OMNeT++ Models for Resource Allocation in Wireless Networks

Doru Todinca
Department of Computers
University Politehnica
Timişoara, Romania
todinca@cs.upt.ro

Dan Pescaru
Department of Computers
University Politehnica
Timişoara, Romania
dan@cs.upt.ro

Mihaela Viţălariu
Department of Computers
University Politehnica
Timişoara, Romania
mikaela_vitalariu@yahoo.com

ABSTRACT

Nowadays Wireless Networks have an important impact in many activity domains. Due to the hardware and bandwidth limitations of these networks, resource allocation represents an important issue, being in the same time a very active research area. Considering the complexity of the wireless networks, analytical solutions are hard to be found. Hence, simulation becomes the most used methodology and OMNeT++ proves to be a very useful tool for investigation. This paper presents our work concerning the performance of different resource allocation techniques for EGPRS and Wireless Sensor Networks.

Categories and Subject Descriptors

I.6.5 [**Simulation and Modeling**]: Model Development; C.2.1 [**Network Architecture and Design**]: Wireless Communications

General Terms

Experimentation

Keywords

OMNeT++, EGPRS, wireless sensor networks

1. INTRODUCTION

Cellular networks like GSM (Global System for Mobile evolution) and UMTS (Universal Mobile Telecommunication System), although initially designed for voice calls, encounter an increase of the number of users that use them for data services. The GSM networks have been extended with a new service, namely GPRS/EGPRS (General Packet Radio Service/Enhanced GPRS) in order to provide efficient and cost-effective data transfer capabilities. The data applications on mobile networks range from e-mail transfer to real-time applications like audio- and video-streaming and video conferences, having different Quality of Service (QoS) requirements. As a consequence, the users of EGPRS networks belong to different QoS classes and the network aims to provide quality of service differentiation between users.

We developed a simulation model in OMNeT++ and used it to study the problem of resource allocation in EGPRS. Because of its complexity, we split this problem into two subproblems: transmission control (TC) and admission control (AC). AC concerns the methods used for admitting new user in the system, while TC involves the allocation of the network resources between the admitted users.

The problem of resource allocation in (E)GPRS is very complex, and because of that the pure analytical approaches do not provide reliable results, as shown in [13]. Hence, most researchers have chosen to develop simulation models for studying this problem. Industrial research and sometimes even academic research uses complex emulators, like POTOMAC, described in [5], or GPRSim ([18], [13]). Our goal was to avoid the complex interactions that appear in such an emulator between the GPRS protocols, and to develop a simulation model that allows different levels of details in modelling (E)GPRS. In this way we can focus on the problems that we want to study (i.e. AC and TC) and we can approach them one by one, being also easier to give an interpretation to the simulation results.

Another example of wireless networks that have special needs in terms of resource allocation are represented by wireless sensor networks (WSN) [2]. Since a WSN must operate under hard energy constraints the resource management is a key issue, starting from routing protocols to applications level. Moreover, considering the particular case of Video-based Wireless Sensor Network (VWSN) that involves video streaming the problem is even harder [25]. Considering the ad-hoc nature of that networks, an analytical solution is difficult to be handled. Indeed, most of the proposed solutions involve simulation models [15],[12]. In our work we concentrate on using simulation on VWSN routing protocols design.

Next section describes our OMNeT++ simulation models for EGPRS and VWSN, section 3 presents the simulation results that we have obtained for resource allocation in radio networks, section 4 gives a comparison of OMNeT++ with other simulators that we have used, and the paper ends with a section of conclusions.

2. THE SIMULATION MODELS

2.1 GPRS/EGPRS

GPRS [3], [6], [7] is a packet switching service imple-

OMNeT++ 2008 March 3, 2008, Marseille, France

mented in the existing GSM networks in order to deal with data transfer applications like e-mail, file transfer, web browsing, audio and video-streaming, video conferences, etc. Packet switching is more suitable for data transfer than the circuit switching technique used for GSM, because of the bursty nature of most data applications. For example, during a WWW session, there are ON periods, when the user downloads information from a web page, and OFF periods, when the user reads the information downloaded and no data is transferred. Even during the ON periods, the flow of information is not continuous, because a web page may contain different files, and each file is transferred as IP (Internet Protocol) packets.

Some changes were necessary in the existing GSM networks in order to introduce GPRS: the Base Station Subsystem (BSS) is updated with a new hardware element, called Packet Control Unit (PCU), while the core network of a GPRS network is based on new network elements, the most important being the SGSN (Serving GPRS Support Node) and the GGSN (Gateway GPRS Support Node).

The radio resources in a GPRS/EGPRS network consist on 8 timeslots on each frequency, and the PCU runs the scheduling algorithms that assign those radio resources to users during a PCU cycle of 20ms, called block cycle. Each user can receive one or several time-slots, called radio blocks, and a user will put a block of data in each radio block.

SGSN and GGSN are involved in routing the data packets inside a GPRS network and also in the process of QoS negociation between a user (a Mobile Station, or MS) and the network.

Enhanced GPRS (EGPRS) [10] is an extension of GPRS that uses the EDGE (Enhance Data Rates for Global Evolution) technology. This technology ensures higher data rates (in theory 3 times higher than in GPRS) by using the 8 PSK (eight Phase Shift Keying) modulation. In EGPRS the users' data are encoded using nine Modulation and Coding Schemes (MCS1-MCS9), while in GPRS there are four coding schemes, CS1-CS4. Lower coding schemes offer more data protection by strong encoding and are used when the radio conditions are not very good, while in good radio conditions the users can have a higher CS or MCS, that ensures a high throughput.

The users of a mobile data networks experience higher error rates than in wireline networks, and the errors are location dependent and time varying due to fading. In order to accommodate these characteristics of the radio channels, the network operators can assign different priorities to different users, the users with a better radio link receiving higher priority. If a mobile user experiences a bad radio channel, it will use a stronger coding scheme in order to protect the integrity of its data, which means a lower transfer rate. Moreover, it is possible that part of the data transferred over the radio interface will need retransmission, reducing even more the throughput of the network. As a consequence, the mobile operator will reduce the priority of those users, increasing the priority of the users that have a good radio link so that more data will be transferred per time unit, increasing the network throughput and the revenue of the network operators.

2.2 The OMNeT++ simulation model for EGPRS

Figure 1 shows our OMNeT++ simulation model for an EGPRS cell. It consists of k *user modules*, a *user generator*, an *admission controller*, a *stat* node, and the *PCU* node that corresponds to the EGPRS Packet Control Unit. A *WA* module implements a fuzzy logic-based weight adaptation algorithm for adjusting users weights according to their QoS class and to their radio link quality, while the radio conditions are modelled by the *gen radio cond* module. A *stat node* collects statistics about users.

Each user from a cell is modelled by a user module, which is an OMNeT++ composed module. Although in OMNeT++ it is possible to dynamically create and destroy modules, this procedure is quite complex when it involves composed modules, hence we have chosen to have a fixed number of user module in the system. In this way we avoid creating and destroying OMNeT++ modules. A user module can be occupied by a user, or it can be free. The number of user modules in our model was set to 50, which is enough for simulating a GPRS cell, but this number can be increased, the only condition for such an increase is to adjust correspondingly the stack for the simulator.

The user generator module creates users at certain time intervals. The users are OMNeT++ messages having different parameters, which describe the QoS class of the user, its traffic characteristics, if the user comes from another cell (it is a handoff user) or it starts its data transfer session in the current cell, etc.

After creation, each new user requests to be admitted in the system. The admission decision is taken by the admission controller module, which, in our model, implements a fuzzy logic based admission control (AC) algorithm [21]. If the user is admitted, it will occupy a free user module. It is an error if there are no free user modules. A user releases an user node in two situations: either its data transfer session ends (all data created has been transferred), or the user leaves the current cell (since we model a single cell, when a user moves to another cell, it has to leave the user module). In both cases, the user has to inform the user generator module, which keeps the evidence of the user modules.

Figure 2 presents our model for a user. It is a composed OMNeT++ module, consisting of the following simple modules: a *source* node, a *file buffer*, a *data packet buffer*, a *loop node* and a *sink*. A *handoff node*, not represented in the figure, models the situation when a user leaves the cell before the data transfer session is finished.

After the *start user* command is received from the user generator module, the source will generate one or several "files", at certain time intervals, each file having a certain length. A "file" in our model corresponds to a real file (e.g., during an FTP session) or to a group of files (a web page, in case of an WWW session), or even to a part of a file, in a video-streaming session.

The source node can work in either *interactive* or *streaming* mode. In interactive mode, a new file will be created only after the previous file has been successfully transferred, i.e., only after the file reached the sink node and the command *receive ok* has been activated by the sink node. In this way we model a WWW session, when the user will transfer a new web page only after the previous page has been received, or an FTP or e-mail transfer session, when a new file is requested only after receiving the previous file. For the streaming mode, which corresponds to an audio or video streaming session, new data is generated no matter if the previous data reached the destination or not.

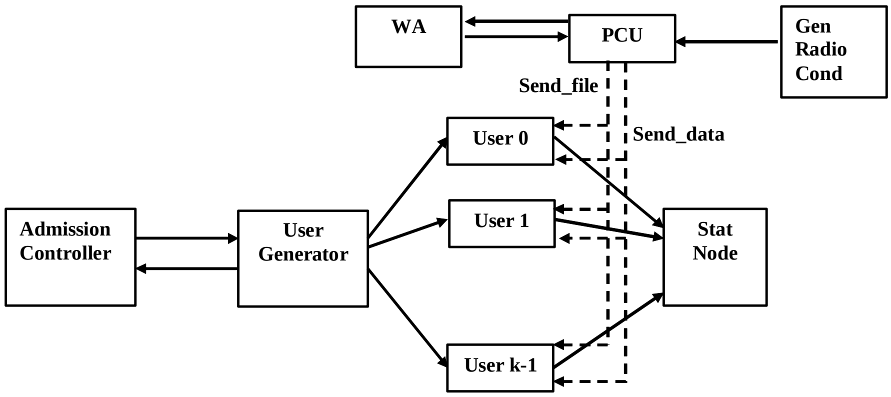

Figure 1: The simulation model for EGPRS.

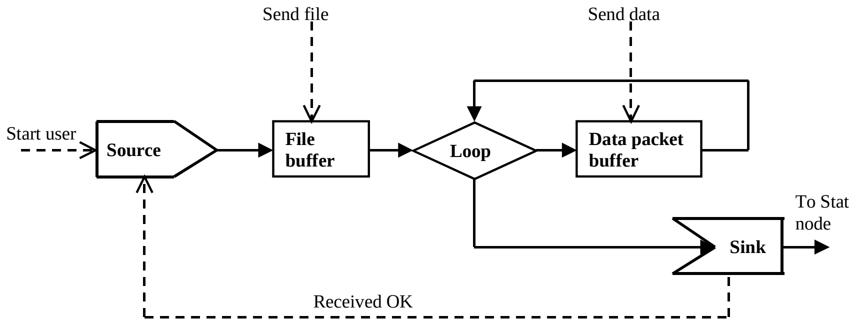

Figure 2: The model for a user.

A file is an OMNeT++ message, having different parameters like the file length, the time when the file was created, the time when the file has reached or leaved a certain node, the order number of the file in the current data transfer session.

After its creation, a file is stored in the *file buffer*. This module models the (E)GPRS process of segmentation of data files into radio blocks using the GPRS coding schemes or the EGPRS modulation and coding schemes: when the file buffer receives the command "send file" from the PCU, it sends the file to the *data packet buffer*, where the file is segmented into radio blocks, according to the user's CS or MCS.

The data packet buffer stores the data blocks resulted from the segmentation of a "file". When the scheduling algorithm decides that a certain user is allowed to transfer a number of data blocks over the radio interface, it sends the command "send data" to the data packet buffer of that user, specifying the number of data blocks that will be transferred. The data blocks affected by errors can be retransmitted, if the retransmission mode is used.

In order to model the transmission and retransmission of data blocks without increasing the simulation overhead by creating too many OMNeT++ messages, we use a *loop node* and modify the file length f_l of the corresponding file according to the amount of data transferred over the radio interface, as shown in the following equation:

$$f\_l = f\_l - (no\_of\_data\_pck - no\_of\_err\_data\_pck) \cdot u\_cs$$

Here, $no\_of\_data\_pck$ is the number of data blocks that are transferred over the radio interface in the current 20ms PCU cycle, $no\_of\_err\_data\_pck$ is the number of data blocks that must be retransmitted, and u_cs is the number of user's bits contained in a data block, according to user's CS or MCS, e.g., 181 bits for CS1 or 428 bits for CS4. The number of retransmitted blocks is a percentage of the total number of transmitted blocks, percentage given by the Block Error Rate (BLER), which is a parameter of the system. If the file length computed according to the above equation is greater than zero, the loop node will re-sent the file to the data packet buffer, meaning that the transfer of that file is not completed, while if the $f\_l = 0$, the file will be sent to the sink, meaning that the file has been successfully transferred over the radio interface.

In the earlier versions of the model, the sink node collected statistics about files and deleted the OMNeT++ messages corresponding to the files, but now it only forwards the "files" (i.e. the OMNeT++ messages) to the *stat node*, which collects statistics for all users. When the source works in interactive mode, the sink informs the source node when a file was completely transferred and a new file can be generated.

The Packet Control Unit, presented in figure 3, contains a source that generates a single message, a *file admission con-*

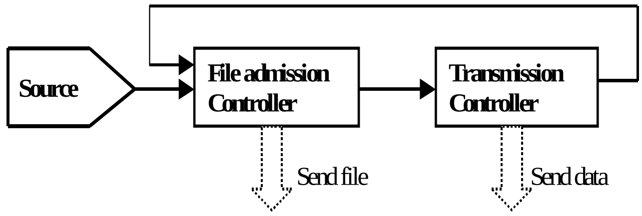

Figure 3: The Packet Control Unit.

troller and a *transmission controller*. The file admission controller module determines which users are active (i.e, they have data in their queues) and sends them the "send file" command, which will determine the transfer of a "file" from user's file buffer to the data packet buffer, where the file is segmented into data blocks. In (E)GPRS there are several levels of admission control: the session level admission control, called PDP context activation (PDP stands from Packet Data Protocol), and a lower level admission control, called Temporary Block Flow (TBF) establishment. The file admission controller module models the TBF establishment, which is used in GPRS/EGPRS in order to avoid wasting the scarce radio resources, by allocating radio resources only to the active users.

The transmission controller module contains the scheduling algorithms that implement the transmission control (TC), i.e., they allocate radio blocks to the active users according to a certain scheme (i.e., weighted round robin or WRR). Every PCU cycle the TC algorithms allocate the available radio channels to the active users. Not all active users receive resources every controller cycle. The number of radio blocks received by a user can be determined by its weight, i.e., in the WRR algorithm, or, a user can receive all the available channels (up to 8), according to the other algorithms that we have implemented in our model: Oldest Queue (OQ), Longest Queue (LQ), Total Queue Length (TQL) or Total File Length (TFL). Those algorithms rank the users according to a criterion, and the winning user will take all the available radio resources. The criterion is the time elapsed since the user has been served last time for OQ, the amount of data in user's data packet buffer (LQ), or the amount of data in both file buffer and data packet buffer for TQL and TFL. The ranking criteria takes into account the user's weight. For example, in the LQ algorithm, the real queue length of a user, expressed in data blocks, is multiplied by the user's weight, so that a user with a higher weight will be served more often. In our simulations the weights are positive integer numbers. The output of this module is the "send data" command, sent every 20ms PCU cycle to the data packet buffer of the active users, specifying the number of data blocks that will be transmitted by those users in the current PCU cycle, i.e. the parameter $no\_of\_data\_pck$.

The *admission controller* module implements the session level admission control (AC). In [21], we have proposed a new AC algorithm, based on fuzzy logic. The OMNeT++ module *admission controller* contains a functional implementation of a FLC, being a "translation" of the VHDL code from [19].

The WA module contains another description of a FLC, but, in order to increase the adaptability of the implementation, the FLC parameters are coded in an XML configuration file. To handle this file we take advantage of the new XML API integrated in recent versions of OMNeT++ [1]. The file integrates membership functions through a <fuzzy_set> tag and fuzzy rules described in a <rule> tag. This structure allows straightforward adjustments in the FLC tuning phase.

The *gen radio cond* module models the quality of the radio link for each user, using a Markov process that determines the change of the MCS for each user.

The *stat node* collects statistics about the simulation. For transmission control, the statistics concern the average, minimum and maximum values for the *waiting delay*, *sending delay* and *total delay* of user's file, the number of blocks retransmitted due to errors, the amount of data dropped (because the file length queue was exceeded), the duration of the data transfer session, etc. Waiting delay is the time spent by a file in the file buffer, sending delay is the time spent in the data packet buffer, and the total delay is the sum between the waiting and sending delay. The statistics for AC concern the probability of dropping a session in progress and the probability to block a request for a new data transfer session.

2.3 Resource allocation in VWSN

A wireless sensor network is a special kind of ad-hoc network consisting of a large number of small and cheap sensor nodes and one or more sinks acting as data collecting points. The basic sensor node architecture is based on three modules. One is the sensing module that collects information from the environment. Another one is the communication module designed to allow wireless multi-hop data communication between nodes and sink. The last one is the processing module, which is involved mainly in local data processing and data aggregation with information coming from other nodes.

Video-based Wireless Sensor Network is a special case of wireless sensor network based on low power video-camera sensors. In a VWSN large amounts of video data are sensed, processed in real-time, and then transferred using wireless communication.

The most important characteristic of WSN is the power consumption. Indeed, a WSN is required to be operational for a long period of time ranging from month to years. Therefore, resource allocation in such a network concentrates on that characteristic. The most energy-consuming part of a sensor node is the communication module. To reduce the energy consumption the duty cycle of the transceiver has to be as low as possible. Moreover, in case of VWSN, data transmission involves large video streams. Considering that, we concentrate on designing a routing algorithm suitable for video transmission on multi-hop wireless communication.

Ad-hoc network routing algorithms can be classified in two main categories as proactive and reactive. In case of proactive routing protocols, like Adaptive Distance Vector [4] or Path-Finding Algorithm [9], all routing information are preserved at the level of each network node. By contrast, reactive routing protocols, like Dynamic Source Routing (DSR)[14] or Ad-hoc On-demand Distance-Vector [16], calculate routes only when communication is needed and they maintain these routes as long as the connection is needed.

In case of wireless sensor networks, existing ad-hoc routing protocols have two major drawbacks: communication overhead and complex hardware needs. More suitable data-centric routing algorithms, like Directed Diffusion [12], were tailored for these networks. Many of them were derived from a basic Flooding algorithm [11]. However, in case of a Video-based WSN they have to be adapted for video-streaming communication. In [8], we propose an algorithm named Adaptive Routing (AR) designed for this case. It combines advantages from both proactive and reactive routing. After network deployment it tries to fill each node routing table with information about proximity and hop count to the central point. Then it will query all nodes for synchronously image information used to extract topology. The central server broadcasts a setup message containing empty route path information. Each node that receives a setup message will determine if it is included or not in the route information. If true, it simply drops the message. If it is not included, it refreshes the routing table and broadcasts the message further. For route optimization, each node keeps additional information describing the energy level and hop-count for all its neighbors. To improve the energetic efficiency the current node elects the neighbor with optimum energy level and hop-count as future node in the path.

2.4 The OMNeT++ simulation model for VWSN

The AR protocol was tested using an OMNeT++ simulation model that covers both networking and distributed computing aspects. OMNeT++ uses NED files to store the network structure. Writing a NED file for a large WSN is a laborious work. Consequently we decided to design a C++ NED generator. It uses a compact XML configuration file describing network topology. The configuration file includes sink description, sensor nodes topology and channel implementation for wireless broadcast. The protocol uses broadcast communication to achieve interaction inside the network. The broadcast is implemented using channels that connect pairs of node entities. Each node is connected to a number of channels corresponding to the number of its neighbors. Therefore, message received by a sensor node is cloned for each of its associated channels and forwarded to the corresponding neighbor. The message encapsulates the following information: the type of message as sink request or node response, the hop-count from/to the sink depending of its type, the network route, an unique identifier to avoid loops and a bitmap representing the image captured by node's camera in case of node response messages.

3. SIMULATION RESULTS

3.1 Results for EGPRS

In this section we present the results obtained with the OMNeT++ models that we have developed for the problem of resource allocation in EGPRS. The research problem was separated into admission control (AC) and transmission control (TC).

For AC, we have developed a new algorithm, based on fuzzy logic, and we have shown its performance and flexibility in [21], [20].

For TC, we compared the performance of the scheduling algorithms described in section 2.2 (i.e. WRR, OQ, LQ, TQL and TFL), using different simulation scenarios. The results, published in [22], [23], show that WRR and OQ are

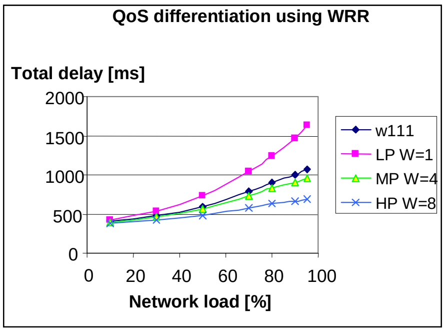

Figure 4: Implementing QoS differentiation with WRR.

the most efficient, while TQL, TFL and LQ are not suited for implementing TC in EGPRS.

Also, we studied the efficiency of implementing QoS differentiation between users, based on their weights. Figure 4 shows the delays for three classes of users: high priority (HP), having a weight $W = 8$, medium priority (MP), with $W = 4$ and low priority (LP), with $W = 1$, for the WRR algorithm, when network load increases from 10% to 95%. The simulations are for a system with 3 HP users, 5 MP and 2 LP users. In the figure w111 corresponds to the case when all users have the same weight, $W = 1$. It can be observed that there is a clear QoS differentiation, the MP and especially HP users having a smaller delay than in the case with equal weights for all users. More details can be found in [22].

By allocating user's weights based on the quality of their radio link, we have demonstrated in [23] how this can reduce the congestion in the cell.

An interesting question is how to combine these two approaches, i.e., how to allocate users' weights taking into account both their QoS class, and the quality of their radio link? In order to answer this question, we propose a new weight adaptation algorithm, based on fuzzy logic. It uses fuzzy IF-THEN rules having in the premises the linguistic variables "QoS class" with the terms HP, MP and LP, "network load" with the terms Low (L), Medium (M) and High (H), and "link quality" with the terms Bad (B), Fair (F), Good (G) and Very Good (VG). The linguistic variable in conclusion is "user's weight" with the terms Very Small (VS), Small (S), Medium (M), Big (B) and Very Big (VB). For HP users we have 12 rules in the form:
If *QoS class* is HP AND *network load* is H AND *link quality* is VG THEN *user's weight* is VB.

The complete set of rules for HP, MP and LP users is given in the Tables 1, 2 and 3 respectively.

Table 1: The fuzzy rules for HP users

		Link quality			
		B	**F**	**G**	**VG**
Network load	**L**	M	B	VB	VB
	M	M	B	VB	VB
	H	S	M	B	VB

It can be noted that the weight for LP users is in most cases Very Small or Small, being Medium only for low network load and very good radio conditions, while, on the op-

Table 2: The fuzzy rules for MP users

		Link quality			
		B	**F**	**G**	**VG**
Network load	**L**	S	M	B	VB
	M	S	S	M	B
	H	VS	S	S	M

Table 3: The fuzzy rules for LP users

		Link quality			
		B	**F**	**G**	**VG**
Network load	**L**	VS	VS	S	M
	M	VS	VS	S	S
	H	VS	VS	VS	S

posite, for HP users the weight is in most cases Big of Very Big, being reduced to Small only for high network load and bad radio conditions. The linguistic terms for user's weight (VS, S, M, B and VB) are mapped to values in the interval [1, 10]. For WRR the values must be integers, while for the OQ algorithm users weights can be real numbers.

We have validated by simulation ([24]) the functionality of the fuzzy weight adaptation algorithm, but since the work is in progress, more simulations are necessary for a quantitative estimation of the performance of our algorithm.

All the simulation results for EGPRS have been obtained on a PC AMD Athlon XP 1700+ with 512 MB RAM, under Linux or Windows XP operating systems. For AC we have generated 4000 users at different time intervals, and we determined the call blocking and call dropping probability. The simulations concerning TC were performed for a set of 10 users, for 5000 seconds simulation time. The duration of such a simulation on our computer was less than one minute. In order to obtain reliable simulation results we used confidence intervals and the OMNeT++ feature to detect the simulation accuracy.

3.2 Results for VWSN

All experiments were performed on an AMD Athlon 64 2800+ PC with 1 GB RAM and 120 GB HDD. The main goal of the performed experiments was to compare AR protocol against standard Flooding algorithm, as presented in [8] and [17]. Also the experiments allow performance impact estimation in case of node failures. We have chosen three metrics to analyze the performance of proposed protocol. They are the number of transmitted messages (NTM), the total amount of received messages (NRM) and the average dissipated energy (ADE). The data set is based on four different topologies consisting in one sink and three to thirty nodes, as presented in Figure 5.

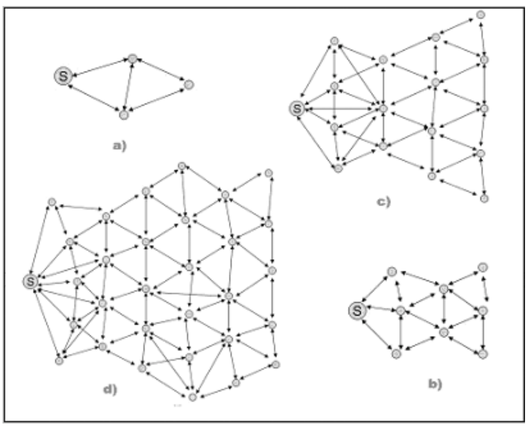

Figure 5: Networks used in tests: a) 3 nodes network; b) 8 nodes network; c) 16 nodes network; d) 30 nodes network.

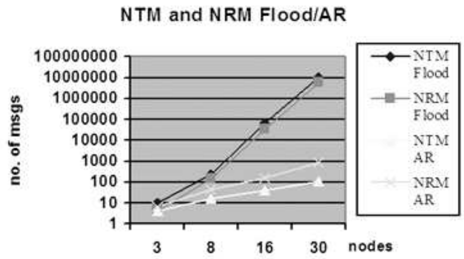

Figure 6: NTM and NRM on Flooding and Adaptive Routing on a Logarithmic Scale.

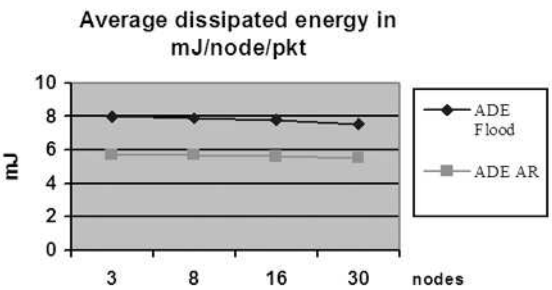

Figure 7: Estimated Average Dissipated Energy on Flooding and Adaptive Routing.

The results of simulation are presented in Figure 6. It shows the total amount of transmitted messages (NTM) and the total amount of received messages (NRM) for both flood-

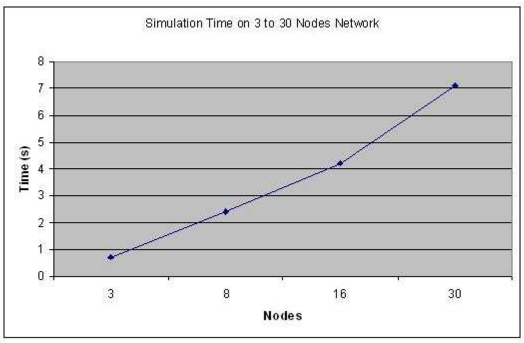

Figure 8: Simulation Time on 3 to 30 Nodes Networks.

ing and adaptive routing protocol on a logarithmic scale. Figure 7 presents the simulation result in terms of average dissipated energy. The simulation time variation is depicted by figure 8. It depends exponentially on network size and on numbers of link between nodes. The number of links results from network topology. In our experiments we consider a 100m x 100m deployment filed size and the wireless range was set to 20m. The resulting average links number for each node was 2 to 6, depending on nodes density.

4. COMPARISON WITH OTHER SIMULATION TOOLS

The firsts versions of our GPRS/EGPRS models ([22], [23]) have been developed at the Dublin City University using the SES/*workbench* simulator from Hyperformix. The SES/*workbench* simulator provides a number of predefined modules, like sources, servers, sinks, etc, the connection between modules being realized by a graphical editor. Each predefined module has a set of parameters that can be easily changed and the possibility to include C++ code into them.

The advantage of this approach (many predefined modules) is an easy development of simple models, but the drawback is that, when the functionality provided by a predefined module has to be changed, this is much more difficult than in OMNeT++ because a part of the functionality of the SES modules is "hidden" and not easy accessible. For example, in our model we need a source module which is controllable (it has an input command, that can block the generation of new data), but the source node from SES/workbench has no inputs. Making a controllable source in SES is not an easy task, involving the changing of internal code, expressed as macro-instructions, not very well documented. In OMNeT++ it was no difficulty at all to model such an interactive source node!

Because both SES and OMNeT are based on C++, the task of "translating" the SES model into OMNeT++ was achievable in only a few weeks.

For the fuzzy logic controller (FLC) modelling, we have initially used VHDL, which is a hardware description language. VHDL has the same philosophy like OMNeT++, being based on open code, with no predefined modules, but everything has to be done by the designer. While VHDL has proved to be a good choice for studying the performance of the FLC ([19]), it is not suited for network modelling, but the translation of the VHDL behavioral description of the FLC into C++ code was straightforward.

5. CONCLUSIONS

In this paper we have presented our research concerning the problem of resource allocation in EGPRS and VWSN. We based our investigations on simulation models developed in OMNeT++. Our results show that OMNeT++ is a very powerful and versatile simulator, offering the possibility to work at different levels of abstractions. Also, we compared OMNeT++ with other simulation tools and revealed the advantages of using OMNeT++.

6. REFERENCES

[1] ***. OMNeT++ NEDXML API reference. http://www.omnetpp.org/doc/nedxml-api/index.html, 2007.

[2] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci. Wireless sensor networks: a survey. *Comput. Networks*, 38(4):393–422, 2002.

[3] C. Bettstetter, H.-J. Vogel, and J. Ebersacher. GSM phase 2+ General Packet Radio Service GPRS: Architecture, protocols, and air interface. *IEEE Communications Surveys*, 2(3), 1999.

[4] R. V. Boppana and S. P. Konduru. An adaptive distance vector routing algorithm for mobile ad-hoc networks. In *the 20th Annual Joint Conference of the IEEE INFOCOMM Proceedings*, volume 3, pages 1753–1762, 2001.

[5] M. Bourgois, E. Vivier, M. Terré, N. Bret, and B. Ridard. Necessity of an emulator to validate new resources allocation protocols. In *Proceedings of IEEE Spring Vehicular Technology Conference VTC 2001*, Rhodos, Greece, 2001.

[6] G. Brasche and B. Walke. Concepts, services, and protocols of new GSM phase 2+ General Packet Radio Service. *IEEE Communications Magazine*, 35(8):94–104, August 1997.

[7] J. Cai and D. J. Goodman. General Packet Radio Service in GSM. *IEEE Communications Magazine*, 35(10):122–131, October 1997.

[8] M. Cosma, D. Pescaru, B. Ciubotaru, and D. Todinca. Routing and topology extraction protocol for a wireless sensor network using video information. In *the 3rd Romanian-Hungarian Joint Symposium on Applied Computational Intelligence (SACI'06) Proceedings*, May 2006.

[9] J. J. Garcia-Luna-Aceves and S. Murthy. A path-finding algorithm for loop-free routing. *IEEE/ACM Transactions on Networking*, 5(1):148–160, 1997.

[10] M. Hakaste, E. Nikula, and S. Hamiti. GSM/EDGE standards evolution (up to Rel'4). In T. Halonen, J. Romero, and J. Molero, editors, *GSM, GPRS and EDGE Performance*, pages 3–63. John Wiley & Sons, LTD, 2002.

[11] W. R. Heinzelman, J. Kulik, and H. Balakrishnan. Adaptive protocols for information dissemination in wireless sensor networks. In *the ACM/IEEE International Conference on Mobile Computing and Networking Proceedings*, pages 174–185, August 1999.

[12] C. Intanagonwiwat, R. Govindan, D. Estrin, J. Heidemann, and F. Silva. Directed diffusion for wireless sensor networking. *IEEE/ACM Transactions on Networking*, 11(1):2–16, 2003.

[13] T. Irnich and P. Stuckmann. Analytical performance evaluation of Internet access over GPRS and its comparison with simulation results. In *Proceedings of PIMRC*, 2002.

[14] D. Johnson, D. Maltz, and J. Broch. *DSR The Dynamic Source Routing Protocol for Multihop Wireless Ad Hoc Networks*, pages 139–172. Addison-Wesley, 2001.

[15] P. Pahalawatta, T. Pappas, and A. Katsaggelos. Optimal sensor selection for video-based target tracking in a wireless sensor network. In *the International Conference on Image Processing ICIP 2004 Proceedings*, pages 3073–3076, October 2004.

[16] C. E. Perkins, E. M. Belding-Royer, and S. R. Das.

Ad hoc on-demand distance vector routing protocol. Technical report, IETF MANET Working Group, February 2003.

[17] D. Pescaru, D. Fuiorea, V. Gui, C. Toma, G. Muntean, and A. Doboli. Image-based node localization algorithm for wireless video sensor networks. In *the IT&T 2006 Conference Proceedings, Ireland*, pages 139–148, October 2006.

[18] P. Stuckmann and O. Möller. Advanced scheduling and admission control techniques for cellular packet radio networks. In *Proceedings of European Personal Mobile Communications Conference EPMCC*, Glasgow, UK, 2003.

[19] D. Todinca, A. M. Badulescu, and R. E. Precup. VHDL approach to performance analysis of fuzzy logic controllers. In *Proceedings of 8th Information Processing and Management of Uncertainty in Knowledge-Based Systems Conference (IPMU 2000)*, pages 896–901, Madrid, Spain, July 2000.

[20] D. Todinca, I. Şora, P. Perry, and J. Murphy. Supporting flexible network operator policies in EGPRS trough admission control. In *Proceedings of the 10th IFIP Conference on Personal Wireless Communicaions (PWC05)*, pages 346–353, Colmar, France, August 2005.

[21] D. Todinca, H. Graja, P. Perry, and J. Murphy. Novel admission control algorithm for GPRS/EGPRS based on fuzzy logic. In *Proceedings of IEE 5th International Conference on 3G Mobile Communications Technologies 3G2004*, London, UK, October 2004.

[22] D. Todinca, P. Perry, and J. Murphy. Algorithms for resource allocation in data transfer over EGPRS networks. In *Proceedings of the 2nd European Conference on Universal Multiservice Networks (ECUMN'2002)*, pages 246–250, Colmar, France, April 2002.

[23] D. Todinca, P. Perry, and J. Murphy. Novel prioritised EGPRS medium access regime for reduced file transfer delay during congested periods. In *Proceedings of IEE 3rd International Conference on 3G Mobile Communications Technologies*, pages 550–554, London, UK, May 2002.

[24] M. Viţălariu. QoS ensurance through dynamic weight adaptation in an EGPRS network. Diploma Project, Technical Report LD645-07 "Politehnica" University of Timişoara, Romania, 2007.

[25] Y. Weng and A. Doboli. Smart sensor architecture customized for image processing applications. In *the 10th IEEE Real-Time and Embedded Technology and Applications Proceedings*, May 2004.