messages exchanged among members and cluster heads are not contextual values. Instead, they correspond to context inference represented by DoDs. The overall architecture of the proposed mechanism is depicted in Figure 1(Left). In Figure 1(Right), we present the architecture of a node. We depict the processes adopted by nodes and the aggregation of the DoDs by the knowledge-centric clustering and inference.

We propose a mechanism that builds on top of a localized FL-based inference system (Type-2 FL system; introduced later) for fusing (i) the *current* context, (ii) the *predicted* context, and (iii) the *deviated* context (defined later). Our mechanism locally derives the DoD_i for every n_i every time a contextual value is captured. Each n_i orchestrates a number of processes to handle incoming contextual data and derive the appropriate context inference:

- *Context Prediction* utilizes the trend of historical aggregated contextual data for forecasting short-term contextual data.
- *Statistical Learning*, incrementally, learns the *probability distribution function* (pdf) of each contextual parameter captured by n_i based on historical measurements;
- *Context Inference*, realized by a *Type-2 Fuzzy Logic System* (T2FLS), fuses the outputs of the context prediction and statistical learning processes. The goal is to derive the DoD_i for n_i. The DoD_i provides a localized inference of the event based on (i) the current measurements, (ii) the deviation of the current measurements from their expected values and, (iii) the predicted measurements. When the DoD_i exceeds a pre-defined threshold, n_i ensures the occurrence of an event and, then, initiates alerts to its assigned cluster head;
- *Knowledge-centric Clustering* based on DoDs of neighboring nodes. The resulted clusters contain nodes that have similar localized context inference results for the event. After clustering, each cluster head delivers aggregated localized knowledge (including its own inference) to the concentrator.

4. Localized Context Inference

Assume a discrete time domain $t = 1, 2, \ldots$. A context vector $\mathbf{x} = [x_1, \ldots, x_d]$ consists of d contextual parameters $x_j \in \mathbb{R}$ corresponding to sensor measurements. A node n_i at t captures context $\mathbf{x}_i[t]$ and locally processes it to infer an event. We assume that n_i can locally store the most recent M vectors $(\mathbf{x}_i[t-M], \ldots, \mathbf{x}[t-1])$. Based on this recent history, n_i is capable of predicting the context vector at t, $\hat{\mathbf{x}}_i[t]$. That is, the context predictor estimates the expected vector $\mathbf{x}_i[t]$ at t based on all the recent measurements from $t-M$ to $t-1$, i.e., $\hat{\mathbf{x}}_i[t] = \mathbb{E}[\mathbf{x}_i[t]|\mathbf{x}_i[t-1], \ldots, \mathbf{x}_i[t-M]]$. Once n_i has captured $\mathbf{x}_i[t]$, it can derive the prediction difference $e_i[t]$ between the estimated context and the actual (captured) one: $e_i[t] = \|\mathbf{x}_i[t] - \hat{\mathbf{x}}_i[t]\|_2$, where $\|\cdot\|_2$ denotes the Euclidean norm. As it will be shown later, this difference gives a first insight of how the actual vector is deviated from the expected vector; important for context inference.

Moreover, n_i incrementally learns the pdf of the context vector $f_i(\mathbf{x})$ as receiving $\mathbf{x}$. Through the incremental learning, n_i is capable of reasoning whether an instantaneous $\mathbf{x}_i[t]$ deviates significantly from the up-to-now estimated expectation $\mathbb{E}_i[\mathbf{x}; t]$, i.e., detects if $\mathbf{x}_i[t]$ lies outside the overall pattern of $f_i(\mathbf{x})$. The difference from the current estimated expectation is: $\mu_i[t] = \|\mathbf{x}_i[t] - \mathbb{E}_i[\mathbf{x}; t]\|_2$. The basic idea of the localized inference is that n_i combines (i) how much the current context vector is deviated from the predicted vector, and (ii) in what degree the current context vector is considered as an *outlier* given an overall (up-to-now) estimation of its statistical distribution patterns. The former indicator exploits short-term knowledge for reasoning, while the latter exploits an overall knowledge about the statistical patterns of the context vectors captured locally on n_i. The fusion of these pieces of knowledge results to more sophisticated context reasoning about an event occurrence.

Remark In this paper, we provide a solution dealing with univariate context, i.e., dimension $d = 1$. Hence, $\mathbf{x} \in \mathbb{R}^d$ reduces to the scalar $x \in \mathbb{R}$. The case of multivariate contextual data is on the agenda of our future research.

4.1. Localized Context Prediction

The proposed mechanism engages a time series prediction algorithm for locally predicting the upcoming contextual data. The mechanism should derive a predicted value in the minimum time, i.e., the provision of the estimation in (near) real time due to the criticality of IoT applications. We adopt a linear time-series predictor with low (linear) computational complexity. Specifically, let a context history of the recent M values $x_i[t-1], \ldots, x_i[t-M]$, with $x_i[t-m]$ be the contextual value corresponding to the m-th measurement, $m = 1, 2, \ldots, M$. We predict the context value $\hat{x}_i[t]$ through a linear combination of the $x_i[t-m]$ historical measurements with a_m coefficients, such that: $\hat{x}_i[t] = a_0 + \sum_{m=1}^{M} a_m x_i[t-m]$, satisfies the prediction equations $\mathbb{E}[\hat{x}_i[t] - x_i[t]] = 0$ and $\mathbb{E}[(\hat{x}_i[t] - x_i[t])x_i[t-m]] = 0$. That is, the set of coefficients $\{a_m\}$, are estimated

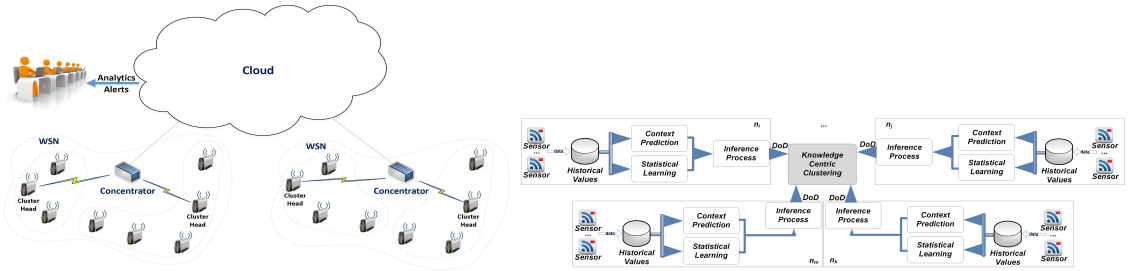

Figure 1. (Left) The inferred context flow from IoT sensing and computing nodes to the concentrators via the cluster heads; (Right) The architecture of an IoT node.

to minimize the error between the predicted $\hat{x}_i[t]$ and the actual contextual value $x_i[t]$. To estimate the a_m coefficients, ee adopt the Levinson-Durbin algorithm [8], [21].

4.2. Incremental Expectation Learning

The pdf of the parameter x_i, $f(x_i)$, captured by n_i is considered unknown. We rely on an incremental estimation of $f_i(x_i)$ based on historical values of x_i to provide an insight on the hidden statistics of the unknown distribution. The incremental estimation of the expectation $\mathbb{E}_i[x;t] = \int_{\mathbb{R}} x f_i(x)\mathrm{d}x$ up to t provides a view on the recent contextual observations to be compared with the current measurements. In this context, n_i can identify whether $x_i[t]$ significantly deviates from previous observations. To estimate $\mathbb{E}_i[x;t]$, we rest on an approximation based on the finite set of measurements and on an approximation of the $f_i(x)$ [3]. We adopt the widely known *Kernel Density Estimator* (KDE) [35]. KDE estimates the hidden distribution of the incoming samples. Let $x_i[1], x_i[2], \ldots, x_i[t]$, be the context values captured by n_i up to t. Since n_i has limited memory capacity, we have to estimate $f_i(x)$ on-the-fly, thus, estimate $\mathbb{E}_i[x;t]$ in an on-line mode. The cumulative KDE, up to t, is defined as: $\hat{f}_i(x;t) = \frac{1}{t \cdot h}\sum_{m=1}^{t} K\left(\left|\frac{x - x_i[t-m]}{h}\right|\right)$, where $h > 0$ is the bandwidth of a Kernel function $K(\cdot)$, which is symmetric and integrates to unity. For $K(\cdot)$, we adopt the Gaussian kernel, i.e., $K(u) = \frac{1}{\sqrt{2\pi}}\exp\left(-\frac{1}{2}u^2\right)$. $\hat{f}_i(x;t)$ is incrementally estimated by its previous estimate $\hat{f}_i(x;t-1)$ and the current context value $x_i[t]$. Specifically, let $u[t] = |\frac{x - x_i[t]}{h}|$. We have,

$$\begin{aligned}\hat{f}_i(x;t) &= \frac{1}{th}\Big(\sum_{m=1}^{t-1} K(u[t-m]) + K(u[t])\\ &= \frac{1}{th}\big((t-1)\hat{f}_i(x;t-1) + K(u[t])\\ &= \frac{t-1}{th}\hat{f}_i(x;t-1) + \frac{1}{th}K(u[t]) \qquad (1)\end{aligned}$$

Hence, upon capturing $x_i[t]$, $\hat{f}_i(x;t)$ is incrementally estimated by $\hat{f}_i(x;t-1)$ plus the Kernel function over the difference $|x - x_i[t]|$. Based on this incremental KDE estimation of $\hat{f}_i(x;t)$, we can approximate the expectation $\mathbb{E}_i[x;t]$ by $\mathbb{E}_i[x;t-1]$. By taking the integrals in both sides of the Eq(1), we obtain that

$$\begin{aligned}\mathbb{E}_i[x;t] &= \frac{t-1}{t}\mathbb{E}_i[x;t-1] + \frac{1}{t}\int_{\mathbb{R}} xK(|x - x_i[t]|/h)\mathrm{d}x\\ &= \frac{t-1}{t}\mathbb{E}_i[x;t-1] + \frac{1}{t}x_i[t] \qquad (2)\end{aligned}$$

since the kernel function (by definition) integrates to unity and $\int_{\mathbb{R}} uK(u)\mathrm{d}u = 0$. By that way, we adopt the incremental mean calculation in Eq(2) to determine the $\mathbb{E}_i[x;t]$.

4.3. Context Inference under Uncertainty

4.3.1. Fuzzy Context Inference. Context event inference is achieved locally on n_i at t by fusing the current context $x_i[t]$ with the deviation measures $e_i[t]$ and $\mu_i[t]$. The $\mu_i[t]$ is an indication whether the current context refers to an event-related observation or not. The $e_i[t]$ is an indication whether the current context significantly deviates from its predicted (short-term) trend. Such fusion is achieved through a finite set of *Fuzzy Inference Rules* (FIR). Each FIR reflects the DoD based on current context and the deviation measures.

Under the principles of fuzzy inference, we propose a T2FLS, which defines the fuzzy knowledge base for n_i, e.g., a set of FIRs like: '*when the local sensed temperature is high then the DoD for a fire event might be also high*'. We do not rely on a *Type-1 FLS* (T1FLS) as such an inference model has specific drawbacks when applied in dynamic environments and, more interestingly, when the construction of the FIRs involves uncertainty due to partial knowledge in representing the output of the inference [28]. In our case, this corresponds to the uncertainty of defining the occurrence of an event based only on the local knowledge. The limitation in a T1FLS is on handling uncertainty in representing knowledge through FIRs [12], [28]. In a T1FLS, experts define exactly the membership degree of input and output variables, e.g., the characterization of a context values as 'high' or 'low'. When the definition of a membership function involves also uncertainty, experts cannot be certain about the membership grade. In such cases, uncertainty is observed not only in the environment, e.g., we classify the DoD as 'high', but also on the description of the term, e.g., 'high', itself. In a T2FLS, membership functions are themselves 'fuzzy', which leads to the definition of FIRs incorporating such uncertainty [28].

4.3.2. Fuzzy Knowledge Base. FIRs refer to a non-linear mapping between three inputs: (i) x_i, (ii) e_i, and (iii) μ_i and one output, i.e., the DoD_i. The antecedent part of FIRs is a (fuzzy) conjunction of inputs and the consequent part of the FIRs is the DoD indicating the belief that an event *actually* occurs. The proposed FIRs have the following structure:
IF x_i is A_{1k} **AND** e_i is A_{2k} **AND** μ_i is A_{3k}
THEN DoD_i is B_k,

where A_{1k}, A_{2k}, A_{3k} and B_k are membership functions for the k-th FIR mapping x_i, e_i, μ_i and DoD_i (*values into unity intervals*), by *characterizing* their values through the terms: *low*, *medium*, and *high*. The structure of FIRs is the same as in a T1FLS. If a linguistic term, e.g., *high*, was represented by one fuzzy set in a T1FLS, we would use one membership function $g(x) \in [0, 1]$ mapping the real value (input) x to a discrete set of pairs $(x_j, g(x_j))$, e.g., $\{(0, 0); (0.25, 0.1); (0.5, 0.75); (1, 1)\}$, where $(0.25, 0.1)$ means that $x = 0.25$ has a membership degree $g(x) = 0.1$. In a T2FLS, A_{1k}, A_{2k}, A_{3k} and B_k are represented by two membership functions corresponding to *lower* and *upper* bounds [27]. For instance, the term '*high*', unlike in a T1FLS, whose membership for x is a number $g(x)$, is represented by two membership functions. Hence, x is assigned to an interval $[g_L(x), g_U(x)]$ corresponding to a lower and an upper membership function g_L and g_U, respectively (e.g., the membership of $x = 0.25$ is the interval $[0.05, 0.2]$). The interval areas $[g_L(x_j), g_U(x_j)]$ for each x_j reflect the uncertainty in defining the term, e.g., '*high*', useful to determine the exact membership function for each term. Obviously, if $g_L(x) = g_U(x), \forall x$, we obtain a FIR in a T1FLS. The interested reader could refer to [27] for information on reasoning under Type-2 FIRs.

4.3.3. Degree of Danger Determination. Without loss of generality, we assume that $x_i, e_i, \mu_i \in [0, 1]$ are normalized based on the minimum and maximum values depicted by the application domain. For instance, if the mechanism reasons about the existence of a fire event, we can set a maximum temperature value that a node can measure. We also define $DoD_i \in [0, 1]$, which conceptually aligns with the contextual parameter x_i. A DoD_i close to unity denotes the case where the danger is at high levels, i.e., there is a high belief that a hazardous phenomenon, like fire, actually occurs. The opposite stands when DoD_i tends to zero. For inputs and the output, we consider three linguistic terms: *Low*, *Medium*, and *High*. *Low* represents that a variable (input or output) is close to zero, while *High* depicts the case where a variable is close to unity. *Medium* depicts the case where the variable is around 0.5. For instance, *Low* e_i indicates that the current and predicted context are close enough, thus, current context *follows* the trend of its historical context. *Low* μ_i denotes that the current context does not significantly deviate from the regular pattern. Similar rationale stands for the remaining terms. For each term, human experts define the upper and the lower membership functions. Here, we consider triangular membership functions as they are widely adopted in the literature. However, our T2FLS is generic enough, thus, any type of membership functions can be adopted.

FIRs incorporate the human knowledge on the inference process. Table 1 shows the proposed FIRs, designed for scenarios where contextual data reaching the upper limit exhibit a '*danger*' case. Upon receiving $x_i[t]$ and its corresponding deviation measures $e_i[t]$ and $\mu_i[t]$, the mechanism is activated as follows: (Step 1) calculation of the interval (based on the membership functions) for each input; (Step 2) calculation of the active interval of each FIR; (Step 3) performance of 'type reduction' to combine the active interval of each FIR and the corresponding consequent. Step 3 produces the interval of the consequent, and accordingly, the defuzzification phase[1] determines a (crisp) value for the DoD_i at t. The most common method for 'type reduction' is the *center of sets type reducer* [28], which generates a Type-1 fuzzy set as output, which is, then, converted in a scalar value for the DoD_i after defuzzification. When the DoD_i is over a pre-defined threshold $\theta \in [0, 1]$, the mechanism triggers an alert, i.e., infers the occurrence of an event.

TABLE 1. THE FUZZY INFERENCE RULES

Rule	x_i	e_i	μ_i	DoD_i
1	Low	Low or Medium	Any	Low
2	Low	High	Any	Medium
3	Medium	Low	Low or Medium	Medium
4	Medium	Low	High	Low
5	Medium	Medium or High	Any	Medium
6	High	Low	Low or Medium	High
7	High	Low	High	Medium
8	High	Medium	Low or Medium	High
9	High	Medium	High	Medium
10	High	High	Low	High
11	High	High	Medium or High	Medium

5. Knowledge-centric Clustering

5.1. Clustering & Cluster Head Election Objectives

The clustering process refers to the creation of a set of clusters of the WSN nodes (from $\mathcal{N}$) based on their DoDs. The process is repeated at each *clustering era* $T, 2T, 3T, \ldots$ (T being a pre-defined time interval). In each cluster, a node is elected as the cluster head, being responsible to exchange messages (only aggregated DoDs, as discussed later) with the concentrator. Hence, the number of messages circulated in the network is reduced as it is not necessary for each node to relay its messages to the concentrator. The election process concerns a node n_i that it experiences the highest DoD_i among its neighbors. The aim of the cluster head is to notify its members about its appointment, thus, avoiding redundant messages dissemination. After the cluster head appointment, members send their DoDs to their cluster head. This results into enhanced neighborhood knowledge by unanimously inferring a possible phenomenon in the area under consideration.

The primary objectives of the election process are: (i) Appointment of a subset of nodes as cluster heads; (ii) Dynamically changing the cluster head appointment. Evidently,

1. Defuzzification is the process of producing a quantifiable result in FL, given fuzzy sets and the corresponding membership degrees.

this prolongs the WSN lifetime by changing cluster head appointments, thus, balancing energy consumption for the inference process and transmission of messages to the members and the concentrator; (iii) Termination of the election process within a constant number of iterations (exchanged messages). It should be noted that the description of the cluster head replacement process (i.e., objective (ii)) is beyond the scope of this paper. It is also worth noting that we do not make any assumption about the spatial distribution of nodes. In our model, every node can act as both a cluster head and a member, which motivates the need for efficient head election algorithms.

5.2. Cluster-head Election & Knowledge Exchange

A baseline solution for the election process involves nodes flooding their DoDs to their neighbours. Hence, the node with the highest DoD is elected as the cluster head. However, this solution requires a significant number of messages. Moreover, since the election process is re-initiated after T, a high energy budget is required for that type of communication. There are certain election algorithms which could be adopted. In our case, nodes exchange their DoDs and, then, 'elect' the head. To this end, we adopt the 'cluster-head' election strategy discussed in [37] to elect the cluster head and modify the election criteria to reflect the knowledge exchange.

At each node, the election process requires a number of iterations $K > 0$. In every step, nodes send and receive specific small-sized messages from neighbors containing their local DoDs. Before n_i starts the election process, it configures a (local) probability of becoming a cluster head ξ_i, hereinafter referred to as *election probability* (EP). ξ_i is considered a function of the DoD_i, i.e., $\xi_i \sim \max(\xi_{\min}, DoD_i)$, where ξ_{min} is the minimum EP for each node. ξ_i is not allowed to fall below the pre-defined threshold $\xi_{\min}$, e.g., 10^{-3}. This restriction is essential for terminating the election process in $K = O(1)$ iterations, as proved below.

A node n_i with a relatively high ξ_i starts the following process: it sends announcement messages of the form $\langle \xi_i, n_i \rangle$ to the $\mathcal{N}_i$ neighbours to be a cluster head. On the other hand, a node n_j with a low ξ_j delays the transmission of announcement messages and considers itself 'non-cluster head' if it has heard $\langle \xi_i, n_i \rangle$ with $\xi_i > \xi_j$. Specifically, during iteration $k \in [1, K]$, n_i decides to become a cluster head with EP ξ_i. Through the process, n_i can either be elected to become a cluster head according to ξ_i or remain at the same status (i.e., non-cluster head) according to overheard announcements within its communication range. n_j selects its cluster head n_i to be the node with the highest DoD_i; this is achieved by the comparison of ξ_i and ξ_j. Every node n_i, then, multiplies its EP ξ_i with a factor of $\chi > 1$, goes to the next step $k+1$ and so on, i.e., $\xi_i(k+1) = \min(\chi \xi_i(k), 1)$. If n_i decides to become a cluster head since its EP ξ_i has reached unity, it sends an announcement 'cluster head n_i' to its neighbours in $\mathcal{N}_i$. $n_j \in \mathcal{N}_i$, then, considers itself 'non-cluster head' if it has heard from n_i a '*cluster head* n_i' and terminates the election process. Note that, this election process is completely distributed. A node can either decide to become a cluster head, since its DoD_i is the highest among its neighbours, or be a member which awaits for the corresponding messages by its unique cluster head.

For clarifying the election algorithm, we provide an example. Assume two nodes n_1, n_2 in a cluster with IDs 1 and 2, respectively. Initially, both nodes start with $\xi_1(1) = \xi_2(1) = 0.001$ and $\chi = 1.2$ ($\xi_{min} = 0.001$). Starting the algorithm, they send to each other the following messages: $\langle 0.001, 1 \rangle$, $\langle 0.001, 2 \rangle$. As $\xi_1 = \xi_2$, they both consider that they are cluster heads and they do not delay the announcement of messages. Hence, they perform the following calculations: $\xi_1(2) = \min(\chi \xi_1(1), 1) = 0.0012$, $\xi_2(2) = \min(\chi \xi_2(1), 1) = 0.0012$. In the second step, n_1 observes $DoD_1 = 0.5$ and n_2 observes $DoD_2 = 0.2$, Hence, we have: $\xi_1(2) = \max(\xi_{min}, DoD_1) = 0.5$, $\xi_2(2) = \max(\xi_{min}, DoD_2) = 0.2$. Now, the exchanged messages are: $\langle 0.5, 1 \rangle$, $\langle 0.2, 2 \rangle$. Nodes observe $\xi_1 > \xi_2$, thus, n_1 considers itself as cluster-head and n_2 being a member delaying the announcement of messages. In the next step, we get: $\xi_1(3) = \min(\chi \xi_1(2), 1) = 0.6$, $\xi_2(3) = \min(\chi \xi_2(2), 1) = 0.24$. If new DoDs are observed (this process could not be synchronized for the two nodes), their values will affect the calculation of ξ_is, thus, leading to, potentially, a new cluster head. The process continues as explained above. The result is that, after K steps, the node having the highest DoD will become the new cluster head.

Lemma 1. The election process requires $O(1)$ iterations.

Proof See Appendix 1.

Note that the number of iterations for each node does not depend on the number of neighbours and is bounded by a constant. Indicatively, when $\xi_{min} = 10^{-3}$ and $\chi = e$, a node needs at most eight iterations to elect/be elected as cluster head.

Lemma 2. The message exchange complexity in the election process is $O(1)$ per node and $O(|\mathcal{N}|)$ for the network.

Proof See Appendix 2.

After the appointment of n_i as a cluster head, n_i locally calculates the difference of its DoD_i with DoDs of its member nodes $n_j \in \mathcal{N}_i$, i.e., $\Delta DoD_i = \left| DoD_i - \frac{1}{|\mathcal{N}_i|} \sum_{n_j \in \mathcal{N}_i} DoD_j \right|$, where $\mathcal{N}_i \subset \mathcal{N}$ is the set of n_i's members and $|\mathcal{N}_i|$ is the cardinality of $\mathcal{N}_i$. This difference reflects the degree of consensus of the cluster on the context inference result. n_i delivers to the concentrator the pair $\langle DoD_i, \Delta DoD_i \rangle$ iff $DoD_i \geq \theta$ and $\Delta DoD_i < 0.5\theta$. The concentrator then acquires knowledge for a specific region about the appearance of an event and to what extend this local inference result is of high belief supported by other nodes other than n_i itself.

6. Performance Evaluation

6.1. Performance Metrics & Comparison Models

We focus on the performance of our model concerning the inference of specific phenomena and define the following performance metrics:

Rate of false alerts (RoFA). The RoFA $\in$ [0,1] represents the rate of false alerts that our mechanism derives. When RoFA $\rightarrow 1$, the mechanism results many false alerts and no efficient conclusion could be drawn for the true state of the phenomenon. When RoFA $\rightarrow 0$, the mechanism decreases the rate of false alerts. RoFA is defined as the ratio of false alerts out of a total number of inferences.

Index of Alert (IoA). The IoA refers to the (index) identifier of the measurement that triggers an alert (i.e., IoA $\in \{1, 2, \ldots\}$). Through IoA, we examine how 'close' to the real case an event is triggered (not at early stages to avoid false alerts and not many stages after the real event). IoA is evaluated by defining (annotating) events over the adopted datasets.

Number of messages ϵ. It indicates the total number of messages sent to the concentrator by cluster heads. The lower the ϵ is, the lower resources are spent. ϵ does not refer to messages circulated during the election process. We consider the resources spent for intra-cluster communication as negligible compared to the resources spent for delivering information from cluster heads to concentrators. This is because such messages are circulated in relatively lower distances than the communication with the concentrator. Assume the lifetime of the network (in term of energy) equal to $\mathcal{S}$ and cluster heads depicted by the set $\mathcal{C}_H$. At each *clustering era* $T, 2T, \ldots$, our mechanism generates clusters and $\mathcal{C}_H$, respectively. Hence, in the network lifetime, $\lfloor \frac{\mathcal{S}}{T} \rfloor$ clustering eras are realized. During clustering, the entire set of nodes send their DoDs to keep the concentrator informed about the status of the phenomenon, i.e., $|\mathcal{N}|$ messages are delivered. In addition, when cluster heads are generated, then only $\mathcal{C}_H$ messages are delivered to the concentrator. Hence, the following equation holds true: $\epsilon = \lfloor \frac{\mathcal{S}}{T} \rfloor (\mathcal{C}_H(T-1) + |\mathcal{N}|)$.

We experiment with two real datasets. The first dataset is available in Intel Berkeley Research Lab[2]. It contains measurements from 54 sensors deployed in a lab. We get 54,000 measurements such that $|\mathcal{N}| = 54$ sensors each one producing 1,000 measurements related to the temperature of the lab. All measurements are scaled in [0, 1]. As no hazardous event is identified by these measurements (i.e., the probability of a true event is equal to zero), we consider the injection of faulty values to see whether the proposed mechanism produces false alerts. We assume that a high temperature (e.g., around 600 Celsius) defines the case of a fire incident and inject faulty measurements as indicated in [31]. The faults rule defined in [31] indicates that every actual measurement x_i is replaced as follows: $x_i \leftarrow (1+a)x_i, a \in \{2, 3, 5\}$. We provide experiments with scenarios where a portion of measurements $p \in \{1\%, 5\%, 10\%, 20\%, 40\%, 60\%, 80\%\}$ are considered as faulty. p represents how many faulty measurements are included into our dataset. The higher the p is, the higher the number of faulty measurements. If $p = 1\%$, we inject 150 faulty measurements, when $p = 5\%$, we inject 750 faulty measurements, and so on. For cluster head election, we adopt the methodology described in Section 5.2 and set $\xi_{min} = 10^{-3}$, $\chi = e$ and $K = 100$.

We also use contextual data from a real flood event reported by $|\mathcal{N}| = 10$ water level sensors[3]. Sensors were located in the shore of a river to monitor the water level and infer floods. In this dataset, the probability of the alert is equal to unity as the flood event was realized in the past. We adopt 265 measurements just before and during the flood event and scale them in [0,1]. Based on these data, we examine whether the proposed mechanism is capable of identifying the event at the right time (not too early to avoid false alerts and not after the real event to avoid disaster).

Comparison Models: We compare our mechanism, referred to as Model, with the *Single Sensor Alerting* (SSA) mechanism. The SSA mechanism delivers an alert when a single measurement is over a pre-defined threshold, i.e., n_i infers an event at t if $x_i[t] \geq \theta$. Through simulations, we set $\theta = 0.7$, for the entire set of the examined models. SSA mechanism is not based on any reasoning to derive the final decision. We also compare our Model with the *Average Measurements Alerting* (AMA) mechanism. The AMA mechanism produces alerts when the average measurement is over θ (for each sensor's historical values). The AMA mechanism realizes a linear opinion pool [27], where observations are of equal weight. The decision process in AMA is applied in every sensor as we focus on a distributed scheme. We also compare the T2FLS with a model that is based on a *Simple Prediction Model* (SPM) over the sensors measurements. The SPM is applied on the historical values reported by each sensor and if one of them exceeds θ, the SPM triggers an alert. The SPM adopts the linear predictor for the historical values of each sensor. In addition, we compare our mechanism with the *Moving Average Model* (MAM) discussed in [33]. The MAM calculates the mean of a set of sensors measurements to produce each point of the output. In general, if $z = (z(1), z(2), \ldots)$ is the sensors input, the output signal is $x = (\hat{x}(1), \hat{x}(2), \ldots)$, with $\hat{x}(k) = \frac{1}{W} \sum_{i=0}^{W-1} z(k-i)$ and W be the number of measurements for averaging. Furthermore, we compare our model with the centralized models in [19] and [20]. Both models (i.e., [19] & [20]) adopt a Type-1 FLS (T1FLS) in a concentrator that receives measurements by nodes. The concentrator is responsible for context event inference. The T1FLS in [19] combines the aggregated contextual data and the predicted context values to infer DoD centrally on the concentrator. The model in [20] combines aggregated and predicted contextual values along with the consensus value as realized by the unanimity of nodes.

2. Intel Lab Data, http://db.csail.mit.edu/labdata/labdata.html

3. http://www.pegelonline.wsv.de

6.2. Performance & Comparison Assessment

We execute a set of experiments based on the Intel Berkeley Research Lab dataset. In Table 2, we present our results for different p. For $p \in \{1\%, 5\%, 20\%\}$ our Model does not result any false alerts while for $p = 10\%$, it produces only a single false alert. The SSA, SPM mechanisms exhibit higher RoFA than our Model. When $p \in \{40\%, 60\%, 80\%\}$, our Model results to RoFA = $\{0.010, 0.286, 0.757\}$ while, the SSA, SPM mechanisms result false alerts that are above the half of the total inference results. When $p \in \{60\%, 80\%\}$, the majority of the measurements are considered as faulty. Our Model minimizes the number of false alerts when compared to SSA, SPM especially for $p \leq 40\%$. SSA and SPM are based only on single (predicted) measurements (when current context is over θ), thus, do not take into consideration possible negative effects on the reporting process (e.g., interference, hardware problems). The Model proceeds with the knowledge fusion of the current and the predicted context along with the deviation of the regular patterns before inferring an event. Through this approach the Model does not purely rely on context values, and infers an event taking into account also historical context accompanied by similar future estimations. AMA and MAM ($W = 10$) does not produce significant RoFA except when $p = 80\%$ (MAM results RoFA = 0.990), however, as we discuss below, this has a negative effect in events identification.

TABLE 2. ROFA METRIC COMPARISON

p	$RoFA_{SSA}$	$RoFA_{AMA}$	$RoFA_{SPM}$	$RoFA_{MAM}$	$RoFA_{Model}$
1%	0.058	0.000	0.000	0.000	0.000
5%	0.179	0.000	0.179	0.000	0.000
10%	0.356	0.001	0.350	0.001	0.001
20%	0.535	0.000	0.525	0.000	0.000
40%	0.755	0.000	0.755	0.000	0.010
60%	0.831	0.002	0.831	0.002	0.286
80%	0.915	0.006	0.920	0.990	0.757

We further perform a set of experiments with real contextual data retrieved by a real flood event. In this dataset, the flood event is actually identified by starting from 45th measurement ($t = 45$, i.e., IoA = 45). By taking into consideration only $|\mathcal{N}|$ = 5 nodes, the Model derives an alert at $t = 49$ and $t = 43$ for $|\mathcal{N}|$ = 10. We observe that our Model is able to infer the event just in its beginning when the number of nodes is large enough (e.g., $|\mathcal{N}|$ = 10). The SSA and the SPM mechanisms have similar behaviour resulting to IoA $\in \{34, 35, 36, 39\}$ for the same data. This indicates that the SSA and the SPM mechanisms derive false alerts many stages before the real event commences. The interesting is that the remaining mechanisms, i.e., AMA, MAM, cannot identify the event. In both mechanisms, the averaging scheme adopted to derive the final result cannot conclude the violation of θ.

We perform a set of experiments for different θ to reveal the impact of θ on our Model. We run experiments for the Intel Berkeley Research Lab dataset and $\theta \in \{0.5, 0.9\}$. In Table 3, we present the comparison between our Model and the remaining models for different p. We observe that the Model achieves less false alerts than the SSA and the SPM even for low θ. False alerts are also limited for AMA and MAM except the case where $p = 80\%$ (MAM results to RoFA = 0.990). Evidently, the lower the θ is, the higher the RoFA becomes. When $\theta = 0.9$, the Model does not result any false alerts. Actually, it produces the lowest number of false alerts among the examined models. Nonetheless, a high θ can possibly lead to missing alerts, i.e., the event cannot be identified. We now set $\theta = 0.9$ and experiment with the real flood data. As expected, the Model is not able to identify the event due to the high θ. This makes the Model very conservative in concluding on the occurrence of an event. When $\theta = 0.5$, the Model results an IoA $\in \{43, 32\}$ for $|\mathcal{N}| \in \{5, 10\}$, respectively, while AMA and MAM cannot identify the event. In any case, the proposed T2FLS can be 'calibrated' through the adopted FIRs to be more sensitive/strict in the event identification, thus, affecting the RoFA. In the first place of our future research agenda is the provision of a model that will adapt the FIRs on the needs of the environment (e.g., use a reference model that depicts the most efficient behaviour).

TABLE 3. ROFA METRIC COMPARISON VS. θ

	p	$RoFA_{SSA}$	$RoFA_{AMA}$	$RoFA_{SPM}$	$RoFA_{MAM}$	$RoFA_{Model}$
$\theta = 0.5$	1%	0.088	0.000	0.088	0.000	0.052
	5%	0.340	0.000	0.335	0.000	0.171
	10%	0.576	0.002	0.576	0.002	0.326
	20%	0.796	0.000	0.790	0.000	0.666
	40%	0.944	0.002	0.940	0.002	0.923
	60%	0.983	0.004	0.980	0.004	0.956
	80%	0.995	0.065	0.990	0.990	0.984
$\theta = 0.9$	1%	0.045	0.000	0.450	0.000	0.000
	5%	0.102	0.000	0.100	0.000	0.000
	10%	0.197	0.001	0.190	0.001	0.000
	20%	0.324	0.000	0.324	0.000	0.000
	40%	0.465	0.000	0.465	0.000	0.001
	60%	0.561	0.001	0.561	0.001	0.001
	80%	0.678	0.003	0.670	0.003	0.001

In Figure 2, we plot our results for ϵ. These experiments refer to $|\mathcal{N}| = 100$ and are not related to any simulation of nodes location and connectivity radius. We observe that ϵ is mainly affected by the number of clusters (cluster heads) and not by the number of clustering eras $T > 1$. Obviously, when $T = 1$, the Model requires a significantly high number of messages and performs as a baseline model, where at each t, all nodes report their DoDs to the concentrator. The baseline model requires $\mathcal{S} \cdot |\mathcal{N}|$ messages. If we set $T > 1$, our results show that the higher the $|\mathcal{C}_H|$ is, the higher the ϵ becomes. This is as $|\mathcal{C}_H| \to |\mathcal{N}|$, more messages are required for the communication between nodes and concentrator.

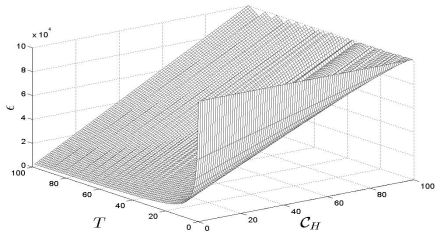

Figure 2. Results for the ϵ metric.

Finally, Table 4 shows the comparison between the Model and models discussed in [19] & [20] using the Intel Berkeley Research Lab dataset. In general, our Model outperforms the remaining mechanisms for $p \in \{5\%, 10\%, 20\%, 40\%\}$. The opposite stands when $p \in \{60\%, 80\%\}$. These results show that our distributed model is heavily affected by fluctuations in the contextual measurements being more sensitive to produce alerts. Concerning the required messages, for $\mathcal{S} = 1000$, $T = 10$ and $|\mathcal{N}| = 54$, we get that both models in [19] and [20] require 54,000 messages to be sent to the concentrator. In our Model, we get that the average $|\mathcal{C}_H| = 4$, thus, the Model requires 9,000 messages for the aforementioned setup.

TABLE 4. RoFA METRIC COMPARISON: T1FLS VS. MODEL

p	$RoFA_{T1FLS}$ [19]	$RoFA_{T1FLS}$ [20]	$RoFA_{Model}$
1%	0.000	0.000	0.000
5%	0.004	0.001	0.000
10%	0.000	0.000	0.001
20%	0.012	0.017	0.000
40%	0.026	0.020	0.010
60%	0.040	0.014	0.286
80%	0.047	0.014	0.757

7. Conclusions

We propose a distributed event analytics mechanism in IoT, where nodes locally infer events from data streams. Our mechanism performs distributed reasoning based on contextualized knowledge-centric clustering, where clusters are formed according to nodes' belief on the presence of phenomena. The localized opinion of each node is derived through Type-2 Fuzzy Logic inference to handle the uncertainty related to knowledge representation of an event. We evaluate our approach in terms of the rate of false alerts using real data and provide a comparative assessment with other event inference mechanisms. Our future agenda involves the enhancement of our mechanism with multivariate contextual vectors in the inference process and the knowledge-centric clustering.

Acknowledgement

This work is funded by the European Commission (FIRE+ challenge, H2020) that aims to provide research, technological development and demonstration under the grant agreement no 645220 (RAWFIE).

References

[1] Abu Safia, A., Al Aghbari, Z., Kamel, 'Phenomena Detection in Mobile Wireless Sensor Networks', Network Systems Management, 2016, pp. 24-92.

[2] Ali, K., Ali, S. B., Naqvi, I. H., Lodhi, M. A., 'Distributed Event Identification for WSNs in Non-Stationary Environments', GLOBECOM, 2015.

[3] Bishop, C. M., 'Pattern Recognition and Machine Learning', Springer, 2006.

[4] Chehri, A., Fortier, P., Tardif, P. M., 'Security Monitoring Using Wireless Sensor Networks', 5th Conf. on Communication Networks and Services Research, 2007.

[5] Di Palma, D., Bencini, L., Collodi, G., Manes, A., 'Distributed Monitoring Systems for Agriculture based on Wireless Sensor Network Technology', Int. J. on Adv. in Networks and Services, 3(1-2), 2010.

[6] Dima, M. S., Antonopoulos, C., Koubias, S., 'On Event/Time Trigerred and Distributed Analysis of a WSN System for Event Detection, Using Fuzzy Logic', Journal of Sensors, vol. 2016, 2016.

[7] Dressler, F., Nebel, R., Awad, A., 'Distributed Passive Monitoring in Sensor Networks', in Proceedings of the INFOCOM, 2007.

[8] Durbin, J., 'The fitting of time series models', Rev. Inst. Int. Stat., vol. 28, 1960, pp. 233-243.

[9] Fernndez-Berni J., Carmona-Galn R., Martnez-Carmona J. F., Rodrguez-Vzquez ., 'Early forest fire detection by vision-enabled wireless sensor networks', Int. Journal of Wildland Fire, vol. 21, 2012.

[10] Fisne, A., Kuzu, C., Hudaverdi, T., 'Prediction of Environmental Impacts of Quarry Blasting Operation Using Fuzzy Logic', Environmental Monitoring Assessment, vol. 174, 2011.

[11] Gouveia, C., Fonseca, A., 'New Approaches to Environmental Monitoring: the Use of ICT to Explore Volunteered Geographic Information', GoeJournal, vol. 72, 2008.

[12] Hagras, H., 'A hierarchical type-2 fuzzy logic control architecture for autonomous mobile robots', IEEE TFS, vol. 12, 2004.

[13] Hagras, H., Doctor, F., Callaghan, V., Lopez, A., 'An Incremental Adaptive Life Long Learning Approach for Type-2 Fuzzy Embedded Agents in Ambient Intelligent Environments', IEEE TFS, vol. 15, 2007.

[14] Hao, Z. Q., Zhang, Z. J., Chao, H. C., ' A Cluster Based Fuzzy Fusion Algorithm for Event Detection in Heterogeneous Wireless Sensor Networks', vol. 2015, 2015.

[15] Hardas, B. M., Asutkar, G. M., Kulat, K. D., 'Environmental Monitoring Using Wireless Sensors: A Simulation Approach', 1st Int. Conf. on Emerging Trends in Engineering and Technology, 2008.

[16] Hsin, C. F., Liu, M., 'A Distributed Monitoring Mechanism for Wireless Sensor Networks', WiSe, 2002.

[17] Hubell, N., Han, Q., 'DRAGON: Detection and Tracking of Dynamic Amorphous Events in Wireless Sensors Networks', IEEE TPDS, 23(7), 2012.

[18] Kausar, F., Al Eisa, E., Bakhsh, I., 'Intelligent Home Monitoring Using RSSI In Wireless Sensor Networks', Int. J. of Computer Networks and Communications, 14(6), 2012.

[19] Kolomvatsos, K., Anagnostopoulos, S., Hadjiefthymiades, S., 'An Efficient Environmental Monitoring System adopting Data Fusion, Prediction and Fuzzy Logic', 6th IISA, Greece, 2015.

[20] Kolomvatsos, K., Anagnostopoulos, C., Hadjiefthymiades, S., 'Intelligent Contextual Data Stream Monitoring', 8th PETRA, Greece, 2015.

[21] Levinson, N., 'The Wiener RMS error criterion in filter design and prediction', J. Math. Phys., vol. 25, 1947.

[22] Liu, C., Cao, G., 'Distributed Monitoring and Aggregation in Wireless Sensor Networks', INFOCOM, 2010.

[23] Lozano, J., Suarez, J. I., Arroyo, P., Ordiales, J. M., Alvarez, F., 'Wireless Sensor Network for Indoor Air Quality Monitoring', Chemical Engineering Transactions, vol. 30, 2012.

[24] Mainwaring, A., Polastre, J., Szewczyk, R., Culler, D., Anderson, J., 'Wireless Sensor Networks for Habitat Monitoring', WSNA, 2002.

[25] Martinez, N. L., Martinez, J. F., Diaz, V. H., 'Virtualization of Event Sources in Wireless Sensor Networks for the Internet of Things', Sensors, 14(12), 2014, pp. 22737-22753.

[26] Mehdiyev, N., Krumeich, J., Enke, D., Werth, Loos, P., 'Determination of Rule Patterns in Complex Event Processing Using Machine Learning Techniques', Procedia Computer Science, vol. 61, 2015, pp. 395-401.

[27] Mendel, J. M., 'Type-2 Fuzzy Sets and Systems: An Overview', IEEE Computational Intelligence Magazine, 2(2), 2007.

[28] Mendel, J. M., 'Uncertain Rule-Based Fuzzy Logic Systems: Introduction and New Directions', Upper Saddle River, Prentice-Hall, 2001.

[29] Ramadan, A. B., El-Garhy, A., Zaky, F., Hefnawi, M., 'New Environmental Prediction Using Fuzzy Logic and Neural Networks', Int. J. of Computer Sciences Issues, 9(3), 2012.

[30] Rothenpelier, P., Kruger, D., Pfisterer, D., Fischer, S., 'FleGSens Secure Area Monitoring Using Wireless Sensor Networks', Int. Conf. on Sensor Networks, Information, and Ubiquitous Computing, 2009.

[31] Sharma, A., Golubchik, L., Govidnan, R., 'On the Prevalence of Sensor Faults in Real World Deployments', SECON '07, 2007.

[32] Shell, J., Coupland, S., Goodyer, E., 'Fuzzy data Fusion for Fault Detection in Wireless Sensor Networks', UK Workshop on Computational Intelligence, 2010.

[33] Smith, S., 'The Scientist and Engineers Guide to Digital Signal Processing', 2nd Ed., California Technical Publishing, San Diego, CA, 1999.

[34] Sun, X., Zhuang, A., Wang, B., Li, T., Liu, Q., 'ISFLE:An improved neighbor-based fuzzy logic event detecting algorithm for wireless sensor networks', Int. J. of Future Generation Communication and Networking, 7(3), 2014.

[35] Wand, M. P., Jones, M. C., 'Kernel Smoothing', Chapman and Hall, 1995.

[36] Wu, H., Cao, J., Fan, X., 'Dynamic Collaborative In-Network Event Detection in Wireless Sensor Networks', J. of Telecommunications Systems, 62(1), 2016, pp. 43-58.

[37] Younis, O., Fahmy, S., 'HEED: A Hybrid, Energy-Efficient, Distributed Clustering Approach for Ad Hoc Sensor Networks', IEEE TMC, 3(4), 2004.

[38] Zou, P., Liu, Y., 'An Efficient Data Fusion Approach for Event Detection in Heterogeneous Wireless Sensor Networks', Applied mathematics and Information Sciences, vol. 1, 2015, pp. 517-526.

Appendix

Appendix 1: Proof of Lemma 1

Proof Consider a probability multiplication factor $\chi > 1$ and that node n_i starts with the minimum EP of being a cluster head, i.e., $\xi_i = \xi_{\min} > 0$. Since at each iteration step the node multiplies its current ξ_i with χ then, in the worst case, that node will be either a cluster head or a member when the process stops at the first iteration step K such that $\chi^{K-1}\xi_{\min} \geq 1$. That is, the maximum number of iteration steps are $K = \min\{k > 0 : \chi^{k-1}\xi_{\min} \geq 1\}$. Hence, the required number of iterations is $K = \lceil \log_{\chi} \frac{1}{\xi_{\min}} \rceil + 1$, which maps to $O(1)$ iterations. Now, if node n_i starts the election with $\xi_i > \xi_{\min}$ then $O(1)$ iterations are the maximum number of steps for the election process. ∎

Appendix 2: Proof of Lemma 2

Proof In the election process, a node which is about to become a cluster head generates at most $K = O(1)$ messages. On the other hand, a node which is about to become a member delays in sending messages and sends one message to just join its cluster head after considering itself as 'non-cluster head'. Obviously, the number of those messages (member messages) is strictly less than $|\mathcal{N}|$, since at least one node will decide to be a cluster head. Hence, the number of messages exchanged in the network is upper-bound by $K \times |\mathcal{N}|$, which is $O(|\mathcal{N}|)$. ∎