Energy Impact Analysis on DTN Routing Protocols

Denis Rodrigues-Silva
Centro ALGORITMI
Universidade do Minho
Braga, Portugal
denis.inf@gmail.com

Antonio Costa
Centro ALGORITMI
Universidade do Minho
Braga, Portugal
costa@di.uminho.pt

Joaquim Macedo
Centro ALGORITMI
Universidade do Minho
Braga, Portugal
macedo@di.uminho.pt

ABSTRACT

In this paper, a mechanism designed and implemented on a simulator module is used to study the energy consumption of a network node in a DTN. The mechanism accounts energy consumption in transmission and reception of a message and also in neighboring nodes discovery. These are routine operations performed by nodes as cell phones or PDAs that traditionally integrate the Delay Tolerant Networks. This module includes a battery recharging option, which makes the simulation environment more realistic. The energy module enables the study of the energy consumption impact on the routing protocols used by Delay Tolerant Networks.

The experimental results show that, when energy consumption is considered, the optimal operating parameters of the protocols change substantially. As the search for neighboring nodes is mainly responsible for the energy consumption of the nodes, the design of new protocols whose strategy is based only on the reduction of messages exchanged may have little impact in terms of actual network performance.

Categories and Subject Descriptors

C.2.1 [**Computer-Communication Networks**]: Network Architecture and Design—*Store and forward networks*; C.2.2 [**Computer-Communication Networks**]: Network Protocols—*Routing protocols*

Keywords

Delay-Tolerant Networks, Energy Awareness, Routing Protocols

1. INTRODUCTION

Despite the development of the Internet, both in terms of technology and coverage, there are still remote areas and scenarios where connectivity is very difficult to achieve. Environmental, physical, economic and security reasons can also make the use of traditional network architectures not feasible. In such situations where there are long periods of disruption, long delays and lack of connectivity between different network nodes, one possible solution is the use of DTNs. Other terminology used to name these networks are Opportunistic Networks or Challenged Networks.

The end-to-end connectivity between the source and destination nodes is not guaranteed in DTNs. The nodes or computer systems that are connected by unidirectional links can lose the ability to connect at any time. Factors such as mobility of nodes, link failures, discharged batteries, among others, may compromise the connectivity of nodes in the network [5]. Once a node is disconnected, one can not predict when it will be available again. Because the DTNs depend on the availability of its nodes to perform routing from source to a destination node, it is necessary to maximize the availability of the nodes in order to increase routing success and message delivery rate.

According to [2] when a connection is established, a given source node has a opportunity to forward its messages to a destination node. In DTNs, this opportunity is called contact. On some occasions, more than one contact may be available between a pair of nodes. For example, a link with a higher cost and higher performance and. a link with a lower cost and lower performance. Both links can be simultaneously available to communicate with a common destination. As the DTNs rely heavily on the availability of their nodes to perform routing of messages and such availability is never guaranteed, what happens when is not possible to establish a direct contact between the source and the destination nodes of a given message? Simply discard the message? Considering the DTN operation scenarios this can be a common situation. To overcome this problem, one solution is to store message into node persistent storage until it can be delivered to the recipient or to an intermediate node toward destination. Moreover, the opportunistic routing is another factor that contributes to the successful delivery messages, taking into account the mobility of network nodes [5, 7].

By analyzing the scenarios for which the DTNs are designed, one can see that the mobility of nodes has a large influence on the amount and quality of contact opportunities. So, the message routing and nodes mobility have a determinant impact on the energy consumption. Furthermore, at DTN typical environments the access to a external power supply is very limited. Therefore, in the design of DTN routing algorithms energy consumption must be carefully considered as an important evaluation measure.

In order to analyze the impact of energy on the performance of DTNs for different routing protocols proposed for these networks, a mechanism that accounts for energy con-

ExtremeCom '12, March 10-14, 2012, Zürich, Switzerland.

sumption was designed and implemented for *The ONE* (Opportunistic Networking Environment) simulator [4, 6]. The energy module receives as parameters the amount of energy expend in search for other nodes, transmission and reception of messages. Typically in units per second. This means that for each operation, the total energy spent is a function of its duration. Furthermore, the node may recharge its battery.

During the simulation, the node has its battery charge reduced with each operation performed. Before starting the simulation, the user must define the battery capacity, and how much energy is consumed per second in different operations. Furthermore, there is the need to define the time interval between battery recharges. This means that the simulation can be considered for long term scenarios, where a node whose battery got discharged can return to the normal network operation after a recharge.

The designed mechanism is compatible with all routing protocols implemented in *The ONE* simulator. This feature enabled us to easily assess the energy behavior of all routing protocols available for this simulator. It will also allow to make energy evaluations for any new protocol implemented in the same environment. Furthermore, internal logic of designed protocols can make decisions to save power, like turning on and off communications interfaces.

The remainder of this paper is organized as follows. In Section 2, there is a brief description of the simulator where the mechanism of energy has been implemented. In Section 3, it is provided a complete description of the energy mechanism, changes performed in the original simulator and how to use the proposed energy module. The following Sections (4 and 5) are the evaluation results and related work, respectively. Finally, section 6 presents major conclusions and possibilities for future work.

2. THE *ONE* SIMULATOR

The ONE [4] is a discrete event simulation platform, based on agents and implemented in Java. The simulator has several models of movement that can vary from synthetic models to real movement traces. In node connectivity, *The ONE* has the notion of location, communication range and throughput of the connections. The event generators create unicast messages, always with a source and a destination. Finally, message routing is implemented by routing modules, with responsibility for deciding which messages are routed through the available contacts. Figure 1 shows simulator modules interaction.

3. PROPOSED ENERGY MECHANISM

This section describes the proposed energy mechanism, the needed changes in the simulator, and how to use it on a given simulation scenario. The mechanism aims to enable the analysis of the impact of energy consumption in the performance of DTNs. Moreover, it can also be used in the design of energy aware routing protocols.

The simulator has a basic module of battery and energy consumption implemented as a variant of the Epidemic protocol. Unfortunately, the module is embedded in the protocol implementation EnergyAwareRouter and can not be used in other protocols. Another limitation is not to consider energy spent during message reception. Furthermore, the battery recharge option is unavailable. So, the model is not usable for a energy consumption impact analysis.

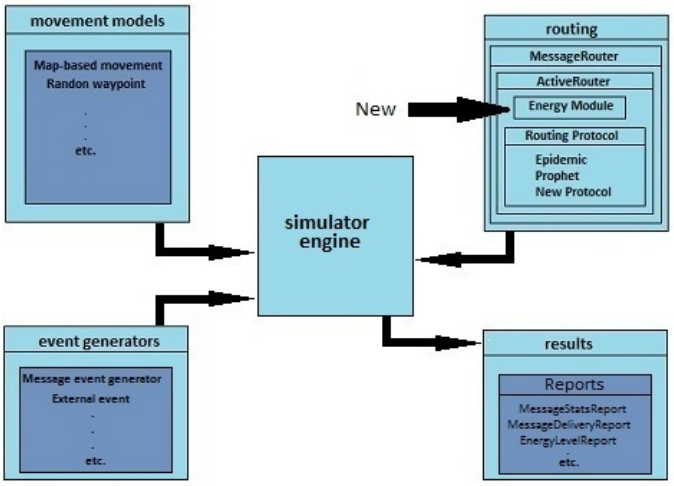

Figure 1: Interaction of *The ONE* modules

Given the situation, the decision was to design and implement a new module for the latest version of *The One*: version 1.4.1 distributed under license GPLv3.

3.1 Energy Module

The energy module introduces the concept of energy consumption in the nodes. In this work, it is assumed that each node has five states:

Off: In this state the node has the network interface off. This state can be reached by lack of energy (battery discharged) or by decision of the internal protocol logic to stay a while without spending energy. In this state, the node can't establish connections with other nodes.

Inactive: In this state the node has its network interface sleeping. However, it can be detected by other nodes for subsequent contacts and exchange of messages. The amount of energy spent per unit of time is very reduced. Here, this amount is considered negligible.

Scan: In this state the node spends energy in order to detect the presence of neighbors. nce the contact is established, the node can transmit and receive messages. Each node, switches periodically to this state, according to the *scanInterval* parameter.

Transmission: The node is sending messages.

Reception: The node is receiving messages.

According to the state, the mechanism uses the method *checkEnergy* to perform the operations that govern the control of energy. When the node is inactive, the energy required is zero. When the node is transmitting or receiving, the total time spent in that state is multiplied by the predefined amount of energy spent per unit of time in the corresponding operation. The spent energy by unit of time for each operation is supplied as parameter for each class of nodes in the configuration of the simulator. That should also happen when the node is in scan state. However, due to a current limitation of *The One*, the scan operation occurs instantaneously, at a given simulation time. The scan duration is not considered. Therefore, the *ScanEnergy* parameter is the total amount of energy expended in each scan operation. The method used to discount the spent energy is *reduceEnergy*.

As an example lets suppose that the amount of energy spent for message transmission is 0,9 units per second and the node battery charge is 500 units. So, if the node is sending messages during 10 seconds, the battery charge becomes 491. To actually perform the reduction of the energy level, the method *reduceEnergy* is invoked with elapsed time and the amount of energy spent by unit of time as parameters. The same approach is used for message reception.

The method *checkEnergy*, invoked from times to times, verifies when is the time for recharge the battery. It returns true if this option is activated in the configuration and if the time value specified in the recharge time interval parameter was already elapsed. Since the simulation time is measured in seconds, the recharge time parameter must also be defined using seconds. For instance, if the recharge interval is one day, its value should be set as 86400 seconds. When the time to recharge the node battery is reached, the method *increaseEnergy* is invoked to actually increase the energy level.

In order to avoid the recharge of all batteries at once, or all at very regular time intervals, which is a situation very unlikely to happen in a realistic scenario, another configuration parameter called *rechargeEnergyRandom* was added. The value is a time interval in which each node performs its recharge at a random moment. Lets suppose for instance that the battery recharge occurs once every day. In this case, the parameter *rechargeEnergy* should be set to the value 86400. To avoid all nodes to recharge just after 86400 seconds, the parameter *rechargeEnergyRandom* could be set to (0, 14400) seconds, for example. Thus, each node must generate a random value between 0 and 14400 seconds, forcing each node battery to be recharged at a different time.

The simulator allows different settings for different groups of nodes. For this reason, different groups can have distinct values for each parameter. For instance, each group of nodes can consume different amounts of energy for the various operations and also have different values of battery charge. It is important to note that having exhausted its energy, the node has its interfaces off and it is unable to connect to other nodes. So, it is not possible for this node to exchange messages with other nodes that still have battery. To ensure that premise, when the method *reduceEnergy* is executed and energy values updated, the current value of the energy is checked. If is equal or less than zero, it is invoked the method *comBus.updateProperty*. This method with parameters (NetworkInterface.RANGE_ID, 0.0) turns off the node interfaces. At battery recharge, the same method is invoked with parametrs (NetworkInterface.RANGE_ID, 1.0) to turn on the node interfaces. Note that this method works like a switch. The values (RANGE_ID, 0.0) and (RANGE_ID, 1.0) do not mean the range or scope of the interface. In practice when the parameters (RANGE_ID, 0.0) are used the range of the interface is set to zero, meaning that the power is off. When the parameters used are (RANGE_ID, 1.0) the interface assumes its initial value. For Bluetooth interface the range value is 10 meters.

The proposed energy module is fully compatible with the addition of new routing protocols to the simulator, without the need for any change in the software code. So, for the design and implementation of a new energy aware routing protocol, the user must be only concerned with the internal logic of the protocol. The mechanism only requires the preconfiguration of all energy related parameters. Furthermore, the new protocol may turn off and on the network interfaces at given instants, in order to save energy. To provide this feature, the module offers the *interfaceOn* and *interfaceOff* methods, which turn on and off the radio interfaces.

Table 1: General Parameters of Simulation

Parameters	Settings
Simulation Time	14 days
Number of Nodes	51
Generation Rate of Messages	1/node/ each 25-30 seconds
Size of Messages	0,5-1MB
Routing Protocols	Epidemic Prophet TFT

In order to have the energy module running for the routing protocols already implemented for the ONE simulator, it was included additional logic into the class ActiveRouter, of the Routing package as shown in Figure 1. This strategy also allows the module usage by any new protocol. To get statistics on the energy level of nodes, the report *EnergyLevelReport* associated with EnergyAwareRouter protocol was used as a starting point. The link to it was changed from the class EnergyAwareRouter to the class ActiveRouter. As all routing protocols are derived from ActiveRouter, the report became automatically available to any routing protocol.

4. EXPERIMENTAL RESULTS

In this section, the operation of the energy mechanism is evaluated. The evaluation was carried in an example scenario by analyzing the energy impact into the performance of two chosen routing protocols: Epidemic [11] and Prophet TFT [9]. The latter is based on the Prophet protocol [7] and uses pair-wise tit-for-tat (TFT) as a practical incentive mechanism for DTNs. Both Epidemic and Prophet are reference routing protocols for DTNs.

For this analysis, several simulations were performed in *The ONE*. The goal is a comparative evaluation, using the amount of messages delivered as metric. The general parameters of the simulation are presented in Table 1.

4.1 Simulation Scenario and Definitions

The DTN network used for evaluation consists of two groups of pedestrians (15 elements each), a group of cars (15 elements) and three groups of trains with two trains each (one node in the train). The sum of all groups equals 51 nodes. The used scenario corresponds to the map of the city of Helsinki, in Finland, which includes streets and crosswalks.

All nodes were considered with the same equipment type, although in the case of trains that option is a bit forced. The equipment used is a mobile phone with Bluetooth interface, communication range of 10 meters, transmission speed of 250Kbps and storage capacity of 5MB. This phone has a battery capacity of 4,8 joules. As the focus of work is the analysis of energy consumption, it was assumed that during the simulation, the battery will be discharging as nodes carry out operations like neighbor discovery, sending and receiving messages.

Each pedestrian group uses the model Shortest Path Map-Based Movement. In this model, the node walks always in the available map roads by choosing a random point and

using the shortest route to this destination from its current location. The used speed can vary between 0.5 to 1.5 meters per second. Cars are forced to follow only the roads, traveling at a speed between 10 and 50 km / h. They also follow the same pattern of movement of pedestrians. For simulation purposes, it is assumed that each car is driven by a person who has the same mobile phone model of the pedestrians. The three groups of trains use the Routed Map-Based Movement model. In this model, each node moves always through the same route. Each group has a different route and stops between ten to thirty seconds at certain locations on the map. Their velocities vary between 25 and 36 km / h. As the simulator have the limitation that people can not leave or enter the train, it was assumed that train drivers have the same mobile phone model of the other groups.

As we are now interested in analyzing the energy consumption in Bluetooth devices, all nodes use the same equipment similar to a smartphone with Bluetooth interface. To try to unlink the test results to a specific scenario, the scenario used was provided with very heterogeneous groups of nodes with different speeds and movements.

Table 2: Energy Parameters

Parameters	Settings
Battery Capacity	4,8 joules
ScanEnergy	0,061 mW/s (usually for 15s)
TransmitEnergy	0,08 mW/s
ReceiveEnergy	0,08 mW/s

For energy configuration settings, a mobile phone Nokia E52 with operating system Symbian S60 3rd edition, was used as a reference. The values of energy consumption were obtained using the Nokia Energy Profiler application available here [8]. This application is capable of measuring the average current energy consumption for a given operation on the device. The first step consisted of setting the display brightness to the minimum value and disabling all phone functions except the Bluetooth interface. Then, it was measured the average energy consumption by doing operations such as neighbor discovery, send and receive files via bluetooth. The achieved results were used as parameters of energy values for the simulation (see Table 2). As noted in this table, the scan process consumes a little more than ten times the amount of energy consumed in sending or receiving messages. in fact, according to tests made using the tool Nokia Energy Profiler, the device remains in scan mode for 15 seconds or until it encounters a neighbor node. Unfortunately the simulator performs this operation instantaneously. Therefore, the amount of energy spent in the scan (0.061 mW / s) is multiplied by 15 although operation is carried out only once.

4.2 Discussion

In this section we discuss the results of an exhaustive battery of tests to analyze the impact of energy consumption in network performance. It was assumed that the performance is measured relative to the number of messages delivered to final destination, since this is the main goal of routing protocols. In all tests Epidemic and Prophet TFT protocols were compared. Both use the technique of flooding however, while the Epidemic floods the network with copies of the messages the extent to which it meets other nodes, the Prophet uses

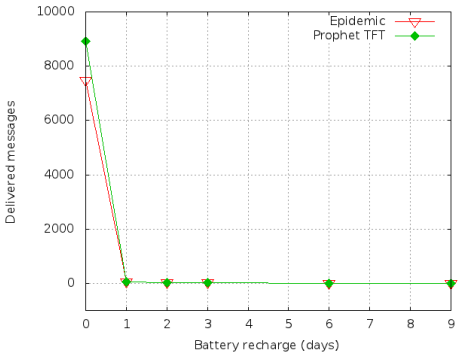

Figure 2: Scan Interval = zero seconds

a history of encounters to leave copies of messages only on nodes that are more likely to deliver the message. In addition, the Prophet TFT differs from the original Prophet because it has an incentive mechanism for routing.

In the first test the energy values were defined according to Table 2. Moreover, it was assumed that the scan interval is zero, ie nodes are always looking for other nodes. The test was conducted in order to evaluate the performance of the two protocols varying the recharge time of batteries over 14 days of simulation. In Figure2, on axis of abscissa, zero indicates that the batteries are always charged. The remaining values represent the interval between two charges, in days. According to Figure 2, when the nodes' batteries are always fully charged, the Prophet TFT takes a considerable advantage over the Epidemic. When charging occurs at an interval of one day, this difference is greatly reduced and the number of messages delivered for both protocols is close to zero. The situation remains very similar for the other recharge intervals. This shows that the two routing strategies do not perform well when there is energy expenditure. Moreover, the gain of the Prophet TFT when no energy is comsumed is not the same when there is consumption.

To try to understand this behavior, we assessed the operations that involve energy consumption separately. The Figure 3 shows the performance of the protocols when only send and receive operations involve energy consumption. For this test battery power was reduced to 1.5 joules, and without making recharging. Otherwise, the battery would never end throughout the 14 days of simulation. This occurs because our model considers the expenditure of energy only when sending, receiving or seeking to neighboring nodes. As for the Scan Interval, we kept the same as the previous test. Figure 3 shows that the delivery rate of both protocols remained high, and the Prophet TFT performance is clearly superior to the Epidemic. Also, the more we increase the energy consumption for sending and receiving messages, the greater the advantage of the Prophet TFT over the Epidemic. This makes sense, since the protocol Epidemic generates copies of the messages whenever it encounters a node, causing higher energy consumption at the nodes.In contrast, the Prophet TFT controls the number of copies, relaying only for nodes that are more likely to deliver the message to the destination. This power-saving Prophet TFT behavior becomes more clear when we look at the report of the energy level of nodes. For example, when energy expenditure in sending and receiving is 0.08 mW at the end of the

simulation protocol Epidemic exhausted the energy of 21/51 nodes. But the Prophet TFT ended the simulation with all nodes with even energy.

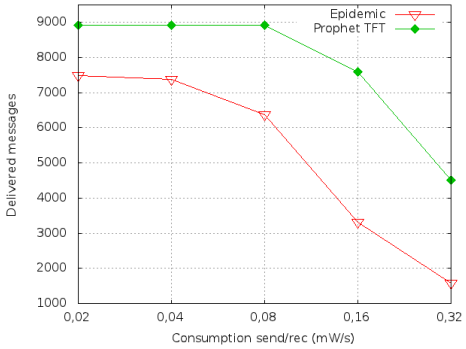

Figure 3: Comparison of Epidemic and Prophet TFT (energy consumption only when sending/receiving)

When we consider the energy consumption only when the node makes search operations, the situation is very similar to that of Figure 2, where the network performance is very close to zero. In fact, using the batteries with their capacity reduced to 1.5 joules, none of the protocols could deliver messages, even declining in four times (0.23 mW) the energy consumed in the search operations. This shows that most of the nodes' energy is consumed in search operations. For this reason, it was found that the performance of the protocols would keep very close in conditions where there is energy consumption.

Thus, the routing strategies aimed at saving energy in reducing in message exchange has little impact on network performance. Another conclusion reached is that a Scan Interval off zero (always scanning) may not make much sense in DTNs, because the opportunity for contact is never guaranteed. Therefore, if nodes are constantly running search operations it is very likely that your energy is rapidly consumed without use in routing the messages.

An effective alternative to improve network performance by reducing the energy consumption is to change the Scan Interval. So the next test aimed to find a value of Scan Interval that increases network performance by reducing the energy consumption. For this, we performed new simulations considering all the energy parameters of Table 2. As interest is to find a good Scan Interval in terms of energy consumption, we performed 10 simulations at each point of scan interval (0, 1, 2, 4, 8, 16, 32, 64, 128, 256, 512, 1024 and 2048 seconds). At each simulation different seeds were used, generating different movements of the nodes. Thus, 130 simulations were obtained for each of the protocols, totaling 260 different simulations. Since there are 10 different simulation results for each Scan Interval, the graph presents only the value corresponding to the result of calculating the simple arithmetic average of 10 simulations performed at each point. The results of these simulations can be seen in Figure 4 and show that no matter how the nodes move, the ideal Scan Interval continues for 32 seconds. Therefore, we conclude that the randomness of movement of the nodes did not alter the behavior of the scan interval.

In order to assess the impact of the ideal Scan Interval

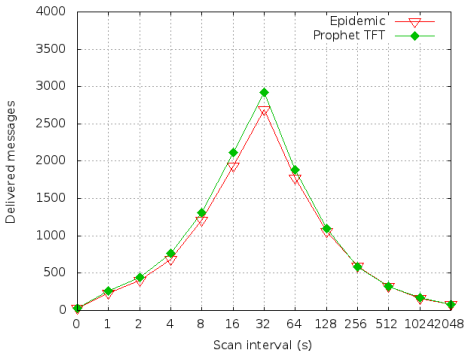

Figure 4: Obtaining the value of optimal Scan Interval

found, Figure 5 shows the result of network performance under the same conditions of simulation of Figure 2, except the Scan Interval. As you can see, despite the reduction in performance when the batteries are always charged, along the simulation where nodes begin to expend energy the number of messages delivered remains high, with a gradual reduction in the extent to that the recharge time of batteries increases.

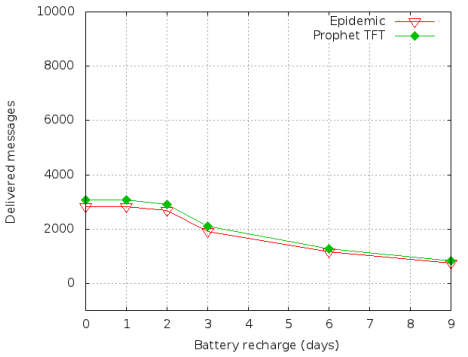

Figure 5: Scan Interval = 32 seconds

5. RELATED WORK

The DTN implicit mobile environment models a situation with two conflicting goals. In one side maximize the number of delivered messages and in the other minimize the energy spent. The greater the number of messages, greater is the amount of energy spent. If the option is save energy, the number of delivered messages can be seriously compromised. Remember that DTNs depends on node availability to forward messages.

This situation can result in a selfish behavior of nodes, which may become reluctant to spend energy on the transmission of other nodes messages. To overcome this problem, there are several incentive algorithms that compel the node cooperation as a way to increase its message delivery chances. However, it is important to note that a permanent cooperation behavior can promote a quick battery discharge, with negative impact on network performance.

In the work described in [10], it is proposed a mechanism that introduces a ranking of the nodes. In this proposal, a node has its score increased when forwards messages from

third parties and decreased when his messages are sent by others. To gain higher ranking, the node prioritize the expedition of messages originated from nodes with higher rank, in detriment of low rank origins. To control the message delivery ratio versus energy, this mechanism sets an superior rank limit that, when reached, the node do not forward messages to save energy. In the other side, when rank score reaches a inferior limit, the node forwards third part messages to increase the rank.

In the work described in [1], it is assumed that the device turns to "high power" mode when sending messages. In the "high power", energy consumption is much higher. Suppose, for example, that a device receives a first message at time x and another at time x +10. If messages are sent immediately, the device will stay for two periods in high power mode. However, in delay tolerant scenarios as DTNs, this is not necessary. The two messages could be sent in a single period. This means a considerable reduction in energy consumption. Then, it was proposed a mechanism to schedule the transmission of messages, minimizing the number of times the energy consumption is high.

For the analysis of energy issues,[3] presents a model that evaluates the behavior of energy consumption in mobile ad hoc networks. In this model it is assumed that the device interface has four states. Sending and Receiving are the states where the node is transmitting or receiving messages. Idle is the state where the interface can change to all other states. Sleeping is the state where the interface needs to be receive a signal to change to idle, in order to perform other operations. The default state of the interfaces is Idle, since traffic transmission and reception can not be predicted. The model calculates the energy consumption in idle mode and does not provide an arbitrary transition to sleeping state. Moreover, it is assumed that all operations of the link layer have the same energy cost. In this case, it is not considered the signal strength as a factor affecting energy consumption.

The work presented in this paper is different from those previously described because it focuses on a very important parameter that is strangely ignored by others. This parameter is the interval between scans. From the simulation experiments, it was found that if the scan interval is too small most energy is consumed in this operation. If it is too big limits the opportunities for contacts and messages delivered to recipients. So it is necessary to find an intermediate value which optimize the relationship between energy and delivery of messages rate. In addition, this proposal introduces the option to recharge the batteries. This is important because it makes the scenario more realistic simulation, allowing to have longer simulations.

6. CONCLUSIONS AND FUTURE WORK

In this paper, we proposed a simulation module for manipulation and analysis of energy consumption in DTNs. It was shown that the same module can be used with different routing protocols and with different values for its parameters. The mechanism is designed so that it can easily be incorporated in any new protocol. The protocol can control the energy consumption, according to its internal logic.

With the analysis of different protocols, it was showed that the energy spent in neighbor search operations is much greater than with the operations of sending and receiving messages. Moreover, it was observed that the performance of the protocols becomes very similar when considering energy expenditure in search operations by neighboring nodes. This shows that DTNs routing strategies that are based only on optimization of message exchange, may have little impact on network performance. In DTN applications, it is necessary to take into account that there is not guaranteed connectivity and, therefore, the time interval between neighbor search operations must be carefully defined. Its range can vary, depending of the application scenario.

However, the used mechanism need some improvements. First of all values of energy transmission, reception and search should be indexed to the interface, not to the node. The amount of energy expended should depend on the range of communication, which is not yet true. Regarding the energy, the module should be able to identify points of interest (restaurants, bars, offices, house, etc ...) where the battery can be recharged. Instead of having a fixed value, the scan interval should be adaptable according to the prediction of new contacts. In the future, we plan to address these and other issues that may increase the degree of realism in the simulation and provide a more detailed energy analysis.

7. REFERENCES

[1] N. Balasubramanian, A. Balasubramanian, and A. Venkataramani. Energy consumption in mobile phones: a measurement study and implications for network applications. In *Proc of IMC'09*, pages 280–293, New York, NY, USA, 2009. ACM.

[2] K. Fall. A delay-tolerant network architecture for challenged internets. In *Proc of the 2003 conf on Applications, technologies, architectures, and protocols for computer communications*, SIGCOMM '03, pages 27–34, New York, NY, USA, 2003. ACM.

[3] L. M. Feeney. An energy consumption model for performance analysis of routing protocols for mobile ad hoc networks. *Mobile Networks and Applications*, 6:239–249, 2001. 10.1023/A:1011474616255.

[4] N. L. Helsinki University of Technology. The one the opportunistic network environment simulator @ONLINE, July 2011.

[5] E. P. Jones and P. A. Ward. Routing strategies for delay-tolerant networks, 2006.

[6] A. Kernen, J. Ott, and T. Karkkainen. The one simulator for dtn protocol evaluation. In *SIMUTools '09: Proc of the 2nd Int Conf on Simulation Tools and Techniques*, New York, NY, USA, 2009. ICST.

[7] A. Lindgren, A. Doria, and O. Schelén. Probabilistic routing in intermittently connected networks. In *Service Assurance with Partial and Intermittent Resources*, LNCS 3126, pages 239–254, 2004.

[8] Nokia. Nokia developer - nokia energy profiler quick start @ONLINE, Sept. 2011.

[9] U. Shevade, H. H. Song, L. Qiu, and Y. Zhang. Incentive-aware routing in dtns. In *IEEE Int Conf on Network Protocols (ICNP 2008)*, pages 238 –247, Oct. 2008.

[10] M. Y. S. Uddin, B. Godfrey, and T. Abdelzaher. Relics: In-network realization of incentives to combat selfishness in dtns. In *IEEE Int Conf on Network Protocols (ICNP 2010)*, Oct. 2010.

[11] A. Vahdat and D. Becker. Epidemic routing for partially connected ad hoc networks. Technical report cs-200006, Duke University, April 2000.