Environmental Engineering and Management Journal

June 2020, Vol. 19, No. 6, 1007-1016
http://www.eemj.icpm.tuiasi.ro/; http://www.eemj.eu

"Gheorghe Asachi" Technical University of Iasi, Romania

NOVEL APPROACH FOR FINDING SHORTEST ROUTE USING DIJKSTRA'S ALGORITHM AND FUZZY LOGIC IN A WIRELESS SENSOR NETWORK INTEGRATED IN A FOREST FIRE DETECTION SYSTEM

Nikola Manev*, Boban Temelkovski, Nevena Serafimova, Jugoslav Achkoski

Military Academy "General Mihailo Apostolski" – Skopje, Vasko Karangeleski bb, Skopje, Republic of North Macedonia

Abstract

This paper proposes a Fuzzy Logic Controller/ Dijkstra's Algorithm based software that calculates the most reliable communication link between a WaspmotePlug&Sense Sensor Node and a Meshlium Router in a Wireless Sensor Network (WSN). The algorithm implements the effect of three parameters important for the functioning of the WSN: Waspmote involvement, the received signal strength indicator (RSSI) and the distance of the Waspmotes, for achieving optimal work capability of the system. Due to the inherent weaknesses of the conventionally used Star and Tree topologies which provide a single route with no alternatives on the forwarding of data, the lack of a software or algorithm that would select the optimal route and the fact that signal quality does not necessarily indicate optimal route employment, we propose an application of a Mesh topology along with a Fuzzy Logic Controller/ Dijkstra's Algorithm based software. Mesh topology allows each controller to be individually connected to at least two Meshlium routers, thus providing an alternative transmission solution in case of damage to certain links between the nodes and selection of a more efficient link for transmission of information. The Fuzzy Logic Controller/ Dijkstra's Algorithm setup reduces energy consumption of the WSN fire detection system by calculating and determining which routers should start up, instead of all of them working.

Keywords: Dijkstra's algorithm, energy consumption, fuzzy logic controller, software defined network, WSN

Received: March, 2019; Revised final: January, 2020; Accepted: February, 2020; Published in final edited form: June, 2020

1. Introduction

Forest fires are a dangerous phenomenon that imposes a severe threat to the forested areas. In order to decrease the risk of serious environmental and economic damages and degradation of the natural forest habitat, a variety of early prevention activities as well as plans for a timely reaction are needed (Faivre et al., 2018; Nasi et al., 2002). These include: fire prevention by changing the consciousness of the population (people being the main cause of forest fires) (Hirschberger, 2012), better fire control, increasing the resistance of the assets to a fire (Parisien et al., 2020), relocating resources away from the path of the fire etc. (Al-Dhief et al., 2019). Fire detection and signaling of an appropriate alarm is deemed to be the most significant factor for preventing vast damage (Bernabeu et al., 2004; Gottuk et al., 2020).

Today, there exist at least a dozen systems in the world that are utilized in early wildfire detection. They are all expected to detect an occurred fire in a faster, easier and simpler manner, in order to successfully deal with the impending threat. The lessons learned from their usefulness, reliability and accuracy in localizing the fire conclude that the future of forest fire detection is the use of Wireless Sensor Networks (WSNs). The sensors provide environmental insight, are able of computing the

* Authors to whom all correspondence should be addressed: e-mail: manev.nikola@yahoo.com; Phone: +389 78591578

collected data and can be interfaced with wireless networks. A wireless sensor network collects data at the wireless sensor node and sends it to a router. The transmitted data is then presented to the system servers (a base station). With WSNs being a relatively "new" technology, their inherent weaknesses are being constantly addressed and overcome. This opens up a field of experimentation, research and application where previous knowledge is applied to WSNs. Such is the use of Dijkstra's Algorithm and Fuzzy Logic, with the improvements being applicable to any WSN.

WSNs usually employ a large number of sensor nodes, each equipped with a battery whose replacement and recharging is a difficult task in most application environments. Hence, improving the energy savings in WSN's applications is an important challenge. In this paper, we mainly focus on energy consumption savings and observe a shortest route approach to minimizing the signal path within the network. Through the application of a Fuzzy Logic Controller/ Dijkstra's Algorithm software for calculating the most reliable communication link between the nodes, we are achieving an optimal work capability of the system.

The issue of limited resources of the Wireless Sensor Networks (WSNs) is a key concern that should be given careful consideration. Energy demands and radio bandwidth utilization are addressed as a result. Musznicki et al. (2012) categorize the most common WSN multicast procedures, depending on the targets' group identification according to their geographic position. Routing procedures must overcome the challenge of non-optimal routing paths in order to successfully transport messages. The Dijkstra-based Localized Energy-Efficient Multicast Algorithm (DLEMA) addresses this challenge by focusing on discovering energy shortest paths that provide maximum geographical advance towards the desired destinations.

In addition Ya-qiong and Yun-rui (2016) focus on the use of a uniform clustering routing algorithm that takes both from the Dijkstra Algorithm and K-means. This algorithm mitigates the problem of too many nodes in the random clustering of a cluster, which induces increased energy consumption, thereby shortening the network's lifetime. In KDUCR (Uniform Clustering Routing algorithm based on K-means and Dijkstra algorithm), the sink node uses Dijkstra's Algorithm with heads' position to calculate the shortest paths from every head to the sink node. The KDUCR algorithm balances the nodes' energy consumption better and prolongs the lifetime of the WSN.

Zhang (2013) presents an Energy Saving-oriented Routing Algorithm Based on Dijkstra (ESRAD), designed to be economic in energy consumption during information transmission between any two sensor nodes in a WSN. By evaluating and assigning a weight index to each edge (communication link) whose value represents the energy consumption of that specific link, the ESRAD uses a Dijkstra-based routing algorithm that can reliably detect the path that will consume the least energy and therefore add to the efficiency of the entire network. This idea is further employed in Ganda, (2016), through the use of Fuzzy Logic in creating simulation software which can give an optimal route based on the condition given by the user. The software (fuzzy logic controller) is developed in MATLAB. Fuzzy logic processes the current condition of the map which becomes an input for the Dijkstra's Algorithm. The result of the program is an optimal route that satisfies criteria from three standpoints: distance, density, and confusion.

Musznicki et al. (2012), Ya-qiong and Yun-rui (2016) and Zhang (2013) address and focus on energy consumption in WSNs in terms of sensor relative distance and hopes, and geographical coverage, while Ganda (2016) using fuzzy logic manages in defining the weights of the edges to depict a city transportation network. These two software methods are integrated for achieving a more thorough path selection process, by considering the effect that the Waspmote (a wireless sensor platform) involvement, the received signal strength indicator (RSSI) and the distance of the Waspmotes have on achieving an optimal work capability by the system. The presented Fuzzy Logic Controller/ Dijkstra's Algorithm reduces energy consumption of the fire detection system, by calculating and determining which routers along the chosen path should start up, instead of all of them working.

While the aforementioned papers provide only partial insight into Dijkstra's Algorithm and its applicability to WSNs in forest fire detection, they lack an actual real life experiment to complement their research. This paper provides on-field testing in a simulated fire environment to check the functionality of the system, as well as to optimize some of its functions.

At the start, the paper focuses on related research and experiment in the field of Wireless Sensor Networks, emphasizing the use of Dikjstra's Algorithm and Fuzzy Logic with WSNs. This is expanded by giving a description on network topologies and topologies that are most applicable to WSNs. Additionally, the paper examines the integration of Fuzzy Logic in the Dijkstra's Algorithm by building a controller that provides link weights that are applied in the weighted graph simulating a Mesh - organized WSN. The sensors and the Meshliums are the nodes of this graph, while the virtual links of the network are the paths. We set up the experimental environment and discuss the results obtained by application of the algorithm. In conclusion, we refer to the focal points of the paper and give an insight of the future work.

2. Materials and methods

In this section, the development and the implementation of a fuzzy logic algorithm is described, relative to the established WSN topology. The ultimate goal of its application in WSN systems

for detection of forest fires is to find the shortestroute for reliable data transfer and consequently, to reduce the energy level consumption. A fuzzy logic controller is developed to do assessment of the value of the link or more precisely, to calculate the weight of each link that is exploited in data transmission. Then, the Dijkstra Algorithm uses the obtained link weights in order to find the shortest route and to help in avoiding links that are not reliable for data transmission or are costly in terms of energy consumption.

Our goal is to determine the applicability of this approach that relies on accurate calculation of the network's link weights and their use in determining the shortest route for data transmission, by exploiting different input parameters of the established forest fire detection system.

Integration of Dijkstra's Algorithm. Network topologies are vital for understanding the applicability of Dijkstra's Algorithm in WSNs. Topologies can be defined as a geometric interconnection pattern by which the stations (nodes/computers) are connected, using transmission media. Physical topologies emphasize the hardware in the network, while logical topologies represent the data flow between nodes (Santanu and Pinaki, 2013). The organization of each topology is suited to specific tasks and has its own advantages and disadvantages. There are eight basic topologies: Point-to-Point; Bus; Star; Ring; Tree; Mesh; Hybrid and Daisy Chain topology. Because of their simplicity, obsoleteness and the fact that the rest of these topologies do not apply to WSNs (require land lines), only the Tree, Star, Mesh and Hybrid topology have been successfully used with WSNs.

However, the biggest disadvantage of the Tree topology is that the existence of only one route between any two nodes of the network means that a failure in one of the nodes, results in a failure of the entire branch. With Star topologies, a failure of the central node leads to a failure in the entire network. Mesh and Hybrid topologies circumvent this by having devices connected with many interconnections. In a well-connected Mesh (Fig. 1) every node is connected to every other node in the network (Pandya, 2013). This is the most efficient connectivity solution, especially in a forest fire detection system where it is critical for the data to reach its final destination, timely activate an alarm and provide the user with an updated situational report from the field.

Here, a parallel can be made between communication and road networks, since planning efficient routes has grown essential with people, products or services having to be transported or delivered on time at the best price, using the shortest available route. The tools and applications required usually rely on a Dijkstra's Algorithm or some of its variations. The Dijkstra's Algorithm finds the shortest path between a source node ***s*** and every other node in the weighted graph that represents the network This same general principle can also be applied in finding the shortest path from a single starting node to a single destination node. Once the shortest path to the destination node has been determined the algorithm will stop calculating.

Dijkstra's Algorithm is an iterative procedure that maintains a distance (weight) parameter for each vertex (node, point) of the weighted graph. These distances (edges, imaginary lines that connect the vertices) are initially set to infinity for all vertices except the source vertex *s*. This convention does not imply an infinite distance, but denotes that those vertices have not been visited yet. At each iteration (repetition), we have a "tree" rooted at *s*. For the first iteration, the tree will be the single vertex *s*, and the distance to it will be zero. The next vertex *v* to be added to the tree will be the closest neighboring vertex to *s*.

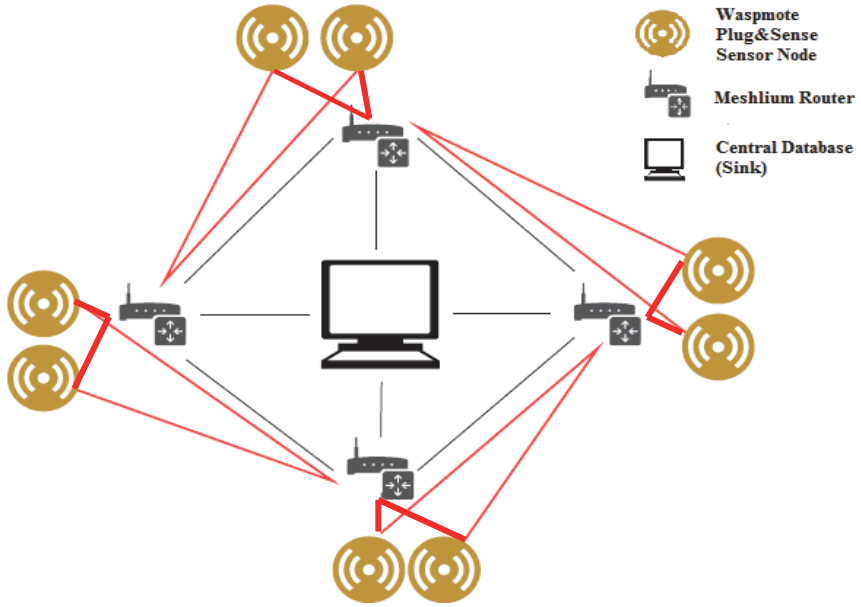
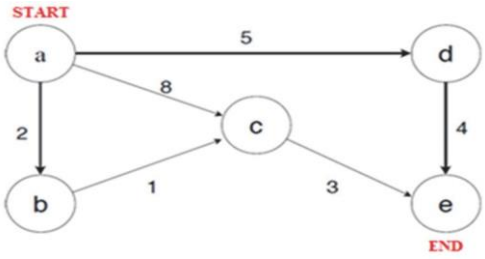


Fig. 1. WSN Mesh topology (each node or sensor can be reached through at least two paths)

Once found, *v* is added to the tree and checked for each of its neighbors *u*, to see if the path from *s* to *v* along with the edge *uv*, gives a shorter distance to *u* than the already stored. If yes, the distance of *u* is updated to the smaller value. We also maintain for each vertex *v* a parent vertex, which determines the last edge that is used to get to *v* from *s*.

Let $G = (V, E)$ be a weighted graph, with weight function $w: E \to \mathbb{R}$ that maps edges to real-valued weights. If $e = (u, v)$, we write $w(u, v)$ for $w(e)$. *Shortest path* between two vertices is a path that has shortest length, also called *link-distance*. The length of a path $p = \langle v_0, v_1, ..., v_k \rangle$ is the sum of the weights of its constituent edges, as depicted in Eq. (1):

$$lenght(p) = \sum_{i=1}^{k} w(v_{i-1}, v_i) \quad (1)$$



$length(\langle a, b, c, e, d \rangle) = 6$; Distance from a to e is 6

Fig. 2. A weighted graph with 5 vertexes

The distance from *u* to *v*, denoted by $d(u, v)$ is the length of the minimum length path if there is a path from *u* to *v*, and is ∞ otherwise.

Fig. 2 is not a schematic of the testbed and only serves to describe Dijkstra's Algorithm in terms of selecting the shortest path between two vertices, vertex $\langle v \rangle$ and vertex$\langle v \rangle$, in a weighted graph with a total of 5 vertices $\langle v, v, v, v, v \rangle$. We can see that there are three possible paths leading from $\langle v \rangle$ to $\langle v \rangle$, however the Dijkstra's Algorithm would choose the path with $w(h) = w(\langle v, v, v, v \rangle) = 6$, since this is the shortest path from $\langle v \rangle$ to $\langle v \rangle$.. Dijkstra's Algorithm is actually very intuitive, having that any sub-path of a shortest path must also be a shortest path. In the graph represented in Fig. 2, $\langle v, v, v, v \rangle$ is a shortest path and the sub-path $\langle v, v, v \rangle$ is also a shortest path. All characteristics of a road network relate to a wireless sensor network as well, since their representation as weighted graphs incorporates a common information structure. This analogy further applies to choosing the most efficient and optimal path from a starting node to a final destination.

Fuzzy logic controller. This section describes the structure of the fuzzy logic controller through the MATLAB program package. The projected fuzzy logic controller has three input parameters and one output which results from the fuzzy operation. The input parameters in the fuzzy model include Waspmote involvement, the received signal strength indicator (RSSI) and the distance of the Waspmote.

The working state of the Waspmote is determined by its battery level, which on the other hand depends on the number of sunny hours during the day. The data for this parameter are collected from the Meshlium and have two possible values (states): 0 indicates that the Waspmote does not work and 1 that it sends data.

The RSSI is the strength of the beacon's signal as seen on the receiving device, with value ranging from -255 to 0. It depends on several factors among which are: the distance of the receiver, the broadcasting power, the configuration of the terrain including effects of absorption, interference, diffraction etc. When considered in an environment such as dense forests and unpredictable weather conditions, these external factors make the RSSI even more unstable. In an attempt to emphasize the effect of the environment over the signal strength, the distance has been included into the weights' assessment as an additional influencing factor.

For the RSSI values we have defined 5 membership functions, based on the xBee standards and the description of the strength of the received signal given in dBm. Regarding the distance parameter, having that the maximum transmission distance of the signal is 40 km, we have defined 5 membership functions whose range is proportional to the RSSI value. The membership levels of the input parameters are presented in Table 1 (where RSSI values should be taken with a negative sign).

Table 1. Membership range levels for input parameters

Input Parameter Level	***1***	***2***	***3***	***4***	***5***
Waspmote [On/Off]	-0.5-0.7	0.3-1.5			
RSSI [dBm]	20-52	48-71	68-75	73-87	84-120
Distance [km]	0-9	8-17	15-27	24-36	33-45

The output is a numbered scale with values from 1 to 11 that represent the weight factor of the link through which the packet is transmitted. The output function actually represents a scale of numbers from 1 to 11. The scale range is determined by collecting the highest levels of the individual input parameters. The estimated scale range of 10 is enlarged by adding a plus-membership function representing the state when the Waspmote is off, so the output scale will have 11 triangular functions. These output values from 1 to 11 represent the weight factor of the link through which the packet is transmitted, and will serve Dijkstra's Algorithm to select the most energy-efficient transmission path through the WSN.

In this fuzzy logic controller there are 50 rules that cover for all possible input situations. The number of rules is obtained by multiplying the numbers of membership functions for the individual parameters.

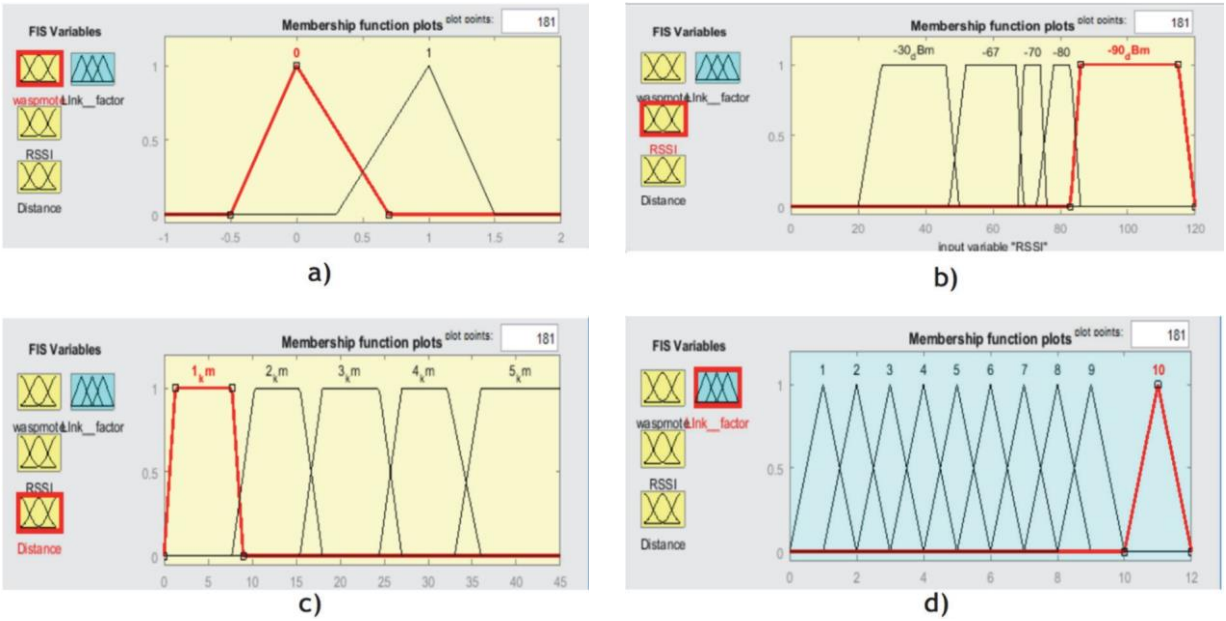


Fig. 3. Description of fuzzy logic membership function: a) Waspmote, b) RSSI, c) Distance and d) link factor

There are several defuzzification methods that lead to the correct result. The defuzzification centroid method is one of the most well known and most widely used methods for defuzzification. In the continuous case, the strength value of the control signal is obtained by the following relationship (Eq. 2):

$$Output = \frac{\sum_{i=1}^{n} center_i \times Strenght}{\sum_{i=1}^{n} Srenght} \quad (2)$$

Experiment setup. This section of the paper relates to the research conducted for development of the Advanced System for Prevention and Early Detection of Forest Fires (ASPires). As a part of the research, a series of tests were conducted in order to check the functionality of the system as well as to optimize some of its functions. Concerning the equipment characteristics, one thing that came to our attention was the disproportion between the system's small battery capacity, in compare to its large consumption demonstrated when all nodes and routers were operational. In a system with multiple routers, each ranked by quality of data transfer, the system will require that the signal takes the router that ranks highest, and therefore this should be the only router working.

In order to minimize this consumption as much as possible, we set up the experiment in a scaled down, laboratory environment with the idea of virtually interconnecting three Libelium's WaspmotePlug&Sense sensor nodes (Fig. 4a) ***with two Libelium's Meshlium multiprotocol routers*** (Fig. 4b).

It should be noted here that, due to the fact that we had at our disposition one Meshlium, the two (multiple) routers effect has been achieved by starting "the first Meshilum router" and connecting it to the Plug&Sense independently from the "second router". The later was started up manually, but only after the first router was shut down. The protocol in use in both cases was xBee. The focus of this research is directed towards an architecture where one transmitter would be connected to at least two receivers (Fig. 1).

Physically, the Waspmote nodes were set up along a line distance of 2m (at a height of 1.5m), then 10m and 30m (at a height of 3m) from the start of the fire (Fig. 5b), simulating as if they were mounted to a tree or a surveillance pillar. According to the local weather forecast provided by AccuWeather, no wind was present while conducting the experiment, which is important for the reliability of the received measurements. After starting the system, the initial values for CO (carbon monoxide), CO_2 (carbon dioxide), Temperature, Humidity and Atmospheric Pressure have been registered. Three fire modules were set up to include fire initialization, development of a small fire and occurrence of a well-developed fire. In a synchronized manner, each of the modules was monitored from start to finish. The schematic of the experimental setup is illustrated in Fig. 6.

During the course of the experiment, the fire intensity has been monitored constantly to register the changes in its development from a smoldering to a well-developed fire. The Meshlium Manager System was monitored in parallel, as the results from the Waspmote Plug&Sense nodes came in and were saved. The sensor data were gathered by the Meshlium and structured in WaspFrames. Connectivity of the Waspmote devices has been monitored with the Meshlium's built-in model for previewing connected nodes. After testing visibility, data exchange and rates of failures were checked. The Gateway connectivity to Internet is tested through checking of 3G networking, remote connectivity to the Meshlium and access to the Meshlium data base. The controlled variation of

certain measured attributes will register possible delay of information, the communication errors, the authenticity of the received data and the allover confidentiality of the system. Because of the inherent weakness of the conventionally used, Libelium proposed Star and Tree topologies which provide a single route with no alternatives to the data forwarding, the signal quality not necessarily reflecting the use of the optimal route and the lack of a software or algorithm that would select the optimal route, we propose using of Mesh topology along with a Fuzzy Logic Controller/Dijkstra's Algorithm based software. The input values for the Fuzzy Logic Controller are derived (and proportionally adjusted to mimic different scenarios) from the experiment's values for the distance, battery consumption, and signal quality (strength) within the presented architecture.

(a) (b)

Fig. 4. (a) The 3 WaspmotePlug&Sense Sensor Nodes; (b) One of the Meshlium Multiprotocol Routers

(a) (b)

Fig. 5. (a) The sensor probes of the Plug&Sense nodes; (b) A Plug&Sense node attached to a pillar at a height of 3m

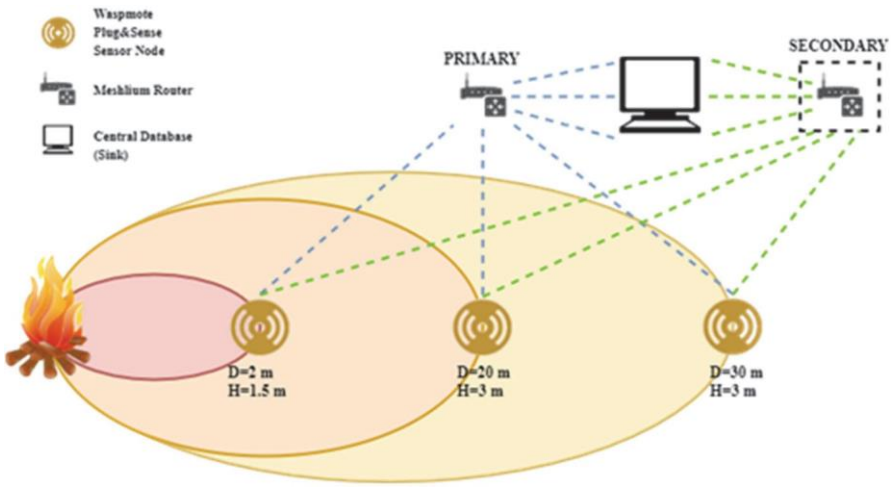


Fig. 6. Schematic of the live experiment setup

3. Results and discussion

For the purpose of verifying the functionality of the fuzzy logic controller, we have written 20 possible locations for setting up the controllers. The input parameters in the controller are the Waspmote involvement (controller), the received signal strength indicator (RSSI) and the distance of the Waspmote. The value of the controller is based on its residual battery, and because it is the key to the operation of the sensor network in 95 percent of scenarios, we assume that the controller has sufficient working power. This means that the RSSI and the distance parameters are those that will determine the degree of output. We have considered different values for the RSSI and the distance, while also checking the performance of the controller and displaying its response to signals of varying strength. Considering the theoretical statements on these parameters and their influence, we are confident that the controller works in line with the theoretical provisions for the signal strength. The values given in Table 2 represent the weight factor of the links and are later used as input values for the Dijkstra's Algorithm.

Using Matlab we have created a scaled model of the actual real-life wireless sensor network, by representing it as a weighted graph. In fact, a slight variation of Fig. 1 and Fig. 6 served as the template for the graph. The graphs weights were defined using an adjacency matrix, with the weights being randomly chosen and imported from Table 2. The nodes are named by capital letters, with "A" being the sink (database), "B, C, D, and E" being the Meshlium Routers and "F, G, H, I, J, K, L and M" serving as each of the Plug&Sense nodes.

As mentioned earlier, each sensor is connected to at least two routers (much like in the experiment architecture). We use Matlab's ***shortest path*** function, to calculate the shortest route between two nodes on this weighted graph. The nodes chosen are the WaspmotePlug&Sense Node "M" and obviously the database "A". The idea behind this choice was to simulate as if a signal (data) from "M" is being sent to the database. The path that the signal (the data) takes is actually the shortest route between these two nodes, and we can observe this from the results located in the Command Window in Fig. 7.

Table 2. Link factor values for different scenarios

Nb	*Waspmote (0 or 1)*	*RSSI (-2 to -120)*	*Distance (0km to 40km)*	*Link factor (1 to 11)*
1	1	-60	22.5	5
2	1	-30	18.5	4
3	1	-45	25	4.26
4	1	-45	12	3
5	1	-52	8	3.15
6	1	-78	31	8
7	1	-84	35	9.44
8	1	-76	35	8.66
9	1	-70	27	7
10	1	-64	28	6
11	1	-33	4	2
12	1	-35	8	2.15
13	1	-35	13	3
14	1	-40	17	3.59
15	1	-47	21	4
16	1	-53	21	5
17	1	-47	2	2
18	1	-58	26	5.59
19	1	-75	32	7.45
20	0	0	7.5	11

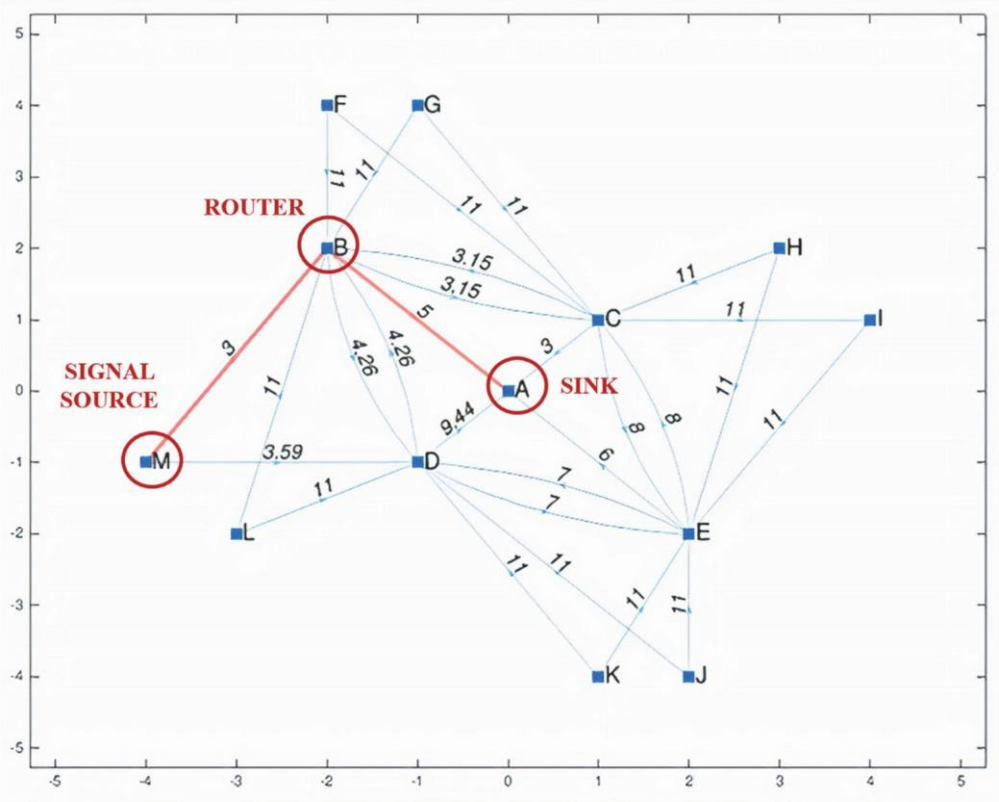


Fig. 7. WSN representation as modeled in MATLAB

Table 3. Available communication links arranged by their link factor

Available communication links from sensor node "M" to database "A" ***(arranged by their link factor from best to worst)***	
1. MBA = 8	8. MBDEA = 20.26
2. MBCA = 9.15	9. MBCEA = 20.15
3. MDBA = 12.85	10. MDECA = 21.59
4. MDA = 13.03	11. MDBCEA = 25
5. MDBCA = 14	12. MBDECA = 25.26
6. MDEA = 16.59	13. MDECBA = 26.74
7. MBDA = 16.7	14. MBCEDA = 30.59

The resulting shortest path value is 8, and we can clearly see that that the signal took the path through the router "B". However, we have 14 available communication links that could have been used against this primary choice, with all of them given in Table 3. Furthermore, all of the links having a route weight 11 signifies that they did not turn on, because there is no communication requirement from them to transmit the data from node "M" to database "A".

Calculation of energy consumption. The classical topology proposed in the Libelium tutorials is the Star topology, since nodes can establish point-to-point wired node connections over a MAC address. In our proposal, we apply a Mesh topology, where each controller is individually connected to at least two Meshlium routers. The aim is to provide an alternative transmission solution in case of damage to certain links between the nodes as well as selection of a more efficient link for transmission of information. The choice of the link will be made by checking the strength of the RSSI value. In the beginning, the controller will send data to the two Meshlium routers to which it is connected, and after the check, the link with a lower weight factor will be turned off. This of course increases the complexity and power consumption of the network. The power consumption of the controller (P_c in mW) for one hour will be calculated using Eq. (3), where *I(mA)* and *U(V)* are the controller's electric current and voltage:

$$P_c = I\,(mA) \cdot U\,(V) = 359mW \tag{3}$$

In the calculations of the controller's power consumption during work state, the values for *I(mA)* and *U(V)* in Eq. (3) have been taken from the manufacturer's technical guide (WTG, 2018).

Since the controller at work state uses only the *Xbee* communication module, which has a consumption of 15 mW (P_{xBee}), the total power consumption of the entire network (*P*) will be given by Eq. (4):

$$P = P_{xBee} + P_c = 374mW \quad (4)$$

Table 4. Energy consumption comparison (in mW) for *Star*, *Mesh standard (st)* and *Mesh with fuzzy logic controller (flc)* topologies

Topology	*2h*	*4h*	*6h*	*12h*	*1 day*	*2 days*	*15 days*	*Per month*
Star	748	1496	2244	4466	8976	17952	134649	269280
Mesh1x2 (st)	1496	2244	4466	8976	17952	34649	269280	538560
Mesh1x2(flc)	749.04	1498.08	2247.12	4472.24	8988.48	17976.96	134838.2	269672.4

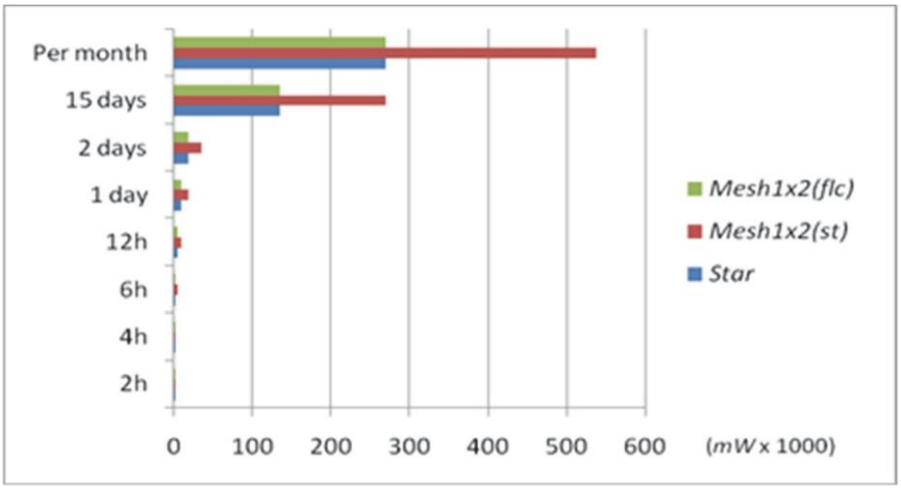


Fig. 8. Graphical comparison of energy consumption for *Star*, *Mesh(st)* and *Mesh(flc)*

Accordingly, as well as having the fact that the link checking will be done every two hours, the additional power that will be consumed in checking another link for a duration of 10 seconds will be calculated with Eq. (5):

$$P_{10\,s} = 374mW / 360 = 1.04mW \quad (5)$$

If a controller needs to check a total of N links within the system, the equation for power consumption will be calculated according to Eq. (6):

$$\square = 1.04\square\square * (\square — 1); (\forall \square \leq 2) \quad (6)$$

Table 4 reports energy consumption using Star and Mesh topologies for various time spans, when connecting one Waspmote controller to two Meshlium routers via Xbee communication protocol. The time step at which the consumption can be calculated is two hours, the period in which the link – checking protocol is activated. Moreover, the measurements presented in the second row of Table 4 (*Mesh1x2(st)*), result from a standard (uncontrolled) application of a two-routers' Mesh connectivity topology. The third row (*Mesh1x2(flc)*) however, presents the energy consumption when applying the algorithm for link selection in the two-routers Mesh topology.

Clearly, a distinct reduction of about 50% in energy consumption can be achieved in this way (Fig. 8). At the same time, the energy consumption in the *Mesh1x2(flc)* network is negligibly higher (less than 1%) than the consumption in the Star topology, which is practically insignificant if compared to the contribution of the Mesh for the improved connectivity of the network.

4. Conclusions

Despite having their fair share of shortcomings, WSNs remain one of the best technologies when it comes to early forest fire detection and signaling of an appropriate alarm, thereby preventing massive fire damages. The proposed Wireless Sensor Network collects data at the Waspmote Plug&Sense wireless sensor node, and sends the data to a Meshlium router. The transmitted data is then presented to system's servers (a base station) and processed to activate the alarm.

With the Fuzzy Logic Controller/ Dijkstra's Algorithm software, we calculated the most reliable communication link between the

WaspmotePlug&Sense Sensor Node and the Meshlium Router in a 13 nodes WSN, interconnected by 28 links in a Mesh network topology. The weights of the links were calculated through the Fuzzy Logic Controller using 3 signal quality parameters: Waspmote involvement, the received signal strength indicator (RSSI) and the distance of the Waspmotes.

The applied Mesh topology has been applied as an alternative to the conventionally used and manufacturer proposed Star and Tree topologies, which provide a single route with no alternatives on data forwarding, whereas the quality of the signal does not necessarily reflect the use of the optimal route. By using Mesh topology paired with the Fuzzy Logic Controller/ Dijkstra's Algorithm software, we achieved optimal work capability of the system by selecting optimal route and saving energy. This approach enhances system's reliability and reduces the overall energy consumption, by determining which routers should start up, instead of all of them working. In the proposed model, the efficiency of the established communication between the nodes is crucial. It should be pointed out that WSN technologies are susceptible to some limitations that are typical for forest environments. The seasonal and environmental conditions immensely influence data transmission in WSNs. The level of the signal strength can be further degraded depending on the type of the forest, where the maximum distance transmission may be substantially decreased in compare to the non-forest environments. A suitable routing protocol designed for communication between the nodes and base station is important in obtaining high accuracy and stability of the detection system. In this sense, a successful implementation of the algorithm would depend upon the reliability of the applied WSN solution and the quality of the local wireless links, having in mind not only the reliability of provider's services but also the physical security and the integrity of the network.

That being said, the presented results are applicable beyond the experimental environment that has been considered in this paper. In terms of future work, while the proposed Fuzzy Logic Controller/ Dijkstra's Algorithm has the potential of serving as a standalone software tool, it can be tested in real environment applications through interfacing with other fire detection software.

Acknowledgments

This paper resulted from a research related to the development of Advanced System for Prevention and Early Detection of Forest Fires, Project Number 2016/PREV/03 Ref. ARES(2016)6878547–09/12/2016, ASPires. The project is funded by the European Civil Protection and Humanitarian Aid Operations Agency - ECHO and its results are widely applicable among the European Union Member States.

References

Al-Dhief F.T., Sabri N., Fouad S., Abdul Latiff N.M., Albader M.A.A., (2019), A review of forest fire surveillance technologies: Mobile ad-hoc network routing protocols perspective, *Journal of King Saud University - Computer and Information Sciences*, **31**, 135-146.

Bernabeu P., Vergara L., Bosh I., Igual J., (2004), A prediction/detection scheme for automatic forest fire surveillance, Digital Signal Processing, 14, 481-507.-

Faivre N., Xanthopoulos F., Rodríguez J., Calzada V., Xanthopoulos G., (2018), *Forest Fires - Sparking firesmart policies in the EU*, European Commission, Brussels, DOI: 10.2777/181450.

Ganda J., (2016), Simulation of routing option by using two layers fuzzy logic and Dijkstra's algorithm in MATLAB 7.0., *Journal of Electrical and Electronics Engineering*, **1**, 11-18.

Gottuk D.T., Peatross M.J., Roby R.J., Beyler C.L., (2020), advanced fire detection using multi-signature alarm algorithms, Fire Safety Journal, 37, 381-394.

Hirschberger P., (2012), *Forests on Fire - The Causes and Consequences of Forest Fires all Around the World*, vol. 6, (in German), WWF Deutschland, Berlin.

Musznicki B., Tomczak M., Zwierzykowski P., (2012), *Dijkstra-Based Localized Multicast Routing in Wireless Sensor Networks*, 8th International Symposium on Communication Systems, Networks & Digital Signal Processing (CSNDSP), Poznan, 1-6.

Nasi R., Dennis R., Meijaard E., Applegate G., Moore P., (2002), Forest fire and biological diversity, On line at: http://www.fao.org/tempref/docrep/fao/004/y3582e/y3582e05.pdf.

Pandya K., (2013), Network structure or topology, *International Journal of Advance Research in Computer Science and Management Studies*, **1**, 22-27.

Parisien M.-A., Barber Q.E., Hirsch K.G., Stockdale C.A., Erni S., Wang X., Arseneault D., Parks S., (2020), Fire deficit increases wildfire risk for many communities in the Canadian boreal forest, Nature Communications, 2121, https://doi.org/10.1038/s41467-020-15961-y

Santanu S., Pinaki P.A., (2013), A study and analysis on computer network topology for data communication, *International Journal of Emerging Technology and Advanced Engineering*, **3**, 522-525.

WTG, (2018), Waspmote Technical Guide, Document version: v8.1 - 11/2018, Libelium Distributed Communications S.L., On line at: http://www.libelium.com/development/waspmote/documentation/waspmote-technical-guide/.

Ya-qiong Z., Yun-rui L., (2016), A routing protocol for wireless sensor networks using K-means and Dijkstra algorithm, *International Journal of Advanced Media and Communication*, **6**, 109-121.

Zhang L., (2013), An energy saving routing algorithm based on Dijkstra in wireless sensor networks, *Journal of Information and Computational Science*, **10**, 109-121.