Exploitation of OpenFlow in Wireless Sensor Networks

Arif Mahmud and Rahim Rahmani
E-mail: arif.mahmud@miun.se and rahim.rahmani@miun.se
Dept. of Information Technology and Media, Mid Sweden University, Sundsvall, Sweden

Abstract— **A novel idea presented in this paper is to exploit the OpenFlow technology to address reliability, which is one of the most important factors in sensor networks. The proposed approach unified with flow-sensor and communication with controller. Hardware to programmable hardware, operating system to network operating system, vendor to owner specific, previous hardware defined networking (device monitoring, traffic controlling, and topology definition) turns to be software defined and where everything leads to a programmable and customized networking system. OpenFlow bears the characteristics to control the flow-sensor and monitor sensor traffic flow that also addresses another two important problems robust routing and load balancing. Flow-sensor displayed much better performances in comparison to typical sensors through generating less number of packets with lower simulation time in ideal scenario and even better results are possible for larger networks.**

Keywords: Reliability, robust routing, OpenFlow, Flow-sensor

I. INTRODUCTION

The OpenFlow can split the traffic path into data packet (maintained by underlying router or switch) and control packet (maintained by a controller or control server) which turn the physical device into a simple one from a complicated mode since complex intelligence programs are removed. So, it becomes easier for the researchers to investigate on experimental traffic, protocol and overall network as well and it is very much expected that in near future these research will lead to a significant achievement in network modernization [1].

OpenFlow control packet rate was compared with the same scenario of OLSR where OpenFlow was found to be little bit higher due to higher header definition rate. It is also true that some additional control packet is added in OpenFlow. But from attainable throughput and feasible result viewpoint, the amount of traffic and header definition rate is reasonably low and almost negligible [2].

OpenFlow is one of hot topics now a day and no one has ever tried this protocol on wireless sensor networks as far as our concern. OpenFlow based sensor, named ´flow-sensor´ is claimed to be much more reliable in comparison to typical sensor since data packets, control packets, packet route and even the sensor nodes can be easily observed, regulated and routed whenever required.

The system is fully a centralized system from physical layer viewpoint but a distribution of service (flow visor could be utilized) could be maintained. One central system can monitor, control all sorts of traffics. It can help to achieve better bandwidth, reliability, robust routing, etc. which will lead to a better QoS. Presently the devices are vendor oriented and individual software maintained that addresses a lack of coordination (especially in service). But OpenFlow can start-stop traffic whenever required through a central coordination system.

Today OpenFlow is supported by several major switch/router vendors (especially a set of functions which are common) and can support all sort of layer (2, 3, and 4) headers. Since packets or group of packets are defined as packet-flows, it can provide a single platform for typical TCP/IP and/or experimental cross layer approach. It is also able to integrate the circuit and packet switching technology and these can be treated separately too. Core network also gain noteworthy benefits due to control, management schemes from cost, energy effectiveness and overall network performances of view [3].

We have proposed the following ideas to implement in sensor networks arena for the sake of achieving increased reliability factor:

1. Introduction of the idea flow-sensor instead of typical sensor.
2. Communication among controller, access point (gateway) and flow-sensor using OpenFlow protocol.
3. OpenFlow messages (control packet) flow without interrupting data packet flows between flow-sensors using TCP/IP layered stack.
4. Achieve a robust routing between sensors and access point where route alternation is possible when required in both upstream and downstream directions.

The paper is organized in the following way: Section 2 describes the Motivation and background; Section 3 presents Design and implementation of the proposed model; Section 4 describes the model checking of the new concept; Section 5 presents the performance evaluation and the conclusions with some ideas for future work are provided in section 6.

II. MOTIVATION AND BACKGROUND

Sensing devices are basically organized with a data processing unit and communicational abilities and whose the duty is to measure the specific constraints from the surroundings and to convert those parameters into electrical signals. These kinds of devices can significantly raise the effectiveness of both military and general functions including battle field inspection, safety, and failure management and in some other areas where usual and normal attempt have proved to be very expensive and uncertain. Unstructured sensors also have the capabilities to observe a large amount of environmental conditions such as sound, pressure, temperature, motion etc. These sensor nodes consume resources such as a small bandwidth, communication range, memory, processing power and capability. The processing of signals and communication activities are felt to be the important factors to improve the data transfer capabilities and reliability turns better thereby.

Reliability is one of the most essential factors and difficult as well to achieve for these sensor nodes due to their low resources and communicational abilities. In the current era, the management and maintenance of these unstructured devices with regards to the undetermined and uncertain number of wireless sensor nodes and their establishment in harsh areas has provided huge challenges for scientists. The network is dynamically self-structured and self-constructed and the nodes, themselves, within the network have the ability to establish and maintain mesh connectivity automatically. But from message transfer view point, node to node or hop to hop or end to end reliability are still unsolved issues. Several researchers have worked with reliability grades or partial reliability where they have compromised with the reliability and total reliability was proven to be almost impossible. So, why should not we choose the idea of flow-sensor where traffic can be remotely controlled and sensor nodes can be remote monitored.

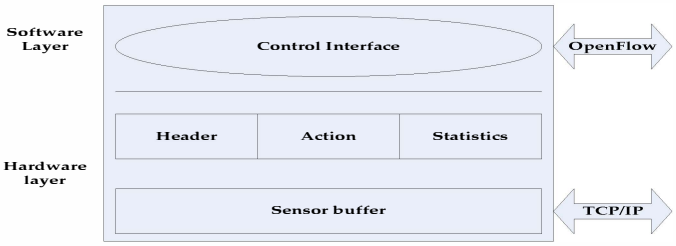

Figure 1. An example of flow-sensor

Messages are transferred among nodes using multi-hopping in a broadcasting scenario especially in upstream directions. Some of the nodes are very important on multi-hopping particularly for the case in which two or more nodes are dependent on it for data transfer. So inactivity of these nodes may cause the network to collapse. In order to succeed robust routing, sensor devices needs to be controlled and data traffic needs to be monitored. A number of algorithms have been proposed to gain robust routing in sensor networking arena. Researchers have tried to solve the problem logically, mathematically and simulate in ideal environment including lots of assumptions. Most of the proposed algorithms are so complex and optimized that their practical implementation in sensor nodes is not viable.

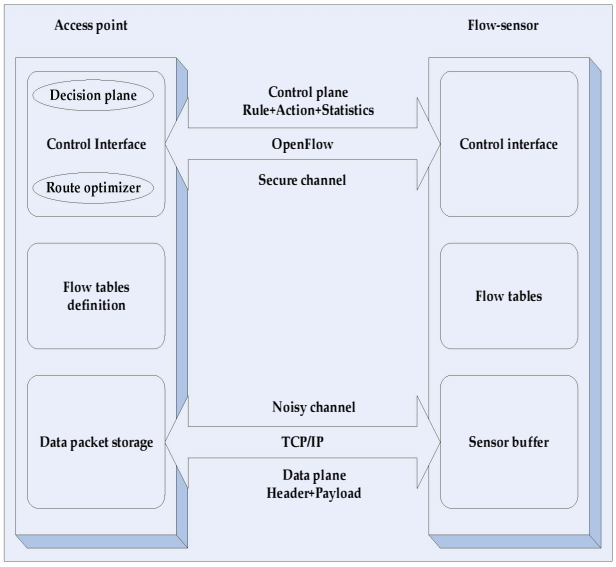

Figure 2. Access point (Gateway) to flow-sensor communication

Flow-sensor is just like a typical sensor associated with a control interface (software layer) and flow tables (hardware layer). A Flow table contains a rule (Header) with source and destination address, action that takes the decision (either to drop or to forward the packets) and a counter that maintains a statistics of control and data packet. Control interface exchanges secure messages (control packet) via OpenFlow and sensor buffer maintain typical TCP/IP with access point (data packet exchange).

OpenFlow can maintain both the flow oriented routing and aggregated routing models. Each flow can be individually maintained and flow data are required to be matched exactly in flow based routing and it can be utilized in campus networks [7]. On the other hand lots of flows could be aggregated and can be used for backbone networks. It's true that OpenFlow is a purely centralized system but a distribution of control could be established through virtualizations [4].

In light of the above observation in the next section, we describe design and implementation criteria.

III. DESIGN AND IMPLEMENTATION

The design criteria can be of three different types while defining a reliability solution for any sensor networks:

- Acknowledgments: Hop to hop or end to end
- Routing: Forwarding or store and forward or store, carry and forward routing
- Type of data: Delay tolerant or real time data.

We will only consider the underlined design factors in our proposed idea.

User can monitor and control total traffic system through a controller (central system). This central system also can be used as the gateway to internet. So an owner can easily maintain and manage all kind of inbound and outbound traffics.

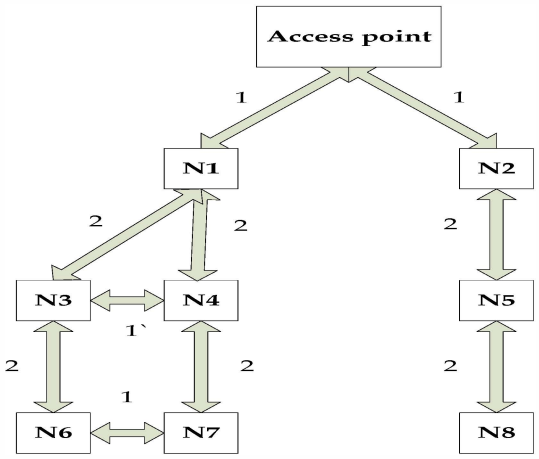

Figure 3. Decision plane

OpenFlow can support different kind of sub-central system such as access point with IP-traffic, access point with non-IP traffic, base station/access point with experimental traffic, access points of heterogeneous wireless nodes/ad-hoc networking, homogeneous wireless nodes/sensor networking [7]. These flow-sensor nodes will carry traffic based on the decision taken by central and sub-central system in both up and downstream direction.

OpenFlow can work with both current/preset infrastructure and experimental topology. Here both the normal traffic and experimental traffic can run together without any obstacles. We will require no complicated algorithm to achieve that where existing algorithms are sufficient enough. Access point will send control packet using OpenFlow protocol whereas data packet will be sent using traditional TCP/IP. Both the packets will reach flow-sensor with different communication protocol but without any interruptions.

Decision plane maintains a tree structure where all routes available in a network associated with route cost will be provided. Route optimizer will be there to decide for the best-optimal route. If any node dies, route can become unusable and decision plane will be updated accordingly. Even for the cases where all the routes with the same cost value, a random one might be chosen by route optimizer. For example, calculation of route and route cost from Access point to N7:

1st route: AP-N1-N4-N7. No. of hops=3. Route cost=1+2+2=5

2nd route: AP-N1-N3-N6-N7. No. of hops=4. Route cost=1+2+2+1=6

3rd route: AP-N1-N3-N4-N7. No. of hops=4. Route cost= 1+2+1+2=6

4th route: AP-N1-N4-N3-N6-N7. No. of hops=5. Route cost=1+2+1+2+1=7

As defined, route optimizer will pick the 1st route. After that any route 2nd or 3rd will be randomly chosen, if 1st route becomes inactive. Different types of data such as block driven, event driven, packet driven can be transferred. Periodic packets can be sent as a block but special packets (event driven packets) can be sent using a shortest link form end point to access points. So, route optimizer can maintain a priority routing too. In order to describe more about the architecture, the illustration of flow-sensor characteristics is given below.

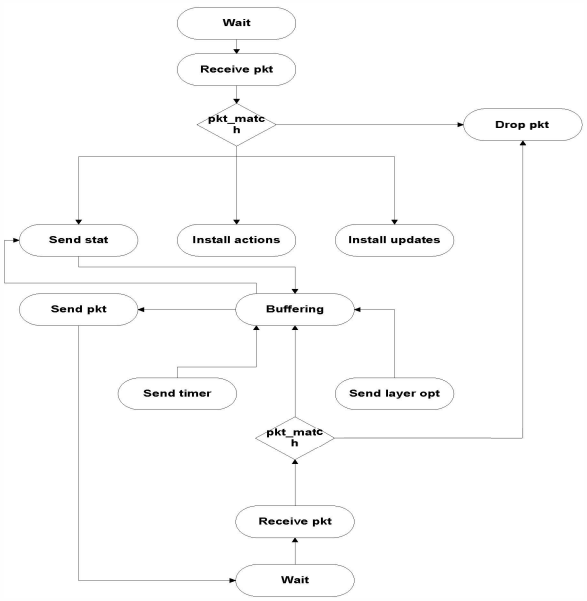

Figure 4. Finite state machine of a flow sensor

Network hierarchy: As a physical layer model a network hierarchy could be maintained. Central, sub-central and nodes can be set up following the top-down approach.

Bandwidth allocation: It is possible to allocate bandwidth for each packet flow through the utilization of hardware abstraction. Flow table can maintain the data rate for each forwarding route.

Topology: Flow-sensor can support both the structured and unstructured topology and we have assumed uncertain topology where sensor nodes are randomly placed in our scenario.

Packet flows: Packet traffic is defined as traffic flows which can contain one or more packets. Flow-sensor is designed to maintain control and data packet flow over secure (OpenFlow) and noisy (TCP/IP) channel respectively.

Flow-sensor processor: Flow-sensor needs to maintain two traffic flows (control and data traffic) and to response accordingly to flow table. That's why processing time will be little bit higher than typical sensors.

Forwarding route: forwarding routes are defined by flow-tables integrated in each flow-sensor. Each flow-sensor should have at least two flow-tables, one for upstream and another for downstream direction [4].

IV. MODEL CHECKING OF THE CONCEPT

The PROMELA and SPIN combination has proved to be a versatile and useful tool in the simulation and verification of software systems [9, 10, 11]. Both have been extensively used in modeling and verifying communication protocols. The paper presents an example of we examine the applicability of SPIN suitability and applicability for a simulation and verification of flow sensor based on OpenFlow. In particular, SPIN shall be applied to simulate exhaustively the correctness of flow sensor and provide verification in Linear Temporal Logic (LTL) with respect to convergence.

A. The model of the Flow Sensor as Finite State Machine

The proposed model of the protocol employs a simple representation of the communicating processes where each process is a finite-state machine. In this case, a pair of communication processes connected by a full-duplex error-free FIFO channel.

As shown in fig.4 control interface of the flow sensor will receive control packet from the access point on a scheduled time provided flow sensor is ready to accept it. As designed packet header will contain at least network layer source and destination address, action rules will set up next step to be followed and statistics will carry detailed packet information. Action rules will be simple and an example rules are following by

- Accept the data packet flow form source-node A.
- Forward the data packet flow to destination-node B.
- Deny any.

Packet body will carry any of the three different type of information such as new action rules, software updates or asking for packet statistics updates. Statistics update of the packet will be sent to access point as data packets from buffer. Data packet will also be received as usual way when medium is ready. Upon receiving data packet, flow table header will be checked to find out the destination address and buffer will release the packet accordingly. Timer and layer optimization are there to support the buffer with maximum time limit to hold the packet and layer optimization values.

Based on design there can be several flow tables where at least two tables needed to be ensured, one for downstream and other for upstream direction. Destination address could be different in these two tables as priority routing needed to be addressed.

B. Packet Flow Algorithm

The following procedure describes the communication for the receive data state in the packet flow algorithm:

```
Proctype flow-sensor (int pkt)
{
#define src_add /*IP address or ID of the
flow-sensor*/
mtype = {stat, action, update};
/*statistics, action rules and soft ware
updates*/
ch ? in_port, dst_add;
 /*channel sends input port number  and
destination address*/
byte state;
xr fromAP;
 /*control packet (receive channel)
      from access point*/
xr fromDS;
/*data packet (receive channel) from
      data storage*/
xs toDS;
/*data packet (send channel) sent to
      data storage*/
xs tobuffer;
/*statistics (send channel) sent to
      data storage*/

if
:: (state ==wait && fromAP? pkt &&
pkt_src_add==src_add && pkt! =Null) → go-
to match_pkt
/*if current state is ready and     packet
received from access    point and authen-
tic address*/
:: (state ==wait && fromDS? pkt &&
pkt_src_add==src_add&& pkt! =Null) → goto
pkt_send
/*if current state is ready and     packet
received from data
storage and authentic address*/
:: else → goto end;
/*drop the packet*/
fi

match_pkt:
if
:: (in_port==1) →tobuffer ! stat; goto
pkt_send;
/*asked for statistics*/
:: (in_port==2) →write (action); /*asked
to install action rules*/
:: (in_port==3) →write (update); /*asked
to install software updates*/
:: else → goto end;
/*drop the packet*/
fi

pkt_send:
read(dst_add);
/*reader the destination address from flow
table*/
read(timer);
/*timer set the maximum time limit  for
buffer to leave the packet*/
read (layer_opt);
/*layer optimizer sends the opti    mized
value to ensure reliability*/
:: toDS ! pkt→write(stat);
/*send the packet to access point and up-
date the statistics*/
}
```

C. The process model of Packet Flow

The complete description and understanding of the model requires the following 4 definitions. Those 4 different states where matching of control and data packets are denoted by π, Π and control packet functions and sending data packet are represented by μ, Γ.

Definition 1: Upon receiving, control packet will be matched based on sensor state, packet source and packet information.

Then the packet will be either dropped or sent to packet function state accordingly.

$\pi|= \neg D \rightarrow CP$ if either $\pi|\neq \neg D$ or $\pi|= CP$

Where receiving of control packet = CP and packet dropping = D.

Definition 2: Data packet will follow the same packet matching algorithm as imposed for control packet. Based on decision data packet will either be dropped or sent to buffering state

$\Pi|= \neg D \rightarrow DP$ if either $\Pi \mid\neq \neg D$ or $\Pi \mid= DP$

Where receiving of data packet = DP.

Definition 3: Control packet will be sent to buffering state and will be forwarded as data packet based on flow table entry if current sensor state is requested. Otherwise it will install action rules or software updates.

$\mu|= F_1 \vee F_2 \vee F_3$ for $\forall$ ip = {1, 2, 3} and we also achieve $\mu|= F_{ip}$

Different control packet functions such as sent to buffer, install actions and install updated are denoted by F_1, F_2, and F_3 respectively along with input port as ip.

Definition 4: Data packet will be sent to data storage or forwarded to next neighbor sensors and packet update will be sent to sensor statistics.

$\Gamma'|= P_1 \upsilon (P_2 \vee P_3)$ iff $\Gamma'|= P_1 \wedge \Gamma'|= (P_2 \vee P_3)$ and we also achieve $\Gamma'|= (P_1 \wedge P_2)$ or $\Gamma'|= (P_1 \wedge P_3)$

Different data packet functions such as sent to buffer, data storage and next nearby sensors are denoted by P_1, P_2 and P_3 respectively.

The following LTL formulas are generated for the definitions:

LTL1: $\Box$ ((CP_receive $\wedge \neg$ Drop) $\rightarrow \Diamond$action ToMatch_pkt)

LTL2: $\Box$ ((DP_receive $\wedge \neg$ Drop) $\rightarrow \Diamond$action ToSend_pkt)

LTL3: $\Box$ (Match_pkt $\rightarrow \Diamond$ (action ToBuffer $\vee$ action ToInstall_Act $\vee$ action ToInstall_Upd))

LTL4: $\Box$ (Send_pkt $\rightarrow \Diamond$ ((action ToDS $\vee$ action ToFwd) $\wedge$ action ToStat))

D. The reference topology model

We can model the common scenario for both typical sensor and flow sensor where network size, sensor density, transmission power are 60*60, $1*10^{-2}$ and -9.03 dBm respectively.

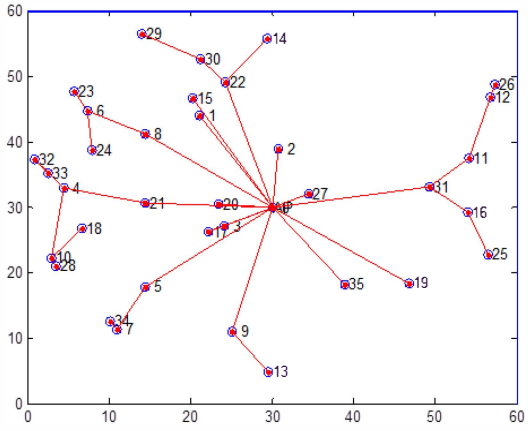

Figure 5. Upstream data packet flow

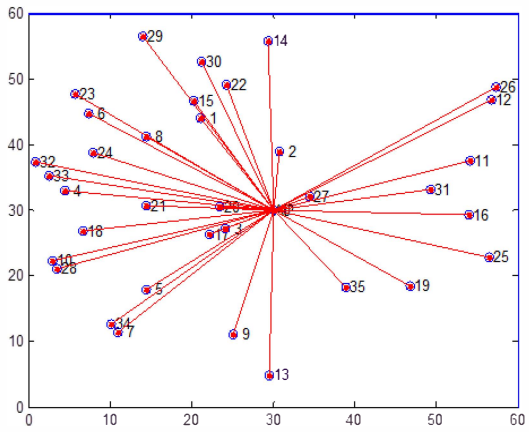

Figure 6. Downstream control packet flow

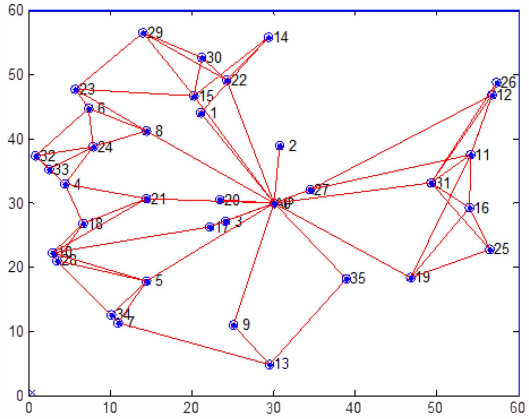

Figure 7. Upstream data packet flow

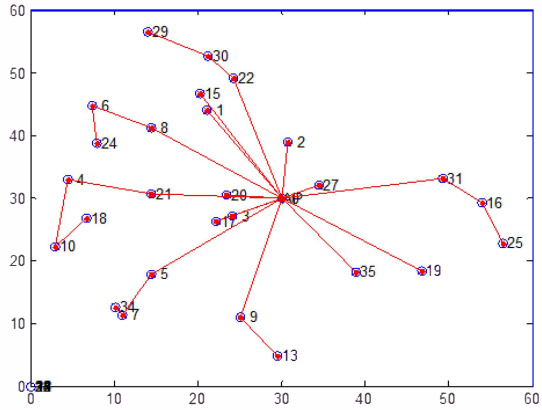

Figure 8. Downstream data packet flow

Fig.5 shows the data packet is flowing in upstream direction where the end nodes are sending data packet to access point via neighboring nodes, route decision taken by flow-tables. This scenario exhibited 56 packets with a simulation time 0.028 sec. In Fig.6 the access point is sending control packets to all the nodes existed in the network. As access point bears the unlimited lifetime, we are not concerned about the amount of packets and to be noted simulation time is negligible.Fig.7 demonstrates the typical sensor is sending data to access point where the end nodes are sending packet to adjacent nodes fol-

lowing the flooding algorithm in a broadcasting scenario. The number of packets gets generated 90 and simulation requires 0.035 sec of time. Fig.8 illustrates the access point is sending data packets to all the required end points via the intermediate nodes following the described tree structure. In this scenario we don't require to send data to 14, 11, 12, 26, 23, 32, 33, 28 nodes. That's why the number of packets gets reduced to 40 and simulation time by 0.027 sec in comparison to upstream direction.

V. PERFOMANCE EVALUATION

The system performance metrics include response time and total number of generated packets for varying topology scenario, sensor density and transmission power.

A. *Simulation assumption and parameters*

- Sensor nodes are sited to be in random but access point is placed in the middle of topology.
- Communication range of sensor nodes and access point are 10 meter and 20 meter respectively.
- We are not concerned about the nodes those are sited out of communication range.
- Receiver sensitivity is -80.5 dBm.
- Total number of packets includes both transmitted and received packets.
- Mobility and reachability of sensor nodes are not considered.
- Topology size, sensor density and transmission power are 100*100, $1*10^{-2}/m^2$ and -10.3 dBm respectively in an ideal scenario.
- Each of the simulation runs for 100 times where we have counted average number of packets with total simulation time.

B. *Simulation results*

We have compared the performance of typical sensor and flow-sensor. We have counted the total number of packets (average) and simulation time (total) based on varying topology sizes, node density and transmission power.

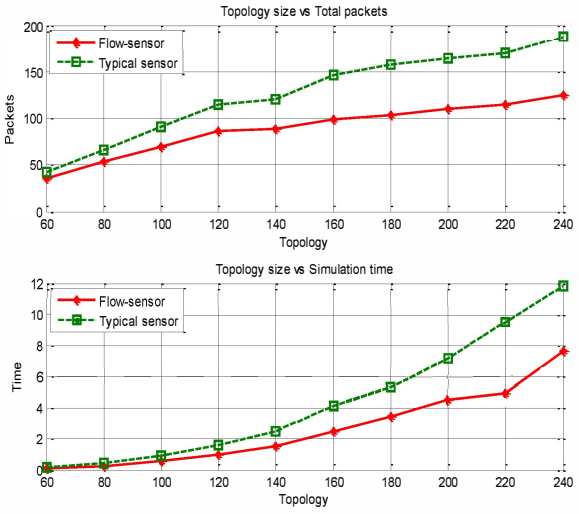

Figure 9. Comparison based on varying topology sizes.

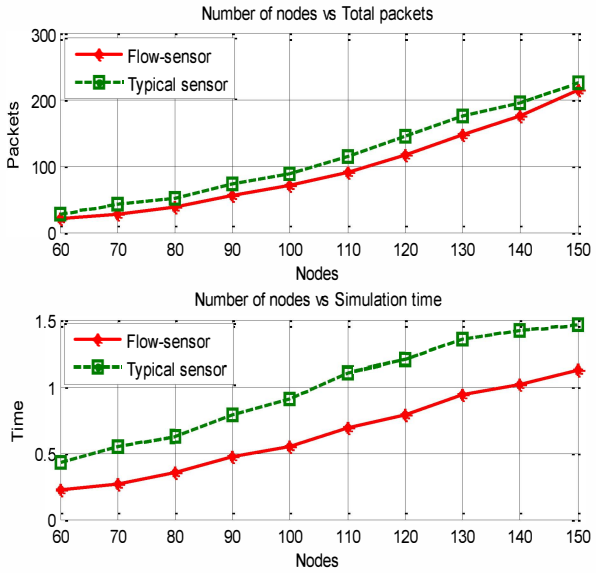

Figure 10. Comparison based on varying number of nodes

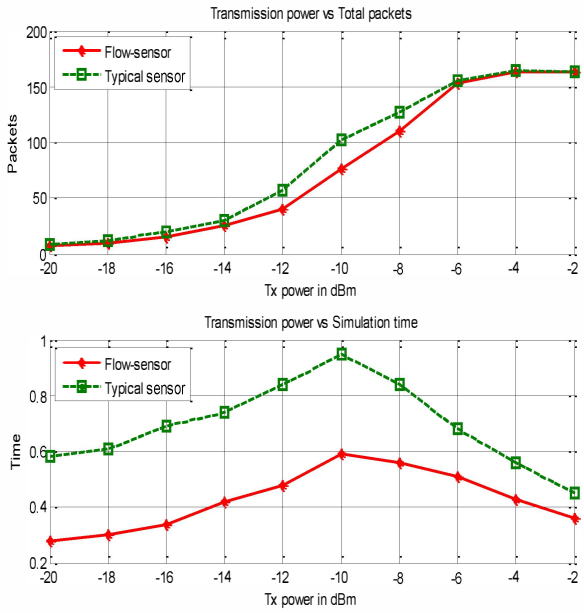

Figure 11. Comparison based on varying transmission power

In fig.9 the total packets and simulation time were measured for varying network sizes. The difference between Flow-sensor and typical sensor increases with the rise of network sizes. For a small network both of them work equally but flow-sensor generated lower amount of packets with a less transmission time in a large scenario. In an ideal scenario flow sensor generated 69.34 packets with a simulation time 0.57 sec whereas typical sensor generated 91.51 packets spending 0.92 sec.

Fig.10 shows the simulation time seems to be increasing but at a slightly higher rate for typical sensor. On the other hand equal number of packets achieved for both small and higher number of nodes but difference is found in medium scale network sizes. As defined in ideal situation flow-sensor

spends 0.55 sec to generate 71.06 packets whereas typical sensor needs 0.91 sec for 88.51 packets.

Fig.11 illustrates the simulation time seems to get equal for higher transmission power but difference continues to rise up to the ideal scenario (-10.3 dBm). Since then they seem to maintain a fixed difference for lower transmission power. On the other hand regarding the total number of packets both of them behave same for higher and lower transmission power. But flow-sensor generated less number of packets (76.42 packets in 0.59 sec) in ideal situation in comparison to typical sensor (101.71 packets in 0.95 sec).

VI. CONCLUSIONS AND FUTURE WORK

The primary goal of this research work in progress has been was to use well-known formal tools for modeling and verification such as PROMELA/SPIN in order to specify, model and verify a successful and packet Flow. The algorithm has been intended for Open Flow for Flow sensor.

Hence evaluate and verify the Packet Flow algorithm and the modelling and verification are important for performance evaluation of Flow Sensor in wireless sensor networks.

The use of PROMELA/SPIN has once again proved the expressive, yet simple and elegant power of, these formalisms for specifying and verifying protocols that are essential for wireless sensor networks. The verification of these algorithms is essential for the credibility and the development of the area of protocol engineering, especially with the increased demand for various services that are based and built on number of complex protocols.

Promising results indicate that flow-sensor displays far better performance where its architecture turns it to be a very much reliable one. From the above three experiments fig.9, 10, 11, we can find that both of them bear the similar performances for higher transmission power and lower network sizes. But for ideal situation and larger networks, our designed flow sensor performed much better than the typical sensors. Most importantly OpenFlow bears the characteristics to control the flow-sensor and monitor sensor traffic flow that addressed another two important problems robust routing and load balancing.

Future work includes studying energy consumption, protocol design and timing, routing initiation and distributed algorithm implementations.

REFERENCES

[1] "OpenFlow Switch Specification", http://www.openflow.org/documents/openflow-spec-v1.1.0.pdf

[2] Peter Dely, Andreas Kassler, and Nico Bayer, "OpenFlow for Wireless Mesh Networks", IEEE International Workshop on Wireless Mesh and Ad Hoc Networks (WiMAN 2011)

[3] Dimitra Simeonidou, Reza Nejabati, and Siamak Azodolmolky, "Enabling the Future Optical Internet with OpenFlow: A Paradigm Shift in Providing Intelligent, Optical Network Services", High Performance Networks Group (HPNG), School of Computer Science and Electronic Engineering, University of Essex, Wivenhoe Park, Colchester, Essex, CO4 3SQ, UK.

[4] R. Sherwood, G. Gibb, K.-King Yap, G. Appenzeller, M. Casado, N. McKeown, and G. Parulkar, "FlowVisor: A Network virtualization Layer", Technical reports, Deutche Telecom and Stanford University, October 2009.

[5] Pavle Vujoševic, "Investigation and validation of the OpenFlow protocol for next generation converged optical networks", Master Thesis Report, July 2011.

[6] Kok-Kiong Yap, Masayoshi Kobayashi, Rob Sherwood, Te-Yuan Huang, Michael Chan, Nikhil Handigol, and Nick McKeown, "OpenRoads: Empowering Research in Mobile Networks", http://conferences.sigcomm.org/sigcomm/2009/posters/sigcomm-pd-2009-final28.pdf

[7] Nick McKeown, Tom Anderson, Hari Balakrishnan, Guru Parulkar, Larry Peterson, Jennifer Rexford, Scott Shenker, and Jonathan Turner "OpenFlow: Enabling Innovation in Campus Networks", March 14, 2008. http://www.openflow.org/documents/openflow-wp-latest.pdf

[8] Magnus Ericsson and Arif Mahmud, "*Dynamic Single Frequency Network in Wireless Multihop Networks – Energy Aware Routing Algorithms with Performance Analysis"*, Dept. Of Information, Technology and Media, Mid Sweden University, "The 10th IEEE Conference on Computer and Information Technology", Bradford, UK; pp 400-406, vol 1; CIT 2010.

[9] Gerard J. Holzmann. " The Model Checker SPIN". IEEE Transaction on software Engineering ,23(5):1-17, May 1997.

[10] Gerard J. Holzmann, SPIN Online Reference. Bell Labs, http://cm.bell-labs.com/cm/cs/what/spin/Man /index.html)August 1997.

[11] Rob Gerth. " Concise Promela Refence". Technical report, Eindhoven University, (http://cm.bell-labs.com/cm/cs/what/spin/Man/Quick.html) June1997.