RPL-based Multipath Routing Protocols for Internet of Things on Wireless Sensor Networks

Quan Le, Thu Ngo-Quynh
School of Information and Communication Technology
Hanoi University of Science and Technology
Hanoi, Vietnam
lequana2@gmail.com, thunq@soict.hust.edu.vn

Thomaz Magedanz
Electrical Engineering and Computer Science Faculty
Technical University Berlin
Berlin, Germany
thomaz.magedanz@tu-berlin.de

***Abstract*—In the last few years, Wireless Sensor Network (WSN) emerges and appears as an essential platform for prominent concept of Internet of Things (IoT). Their application ranges from so-called "smart cities", "smart homes" over environmental monitoring. The connectivity in IoT mainly relies on RPL (IPv6 Routing Protocol for Low Power and Lossy Network) – a routing algorithm that constructs and maintains DODAGs (Destination Oriented Directed Acyclic Graph) to transmit data from sensors to root over a single path. However, due to the resource constraints of sensor nodes and the unreliability of wireless links, single-path routing approaches cannot be considered effective techniques to meet the performance demands of various applications. In order to overcome these problems, many individuals and group research focuses on multi-path solutions for RPL routing protocol. In this paper, we propose three multipath schemes based on RPL (Energy Load Balancing-ELB, Fast Local Repair-FLR and theirs combination-ELB-FLR) and integrate them in a modified IPv6 communication stack for IoT. These schemes are implemented in OMNET++ simulator and the experiment outcomes show that our approaches have achieved better energy efficiency, better end-to-end delay, packet delivery rate and network load balance compared to traditional solution of RPL.**

Keywords—Wireless sensor network, Internet of Things, multipath, RPL.

I. INTRODUCTION

Recently, Internet of Things (IoT) becomes a potential future scenario of the applicability and impact of technology in human life. The elements of the IoT comprise not only those devices that are already deeply rooted in the technological world (such as cars or fridges), but also objects foreign to this environment (garments or perishable food), or even living beings (plantations, woods or livestock). By embedding computational capabilities in all kinds of objects and living beings, it will be possible to provide a qualitative and quantitative leap in several sectors [1] [2]: healthcare, logistics, domotics, entertainment, and so on.

IoT extends the concept of Internet from a network of rather homogeneous devices such as computers to network of heterogeneous devices (home appliances, consumer electronics or sensors nodes of Wireless Sensor Networks). The benefits of connecting both WSN and other IoT elements go beyond remote access, as heterogeneous information systems can be able to collaborate and provide common services. This integration is not mere speculation, but a fact supported by several international companies. Since IoT systems consist of sensor nodes that have weak processing power, to suit with new type of network, techniques used Internet are required adjustment or even new techniques, protocols or mechanisms are also suggested.

For enabling the implementation of IoT over WSN or making IPv6 packets to be carried over IEEE 802.4 feasible, IETF Working Group Routing over WSN investigated a routing protocol so-called RPL [3]. RPL is a proactive, distance vector routing protocol, and proposed because none of the existing known protocols, such as AODV [4], OLSR [5] and DYMO [6]…, could meet the requirements of Low power and Lossy Networks (LLN). RPL is designed to be a very flexible protocol in the sense of providing a de-facto standard of few basic mechanisms, which form the smallest common denominator or functionalities on different WSN applications. Variety of extensions has been provided to tailor RPL framework to the particular requirements.

Participants in networks running RPL are connected by constructing a Destination Oriented Directed Acyclic Graph (DODAG), and routed at *root*(s), in order to eliminate cycles. This DODAG is built by control messages containing useful information. From this information, a node is able to select a best candidate of intermediate node (called *preferred parent, default route*) within a set of next-hop neighbors (called *parent set, parent list*) for forwarding data towards root.

It means RPL is designed based on a *single route* strategy. But due to the resource constraints of sensor nodes and the unreliability of wireless links, single-path routing approach cannot be considered effective techniques. In order to cope with the limitations of single-path routing techniques, another type of routing strategy, which is called *multipath routing* has become as a promising technique. Especially in case of RPL, DODAG differs from a tree mainly in that a node can have a set of parent or more than one parent node. This characteristic helps RPL can maintain multipath routing and react to frequent topology changes easily.

In this paper, we concentrate on developing multipath approaches for RPL and propose three schemes: Energy-awareness Load Balancing ELB, Fast Local Repair FLR and a combination of two previous ones ELB-FLR. We also integrate them in a modified IPv6 communication stack of IoT for WSN

978-1-4799-6956-2/14/$31.00 ©2014 IEEE

and implement them in OMNET++ simulator. Simulation result shows that our approaches achieve better energy efficiency, load balance, end-to-end delay and packet delivery rate compared to traditional solution of RPL. The rest of this paper is organized as follows. We briefly introduce RPL in section II. In section III, we describe in details our proposed enhanced protocols. The performance result of all methods is analyzed in section IV. Finally, section V sums up the paper.

II. IPv6 ROUTING PROTOCOL FOR LOW POWER AND LOSSY NETWORK

RPL [3] is a de-facto standard routing protocol entirely defined for IPv6 WSNs. Because WSN does not typically have predefined topologies, for example those imposed point-to-point wires, RPL has to discover links to form a topology at first. RPL constructs and maintains network as a directed acyclic graph (DAG), which can be divided into several Destination-Oriented DAGs (DODAGs). In addition, it can be considered as a logical routing topology over physical network. For achieving requirements of application, it is very important to select a set of parameters (routing metrics) that will affect routing decision for each DODAG, these rules is called *object function* (OF). An OF defines a set of functions to calculate *rank* (the relative distance of one node towards root) as well as to compare neighbors with aspect of rank. The rank may be concerned with node-quality (hop-count, residual energy, etc.) or link-quality (expected transmission number, delay transmission time, etc.). However, it still has various open issues since no specific objective function, DODAG construction and rank computation, nor RPL optimization is deeply studied and well defined. Within scope of this paper, we use Objective Function 0 [7] as default OF.

RPL also specifies a set of new ICMPv6 control messages [8] to construct DODAG and to aid communication between root and sensor nodes. Within scope of this paper, we only consider upward routing, which only requires two kinds of message:

- DODAG Information Object (DIO): is used for constructing a DODAG. The first DIO, emitted by the root, contains four important parameters (*DODAGInstanceID, DODAGVersion*, *DODAGID* and *MinHopRankIncrease)* and root's rank. Upon reception of DIO, the neighbors of root change the rank in this message and continue to forward to others.
- DODAG Information Solicitation (DIS): is used by nodes that want to join a DODAG. This particular node sends DIS to its neighbors in order to solicit DIO.

RPL provides 3 basic mechanism of *network initialization*, *data forwarding* and *local repair*.

- Network Initialization: The DODAG construction is issued firstly by the root with the DODAG information initialization and DIO multicasting. From this forward, whenever a node receives DIO from neighbors with lower rank, it saves this neighbor info to parent list as potential next-hop node, and multicasts DIO to others. The DIO with the same rank is discarded to avoid loop.
- Upward Routing: The topology information is maintained in DODAG, it contains the paths from the children node to the root through different preferred parent. In order to route the traffic upward, RPL only needs the information in this DODAG. The DODAG tells who the preferred parent of the children node is. When a node wants to send a packet to the root, it simply sends the packet to its preferred parent as next-hop in the tree, and the preferred parent then sends the packet to his preferred parent and so on until the packet reaches the root.
- Local Repair: On operation, if a mode suffers from a broken link, RPL provides local repair mechanism to fix it. This node send DIS message to get the new topology information like the first time it joins the network. Its neighbors, when receiving this DIS message, will reset DIO trickle timers for minimizing the interval to next DIO, note that trickle timer is used for controlling the duration period that DIOs are sent.

However, with this design, RPL still has several limitations and in this paper, we point some of them in which we attempt to enhance.

- As mentioned above, in time of local repair, one node should wait for a DIO message after sending DIS message. This phase lasts for long time (its minimum value is minimum interval of trickle timer). In addition, when the neighbors of this node receive the DIS message, their trickle timers [9] are reset and the number of DIO messages sent is increased. As the result, local repair leads to the raise of both delay and consumed energy.
- In order to avoid loop, RPL does not consider neighbors with equal ranks as potential next-hop. In the initialization phase, only neighbors with lower rank are stored in parent list. In reality, the information from neighbors with the same rank is still useful, especially when parent list is empty, a sensor node can select nodes with equal ranks for next-hop transmitting.
- RPL does not well define energy-awareness rank. Normally, hop-quality and link-quality can be used as routing metrics for objective function to calculate rank. In order to achieve energy efficiency, it is necessary to utilize residual energy as routing metric but this parameter is rarely investigated in the literature.
- RPL does not provide switching mechanism among parents. In other words, a node can be preferred parent of different other children node and is used until only path failure or detection of better route. This parent node has intensively to process different requests of its children and soon runs out energy. The consequence is an unbalanced network and reduction of network lifetime. In the other hand, the second-best rank parent is rarely used. To enhance RPL, one should provide a load balancing mechanism to switch between the best parent node and these second-best ones.

III. RPL-BASED MULTIPATH APPROACHES

To overcome these above disadvantages of RPL, we propose different multipath solutions with the main design features:

- Provide more path redundancy and methodology to switch among neighbors in parent list constructed by RPL. It equivalents to enhance RPL with multipath features. In basic RPL, there are many options for one node to determine next hop to forward data, and all are convergent to the root which have lower rank. Our solutions offer node with more options by considering node with the same rank. In addition, out proposed protocols provide a mechanism to deal with these same rank node to avoid cycle.
- In case of path failure, our approaches provide a faster local repair mechanism by exploiting neighbor with the same rank. This solution reduces the appearance of local repair; and the use of sibling node helps to provide more backup route for transmitting data towards root.
- For balancing load, our algorithms switch route among potential parents that have similar level of residual energy. In many others related work to enhance RPL, only one criteria is chosen to calculate rank, e.g. hop-count, expected transmission number… In this paper, we provide a new way to calculate rank by integrate both hop count and residual energy in one valid integer.

In this section, we introduce three new protocols based on RPL: Energy-awareness Load Balancing (ELB), Faster Local Repair (FLR) and a combination of two former protocols (ELB-FLR).

A. *Energy-awareness Load Balancing*

To resolve unbalanced load in RPL, we propose a new protocol with energy-awareness objective functions, associated with a simple load balancing mechanism. The modified protocol is called energy-awareness load balancing (ELB). ELB is more load balanced than RPL because one node chooses preferred parent by considering not only residual energy but also using frequency of parents according to one node. To achieve this goal, we propose a new set of objective functions to calculate rank based on both hop-count and residual energy. We also modify some mechanism of RPL to support energy load balancing.

The first RPL mechanism ELB modified is network initialization. Initially, all sensor nodes are awake and have their transceiver in listen mode in order to receive the network construction message (DIO).

- Step 0: The root initializes DODAG information (*DODAGInstanceID, DODAGVersion*, *DODAGID, MinHopRankIncrease* and *its rank)* as following:

The *MinHopRankIncrease* (or *RankInc*) is set to 100:

$$RankInc = 100 \quad (1)$$

The rank of root equals to this *MinHopRankIncrease*:

$$Rank(root) = RankInc \quad (2)$$

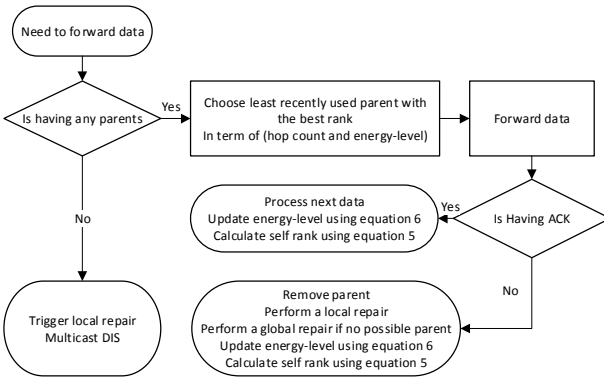

Fig. 1. ELB data forwarding

Then, it encapsulate this information in DIO and multicasts this message to all its neighbors.

- Step 1: Upon reception of DIO, a node records the latest DODAG info (*DODAGInstanceID, DODAGVersion*, *DODAGID* and *MinHopRankIncrease)* and considers rank field of this DIO. If root's rank is higher than or equal to node's rank, this DIO is discarded. In contrast, node puts this root to parent list and uses OF to calculate its rank as following:

From RPL rank increment rule, the *Rank(sender)* is the multiplication of *Hop(sender)* and *RankInc*. Hence, it may extract the hop count *Hop(sender)* of DIO sender:

$$Hop(sender) = \left\lceil \frac{Rank(sender)}{RankInc} \right\rceil \quad (3)$$

The hop count of a node is calculated based on the hop count of DIO sender, by using this formula:

$$Hop(node) = Hop(sender) + 1 \quad (4)$$

In our scheme, we integrate the energy level into the rank value. To avoid ambiguous between hop count and energy level, we select the *RankInc* as 100, so the energy level is never exceed this threshold. We also subtract the *RankInc* by energy level of sender node, so higher energy node has lower rank value, which means more priority when selecting preferred parent.

Then, the node calculates its rank using this formula:

$$Rank(node) = Hop(node) * RankInc - EnergyLevel(node) \quad (5)$$

Where *EnergyLevel(node)* is the energy level of node:

$$EnergyLevel(node) = \frac{ResidualEnergy\ (node)}{MaxmimumEnergy\ (node)} * 100 \quad (6)$$

Due to the depletion of battery charge, whenever the node detects a change in *EnergyLevel(node)*, node re-calculates its rank and updates the DODAG information entry.

- Step 2: Based on the rank calculated in step 1, node changes the rank field of original DIO message, and multicasts this DIO to its own neighbor.
- Step 3: Whenever a sensor node receives a DIO message, it repeats step 1 and step 2.

Fig. 2. FLR package reception

After this initialization phase, every node has a list of parents with lower ranks. Based on rank values in this list, node can select a smaller list of candidates with lowest rank values. These nodes have the same *Hop(node)* and *EnergyLevel(node)*. Among this smaller list, node with the earliest DIO message is selected as best preferred parent for default route to transmit data. The rest list is also queued based on the arriving orders of DIO message and reserved as second best preferred parent for backup route.

One important design feature of ELB is the switching algorithm between best preferred parent (default route) and nodes in the second best preferred parent list (backup route) alternatively in each every packet. Nodes in the rest list of second best preferred parent are also selected alternatively according to this order. By using this switching scheme, the traffic arriving each node of parent list is more balanced. This characteristic leads to better load balance and also network lifetime of ELB compared to RPL (shown in Fig. 1).

B. *Fast Local Repair*

To reduce number of local repairs, we propose a new protocol called Fast Local Repair (FLR) which provides more path redundancy to use in urgent situation. FLR supports sensor node to find more multiple paths compared to RPL and quickly switch to these path on local repair. To achieve this goal, FLR uses a new term called *siblings* defined as follow:

Siblings = neighbors with the same rank

Unlike ELB and RPL, FLR does not only store these better-rank neighbors but also neighbor with the equal rank, in siblings list. In urgent circumstances, node is capable of using siblings to forward its data. Thus, FLR perform a quicker repair when no more parents in parent list are available. And as a result, it reduces a significant amount of overhead (by reducing unnecessary ICMPv6) and also the end-to-end delay when forwarding data (it does not suspend the package streaming to perform local repair). To achieve this goal, FLR

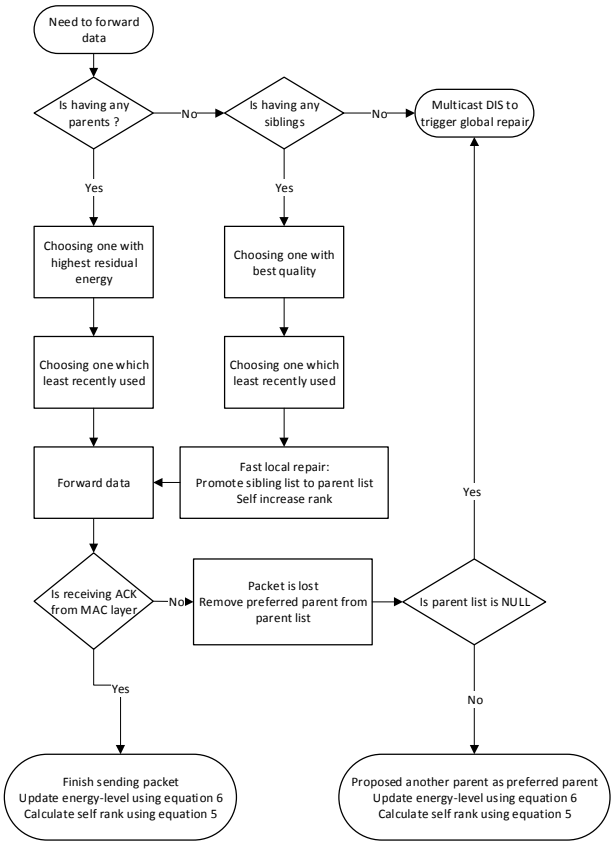

Fig. 3. ELB-FLR data forwarding

modifies only network initialization and local repair phases of RPL while maintaining its data forwarding scheme.

In network initialization phase: FLR saves siblings entry to siblings list as a secondary back-up route (the first one is parent list).

In time of all routes to parent are broken, the faster local repair mechanism of FLR operates as follow:

- Step 1: Node considers the sibling list. If this list is empty, FLR utilizes local repair scheme of RPL. In contrast, FLR implement its own faster local repair mechanism as follow:
- Step 2: Node promotes all its siblings to become its parents in parents-list.
- Step 3: Node clears the siblings-list.
- Step 4: Node increases rank by *MinHopRankIncrease*.

By applying these rules, node with broken path can become children of its old siblings without multicasting DIS and waiting for DIO. This should decrease end-to-end delay and energy usage. Unfortunately, this faster local repair mechanism can cause loop because when two or more siblings simultaneously perform faster local repair schemes and makes themselves as their preferred parents. After faster local repair, they keep forwarding data to each other that consequently causes a loop.

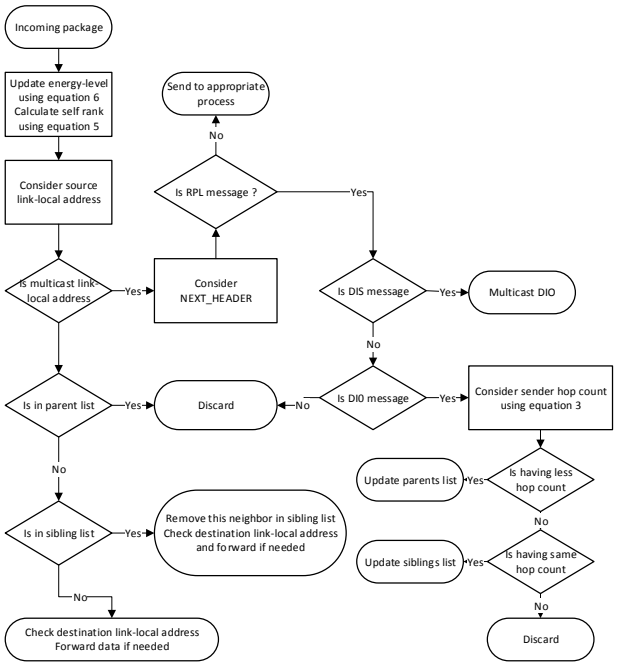

Fig. 4. ELB-FLR package reception

In order to overcome this disadvantage, FLR considers the source kind of a data packet whether it comes from a parent, a sibling or a children node in order to detect and avoid potential loop (shown in Fig. 2)

- First, if the source address of this data packet is not in parent list nor in sibling list, FLR processes this data packet similarly to RPL.
- Secondly, if the packet originates from a siblings (its source address is in the sibling list) the receiver acknowledges that the sender has just triggered a faster local repair mechanism. This sender has just become the receiver children. Then the receiver purges the sibling entry out of sibling list.
- Lastly, node receives a packet from a parent. In this circumstance, FLR clears the sender entry in the parent list as well as does not send back an ACK packet to the sender. By not receiving ACK from receiver, one node is aware that this path is a potential loop and quickly change default route to another to avoid looping.

C. *ELB-FLR*

Finally, we propose a combination of two former methods, called EBL-FLR routing protocol that integrates objective function and load balancing of ELB, faster local repair and loop detection/avoidance of FLR into RPL.In order to merge these two protocols, we propose a new definition of siblings as follow:

Siblings = neighbors with the same hop count

Besides, we have modified the RPL data forwarding and package reception with both ELB and FLR features (shown in Fig. 3 and Fig. 4).

IV. PERFORMANCE EVALUATION

We have implemented the IPv6 communication stacks for WSN on OMNeT++ simulator as follow: UDP in Transport layer, IPv6/RPL/6LoWPAN [10] in Network layer, non-beacon mode CSMA/ContikiMAC [11] in Data-link layer/ framer802.15.4/cc2420 in Physical layer. We also implemented signal propagation model which successfully simulates the hidden and exposed node problems. We also implemented our three proposed protocols along with RPL for evaluating performance of the system.

A number of 144 sensors nodes are uniform deployed in square field of 420mx420m, each node has the transmission range of roughly 47m and is charged with 2AA batteries with capacity of 1000mAh each, and operation voltage of 1.5V. The maximum energy is set at 0.3% capacity to quickly observe the change in energy. The root is located at the border of the region. The non-beacon mode set up is as in 802.15.4 specification, ContikiMAC is as specified in ContikiOS. The simulation scheme is to sampling temperature, humidity and light at the rate of one sample in 65s. Each data payload is 8 bytes. The experiment result is recorded after running in 3600s, with the setup delay of 80s. The simulation stops only when any node completely discharges its battery. The experiment outcome is shown on Table I.

	RPL	ELB	FLR	ELB-FLR
Overhead (%)	46.78	43.57	41.67	40.16
Delay (millisecond)	121.31	118.11	123.95	118.97
PDR (%)	79.84	83.29	81.20	83.98
Half node dead (second)	5456	5456	5458	5457

TABLE I. EXPERIMENT OUTCOMES

A. *Network-layer overhead*

The overhead at network layer is presented in Table I, and is calculated as:

$$Overhead = \frac{Number\ of\ DIO + Number\ of\ DIS}{Total\ number\ of\ packet\ in\ network\ layer} \quad (7)$$

From this table, we realize that the hop-count RPL has the highest overhead, whereas ELB-FLR has the lowest value. ELB has 43.57% overhead, while FLR has 41.67%. The reason FLR-involved protocols have lower overhead: it uses faster local repair mechanism, which reduces the number of DIS and consequently the number of DIO. As a result, the total number of ICMPv6 reduces. In addition, ELB can also decline the overhead compared to RPL. It is caused by the simple load balancing algorithms which switch among the parent, and reduces the traffic to the node with best quality. As a result, the package sent over this link is less likely to be corrupted and so is this route. Last but not least, because ELB-FLR inherits all good mechanism of both ELB and FLR, it reduces a significant amount of overhead than RPL (more than 6%).

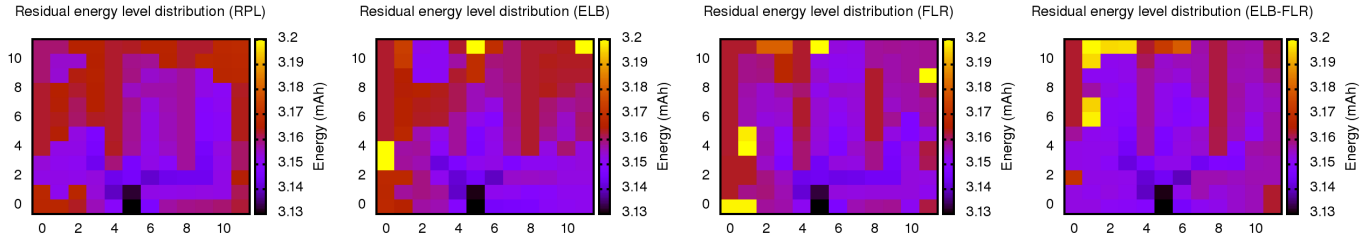

Fig.5 .Residual Energy Distribution

B. *End-to-end Delay*

The end-to-end delay of all packets arrived at the root is presented in Table I. From this table, it is shown that RPL has the highest delay, whereas ELB has lower delay than any of FLR or ELB-FLR. ELB and ELB-FLR slightly reduce the delay by applying the switching mechanism, which decreases the traffic and data forward requests to node with best quality. However, FLR has the faster local repair mechanism but the result is even worse than typical RPL. The main reason of this drawback is the larger number of packet sent by FLR. At network layer, FLR sends 56185 packets while RPL just sends 54516 packets, it means that FLR sends 1669 packets more than RPL does. Because all protocols are executed under a same context, it means that FLR has transferred more packet with further hop-count, which must have more time to transmit and causes higher delay. Another factor is the increasing in transmission hop per packets. In faster local repair mechanism, when a packet sent to a sibling node, the hop-count does not reduce, which will take more time to forward this packet towards root.

C. *Packet delivery rate*

In term of packet delivery rate, RPL also shows the worst performance (79.84%), and ELB, FLR, ELB-FLR show better results, with 83.29%, 81.20% and 83.98% respectively. Similar to delay aspect, FLR is better than RPL but not good as ELB. Fortunately, the combination of ELB and FLR shows higher packet delivery rate. As mentioned before in section III, thanks to energy-awareness OF, simple load balancing mechanism and faster local repair mechanism, ELB and FLR have more effective mechanisms than RPL in term of traffic balancing, path maintaining, etc. And, all leads to a higher success of transferring a packet from an individual node to roots. By combining all these mechanism, ELB-FLR is shown to be the most effective protocol to transfer packet (4%, 1%, 2% higher rate than RPL, ELB and FLR).

D. *Half node dead*

This value is the time it takes to the half of node runs out of energy. From Table I, it is shown that all protocols do not present considerably differences. Our scheme only has 1 - 2 seconds better than RPL. We need to do more research to figure out the reason and improve our solution.

E. *Residual Energy Level Distribution*

Fig. 5 shows the residual energy distribution. It is clearly that three proposed protocols have achieved better performance and have the higher number of node which has more remaining energy. Our proposed schemes provide more balancing mechanism, which have more node with the nearly equivalent energy (shown as same color).

V. CONCLUSION

In this paper, we have briefly described RPL and some research based on its idea. We also points out some of its disadvantages, and propose three multipath protocols based on RPL. The first protocol ELB has lowered overhead, end-to-end delay, and increased packet delivery rate compared to original RPL. The second protocol FLR proposes a faster local repair mechanism, which also decreases overhead, increased packet delivery rate, but has the drawback: rise of end-to-end delay. It is caused by the increasing in number of packet and the hop to transfer packet to root. The final protocol is the combination of two former ones, it inherits the good characteristics of ELB and FLR, and reduces significantly overhead, end-to-end delay while maintaining a more well-balanced network. Nevertheless, the results are good but has limited effect. We need more afford more work to improve our solution.

ACKNOWLEDGEMENT

This research is funded by Vietnam Ministry of Education and Training under grant number B2014-01-83.

REFERENCES

[1] John A. Stankovic, Athony D.Wood, Tian He, Real Applications for Wireless Sensor Networks.

[2] Thang Vu Chien, Hung Nguyen Chan and Thanh Nguyen Huu, *"A comparative study on operating system for Wireless Sensor Networks"*, in proceeding of International Conference of Advanced Computer Science and Information System ICACSIS 2011, December 2011.

[3] T. Winter and P. Thubert, A.Brandt, T. Clausen, J. Hui, R. Kelsey, P. Levis, K. Pister, R. Struik, and J. Vasseur, "RPL: IPv6 Routing Protocol for Low Power and Lossy Networks," in draft-ietf-roll-rpl-19 (work in progress), Mar. 2011

[4] C. Perkins, E. M. Royer and S. Das, "Ad hoc On-demand Distance Vector (AODV) Routing", IETF RFC 3561, 2003.

[5] T. Clausen, P. Jacquet, "Optimized Link State Routing Protocol (OSLR)", IETF RFC 3236, 2003.

[6] C. Perkins, S. Ratliff, J.Dowdell, "Dynamic MANET On-demand (AODVv2) routing", in draft-ietf-manet-aodvv2-03, Feb. 2014

[7] P. Thubert, "RPL Objective Function 0," IETF RFC 6552, Mar. 2012.

[8] A. Conta, S. Deering, and M. Gupta, "Internet Control Message Protocol (ICMPv6) for the Internet Protocol Version 6 (IPv6) Specification," IETF RFC 4443, 2006.

[9] P.Evis, T.Clausen, J.Hui, O. Gnawali, J. Ko, "The trickle algorithm", March 2011

[10] G. Montenegro, N. Kushalnagar, J. Hui, D.Culler, "Transmission of IPv6 Packets over IEEE 802.15.4 Networks", September 2007

[11] A. Dunkels, "The ContikiMAC Radio Duty Cycling protocol", December 2011