Performance analysis of a new SDN-based WSN architecture

Slavica Tomovic, Igor Radusinovic, *Member, IEEE*

***Abstract* — Energy efficiency is one of the key requirements in Wireless Sensor Networks (WSNs). In order to optimize energy usage at sensor nodes, this paper proposes a new WSN architecture that relies on concepts of Software Defined Networking (SDN). Since SDN is a relatively new technology, originally envisioned for wired networks, it can't be expected to get immediately adopted in WSN domain, regardless of potential benefits. For this reason, we consider scenario where SDN nodes coexist with traditional sensor nodes, and propose a new routing algorithm for SDN controller that prolongs the WSN lifetime even when small percentage of SDN nodes is deployed.**

***Keywords* — energy efficiency, routing, SDN, WSN.**

I. INTRODUCTION

WIRELESS sensor networks (WSNs) consist of spatially distributed, low-power devices (SNs-sensor nodes) using sensors to measure specific parameters of the environment. Depending on application, they may include a vast number of SNs. The sensor data are transmitted to external gateway either directly, or by ad-hoc network. In ad-hoc scenario, each SN participates in routing by forwarding data for the other nodes. There is a variety of applications in which such networks might find uses, such as agriculture, transportation, industry, healthcare…[1].

Energy is an important resource constraint of SNs because their power sources are usually batteries with limited capacity. On the other hand, one of the key factors that determines the functionality and the accuracy of the sensing applications is WSN lifetime, which may be prolonged by balanced energy consumption among SNs [2]. These unique features must be taken into account while designing routing protocols and algorithms. If the gateway is too far from SNs, direct transmission has to be avoided due to quite extensive energy loss. However, conventional multi-hop communication such as Minimum Transmission Energy (MTE) routing often results in an equally undesirable effect [3]. All MTE SNs act as routers for the other SNs. Because SNs close to the gateway are most engaged in transmission of data, their energy resources rapidly drain. Various routing protocols have been proposed to alleviate the mentioned problems [4]. However, most of the proposed solutions is too complex with questionable potential for practical implementation.

Many inherent problems of WSNs are deeply rooted in the architecture. In ad-hoc networks, each SN participates in decentralized routing and independently determines the next-hop to send the data towards the destination. This prevents global resource optimization and smart traffic management [5]. Thus, we propose using Software Defined Networking (SDN) in WSNs. SDN is technology initially proposed for wired networks, with separated control and data planes [6]. The control plane is logically centralized at SDN controller, that maintains global view of the network, interacts with data plane devices and provides a programming interface for user-written network management applications. Leveraging centralized intelligence of the SDN controller, it is possible to dynamically alter the network behaviour and deploy new applications in real time [6][7].

In this paper, we consider scenario where SNs with SDN capabilities (SDNSNs) are incrementally deployed in a traditional WSN. The key question we are trying to answer is whether it is possible to do effective traffic engineering and prolong the WSN lifetime when all the nodes in the network cannot be controlled by SDN controller. Particularly, we have focused on WSNs using MTE routing protocol, with only small percentage of SDNSNs deployed. In the paper we show that when SDN controller knows positions and capabilities of SNs, WSN lifetime could be significantly increased.

The rest of the paper is structured as follows. In Section II we outline the network and radio models assumed in the analysis, and propose a new routing model. Simulation results and analysis are presented in Section III. Conclusion remarks are given in Section IV.

II. NETWORK MODEL

We assumed the hybrid WSN (Fig. 1) consisted of:

1. regular sensor nodes that run MTE;
2. SDNSNs which run MTE, but in addition could be controlled by SDN controller.
3. SDN controller which is implemented at externally power supplied gateway (e.g. base station) and makes routing decisions for SDNSNs.

Our initial assumption is that no changes are made to regular SNs, i.e. they are completely unaware of the existence of SDN devices in WSN. SDN controller is not able to control regular SNs. Instead, regular SNs run MTE routing protocol. The basic principle of MTE routing is simple: each node sends data packets to the closest node

This work has been supported by the Montenegrin Ministry of Science and the HERIC project through the BIO-ICT Centre of Excellence (Contract No. 01-1001).

S. Tomovic is with the Faculty of Electrical Engineering, University of Montenegro, Dzordza Vasingtona bb., 81000 Podgorica, Montenegro (phone: 382-69-468583; e-mail: slavicat@ac.me).

I. Radusinovic is with the Faculty of Electrical Engineering, University of Montenegro, Dzordza Vasingtona bb., 81000 Podgorica, Montenegro (phone: 382-20-245873; e-mail: igorr@ac.me).

on the way to the base station [8]. However, to make such a forwarding decision SN needs to know all active SNs within its coverage area and their positions. These information are obtained via periodic exchange of "keep-alive" messages between the nodes. SDNSNs have to participate in this exchange, so we assumed that MTE routing daemon is running on them as well. Note that MTE routing logic makes sense only if the SNs can use power control to vary amount of transmit power. Technological advances in radio hardware make this assumption reasonable [9]. The power level for data transmission is determined based on the location of the next-hop neighbor. For regular SNs that is always the neighbor that requires the minimum communication energy, while for SDN node that could be any node within the radio range. Of course, to avoid loops in distributed routing, the next-hop on the route must be less distant from the base station. Note that real location of neighbors may not be of particular importance for the regular SNs because relative distance to them could be approximately estimated based on received signal strength of the "keep-alive" messages.

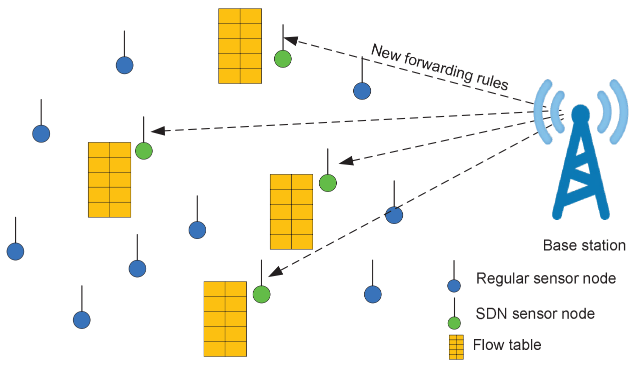

Fig. 1. The proposed WSN environment

The SDNSNs have role similar to OpenFlow switches in wired SDN networks [10]. They perform data forwarding according to the controller's instructions stored in so called flow tables. Each flow table entry is defined by: i) matching rule which describes characteristics of packets belonging to a traffic flow; ii) action which defines the way of processing the flow; and iii) counters which serve for statistical purposes. We assumed flow table format proposed in [11], which is specially tailored for WSN environment. Since we are only interested in routing functionality, in our study the processing action is always forwarding to the next-hop sensor node. SDN-related control communication relies on tree types of messages: beacon packets, packet-in requests and flow-mod responses [11]. The base station periodically broadcasts beacon packets. Beside the header fields, these packets contain an additional byte indicating number of hops required to reach the base station/controller from the transmitting SN. Base station initially sets this byte to zero. Upon receiving a beacon packet, each SN checks whether the incremented byte value is less than the current estimate of the distance to base station (initially ∞). If yes, the current estimate and estimate in the beacon packet are updated. Only if received beacon packet is updated, it will be broadcasted further. Packet-in message is sent by SDNSN to SDN controller when received data packet does not match any of the rules in flow table. It contains header of the data packet, based on which the controller makes routing decisions. To inform SDNSNs on routing decisions, controller generates flow-mod messages, which contain elements of flow entries. For the adopted format of flow table, payload of these messages doesn't exceed 4B if matching is done on one block of packet header bytes [11].

Important assumption made in our study is that SDN controller has knowledge of sensor nodes' positions and capabilities. This is important for two reasons. First of all, to perform traffic engineering controller needs information about the network topology and available energy resources at each node. Then, to calculate feasible routes controller must know which nodes could be manipulated and which are the regular ones. By knowing the geographical position of each sensor node and characteristics such as SDN capability, radio range, energy consumption model and initial energy, the controller can estimate residual energy of nodes and make efficient routing decisions. Node's energy consumption depends on radio characteristics, distance to the next-hop node and amount of data transmitted. We assume that controller knows the first factor in advance. The second factor is calculated based on locations of the node and its next-hop neighbor. SDN controller knows next-hops of SDNSNs. On the other hand, the next-hop of regular node is always the closest node on the way to the base station. Information on number of bits transmitted by SN during some period of time may be derived from flow table counters. Because regular nodes do not have flow tables, we assumed that the base station has flow table with separate entry for each SN. These entries perform matching on the packet source address, so the counters indicate number of bytes originated from each SN which successfully reached the base station. Beside the data generated by them, SNs participate in forwarding of data generated by other nodes in the network. However, SDN controller knows the routes from each SN to the base station that were used during the analyzed time interval. Therefore, by knowing the routes and the flow table counters it can estimate total amount of traffic carried by SN. When SDN controller computes estimate of residual energy for some sensor node, it considers the last estimation, time when the last estimation was computed, the route collection, current and previous state of the corresponding counter in the flow table. Since forwarding rules at SDN nodes could be dynamically changed, computation of residual energy must be triggered each time before a new route is installed in order to keep correct insight into energy consumption. Although SDN controller may not be able to predict when sensor node is "dead" (e.g. it is possible that some transmitted data are lost), estimations of residual energy help in finding the most efficient route.

Algorithm 1 describes routing logic of SDN controller. In the first step controller creates reduced network graph, which contains only links that could be used for the route setup. These are all links originated at SDNSNs (lines 8-9), and links from each regular SN to the nearest neighbor (lines 13-14). Link cost is chosen as a function of residual energy of the source SN and energy needed to transmit and receive packet on the link. Parameters α and β define relative impact of these two factors on total link cost. They can be chosen to find the minimum energy path or the path with nodes having the most energy, or a combination of the above. Once the link costs are determined, Dijkstra algorithm [12] is used to find the least cost route between the source SN and the base station. Then, the controller sends flow-mod messages with new forwarding rules to SDN nodes along the route.

Algorithm 1*: Route calculation*

```
1:# G- reduced network graph used for route calculation
2:# src- source node, BS- base station, nodes- live nodes
3:# Costs - link costs, R - residual energy of source node
4:# EC - energy used to transmit and receive on the link
5: function: find_route(src, BS, nodes)
6:   for node in nodes do:
7:     if node.type is SDN do:
8:       for nb in node.neighbours do:
9:         G.addLink(node,nb)
10:        Costs([node,nb])=EC([node,nb])$^\alpha$/R([node])$^\beta$
11:      end for
12:    else do:
13:      nb=node.MTE_nextHop
14:      G.addLink(node, nb)
15:      Costs([node,nb])=EC([node,nb])$^\alpha$/R([node])$^\beta$
16:    end if
17: end for
18: return leastCostPath(G,src,BS,Costs)
19: end function
```

In our analysis we used the same energy consumption model discussed in [3][9][13], where the transmitter dissipates energy to run the radio electronics and the power amplifier, and receiver only dissipates energy to run the radio electronics. Energy dissipation in other components of the typical SN have been neglected because main objective of this paper was to propose energy-efficient network layer mechanism. Energy consumed for the transfer of a *k* bit message between two SNs separated by a distance of *r* meters is given by following equations:

$$E_T = E_{Tx}k + E_{amp}k \quad (1)$$

$$E_R = E_{Rx}k \quad (2)$$

where E_T denotes the total energy dissipated in the transmitter of the source node and E_R represents the total energy dissipated in the receiver electronics. Parameters E_{Tx} and E_{Rx} are per-bit energy dissipations for transmission and reception respectively. For transmission, additional energy is dissipated to amplify the signal (E_{amp}) according to the distance to the destination. As is the case in [3], we assumed that propagation loss is inversely proportional to r^2 (free-space model) for small distances, while it is inversely proportional to r^4 for long distances (two-ray model). Thus, the overall energy consumed by the radio to transmit a *k*-bit message over distance *r* is calculated as follows:

$$E_T = E_{Tx}k + E_{amp}k = \begin{cases} E_{Tx}k + \varepsilon_{FS}r^2k, \; r \le r_o \\ E_{Tx}k + \varepsilon_{TW}r^4k, \; r \ge r_o \end{cases} \quad (3)$$

where ε_{FS} and ε_{TW} are amplifier parameters for free-space and two-ray propagation models respectively, and r_o is threshold distance given by:

$$r_o = \sqrt{\varepsilon_{FS} / \varepsilon_{TW}} \quad (4)$$

In simulations described in the paper, we used the same set of parameters as in [9]: $E_{Tx} = E_{Rx} = 50nJ/bit$, $\varepsilon_{FS} = 10pJ/bit/m^2$ and $\varepsilon_{TW} = 0.0013pJ/b/m^4$.

III. Simulation results

In order to verify the effectiveness of the proposed solution, we carried out set of simulations in MATLAB. Performance was measured by following metrics: energy dissipation, WSN lifetime, total number of successfully delivered messages and number of nodes that are alive.

In simulations, we used WSN topology with 100 nodes randomly distributed in 200m x 200m area. Location of the base station was chosen to be at least 20 m from the nearest node (x=100, y=220). Each SN was assigned an initial energy of 1J. Following the approach from the literature [3][9][13], we assumed that all SNs are sensing the environment at a fixed rate, and send a 2000-bit data packet to the base station during each time step or "round" of the simulation. Values of α and β parameters in link cost definition were set to 1 and 4 respectively. The influence of random factors such as wireless channel interference was ignored.

The first set of results provides comparison of the traditional WSN using MTE routing protocol, and the hybrid WSN with only 20% of SDN nodes. From the Fig. 2 we can see the number of SNs alive as a function of number of sensing rounds. The results show that the hybrid WSN, for the same initial energy, produces a longer WSN lifetime. In MTE scenario, the first dead node occurs after only 80 rounds, and after 1084 rounds all the nodes are dead. This is because the nodes close to the base station forward the largest amount of data and suffer higher energy losses. These nodes die very fast, causing the increase in energy required by other nodes to reach the base station. On the other hand, in the hybrid WSN the first node died in 166th round. If we define WSN lifetime as number of rounds for which 75% of the SNs remain alive, the proposed solution exceeds the lifetime of MTE network by 57.9%. Number of SNs alive diminishes more slowly because forwarding rules of SDNSNs are dynamically defined by considering estimated values of the remaining energy for each SN. The outcome is more balanced energy consumption among SNs.

To compare quality of service [9], we measured number of data messages received by the base station in hybrid and MTE WSN over the rounds of the network activity (Fig. 3). We can see that the hybrid WSN has a great advantage in data delivery. With only 20% of SDNSNs deployed, it offers improvement by factor of 33.8%. In traditional WSN some nodes drained their batteries quickly, and could not longer transmit or receive the data. Throughout simulations we assumed that each SN is within communication range of each other and the base station. For the shorter radio ranges of SNs, it happens that the nodes in MTE WSN unnecessarily consume energy to transmit and receive data which cannot reach the base station even under ideal wireless channel conditions. For example, when a lot of nodes around base station die, some parts of the network stay disconnected. As MTE routing does not include any centralized control, SNs at the network edge may be completely unaware of this situation. The more distant the base station is from sensor nodes, benefits of MTE routing over the direct transmission are more pronounced. However, due to above-mentioned reasons, the proposed routing model outperforms MTE in each scenario.

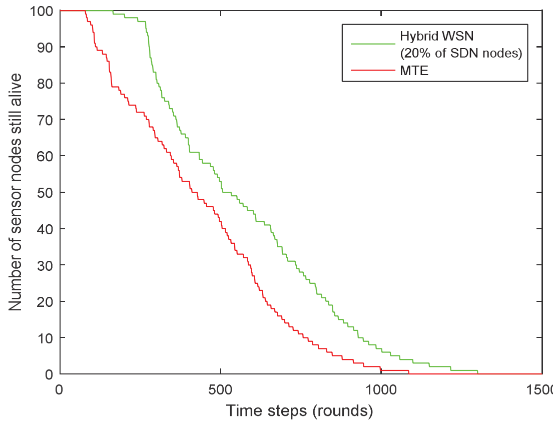

Fig. 2. Lifetime of the hybrid and traditional WSN

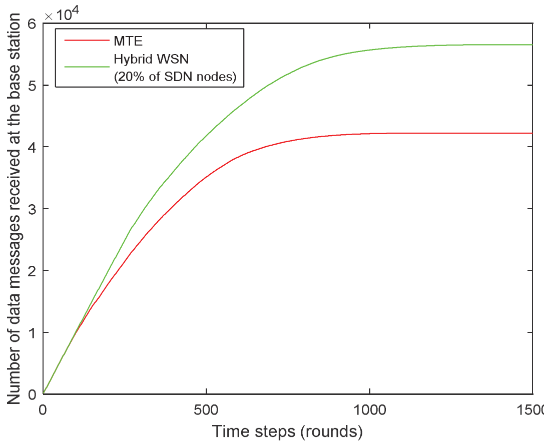

Fig. 3. Total number of data messages received at base station over rounds of operation

The results from Fig. 4 show that the hybrid WSN has a more desirable energy consumption curve. One of the reasons why MTE requires so much energy to transmit the data is that each messages traverses a large number of hops on the way to the base station. On the other side, SDN controller has knowledge of the location and energy resources of all SNs in the network, so it can select routes in more efficient manner.

Finally, we analyzed the impact of number of SDNSNs on the network performance. In set of experiments the percentage of SDN nodes was increased from 0 to 100 by step of 20. As expected, the obtained results showed that WSN lifetime sharply increases with the number of implemented SDN nodes. Due to the lack of the space we can't provide more details in this paper.

IV. Conclusion

In this paper we proposed a routing algorithm for the hybrid WSN model where SDN-enabled sensor nodes coexist with traditional sensor nodes which use MTE routing protocol. The algorithm is based on assumption that SDN controller knows locations and main features of sensor nodes. Through simulations we have shown that if these information are available, deployment of even small percentage of SDN sensor nodes could significantly increase the WSN lifetime and improve overall network performance. In the next phase of our research we will pay more attention on a selection of routing metric, as it could has a deep impact on WSN performance. Also, comparison with state-of-the art routing solutions from literature should be provided.

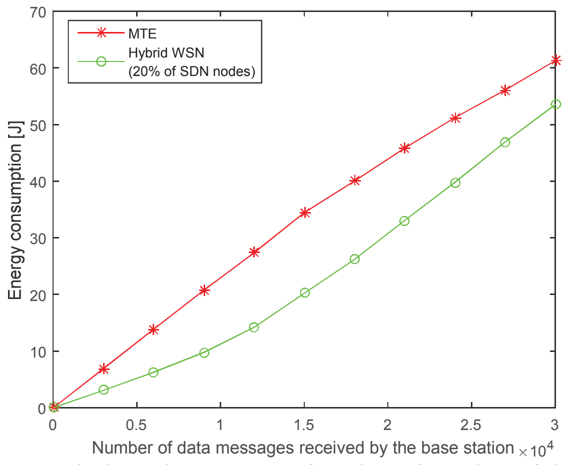

Fig. 4. Dissipated energy as a function of number of data messages received at base station

References

[1] Th. Arampatzis, J. Lygeros, S. Manesis, "A survey on applications of wireless sensors and wireless sensor networks," IEEE Mediterrean conf. on control and automation, pp. 719-724, 2005.

[2] I. Dietrich and F. Dessler, "On the lifetime of wireless sensor networks," *ACM Trans. on sensor networks*, vol. 5, no. 1, pp. 1-38, 2009.

[3] S. D. Muruganathan, D. C. F. Ma, R. I. Bhasin and A. Fapojuwo, "A centralized energy-efficient routing protocol for wireless sensor networks," *IEEE Comm. Magazine*, vol. 43, no. 3, pp. 8-13, 2005.

[4] N. A. Pantazis, S. A. Nikolidakis, and D. D. Vergados, "Energy-efficient routing protocols in wireless sensor networks: a survey," *IEEE Communications Surveys & Tutorials*, vol. 15, no. 2, pp.551-591, 2013.

[5] T. Luo, H.-P. Tan, and T. Quek, "Sensor OpenFlow: enabling software-defined wireless sensor networks," *IEEE Communications letters*, vol. 16, no. 11, 2012.

[6] Open Networking Foundation, "Software Defined Networking: the new norm for networks," Web. White Paper, Retrieved Sept. 2015.

[7] A. Gante, M. Aslan, and A. Matravy, "Smart wireless sensor network management based on software-defined networking," 27th QBSC, pp. 71-74, 2014.

[8] S. Tomovic, M. Radonjic, M. Pejanovic-Djurisic, I. Radusinovic, "Software defined wireless sensor networks", 20th Conference on Information Technologies IT 15, Zabljak, Montenegro, 2015.

[9] W. R. Heinzelman, A. P. Chandrakasan, and H. Balakrishnan, "Energy-efficient communication protocol for wireless microsensor networks," 33rd Hawaii Int'l. Conf. Sys. Sci., Jan. 2000.

[10] OpenFlow switch specification v1.0.0, Retrieved: Sept. 2015, Available at: http://archive.openflow.org/documents/openflow-spec-v1.0.0.pdf

[11] S. Costanzo, L. Galluccio, G. Morabito, and S. Palazzo, "Software defined wireless networks: Unbridling sdns," EWSDN, pp. 1–6, 2012.

[12] E. W. Dijkstra, "A note on two problems in connection with graphs," in Numerische Math, vol. 1, pp. 269-271, 1959.

[13] W. B. Heinzelman, A. P. Chandrakasan and H. Balakrishnan, "An application-specific protocol architecture for wireless microsensor networks," *IEEE Transaction on wireless communications*, vol. 1, no. 4, pp. 660-670, Oct. 2002.