Simplified OPNET node model for developing load balanced routing protocol in WSN

Karl Benkič, *Member, IEEE,* Marko Malajner, Uroš Pešović, Žarko Čučej, *Member, IEEE,*

***Abstract*—Routing protocol development and evaluation is still very interesting research topic in wireless sensor networks (WSN). There are two options for evaluating the routing protocols: evaluation on the real WSN or evaluation in simulator. The evaluation on the real network takes a lot of time and it can be quite expensive, so, evaluation in the simulator is common researchers selection. When using simulations the selection of a simulator and simulation model is crucial. In the article we presents the simplified OPNET model and an evaluation metrics for the routing protocol evaluation. We describe the the modeling, simulation and evaluation part of experiment done in OPNET and SPaRCSoft.**

***Index Terms*—WSN, modeling, OPNET, SPaRCSoft, routing, topology.**

I. Introduction

When developing new routing algorithm it is crucial to test all it's features by simulations and than evaluate the results with the selected metrics. For simulating rotuting protocol we need:

- Model of a Wireless sensor node.
- Model of a wireless sensor network.
- Simulation tool with adjustable traffic source and capability to collect relevant simulation results and if possible to evaluate them using selected evaluation metrics.

A. Wireless sensor network limitations and definitions

In the simulations we are focused on a static asymmetric WSN with one or more base station (BS). BS node(s) are resourceful powerful devices (computational power, amout of memory…) and all the main topology and routing algorithms are calculated on the BS nodes. Data is collected from single nodes and forwarded to BS (traffic sink) using multi-hop routing approach.

B. Node model

WSN is very specific regarding to other compared wireless networks. The cost of node is relatively low, processing capabilities are low and power consumption should be as low as possible. Node position is not always known and data rates between nodes are only few hundred Kbytes/s.

When selecting or creating a node model it is wise to consider commercial available product. On of the commercial wide spread solutions is ZigBee. The foundation for the ZigBee techology is the IEEE 802.15.4 standard covering the physical and MAC part of the ZigBee. IEEE 802.15.4 offers different types of networking: star, mesh and cluster tree. Higher layers, including routing are defined by the consortium of ZigBee alliance. ZigBee defines three basic device types: Reduces Function Device (RFD; no routing capabilities), Full Function Device (FFD; routing capabilities) and Coordinator (basically FFD) and two basic routing types: AODV, a well known on-demand routing protocol and Motorola's Cluster tree routing. Other proposed and discussed routing algorithms for WSN's like SPIN, Directed diffusion, Rumor were left behind.

When planing the simulation we considered three choices:

- Use of an OPNET native ZigBee models.
- Use of an IPP HURRAY ZigBee OPNET model produced by the IPP Hurray Group [2].
- Create our own, simplified WSN PHY and MAC model.

Native OPNET ZigBee model had no source code available, so, it was out of the question. When we examined the IPP Hurray model we found out that model is rather complex. So we decided that we will create our own simplified model, which will not be based on ZigBee technology but it will support simple 802.11 like communication (CSMA-CA option with the use of RTS-CTS packets if necessary).

C. Simulation tools

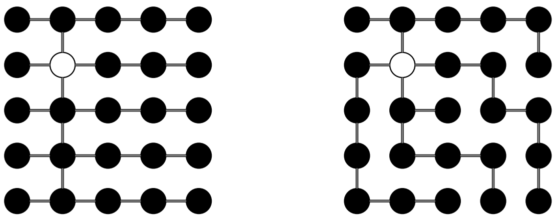
When developing or optimizing routing algorithm, selecting the simulation tool is crucial. There are some network simulators (NS2, OMNET++, OPNET) to chose from but mainly none of them supports WSN natively. After evaluating we selected the OPNET simulation tool for simulating WSN. OPNET supports practically all physical radios, it's modulations and therefore, it is quite easy to implement custom PHY layer for wireless communications. Among other models OPNET supports ZigBee and all three types of devices (RFD, FFD and Coordinator). For PHY and MAC layer there is source code available but for the network and application layer there is only an object code available…OMNET++ on the other hand offers good discrete simulator and 3. party models for the WSN simulations [1] without or with limited support what is expected, because the majority of the simulation models are open source projects done by non profit organizations (faculties) or individuals.

Authors in [3] offered answers which simulator is more appropriate for certain type of simulation (for example: they are recommending the NS-2 simulator for high-precision PHY layers simulation) and how simulator differs in the same setup and scenarios. The same comparison was done in [4] when authors found great differences in MANET simulations

Karl Benkič, Faculty of electrical engineering and computer science Maribor, Slovenija (tel.: +386 2 220 7145, e-mail: karl.benkic@uni-mb.si)

Marko Malajner and Žarko Čučej, Faculty of electrical engineering and computer science Maribor, Slovenija (tel.: +386 2 220 7120, e-mail: {marko.malajner, zarko.cucej}@uni-mb.si)

Uroš Pešović, Technical Faculty Čačak, Srbija (tel.: +381 32 302 220, e-mail: pesovic@yahoo.com)



(a) Shortest Path Algorithm

(b) Load Balanced Algorithm

Figure 1: Shortest Path tree and balanced tree example

executed in OPNET Modeler, NS-2 and GloMoSim, for example: OPNET simulates a very high time delay in MANET compared to NS-2 and GloMoSim [4]. There are many known routing protocols proposed in WSN [5], everyone with some special specific (energy efficient, QoS, low message lateny). Performances was simulated and evaluated by many authors [6], [7], [8], [9], but in general they evaluated different parameters on different setup scenarios.

D. Article organization

In the next section we briefly introduce load balancing routing protocol and it's theoretical background. We define the correlation between shortest path and balanced tree. At the end of the section we define the performance metrics used for evaluating routing performance. The rest of an article is organized as follows. In section III simple OPNET model is introduced, including statistics and modeling phases. In section III-D we show simple result example and in section IV we discuss future plans for out work in progress. Finally, in V we draw the conclusion.

II. Load balanced routing protocol

A. Load balancing in WSN

In asymmetric multi-hop WSN architecture wireless sensor networks it is common that other nodes are rooted to powerful BS with much grater resources available than other nodes (as defined in section I-A. In many early research work regarding routing and QoS in WSN load balancing was ignored [10]. The idea behind load balancing is to achieve evenly distributed packets over different branches of routing tree [10].

There are many solutions known from the 802.11 standard where authors proposed load-balancing routing for wireless networks, but not for the wireless sensor networks [11].

Normally, we are trying to achieve two goals with load balanced routing:

- Minimum number of hops.
- Balancing the tree.

One of the shortest path algorithms can be used to achieve minimum number of hops, but in practice, shortest path algorithm can produce very unbalanced tree. Furthermore, no shortest path algorithm guaranties balanced tree as seen on (Fig. 1). On the other side the algorithms for load balancing can quite good balance the tree but they can not guarantee the shortest path [10].

In [10] Dai at al. proposes Node-Centric load balancing algorithm evaluated by Chebyshev Sum metric. Simulation results performed in there own JAVA simulation tool confirmed that algorithm performs better than shortest path (Dijkstra) and BFS algorithm [10].

B. Performance metrics

When evaluating results of routing protocol performance metrics must be defined. In [8] authors proposed good performance metrics (Fig. 2a) which will be adopted in our work.

Every routing algorithm is evaluated by the performance metrics and compared with others.

III. Simple OPNET model

When planing OPNET model we want to keep it as simple as possible. Currently we are not especially interested on the PHY or MAC layers so we tried to simplified it as possible but still kept all the necessary features and services. Despite the “Energy consumption” metric in performance metrics, no energy aware MAC protocol is implemented, since it impact to the evaluation is not very clear (some MAC protocols implements sleeping sequence which can impact latency and network delay metric). In this case and performance evaluation we prefer low latency over the energy efficiency. The hierarchical assembly in OPNET in done in four main layer:

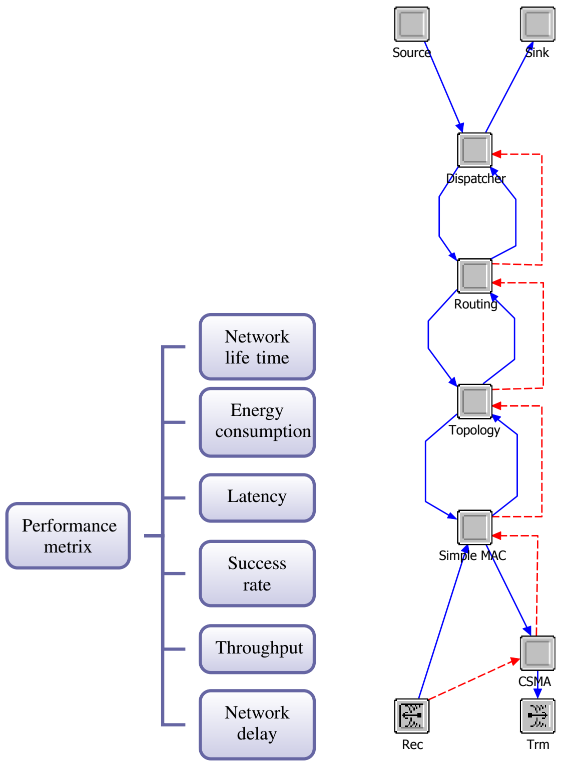
- *Process model* is most deeply nested assembly which represents the C code of and process. In process model we define *states* and *transitions* (Fig. 3) between them.
- *Process* assembles and links C code written in process model and the process model transitions.
- *Model* consists from one or more processes. It defines streams (straight arrow line on (Fig. 2b)), static flows (dashed arrow line on (Fig. 2b)), connectivity with the medium (Rec and Trm processes) etc.
- *Node* is sitting on top of all.

Despite the fact that CSMA-CA mechanism is part of MAC [12] we implemented it as independent model process (Fig. 3). In MAC process we implemented send/recive mechanism and RTS/CTS mechanism to avoid hidden node problem [12]. The *Topology* and *Routing* processes of a model are routing protocol depended. For every protocol we created it's own process. In the dispatcher process is basically the application layer. In the process helps us to establish and setup the network. It creates text files and decides where the next packet from the *Source* process will be sent.

In *Source* process we are using *Simple Source* process model. For the traffic generating size and interarrival time the *Normal distribution* is selected. Selected packet size is 96 bytes with the variance of 32 bytes of data.

A. Using OPNET statistics

OPNET offers integrated functions for the statistics. Using the statistics from native models in OPNET is quite simple but for one want to create he's own statistics things could be a little confusing. OPNET offers statistics implementation on every level of integration. From the node point of view to the process model state. OPNET defines two type of statistics. Global and local one. User can use simple wrapper functions to keep up the statistics. If the statistics are not enough (or



(a) Performance metrics as defined in [9]

(b) Simple node model

Figure 2: Simple nodes and it's OPNET model

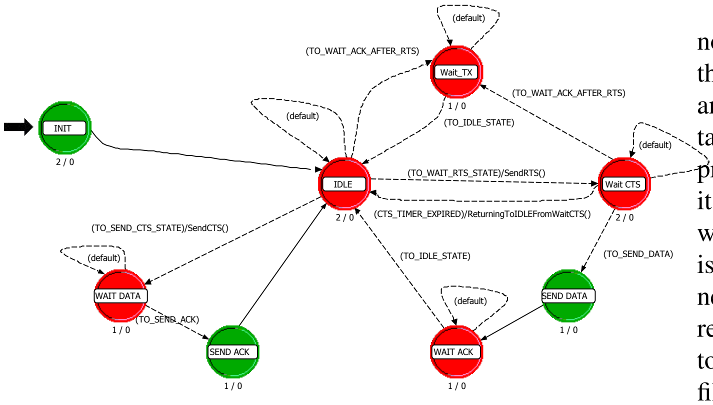


Figure 3: State machine for the MAC Process

for the debugging possibilities) user have the the option to print out the text to console view on the simulation window console. Unfortunately the text from a console can not bi copied so, it is not much of a use in statistics purposes.

Other option natively supported by the OPNET is writing the text to a output file of your choice. Due to the wide range of OPNET APIs and statistics all the statistics can be printed out to a file and then used in another program.

B. Modeling the routing protocol

Studies and researches of routing protocols began few years ago with problem of a WSN model appropriate for routing properties. We form the model in which WSN is modeled as connected graph G=(E,V,W) where W represents model and parameters of a channel [13]. The tool for modeling the WSN was named SPaRCSoft. SPaRCSoft is part of a bigger project AeWSN [14] and it was first introduced in Kendal [15]. In SPaRCSoft tool we are using latest achievements and algorithms known from the graph theory, artificial intelligence (swarm computing) and probabilistic techniques (Ant colony optimization, simulated anealing etc.).

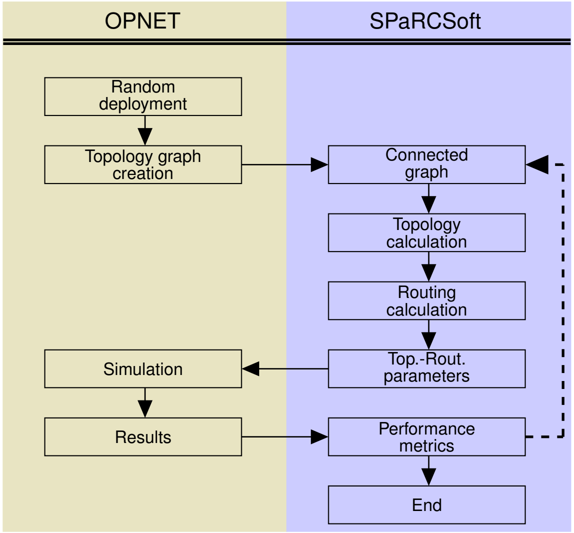


Figure 4: OPNET and SPaRCSoft interaction

C. Putting it all together

For the simulation purposes we will use OPNET random node initial deployment. When deployment is done we save the scenario and that point forward for all the simulations and routing algorithms in session same settings apply (distance, radio parameters such as data rate, modulation type, propagation models...). Propagation model used is static and it's behavior don't change in time. In *initial* network state we first create the *Topology graph* (Fig. 4). Topology graph is a object containing nodes and information about there neighbors. Every node record it's neighbor and neighbor recited power. When initial phase is over text file with the topology graph is written. When topology graph created TXT file is imported into SPaRCSoft program where it is parsed and the connectivity matrix is assembled. If the nodes can communicate the element in matrix differs from 0. Value of the element is the received power from the neighboring node. Due to static propagation model in simulation received powers do not differ from one node to other and vice versa.

When initial topology is transformed into connectivity matrix topology is calculated. SPaRCSoft already supports many topology algorithms, centralized and distributed known in WSN (RNG, MST, Connect, NTC...) [13], [16], [17].

After topology is selected parameters of a routing protocol are calculated (if needed). Currently we are implementing SPIN routing protocol after AODV, Rumor and Direct diffusion already implemented.

When complete parameters are written to a file and ready for transferring in OPNET. In OPNET parameters are read and updated to topology and routing processes.

After parameters are updated simulation is executed and results are written to the statistic file. When done, SPaRCSoft reads the file, converts data to performance metrics and compare it with others. If user is not satisfied with the results

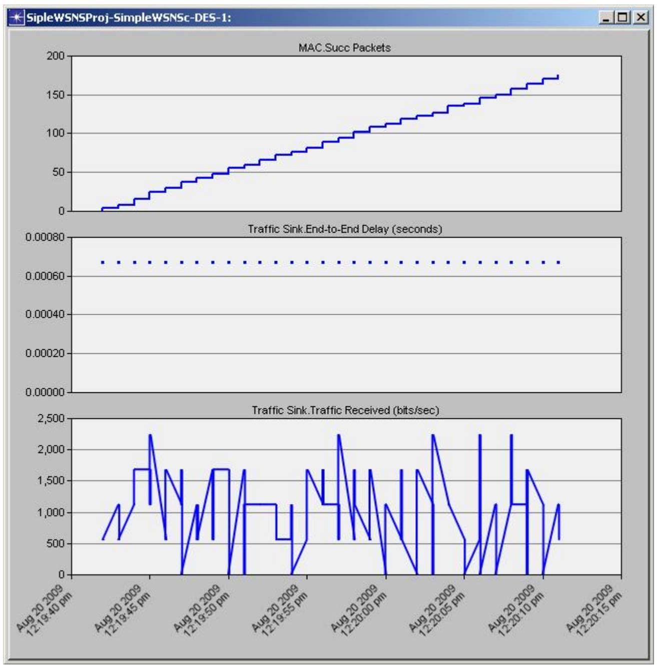


Figure 5: OPNET simple results

it can repeat the sequence from the start of SPaRCSoft part (Fig. 4) trying to change the algorithms properties.

D. Simple results

On the (Fig. 5) simple OPNET result is showed. We made simple scenario with the five active wireless sensor nodes. They were all sending the data to random neighbor once per second with the variance of 0.3s. On the first graph successfully received packets are presented (statistic was made by us) and on the lower two graphs with native OPNET statistics for the traffic sinks are displayed.

IV. Future work

Reader can find that there is one component missing in the node model: Energy or battery process which is for now in development phase. We are facing the lack of data for implementing the process. Currently we are using energy model of a MRF24J40 chip (Microchip IEEE 802.15.4 ZigBee radio), linked on the MAC protocol, meaning, only transmitting and receiving is implemented in the energy consumption statistic. In the future, we want to implement independent power module including CPU calculations and similar, now overlooked energy consumption on a node.

Of course implementation of some routing protocols is missing to, which, we hope will be finished till the paper presentation. Main and hardest reachable goal will be correct implementation of out own routing protocol.

At the end some automatism between OPNET and SPaRCSoft would be great. There are some ideas how to implement wireless sensor nodes in System In The Loop (SITL) in OPNET…

V. Conclusion

In the article simple OPNET node model for developing load balanced routing protocol in WSN is presented. We argumented, describe the model and showed the connection with our previous research work SPaRCSoft. Despite to the early doubt connecting two different programs together was not so hard task.

OPNET it self is very useful and relatively simple to use if one is using native supported model. Developing new, own models is very difficult especially without the support. Editor for code writing is user unfriendly and it “takes you back to the 80’s”. Setting for the interaction with Visual Studio is badly written in documentation and need good programming knowledge to make it work. If one uses OPNET 14.5 version (which we are using) it is necessary to have some dll files for connecting OPNET with the Visual Studio 2008 (natively the VS2005 is supported).

Despite the problems mentioned before we believe we made the right choice using OPNET simulation tool: the results we belie are authentic and realistic and can be compared with others produced by the OPNET.

References

[1] Internet link: http://www.omnetpp.org/models.

[2] Internet link: http://www.hurray.isep.ipp.pt/info/.

[3] Luc Hogie, Pascal Bouvry and Frédéric Guinand: “An Overview of MANETs Simulation”, Electronic Notes in Theoretical Computer Science, Volume 150, Issue 1, 9 March 2006.

[4] Cavin, D., Y. Sasson and A. Schiper: “On the accuracy of manet simulators”, Proceedings of the second ACM international workshop on Principles of mobile computing (2002), pp. 38–43.

[5] Jamal N. Al-Karaki and Ahmed E. Kamal: “Routing Techniques in Wireless Sensor Networks:A Survey”, IEEEWireless Communications, December 2004.

[6] M. Becker, S. Schaust, E. Wittmann: “Performance of Routing Protocols for Real Wireless Sensor Networks”, International Symposium on Performance Evaluation of Computer and Telecommunication Systems (SPECTS 2007), July 2007

[7] A. H. Azni, Madihah Mohd Saudi, Azreen Azman, and Ariff Syah Johari: “Performance Analysis of Routing Protocol for WSN Using Data Centric Approach” World Academy of Science, Engineering and Technology, 2009.

[8] L. Alazzawi and A. Elkateeb: “Performance Evaluation of the WSN Routing Protocols Scalability”, Journal of Computer Systems, Networks, and Communications, 2008.

[9] Abdelghani Bellaachia and Nilkamal Weerasinghe: “Performance analysis of four routing protocols in sensor networks”, International Journal of Ad Hoc and Ubiquitous Computing, May 2008.

[10] Hui Dai, Richard Han.: A node-centric load balancing algorithm for wireless sensor networks, Global Telecommunications Conference, 2003.

[11] P. H. Hsiao, A. Hwang, H. T. Kung, and D. Vlah.: Load-Balancing Routing for Wireless Access Networks, IEEE Infocom, April 2001.

[12] Pablo Brenner: “A technical tutorial on the IEEE 802.11 protocol”, BreezeCom Wireless communications, 2007 (Avaliable at: http://sss-mag.com/pdf/802_11tut.pdf)

[13] Karl Benkič: “Modeling the CDMA wireless sensor networs”, FERI Faculty, Master thesis, April 2008

[14] BENKIČ Karl, MALAJNER Marko, PEULIĆ Aleksandar, ČUČEJ Žarko:“Academic education Wireless Sensor Network: AeWSN”, Proceedings ELMAR-2008, September 2008.

[15] BENKIČ Karl, ČUČEJ Žarko: “Using ant colony optimization for routing algrithm in CDMA wireless sensor networks”, Smart sensing and context : second European conference, EuroSSC 2007, October 2007.

[16] Paolo Santi: “Topology Control inWireless Ad Hoc and Sensor Networks”, ISBN-13: 978-0470094532, September 2005.

[17] Gaurav Srivastava, Paul Boustead, Joe F.Chicharo: “A Comparison of Topology Control Algorithms for Ad-hoc Networks”.