Soler Fernández, D.; Albiach Vicent, J.; Martínez Molada, E.; Manzoni, P. (2013). An algorithm to evaluate routing conditions in smartphones-based wireless networks. Expert Systems with Applications. 40(13):5033-5048. doi:10.1016/j.eswa.2013.02.035.

An algorithm to evaluate routing conditions in smartphones-based wireless networks

David Soler[a], José Albiach[a], Eulalia Martínez[a], Pietro Manzoni[b,*]

[a] *Universitat Politècnica de València, Instituto Universitario de Matemática Pura y Aplicada, Valencia, SPAIN*
[b] *Universitat Politècnica de València, Dept. of Computer Engineering, Valencia, SPAIN*

Abstract

The increasing penetration of smartphones, i.e., smart devices with multiple sensing and communication interfaces, creates the possibility to build novel types of networks. Opportunistic networking and Content-Based networking strongly rely on the use of such devices. Smartphones tend to have an ON/OFF status that strongly depends on the user activity, mobility pattern and energy saving approach. Wireless adaptors are, after the terminal screen, the strongest source of power consumption. It is therefore common for a node to occasionally turn off the networking device to save energy. The impact on routing of the presence of nodes in the off-state must therefore be thoroughly evaluated.

We propose an analytical model based on evolving graphs, which provides an exhaustive evaluation of routing conditions with the aim to determine the best recurring strategy and parameters when dealing with end-devices that show an ON-OFF behavior. Computational results are given, both on static and dynamic scenarios.

Keywords: Network design, Network protocol, Protocol verification, Wireless network, Evolving graph.

*Corresponding author
Email addresses: dsoler@mat.upv.es (David Soler), jalbiach@mat.upv.es (José Albiach), eumarti@mat.upv.es (Eulalia Martínez), pmanzoni@disca.upv.es (Pietro Manzoni)

1. Introduction

The increasing penetration of smartphones, i.e., smart devices with multiple sensing and communication interfaces, creates the possibility of building novel types of applications that leverages the use of wireless ad hoc networks (MANETs), see for example (Mengual et al., 2012; Oliveira et al., 2012; Santiago et al., 2012; Yoon & Cho, 2012). These devices can be intermittently connected and even individual nodes can dynamically be turned off to save energy. Regarding topology, high variability can be expected, from very dense scenarios to highly partitioned ones (Cheng et al., 2012). We will refer to this type of networks as *Smartphones-based wireless networks* (SWNs).

Classical MANETs routing algorithms can provide forwarding in SWNs by building and updating routing tables whenever mobility occurs. There is anyway a novel and critical factor that has a strong impact on the behavior of those algorithms: energy saving. For example, in some recent works (Frantti, 2012; Quintas & Friderikos, 2012) efforts on reducing energy consumption by modifying and adapting the lower layers of the networking architecture are presented. Wireless adaptors are, after the terminal screen, the strongest source of power consumption. It is therefore very common for a node to occasionally turn off the networking device to save energy. The effect of the presence of nodes in the off-state must therefore be thoroughly evaluated.

The definition of a formal model to describe a SWN can be approached in different ways (Frantti & Koivula, 2011; Gutierrez-Reina et al., 2012; Liao et al., 2011; Torres et al., 2012; Zhou et al., 2004). In (Buchegger & Le Boudec, 2002) the authors use game theory to evaluate cooperation in ad hoc networks for energy optimization. In (Tian et al., 2002) the authors propose a graph-based mobility model, which provides a more realistic movement than the random walk model by reflecting the spatial constraints in the real world. Finally, some papers have considered a graph structure called *evolving graph* to study the behavior of some kinds of SWNs, with the aim of designing new routing protocols; see for example (Ferreira, 2004; Monteiro et al., 2006).

We will use this last approach formally defining the concepts of *evolving graph* and *journey* as follows:

Definition 1. *Let $S_G = \{G_i\}_{i=1}^{T}$ be a set of subgraphs of a given graph G such that $G = \cup_{i=1}^{T} G_i$. Let $S_T = \{t_1, t_2, \ldots, t_T, t_{T+1}\}$ be an ordered sequence of time instants. The system $\Gamma = \{G, S_G, S_T\}$, where each G_i is the subgraph in place during $[t_i, t_{i+1}[$ $\forall i \in \{1, 2, \ldots, T\}$ is called an evolving graph.*

Definition 2. *Given an evolving graph $\Gamma = \{G, S_G, S_T\}$, a path $P = \{(u_1, u_2), (u_2, u_3), \ldots, (u_k, u_{k+1})\}$ in G and a time schedule $R = \{\tau_1, \tau_2, \ldots, \tau_k\}$ indicating that edge (u_i, u_{i+1}) is to be traversed at time τ_i, the pair $J = (P, R)$ is called a journey in Γ if and only if $t_1 \leq \tau_1 \leq \tau_2 \leq, \ldots, \leq \tau_k < t_{T+1}$ and a subgraph in S_G containing edge (u_i, u_{i+1}) is placed at instant τ_i $\forall i \in \{1, 2, \ldots, k\}$.*

We propose an analytical model based on an evolving graph (hereinafter EG) associated to the network conditions. This tool provides an exhaustive evaluation of routing conditions, and its aim is to determine the best option for routing parameters taking into consideration the network conditions in SWNs, mainly the possible switching-off mechanisms adopted to save energy.

We describe an algorithm that determines factors like: how many complete messages get to the destination, which is the smallest amount of time required by a packet to get to the destination, and the fluctuations of the number of hops observed by a packet to get to its destination. Thus, for example, applying this algorithm to different switching-off patterns, through the obtained results with respect to the factors cited above, we can determine the best routing strategy to maintain an adequate performance while still saving energy.

We show that the complexity of the algorithm is $O(nT^2)$, n being the number of nodes, and T being the width of the time interval studied, that is, the number of instants of time of the studied period.

The rest of this paper is organized as follows. Section 2 describes our model, i.e. the network conditions and the problem we are going to tackle in this network. Section 3 describes the EG associated to our model as well as a particular

case of journey, Section 4 presents the details of the proposed algorithm. Section 5 offers computational results on a set of 400 randomly generated instances, both in static and dynamic scenarios. Finally, Section 6 gives the conclusions of this work.

2. Model definition

2.1. Network conditions

We represent an SWN as a set of n nodes, $W = \{v_i\}_{i=1}^{n}$, each referring to a device placed on the $\mathbb{R}^3$ space. We define a coordinate function:

$$co_i : [0, T] \to \mathbb{R}^3, \quad co_i(t) = (x_i(t), y_i(t), z_i(t))$$

where $T \in \mathbb{Z}^+$ and $[0, T]$ is the total interval of time during which there is network activity. The coordinate function provides the position of node v_i at any instant of time $t \in [0, T]$.

Each node v_i can be either *active* or *inactive*. A node is active, i.e., it can either send or receive messages, during certain time windows inside $[0, T]$, where we will suppose that time takes integer values (discretized time). The set of activating time windows for node v_i is defined as:

$$tw_i = \{[t^i_{2k-1}, t^i_{2k}]\}_{k=1}^{p_i}$$

where $0 \leq t^i_s < t^i_r \leq T$ if $s < r$ with $s, r \in \{1, 2, \ldots, 2p_i\}$, p_i being the number of activating time windows for node v_i. Thus, v_i can only send messages at time $t \in [t^i_{2k-1}, t^i_{2k}]$ for some $k \in \{1, \ldots, p_i\}$. Outside tw_i node v_i is *inactive* and it can neither send nor receive any message.

The transmission range of a node is a zone in $\mathbb{R}^3$ that depends of the node itself and the sending time t. We define the transmission range of node v_i at instant t as $R_i(t) \subset \mathbb{R}^3$ such that if $co_j(t) \in R_i(t)$ and v_i sends a packet at time t, this packet will be received by v_j at instant $t + t_{ij}(t)$. Note that $R_i(t) = \emptyset \ \forall t \notin \bigcup_{k=1}^{p_i} \left[t^i_{2k-1}, t^i_{2k}\right]$, and that this transmission range can depend on many factors, like the RF technology adopted, the surrounding environment, the antenna type, and so on. For example, it can be an sphere centered at $co_i(t)$

or like a cone with vertex at $co_i(t)$ if the antenna transmits only in one direction, or a disc or a circular sector if we only consider movements in $\mathbb{R}^2$ (which is the usual), etc.

In this paper we will use the term "packet" to indicate the message unit, and we will suppose that all messages consist of one or several packets and that at any instant of time a node can send at most one packet.

On the other hand, function $t_{ij}(t) \in \mathbb{Z}^+$ indicates the units of time it takes for a packet to move from v_i to v_j. The node v_j will receive this packet only if active, that is, only if $t + t_{ij}(t) \in [t^j_{2k-1}, t^j_{2k}]$ for some $k \in \{1, \ldots, p_j\}$. We consider that $t_{ij}(t)$ is small enough so that $co_j(t) \approx co_j(t+t_{ij}(t))$, and therefore, if $co_j(t) \in R_i(t)$ then $co_j(t') \in R_i(t)\ \forall t' \in [t, t + t_{ij}(t)]$. We are basically supposing that, since sending time are so small compared to the nodes mobility, vertices are practically static during each sending operation. Note that the fact that the values $t_{ij}(t)$ must be integer does not involve a strong restriction with respect to the continuous case, because we can define an appropriate unit of time for each SWN.

The sending of a packet from v_i at time t and the reception at node v_j at instant $t + t_{ij}(t)$ if $co_j(t) \in R_i(t)$, implies a nonnegative cost that may depend on many different factors that might be of interest of the protocol designer. For example, it might be proportional to the distance, to the sending time, or it might be equal to one (representing a hop). For each path followed by a packet, our procedure will consider three associated values: time consumed, cost and number of hops.

Once node v_j receives a packet from v_i and determines that it has to forward it, it will do it at time $t+t_{ij}(t)+r_j$, where value $r_j \in \mathbb{Z}^+$ represents the processing time at node v_j. The instant of time $t + t_{ij}(t) + r_j$ must belong to the same activation window of $t + t_{ij}(t)$, otherwise the message will get lost at v_j. To simplify, in this paper we suppose r_j as a fixed value for each v_j, but in general it can be a function $r_j(t)$ that may for example depend on the routing protocol. This last can be included in our procedure without additional changes. In this work we are focussing on FIFO scheduling which is the generally deployed

scheme. We are not saying that this is the best schedule, but it includes most of the real cases. Proving that more transmissions might be feasible with some other schedule is not part of this work. In fact, given that some transmissions are not possible (i.e. the system is above capacity), there may be different ways to choose which packets should be transmitted.

We will suppose that in the network the following circumstances are verified:

- If a node v_i receives more than one packet at the same instant of time, because of interferences all the packets will be lost at v_i.
- A node does not forward a packet originated by itself nor resends a packet whose destination is itself, nor resends a packet sent previously by itself. Basically a packet is resent by a node only if it comes from different paths, thus avoiding loops.
- To save energy, a node v_i resends the same packet (not originated by itself) at most g_i times, ignoring this packet if it comes after the g_i-th resending operation.
- If a node v_j is about to send a locally generated packet at instant t, instant $t - r_j$ belongs to the same activating time window that t, and v_j receives one and only one packet at $t - r_j$, except in the three cases listed below, it will send the packet received at instant t and it will send its own packet in the following available instant t, delaying if it is required any other own packets. If at instant $t - r_j$ v_j receives more than one packet, as explained before these packets will be lost and therefore the node will send its own packet at instant t. The three exceptions are:
  1. If the packet that receives at $t - r_j$ is directed to it;
  2. if the packet received at instant $t - r_j$ was generated by v_j in a previous instant of time;
  3. if the packet received at instant $t - r_j$ is not directed to v_j and v_j already forwarded it g_j times;

  in the above three cases it will send its own packet at instant t.

2.2. Problem statement

Given an SWN consisting of n nodes, $W = \{v_i\}_{i=1}^{n}$, with all the conditions and data cited above, we suppose that during interval $[0, T]$ a subset of nodes $W_0 \subseteq W$ generate (i.e., send) messages. That is, each node $v_i \in W_0$ sends m_i messages $M_1^i, M_2^i, \ldots, M_{m_i}^i$ respectively to nodes $v_{1_i}, v_{2_i}, \ldots, v_{m_{i_i}}$ which can not be two disjunct. Each message M_k^i has a unique destination and it is made of p_k^i packets sent at integer instants of time $t_1^{k_i}, t_2^{k_i}, \ldots, t_{p_k^i}^{k_i}$ where $0 \leq t_1^{k_i} < t_2^{k_i} < \ldots < t_{p_k^i}^{k_i} < T$. Packets related to different messages can be alternated, like for example, $t_1^{k_i} < t_1^{h_i} < t_2^{k_i}$ supposing $h \neq k$, where node v_i after sending the first packet of message M_k^i send the first packet of message M_h^i and afterward the second packet of message M_k^i.

The answers that we will give with our approach are all oriented to offer criteria to determine the impact of the different parameters values involving the network conditions (e.g., inactivity periods, transmission ranges, number of times that a packet can be resend, processing times, etc.), when adopting an energy saving procedure at the device level. The three most important metrics that we will obtain are:

- the number of complete messages that get to the destination;
- the minimum and maximum number of hops made by a packet to get to its destination;
- the amount of time required by a packet to get to the destination;

3. EG associated to our model

For each $t \in [0, T-2] \cap (\mathbb{Z}^+ \cup \{0\})$ we construct a directed graph $G_t = (V_t, A_t)$ such that if $v_i, v_j \in W$ verify that:

- $co_j(t) \in R_i(t)$,
- v_j is active during the time interval $[t + t_{ij}(t), t + t_{ij}(t) + r_j]$ and
- $t + t_{ij}(t) + r_j \leq T$;

that is, if v_j can receive and process a packet sent by v_i at instant t, then vertices $v_i, v_j \in V_t$ and arc $(v_i, v_j) \in A_t$, having this arc two associated costs: $c_{ij}^t = t_{ij}(t) + r_j$ (the sending time plus the processing time at v_j) and $c_{ij}'^t$ corresponding to the sending of a message from v_i at time t and the reception at node v_j at instant $t + t_{ij}(t)$ (see the network conditions).

We consider the EG $\Gamma = \{G_E, S_G, S_T\}$, where $S_G = \{G_t\}_{t=0}^{T-2}$, $S_T = \{0, 1, \ldots, T-1\}$, $G_E = (V_E, A_E) = \cup_{t=0}^{T-2} G_t$, with $V_E \subseteq W$ and G_i is in place in $[i, i+1[$ $\forall i \in \{0, 1, \ldots, T-2\}$. Note that as we consider discretized time, the last condition implies that G_i is only in place at instant i.

It is easy to see that given our SWN with all the cited parameters, the construction of the corresponding EG Γ has complexity $O(n\sigma)$, where σ is the sum of all the instants of time during which nodes are active, that is:

$$\sigma = \sum_{i=1}^{n} \sum_{j=1}^{p_i} (t_{2j}^i - t_{2j-1}^i)$$

Note that σ is upper bounded by nT, the case in which all the nodes are active during the whole interval $[0, T]$, and then, that the construction of the EG Γ has complexity upper bounded by $O(n^2 T)$. We will use this fact later.

To clarify the presentation of our network model and its associated EG, we show an example of an SWN in $\mathbb{R}^2$ with 11 nodes ($W = \{v_i\}_{i=1}^{11}$) with their corresponding activation time windows: $tw_1 = [0, 2]$, $tw_2 = [1, 3]$, $tw_3 = tw_4 = tw_5 = [3, 6]$, $tw_6 = [4, 7]$, $tw_7 = [4, 8]$, $tw_8 = tw_9 = [6, 10]$, $tw_{10} = [7, 12]$ and $tw_{11} = [10, 15]$. To simplify, independently of the movements and position in $\mathbb{R}^2$ of each node at each instant of time, we will suppose that the transmission range of each node does not change along its activation time window. Thus, if we denote by $< i, k >$ the fact that node i is always in the transmission range of node k and vice versa, independently of the movements and position in $\mathbb{R}^2$, in this example we consider the next pairs: $< 1, 2 >$, $< 1, 3 >$, $< 1, 5 >$, $< 2, 3 >$, $< 3, 4 >$, $< 3, 7 >$, $< 4, 5 >$, $< 4, 7 >$, $< 5, 6 >$, $< 5, 9 >$, $< 6, 9 >$, $< 6, 10 >$, $< 7, 8 >$, $< 8, 11 >$, $< 9, 10 >$ and $< 10, 11 >$.

We will assume that all sending operations will last 1 time unit, i.e. $t_{ij}(t) = 1$ for all j in the transmission range of i and for all t in the activation window of

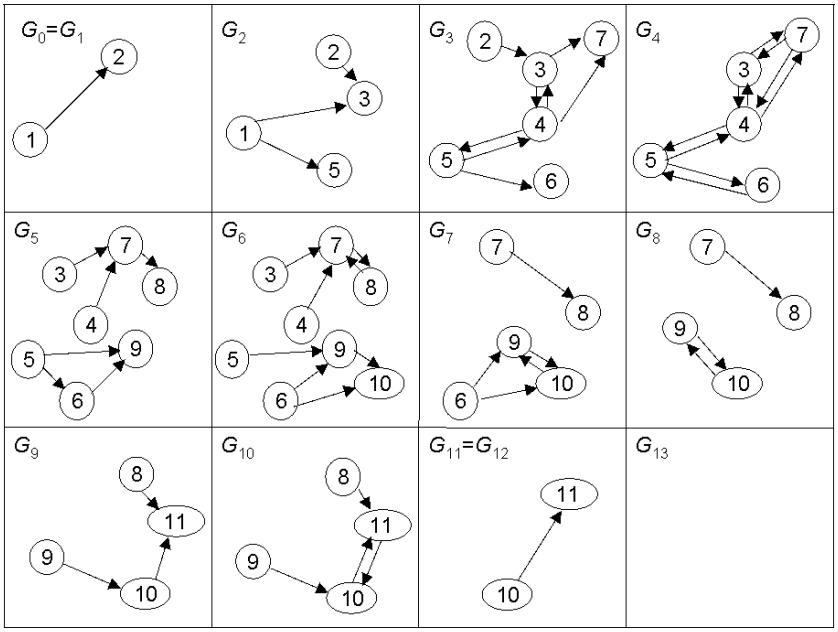

Figure 1: Subgraphs of the EG associated to our example.

i, and that $r_j = 1\ \forall j$.

In this example $T = 15$ and therefore $S_T = \{0, 1, \ldots, 14\}$. Figure 1 shows all the subgraphs G_i $i \in \{0, 1, \ldots, 13\}$. Note that given the size of this figure, inside each vertex we have written i instead of v_i. From the sending times and processing times given above, it is evident that all arcs in all these subgraphs have cost $c_{ij}^t = 2$. In this example we do not consider a different second cost $c_{ij}^{\prime t}$ (we may suppose that both costs are the same). Figure 2 shows the resulting graph G_E of our EG.

In our EG Γ we need to define a particular case of journey:

Definition 3. *Let Γ be an EG as defined above and let $J = (P, R)$ be a journey in Γ with $P = \{(v_{i_1}, v_{i_2}), (v_{i_2}, v_{i_3}), \ldots, (v_{i_s}, v_{i_{s+1}})\}$ with $i_j \neq i_k$ if $j \neq k$ and with $R = \{\tau_{i_1}, \tau_{i_2}, \ldots, \tau_{i_s}\}$, we say that J is a possible journey from v_{i_1} to $v_{i_{s+1}}$ in Γ if $(v_{i_j}, v_{i_{j+1}}) \in A_{\tau_{i_j}}$ and $\tau_{i_{j+1}} = \tau_{i_j} + c_{i_j i_{j+1}}^{\tau_{i_j}}\ \forall j \in \{1, \ldots, s\}$.*

In absence of any kind of interferences, a possible journey from v_{i_1} to $v_{i_{s+1}}$ implies that in our network model, a packet sent from node v_{i_1} at time τ_{i_1}

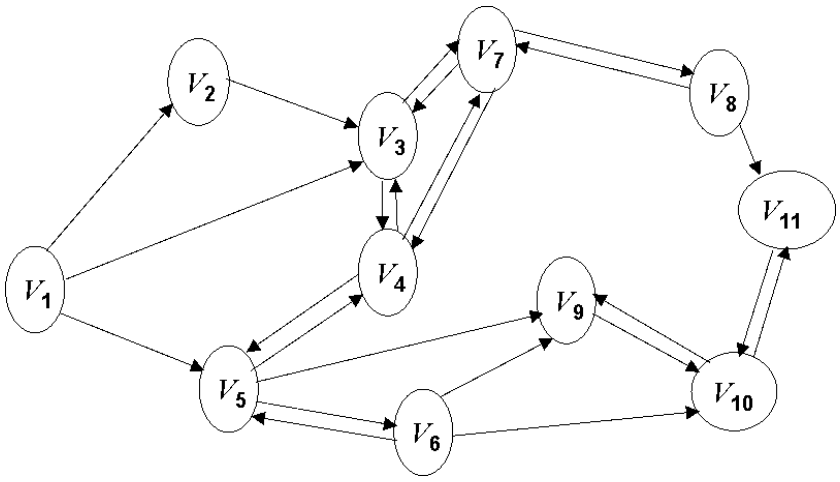

Figure 2: Graph G_E associated to our example.

arrives at node $v_{i_{s+1}}$ at time $\tau_{i_{s+1}} - r_{i_{s+1}}$ and is processed by $v_{i_{s+1}}$.

From figures 1 and 2 it is easy to see that the pair (P, R) with $P = \{(v_1, v_2), (v_2, v_3), (v_3, v_7), (v_7, v_8)\}$ and with $R = \{0, 2, 4, 6\}$ is a possible journey in the EG associated to our example. Also (P, R') with $R' = \{1, 3, 5, 7\}$ is a possible journey. In Section 4, after the application of our procedure to this example, it will be clear that a packet sent by v_1 at time $t = 1$ will arrive at v_8 at time $t = 9$ following the possible journey (P, R'), while a packet sent by v_1 at time $t = 0$ will not arrive at v_8 despite of the existence of a possible journey in Γ between these nodes.

Note that the difference between journey and possible journey takes root in the fact that in our SWN model, a node must forward the packet immediately it has processed the packet; it can not store the packet and to forward it later, each time a node is in its range.

From the definition of possible journey, in absence of any kind of interferences, if we want to know the shortest path followed by a packet sent from a node to another one at a given time, we have to apply a specifically designed modification of the classical Dijkstra's algorithm (Dijkstra, E.W., 1959) to find a shortest path in directed graphs, in a similar way as in (Monteiro et al., 2006).

But from a theoretical point of view, the conditions of our model do not guarantee this absence of interferences; if two or more packets arrive at a node at the same instant of time (which can occur in real SWNs), the node does not forward any of them. Therefore, if we take into account this last fact, we can not use and adaptation of Dijkstra's algorithm individually for each packet.

Moreover, another important difference between our model and those considered in the cited papers involving EGs, is that in those models, each node knows exactly what is going to happen in the network topology, i.e. it knows exactly when some other node will be available or not for communication, so when it receives a packet, as it knows the journey from the source to the destination, if possible, it only has to store the packet until the next node in the journey is available. Although in some SWNs the networks can have a predictable dynamic, in general this assumption is less realistic, and moreover it implies that a node could store hundreds or thousands of packets and that it could forward a lot of packets at the same time, which it is also less realistic. In our model, none of the nodes has information about the rest of them and if a node receives a packet, in absence of interferences it just resends the packet to all nodes inside its transmission range.

Thus we propose a new and more complex procedure that, taking into account the characteristics of our model, in particular the handicaps cited above, it allows to know the path followed by each packet sent by each node during the whole time period $[0, T]$.

4. The Algorithm

This section describes the details of the proposed algorithm, which whole pseudo-code is given in Appendix A.

To make more comprehensible its description, some steps will be executed on the network given in Section 3 to construct an EG. The additional data we need are the messages and the values g_i. We will suppose that $g_i = 2\ \forall i$, that is, each node will forward at most twice the same packet, an that only the following messages are sent:

- v_1 sends to v_{11} a message made of 3 consecutive packets at the instants of time 0, 1 and 2.
- v_3 sends to v_8 a message made of 2 consecutive packets at the instants of time 3 and 4.
- v_5 sends to v_{10} a message made of 2 consecutive packets at the instants of time 3 and 4.

The aim of this algorithm is the construction of a directed graph $G = (V, A)$ with associated vectors to its vertices, from which we can obtain all the desired information.

By default, all variables that are not initially set to a different value must be considered as initialized at 0.

Each vertex v_m^h that appears in the graph $G = (V, A)$ refers to node v_m at the instant of time h. Associated to each vertex v_m^h there is a 4-component vector $te_m^h = (\tilde{m}, \tilde{h}, \hat{m}, \tilde{h}')$. This vector must be interpreted as if node v_m at instant h forwards the packet sent for the first time by node $v_{\tilde{m}}$ at instant $\tilde{h}$, and whose destination is node $\hat{m}$; this packet was initially scheduled to be sent by $v_{\tilde{m}}$ at instant $\tilde{h}'$. Note that $\tilde{h} > \tilde{h}'$ means that the corresponding packet was sent with a delay of $\tilde{h} - \tilde{h}'$ units of time.

The vertices v_m^h that generate packets, will sometime have a second associated vector, ste_m^h, similar to the previous one, and a third one, tte_m^h. These two vectors are used to determine which packet will be eventually sent.

Each vertex v_m^h will have another associated vector $path_m^h$, with at most $n - 1$ components (remember that n is the total number of nodes). The i-th component of $path_m^h$ will indicate the i-th node through which passed the packet that v_m will possibly send at instant h. This vector is used to avoid loops and to count the number of hops made by a path.

Obviously, at the beginning $path_m^h = \emptyset$ for all m and h. The first element of $path_m^h$, when created, indicates the node that generated the packet.

In the algorithm pseudo-code (Appendix A) we will use the symbol $\oplus$ to indicate the addition of a component to vector $path_m^h$; e.g., $(2, 3) \oplus 4 = (2, 3, 4)$.

Also, $c \in path_m^h$ will indicate that one of the components of $path_m^h$ is node c; therefore $m \in path_m^h$ will alert about the creation of a loop.

Each vertex v_m^h will have associated a binary variable $etiq_m^h$ which will be set to 1 if node v_m receives 2 or more packets at instant $h - r_m$. The procedure will only consider this variable for vertices that do not generate a packet. Vertices v_m^h that generate a packet will have associated another variable rec_m^h which will be set to 0 if node v_m does not receive any packet at instant $h - r_m$, or to 1 if it receives a unique packet at instant $h - r_m$, or to 2 if it receives two or more packets at instant $h - r_m$ (interferences). All the $etiq$ and rec variables will be set to 0 at the beginning.

Using $c(v_m^h, v_{m'}^{h'})$ we will indicate the cost of arc $(v_m^h, v_{m'}^{h'})$ in G; in this work this will always coincide with $c'^h_{mm'}$. By $d^+(v)$ we will understand, as normal in graph theory, the number of arcs that comes out from vertex v. Finally a set N will contain all those te_m^h vectors that correspond to packets that could not be sent since their sending instant were delayed to an instant at which the node is inactive and will remain inactive until T.

The algorithm distinguishes two vertex sets, Q and V (V is the vertex set of graph G). In the beginning $V = Q$ and it contains all the vertices v_m^h so that v_m belongs to the subset of nodes that generate their own messages and h is the instant of time at which v_m wants to send one of its packets; therefore Q and V will initially have as much vertices as many locally generated packets that the nodes want to send in the interval $[0, T]$. Figure 3 shows all the possible vertices than can appear throughout the procedure in our example. Initial set V is made of the shaded vertices v_m^h in this figure, with their corresponding vectors te_m^h.

At each iteration the algorithm searches in strictly increasing time order a vertex v_a^b in Q, it removes this vertex from Q and for all nodes v_m such that $(v_a, v_m) \in A_b$, that is, such that the message sent from v_a at instant b will get to v_m and will be processed by v_m, except for interferences, the algorithm adds the vertices $v_m^{b+c_{am}^b}$ to V and Q with their respective vectors $te_m^{b+c_{am}^b}$, and the arcs $(v_a^b, v_m^{b+c_{am}^b})$ to A, or it properly marks them if they already belong to Q, depending on node v_m receives at instant $b + c_{am}^b - r_m$ two or more packets

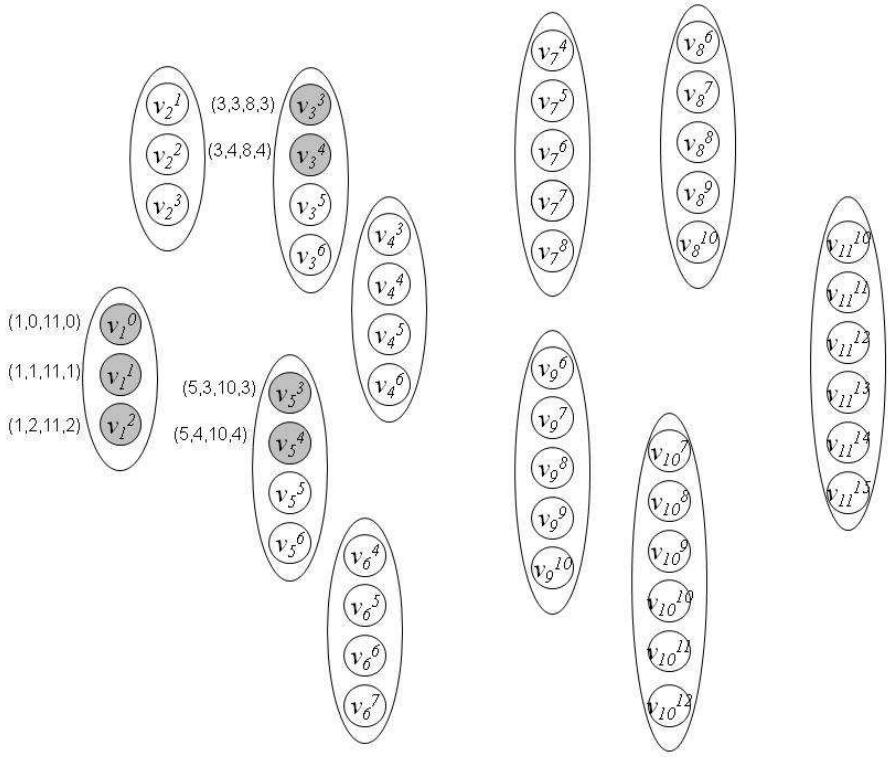

Figure 3: Initial graph (the shaded vertices)

and therefore, due to interferences, it loses the information of those packets or it generate a local packet at instant $b + c^b_{am}$.

If the third vector component of the selected vertex v^b_a is a (i.e., $te^b_a(3) = a$), it means that v_a is the destination of the received packet, then v^b_a will be removed from Q without any further search, unless v_a generates a local packet at $t = b$. Iterations stop when $Q = \emptyset$.

As example of how graph G is being built through the iterations, Figure 4 shows the results on G of the five first iterations (the whole process corresponding to this example is given in Appendix B):

- At Iteration 1, $V \leftarrow V \cup \{v^2_2\}$ and $A \leftarrow A \cup \{(v^0_1, v^2_2)\}$
- At Iteration 2, $V \leftarrow V \cup \{v^3_2\}$ and $A \leftarrow A \cup \{(v^1_1, v^3_2)\}$

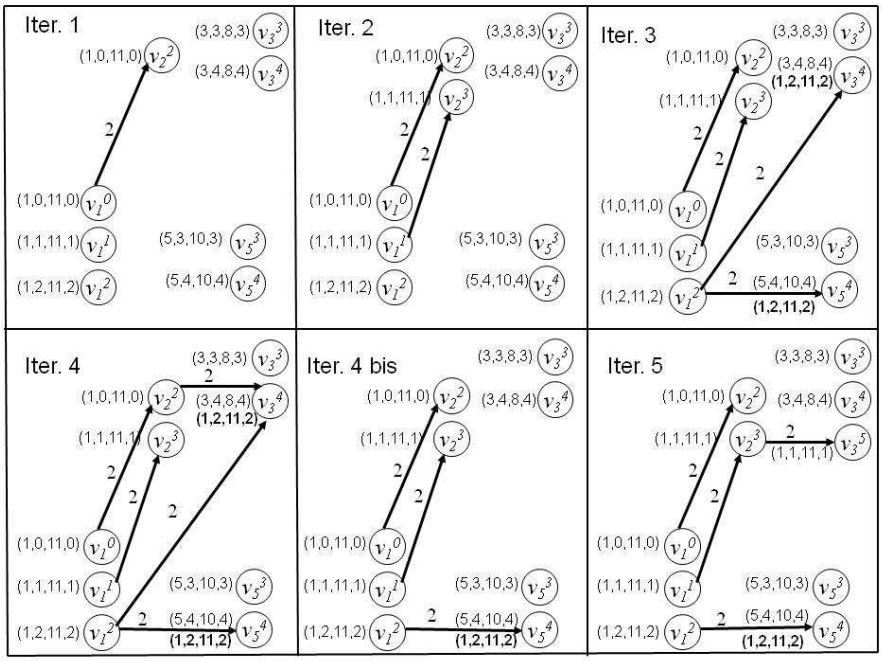

Figure 4: Five first iterations of our procedure over the example.

- At Iteration 3, $A \leftarrow A \cup \{(v_1^2, v_3^4), (v_1^2, v_5^4)\}$, but $v_3^4, v_5^4 \in Q$, then these vertices have associated the additional vectors ste_3^4 and ste_5^4 respectively (in bold font in Figure 4).

- At Iteration 4, we have that two packets arrive to v_3 at the same instant $t = 4$. Therefore both packets are lost at v_3^4, which sends its own packet (at the moment). The result is $A \leftarrow A - \{(v_1^2, v_3^4)\}$

- At Iteration 5, $V \leftarrow V \cup \{v_3^5\}$ and $A \leftarrow A \cup \{(v_2^3, v_3^5)\}$

The whole obtained graph G after 33 iterations corresponding to this first phase is given in Figure 5.

Finally, on a second phase, to reduce as much as possible the size of G we eliminate from V in strict decreasing time order all those vertices v_m^h without leaving arcs ($d^+(v_m^h) = 0$) where v_m does not generate any local packet at instant $t = h$ and $te_m^h(3) \neq m$, which means that v_m is not the destination of

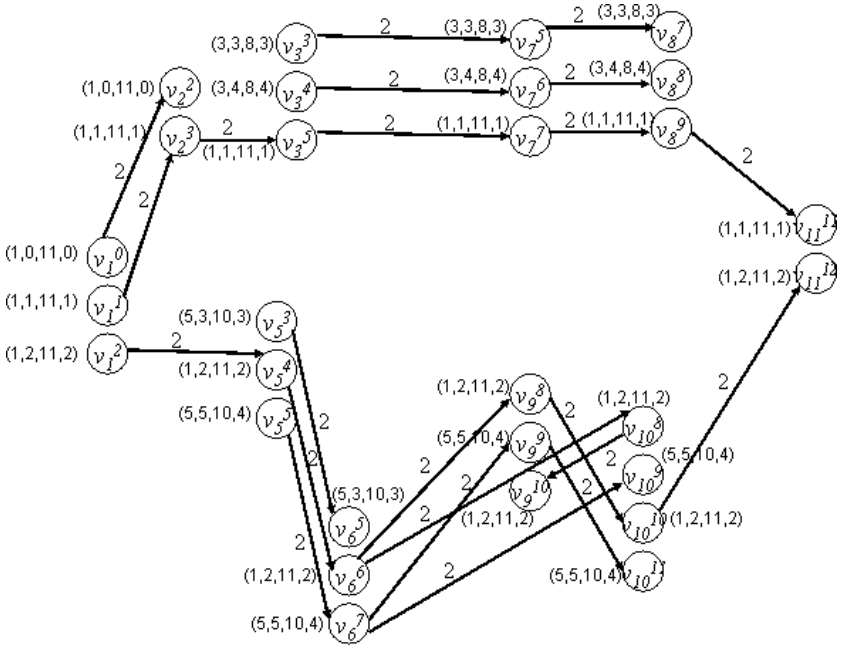

Figure 5: Graph G after first phase.

the received packet at instant $h - r_m$, eliminating at the same time its input arc of A, which can generate more vertices without any leaving arc in G at previous instant. Figure 6 shows graph G in its final form; note that vertices v_9^{10}, v_{10}^{8}, v_6^5 and v_2^2 have been removed.

After the two phases, the resulting directed graph $G = (V, A)$ is acyclic and is formed by:

- Maximum paths (they are not part of a longer path), which go from vertices v_m^h so that v_m belongs to the subset of nodes that generate their own messages and h is the instant of time at which v_m send one of its packets, to vertices $v_{m'}^{h'}$ so that $v_{m'}$ is the destination node of the packet originally send by v_m at $t = h$. Note that a maximum path in G from v_m^h to $v_{m'}^{h'}$ coincides with a possible journey in the EG Γ from v_m to $v_{m'}$ with $\tau_m = h$.

- Various of the previous paths concatenated among them, if it happens that $v_{m'}^{h'}$ is at the same time generator of a local packet.

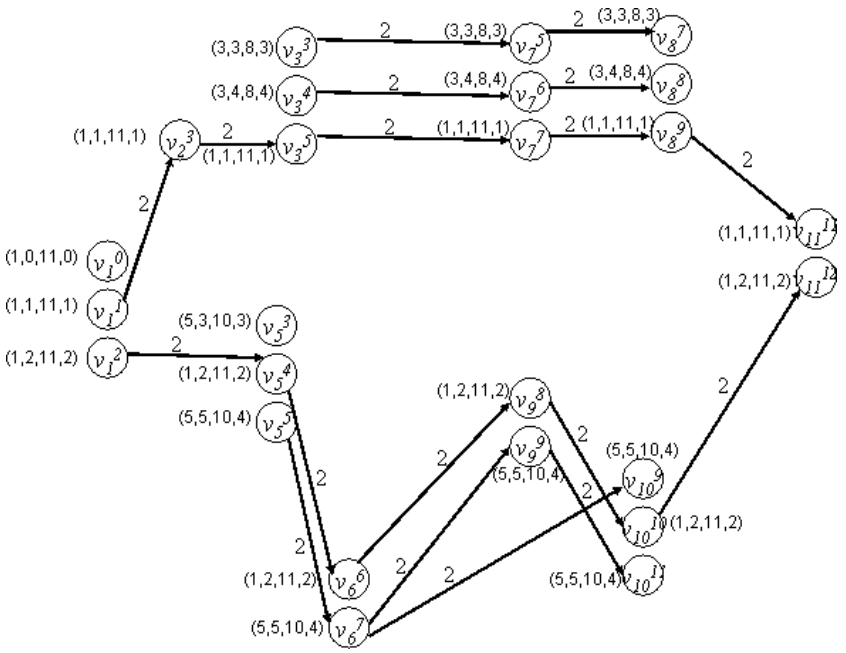

Figure 6: Graph G in its final form (after second phase).

- Isolated vertices v_m^h that correspond to packets sent which did not get to their final destination. With respect to the EG Γ, this fact means either that there is not possible journey in Γ from v_m to its destination with $\tau_m = h$, or that for each possible journey in Γ from v_m to its destination with $\tau_m = h$, there exists a vertex $v_{m'}$ in its path with its corresponding time $\tau_{m'}$ such that in the subgraph $G_{\tau_{m'}}$ there are at least two entering arcs considered by the algorithm, that is, two or more packets arrive to the same node at the same time.

Therefore, two paths in G corresponding to different packets will be vertex-disjoint, except in the case that they are concatenated. In this last case they will have in common the last vertex of the former and the first vertex of the latter.

From the algorithm design and some basic properties of graph theory, it is easy to prove the following statements, which we have included in a unique theorem:

Theorem 1. *Given an SWN, described with all the parameters of Section 2 and given the graph G and the set of vectors N obtained applying the algorithm, it is verified that:*

a) *A packet generated and sent by v_s at instant t will not get to its destination if and only if $v_s^t \in V$ and $d^+(v_s^t) = 0$.*

b) *A packet that was scheduled to be generated and sent by v_s at instant t to v_d, will be definitively sent by v_s at instant t', where $t' \geq t$ if $v_s^{t'} \in V$ and $te_s^{t'} = (v_s, t', v_d, t)$.*

c) *A packet that was scheduled to be generated and sent by v_s at instant t to v_d, will not be sent by v_s at any instant of time if a vector $(v_s, t', v_d, t) \in N$ for some $t' \geq t$.*

d) *If v_s generates and sends a packet at instant t to v_d, this packet will get to its destination at instant t' if it exists in G a path from v_s^t to $v_d^{t'+r_d}$. In this case, the number of components of vector $path_d^{t'+r_d}$ indicates the number of hops made by this path.*

e) *Given a maximum path in G that starts at v_s^t and ends at $v_d^{t'}$, then either $te_s^t(3) = v_d$ or two vertices exist inside that path in G, v_i^h and $v_j^{h'}$ with $t < h \leq h' < t'$ (possibly $i = j$ and $h = h'$) so that $te_s^t(3) = v_i$ and $te_j^{h'}(3) = v_d$.*

Through the information stored into vectors te_d^t we know which, if any, packet gets at each instant to v_d. For example, if we want to know at which instant the packet sent from v_s at instant m will get to v_d, we will have to look for the vertex v_d^h in V, if it exists, so that $h = \min\{l \mid v_d^l \in V$ and $te_d^l = (s, m, d, m')\}$. This packet will get to v_d at instant $h - r_d$. Since it left v_s at instant m, the time required by the packet is $h - r_d - m$ time units. It was initially scheduled to be sent at m', where $m > m'$ means that the packet suffered a delay of $m - m'$ time units. The information about the path followed by the packet is stored in $path_d^h$, and checking the super-indexes of the vertices in the path we can determine at which instant of time the packet has been

forwarded by each node in the path. Also we can easily obtain the cost of this path and the number of hops made.

Item e of the previous theorem means that given a maximum path in G that starts from v_s^t and ends at $v_d^{t'}$, two situations can occur: either v_s generates and send a packet at instant t to v_d, which will receive it at $t' - r_d$, or this maximum path is made of various concatenated paths, as if for example node v_i receives a packet being the final destination at instant $h - r_i$ and at the same time it generates and send a packet at instant h (vertex v_i^h inside the path).

In our example, from Figure 6 we have:

- The packets sent from v_1 at $t = 0$ and from v_5 at $t = 3$ will not get to their destination because v_1^0 and v_5^3 are isolated vertices in G.
- The unique packet that will suffer a delay is the one sent from v_5 at $t = 5$, which was initially scheduled to be sent at $t = 4$. It will get to v_{10} at $t = 8$ $(9 - r_{10})$, through path (v_5, v_6, v_{10}) and at $t = 10$ $(11 - r_{10})$, through path (v_5, v_6, v_9, v_{10}).
- The rest of the sent packets will get only once to their final destination:
  - The packet sent from v_1 at $t = 1$ will get to v_{11} at $t = 10$ $(11 - r_{11})$, through path $(v_1, v_2, v_3, v_7, v_8, v_{11})$.
  - The packet sent from v_1 at $t = 2$ will get to v_{11} at $t = 11$ $(12 - r_{11})$, through path $(v_1, v_5, v_6, v_9, v_{10}, v_{11})$.
  - The packet sent from v_3 at $t = 3$ will get to v_8 at $t = 6$ $(7 - r_8)$, through path (v_3, v_7, v_8).
  - The packet sent from v_3 at $t = 4$ will get to v_8 at $t = 7$ $(8 - r_8)$, through path (v_3, v_7, v_8).

Note that each one of the paths followed by these last packets corresponds to a possible journey in Γ. For example, the packet sent from v_1 at $t = 1$ follows the possible journey $(\{(v_1, v_2), (v_2, v_3), (v_3, v_7), (v_7, v_8), (v_8, v_{11})\}, \{1, 3, 5, 7, 9\})$.

To finish this section, we determine an upper bound for the complexity of the provided algorithm, that is polynomial with respect to the input parameters n and T. To our aim we define $\sigma_i = \sum_{j=1}^{p_i}(t_{2j}^i - t_{2j-1}^i)$ where $i \in \{1, 2, \ldots, n\}$ (then $\sigma = \sum_{i=1}^{n} \sigma_i$).

Theorem 2. *The algorithm has complexity $O(nT^2)$.*

Proof. The number of iterations made by the algorithm in the first phase is bounded above by σ since at each iteration it sets a different v_s^t and it removes it from Q. To know the vertex we have to choose at each iteration, it is enough to order in a list all the vertices v_i^j, first by superindex and then by subindex, and to label a vertex when it enters to Q. It is easy to see that the construction of this list is $O(nT)$.

Once v_a^b is selected, the procedures that impact the complexity of the algorithm are:

- Compute $|\{v_a^l \in V \ \ | \ l < b \text{ and } te_a^l = ste_a^b\}|$ and, if it is the case, delaying the sending of a locally generated packet. The set of this two procedures requires a number of basic operations $O(\sigma_a)$ (we have to check a number of instants l bounded by σ_a).

- Once the EG Γ is constructed (remember that this process is $O(n\sigma)$), the subroutine FORWARD (see detailed pseudo-code) requires just a few operations for each node v_m such that $(v_a, v_m) \in A_b$, therefore the complexity of this subroutine is $O(n)$.

  Thus, once we have the ordered list, the number of basic operations made in the first phase is upper bounded by $O(\sum_{i=1}^{n} \sigma_i^2)$.

Regarding the second phase, a few checks must be done for each vertex in V, where $|V| \leq \sigma$ to determine P (see detailed pseudo-code in Appendix B). P will have less vertices than $|V|$ and to know the vertex we have to choose at each iteration in this phase, we can make use of the ordered list made in the first phase but going from the end to the beginning. As each iteration in the

second phase requires just a few basic operations, the complexity of this phase is $O(\sigma)$.

Note that $\sum_{i=1}^{n} \sigma_i^2 \leq nT^2$ and the construction of the EG Γ is $O(n^2T)$. As in real situations T is much more greater than n, the complexity of the whole process (EG construction and execution of the algorithm) is $O(nT^2)$.∎

5. A study case

We have designed an implementation of the above described algorithm and done a preliminary evaluation of the behaviors of a SWN to better illustrate the possibility of our proposal. The generation of the instances has been done following previous works about the behavior of MANETs (see e.g., (Ferreira et al., 2007; Monteiro et al., 2006)), both in static and dynamic test scenarios. The two basic metrics that we took into consideration were the percentage of delivered packets and the number of hops.

All data corresponding to the generated instances, as well as additional information, have been uploaded to the website `http://www.grc.upv.es/stpa`, to make them easily available to any researcher. The algorithm was implemented in Fortran 95 and was executed on a Pentium Core 2 Quad 2.33 Ghz computer.

5.1. Static case results

The generation of each instance is done randomly. N nodes are generated randomly inside a rectangle of 1000x500 u^2. The chosen radio range is of 250 m. Thus, if the distance between two nodes i and j is less than or equal to 250, node i is in the transmission range of node j and vice versa. Among the n nodes generated m are selected at random to be transmitters and other m to be receivers; m pairs are created so that no node sends to itself, nor we have repeated pairs.

The total study time in each instance is 900 s, being the unit of a millisecond time, so as to have 900,000 time units. Other fixed values for all the instances are: $t_{ij}(t) = 1\ \forall i, j, t$, $r_j = 10\ \forall j$ and $g_j = 2\ \forall j$.

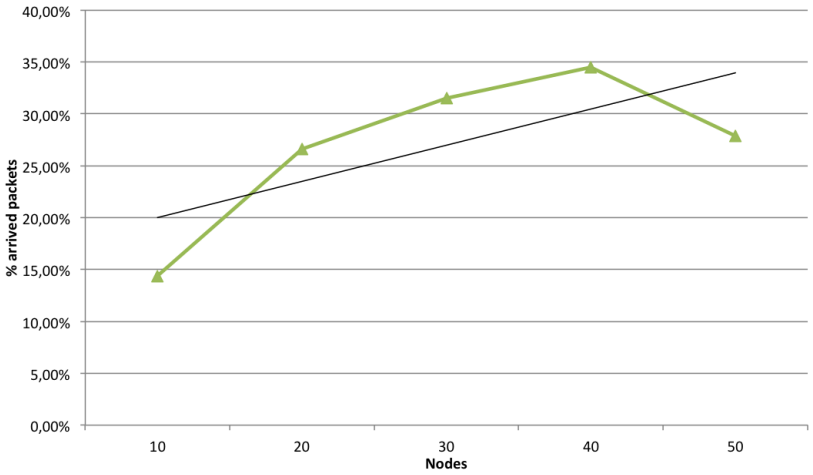

Figure 7: Percentage of delivered packets when varying the overall number of nodes.

A transmission vector of 900,000 positions is created for each node to indicate respectively whether or not to transmit in that moment of time. Each node can transmit its own packets for 10″ and cannot during 50″ for randomly starting a case or the other one. Moreover each node i alternates periods of activity and inactivity, starting with one of them randomly and immediately switches to the opposite state.

First, for a fixed activity/inactivity ratio for nodes of 1.5/0.5 seconds and 5 communication pairs, we generated 5 sets of 20 instances with 10, 20, 30, 40 and 50 nodes respectively. Second, for a fixed number of 50 nodes and 5 communication pairs, we generated 6 sets of 20 instances with activity/inactivity ratios of 1.5/0.5, 3/0.5, 10/0.5, 1.5/1,5, 1.5/3 and 1.5/5 respectively. As in both groups there is a set with the same characteristics (50 nodes and 1.5/0.5 ratio) a total of 200 instances were generated for the static case.

Figures 7 and 8 were obtained from the instance sets corresponding to the variation of the number of nodes; in Figure 7 is represented the average percentage of delivered packets in each set of 20 instances, while in Figure 8 it is represented the average of the maximum, minimum and average number of hops of the delivered packets in each set. We observe that when increasing the

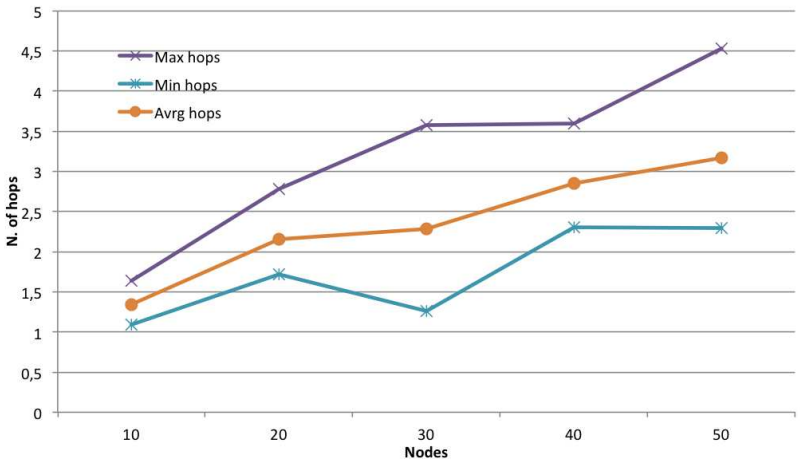

Figure 8: Maximum, minimum and average number of hops of the delivered packets when varying the overall number of nodes.

number of nodes the percentage of delivered packet increases as expected, but the percentage of arrived packet is generally low, the maximum being at 34.5% with 40 nodes. Moreover the path length increases quickly reaching in some situation more that 4 hops, which is a critical value in wireless networks.

Figures 9 and 10 were obtained from the instance sets corresponding to the variation of active/inactive ratios; in Figure 9 it is represented the average percentage of delivered packets in each set, while in Figure 10 it is represented the average of the maximum, minimum and average number of hops of the delivered packets in each set.

We observe that, as a general rule, when keeping the devices active most of the time the percentage of delivered packets increases as expected, even if with a very low percentage of arrived packet. Most importantly, the path length increases when keeping the devices active most of the time, reaching in some situation up to 6 hops, which is a critical value in wireless networks.

Note that the running time of the algorithm was similar in all the instances, and in the worst case less than two seconds.

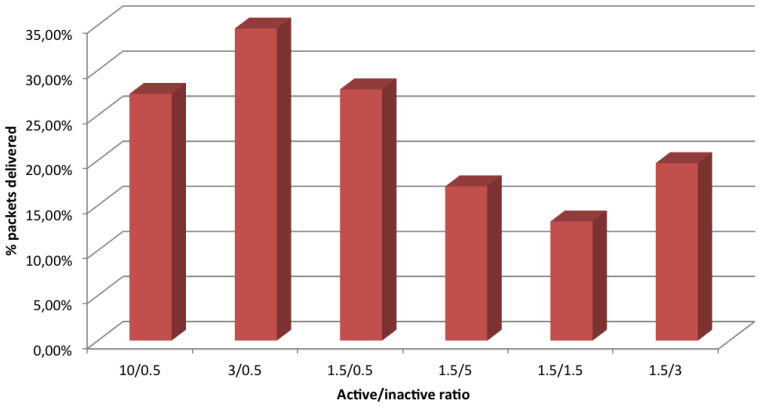

Figure 9: Percentage of delivered packets when varying the ratio between active periods and off periods.

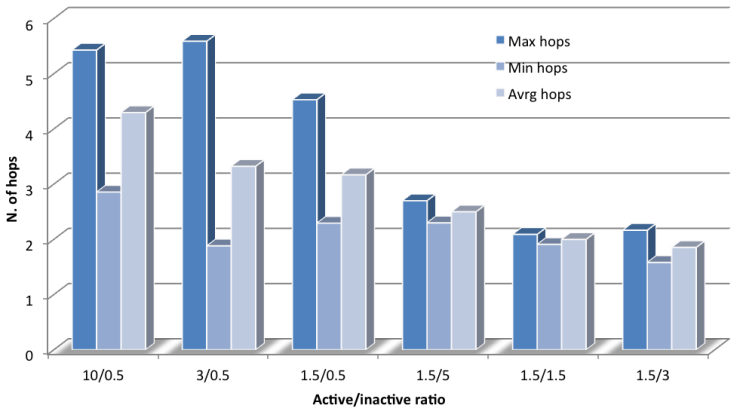

Figure 10: Maximum, minimum and average number of hops of the delivered packets when varying the ratio between active periods and off periods.

5.2. Dynamic case

It is basically the same as in the static case except obviously for connections between nodes that vary every 5″; i.e., every 5″ each position is recalculated for every node taking into account that each node moves with random speed generated by a normal distribution with mean 10 and deviation 4, heading straight to a destination also generated randomly within the grid 1000x500 in which nodes are generated.

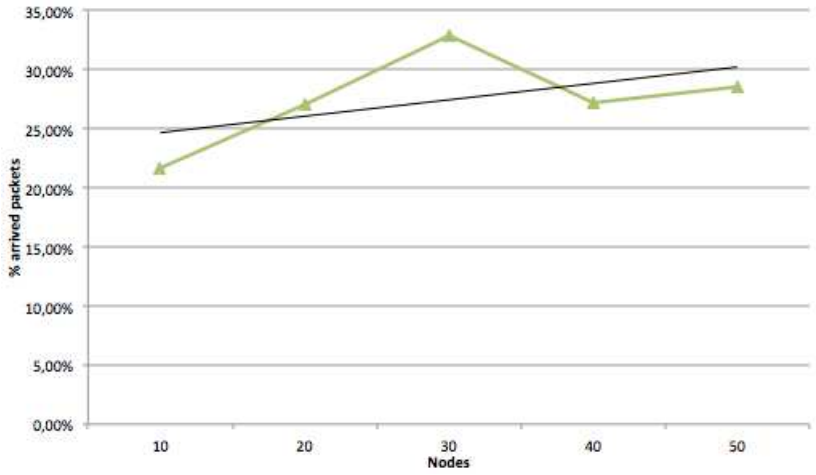

Figure 11: Percentage of delivered packets when varying the overall number of nodes.

We know if a node has reached its destination or not at any given time (multiple of 5″) by determining if the Euclidean distance between the point of origin and the destination is less than or equal to the Euclidean distance between the point of origin and the point at which "find" according to the rated speed. If at that time the node has not reached its destination, will continue in that direction with the same speed, otherwise it will create a new destination and a new speed to start another journey.

As in the static case, we generated 5 sets of 20 instances with 10, 20, 30, 40 and 50 nodes respectively, all of them with fixed activity/inactivity ratio 1.5/0.5 and 5 communication pairs, and also 6 sets of 20 instances with 50 nodes, 5 communication pairs and with activity/inactivity ratios of 1.5/0.5, 3/0.5, 10/0.5, 1.5/1,5, 1.5/3 and 1.5/5 respectively. Thus, a total of 200 instances were also generated for the dynamic case.

Figures 11 and 12 were obtained from the instance sets corresponding to the variation of the number of nodes; in Figure 11 is represented the average percentage of delivered packets while in Figure 12 it is represented the average of the maximum, minimum and average number of hops of the delivered packets. We again observe that, as a general rule, when increasing the number of nodes

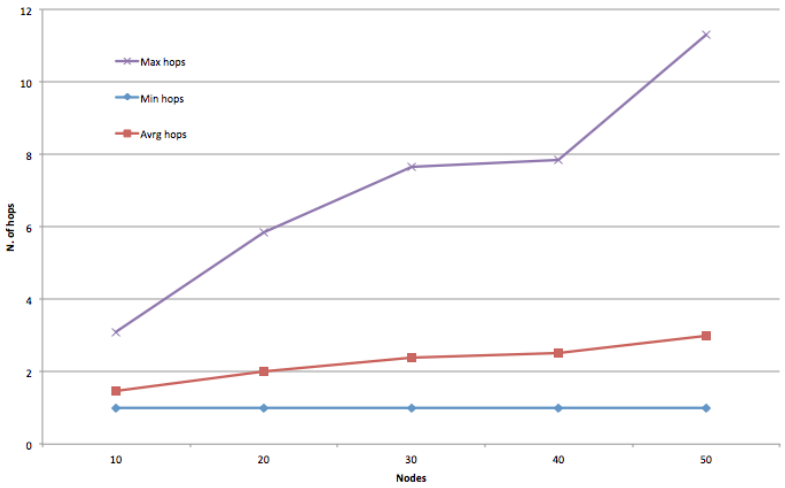

Figure 12: Maximum, minimum and average number of hops of the delivered packets when varying the overall number of nodes.

the percentage of delivered packet increases as expected, but the percentage of arrived packet is generally low, the maximum being at 32,90% with 30 nodes. Moreover the maximum path length increases quickly reaching in some situation more that 11 hops, which is an extremely critical value in wireless networks.

Figures 13 and 14 were obtained from the instance sets corresponding to the variation of active/inactive ratios; in Figure 13 it is represented the average percentage of delivered packets while in Figure 14 it is represented the average of the maximum, minimum and average number of hops of the delivered packets.

Again, we observe that when keeping the devices active most of the time the percentage of delivered packet increases as a general rule, even if with a very low percentage of arrived packet. Most importantly the path length increases when keeping the devices active most of the time, reaching in some situation up to 9 hops.

Note that in the dynamic case, the execution of the algorithm was obviously more time consuming than in the static case, due to the recalculations of positions and connectivity. The worst case had a running time of less than 5 minutes.

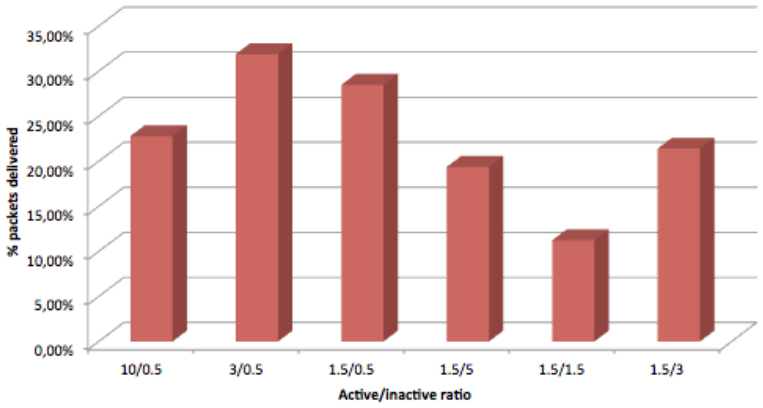

Figure 13: Percentage of delivered packets when varying the ratio between active periods and off periods.

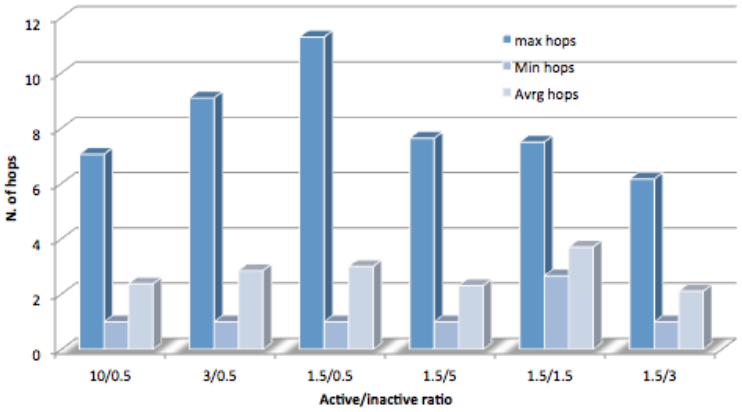

Figure 14: Maximum, minimum and average number of hops of the delivered packets when varying the ratio between active periods and off periods.

What we do extract from these computation results is that in SWNs with the conditions given in this section, routing protocols must deal well with 2 issues: a low percentage of delivered packets and with long paths. Of course, these conclusions cannot be extrapolated to other SWNs with different conditions, but the idea of our model is that it can be configured and "personalized" around a specific problem under study. We have varied the number of nodes and the active/inactive ratios, but the variation of all other parameters involving the

behavior of a SWN cans also be studied with this tool: radio range, transmission range type, periods of transmission, number of times a packet can be resent by the same node, etc. Moreover, the algorithm can be modified to be adapted to other conditions, for example to the existence of buffers, such that a node can store a given number of received packets and resend them, when possible, in the receiving order or in other order depending on preferences.

6. Conclusions

In this paper we proposed an analytical model based on evolving graphs correlated to specifically defined network conditions. This tool provides an exhaustive evaluation of the routing conditions, and its aim is to allow designers to determine the best combination for routing strategies and parameters taking into consideration the network conditions in SWNs, mainly the possible switching-off mechanisms adopted to save energy.

The proposed algorithm determines factors like: how many complete messages get to the destination, which is the smallest amount of time required by a packet to get to the destination, and the fluctuations of the number of hops observed by a packet to get to its destination. Thus, for example, applying this algorithm to different switching-off patterns, through the obtained results with respect to the factors cited above, we can determine the best routing strategy to maintain an adequate performance while still saving energy.

We demonstrated that the complexity of the algorithm is $O(nT^2)$, n being the number of nodes, and T being the width of the time interval studied.

Finally, to better illustrate the possibility of our proposal, we showed some computational results on a set of 400 randomly generated instances, whose data are available to any researcher through the website http://www.grc.upv.es/stpa.

Overall we think that our proposal is a flexible tool that could help routing protocols designers to fine tune their proposals. The flexibility of this work stands in the fact that the assumption that we took for its development can easily be adapted to other different context or scenarios. For example, our

future research will focus on adapting our proposal to model Delay Tolerant Network (DTN) conditions for Intelligent Transport Systems (ITS) settings.

Acknowledgments

This work was partially supported by the *Ministerio de Ciencia e Innovación*, Spain, under Grant TIN2011-27543-C03-01.

Appendix A: Algorithm pseudo-code

Algorithm 1:

First Phase;
`Initialization();`
while $Q \neq \emptyset$ **do**
  $v_a^b := v_m^h$ so that $v_m^h \in Q$ and $h = \min\{l \mid v_s^l \in Q\}$;
  $Q \leftarrow Q \sim \{v_a^b\}$;
  if $te_a^b(1) = a$ *and* $te_a^b(2) = b$ **then**
    if *(*$rec_a^b \neq 1$*) or (*$rec_a^b = 1$ *and* $ste_a^b(3) = a$*) or (*$rec_a^b = 1$ *and* $ste_a^b(1) = a$*) or (*$rec_a^b = 1$ *and* $|\{v_a^l \in V \mid l < b$ *and* $te_a^l = ste_a^b\}| = g_a$*) or (*$rec_a^b = 1$ *and* $a \in path_a^b$*)* **then**
      `Forward();`
    else
      $tte_a^b = te_a^b$ and $te_a^b = ste_a^b$;
      `Forward();`
      delay the sending of v_a at $b' > b$, where b' is the first available instant so that v_a is active at b';
      if $v_a^{b'} \notin Q$ **then**
        $Q \leftarrow Q \cup \left\{v_a^{b'}\right\}$;
        $V \leftarrow V \cup \left\{v_a^{b'}\right\}$;
        $rec_a^{b'} = 0$ and $te_a^{b'} = (a, b', tte_a^b(3), tte_a^b(4))$;
      if $v_a^{b'} \in Q$ **then**
        if *(*$te_a^{b'}(1) \neq a$*) or (*$te_a^{b'}(1) = a$ *and* $te_a^{b'}(2) < b'$*)* **then**
          if $etiq_a^{b'} = 1$ **then**
            $rec_a^{b'} = 2$, $ste_a^{b'} = te_a^{b'}$, $te_a^{b'} = (a, b', tte_a^b(3), tte_a^b(4))$;
            $A \leftarrow A \sim \left\{(u, v_a^{b'}) \mid (u, v_a^{b'}) \in A\right\}$ and $path_a^{b'} = \emptyset$;
          if $etiq_a^{b'} = 0$ **then**
            $rec_a^{b'} = 1$, $ste_a^{b'} = te_a^{b'}$;
            $te_a^{b'} = (a, b', tte_a^b(3), tte_a^b(4))$;
        if $te_a^{b'}(1) = a$ *and* $te_a^{b'}(2) = b'$ **then**
          $tte_a^{b'} = te_a^{b'}$ and $te_a^{b'} = (a, b', tte_a^b(3), tte_a^b(4))$;
          `Delay(`v_a`)`
  if *(*$te_a^b(1) \neq a$*) or (*$te_a^b(1) = a$ *and* $te_a^b(2) < b$*)* **then**
    if *(*$etiq_a^b = 1$*) or (*$etiq_a^b = 0$ *and* $te_a^b(1) = a$*)* **then**
      $V \leftarrow V \sim \{v_a^b\}$;
      $A \leftarrow A \sim \left\{(u, v_a^b) \mid (u, v_m^{b+t_{am}(b)+r_m}) \in A\right\}$;
    if *(*$etiq_a^b \neq 1$ *and* $te_a^b(1) \neq a$*)* **then**
      if $te_a^b(3) \neq a$ **then**
        if $|\{v_a^l \in V \mid l < b$ *and* $te_a^l = te_a^b\}| = g_a$ *or* $a \in path_a^b$ **then**
          $V \leftarrow V \sim \{v_a^b\}$;
          $A \leftarrow A \sim \{(u, v_a^b)\}$ being (u, v_a^b) the unique arc entering in v_a^b;
        else
          `Forward();`
`SecondPhase();`

Procedure `Forward`

begin

forall v_m *with* $(v_a, v_m) \in A_b$ **do**

if $v_m^{b+c_{am}^b} \in Q$ **then**

if *(*$te_m^{b+c_{am}^b}(1) \neq m$*) or (*$te_m^{b+c_{am}^b}(1) = m$ *and* $te_m^{b+c_{am}^b}(2) < b + c_{am}^b$*)* **then**

$A \leftarrow A \cup \left\{(v_a^b, v_m^{b+c_{am}^b})\right\}$;

$etiq_m^{b+c_{am}^b} = 1$;

if $te_m^{b+c_{am}^b}(1) = m$ *and* $te_m^{b+c_{am}^b}(2) = b + c_{am}^b$ **then**

if $rec_m^{b+c_{am}^b} = 0$ **then**

$ste_m^{b+c_{am}^b} = te_a^b$;

$rec_m^{b+c_{am}^b} = 1$;

$A \leftarrow A \cup \left\{(v_a^b, v_m^{b+c_{am}^b})\right\}$;

$path_m^{b+c_{am}^b} = path_a^b \oplus a$;

$c(v_a^b, v_m^{b+c_{am}^b}) := c'^b_{am}$;

if $rec_m^{b+c_{am}^b} = 1$ **then**

$rec_m^{b+c_{am}^b} = 2$, $path_m^{b+c_{am}^b} = \emptyset$;

$A \leftarrow A \sim \left\{(u, v_m^{b+c_{am}^b})\right\}$ being $(u, v_m^{b+c_{am}^b})$ the unique arc entering in $v_m^{b+c_{am}^b}$;

if $rec_m^{b+c_{am}^b} = 2$ **then**

do nothing

if $v_m^{b+c_{am}^b} \notin Q$ **then**

$Q \leftarrow Q \cup \left\{v_m^{b+c_{am}^b}\right\}$;

$V \leftarrow V \cup \left\{v_m^{b+c_{am}^b}\right\}$;

$A \leftarrow A \cup \left\{(v_a^b, v_m^{b+c_{am}^b})\right\}$;

$path_m^{b+c_{am}^b} = path_a^b \oplus a$;

$c(v_a^b, v_m^{b+c_{am}^b}) := c'^b_{am}$;

$te_m^{b+c_{am}^b} = te_a^b$;

end

Procedure `Initialization`

begin

$Q := \{v_m^h \mid$ so that v_m belongs to the subset of nodes that generate local messages and h is the instant of time when v_m should send one of its packets$\}$;

$V := Q$;

$A := \emptyset$;

$te_m^h = (m, h, \tilde{m}, h)$ $\forall v_m^h \in Q$, where $v_{\tilde{m}}$ is the destination node of the packet that v_m wants to send at instant h;

end

Procedure `Delay`(v_a)

begin

Delay the sending of the local packet of v_a forecasted for instant b' to the next instant $b'' > b'$ so that v_a is active at b'', doing exactly as with v_a^b, changing b for b' and b' for b", and so successively until there is no need to delay any local packet of v_a that was to be sent after b;

If, as a consequence of this process, a packet that was to be sent from v_a, cannot be sent because delayed too long and v_a becomes inactive; the vector corresponding to this packet must be stored in N;

end

Procedure `SecondPhase`

begin

$P := \{v_m^h \in V \mid d^+(v_m^h) = 0,\ te_m^h(3) \neq m \ and \ te_m^h(1) \neq m\}$;

while $P \neq \emptyset$ **do**

$v_a^b := v_m^h$ so that $h = \max\{l \mid v_m^l \in P\}$;

$P \leftarrow P \sim \{v_a^b\}$;

$V \leftarrow V \sim \{v_a^b\}$;

$A \leftarrow A \sim \{(u, v_a^b)\}$ being (u, v_a^b) the unique arc entering in v_a^b;

update P;

end

Appendix B: Execution of the algorithm in the example

To simplify the text, we will indicate subroutine Forward with *SF*.

First Phase

Initialitation

$Q := \{v_1^0, v_1^1, v_1^2, v_3^3, , v_3^4, , v_5^3, v_5^4\}$ $V := Q$ $A := \emptyset$

$te_1^0 = (1, 0, 11, 0)$ $te_1^1 = (1, 1, 11, 1)$ $te_1^2 = (1, 2, 11, 2)$ $te_3^3 = (3, 3, 8, 3)$

$te_3^4 = (3, 4, 8, 4)$ $te_5^3 = (5, 3, 10, 3)$ $te_5^4 = (5, 4, 10, 4)$

Iteration 1

$v_a^b = v_1^0$ $Q \leftarrow Q - \{v_1^0\}$

$te_1^0(1) = 1$, $te_1^0(2) = 0$ and $rec_1^0 = 0 \rightarrow SF$

Only $(v_1, v_2) \in A_0$ and $v_2^2 \notin Q \rightarrow$

$Q \leftarrow Q \cup \{v_2^2\}$ $V \leftarrow V \cup \{v_2^2\}$ $A \leftarrow A \cup \{(v_1^0, v_2^2)\}$ $path_2^2 = (1)$

$c(v_1^0, v_2^2) = 2$ $te_2^2 = (1, 0, 11, 0)$

Iteration 2

$v_a^b = v_1^1$ $Q \leftarrow Q - \{v_1^1\}$

$te_1^1(1) = 1$, $te_1^1(2) = 1$ and $rec_1^1 = 0 \rightarrow SF$

Only $(v_1, v_2) \in A_1$ and $v_2^3 \notin Q \rightarrow$

$Q \leftarrow Q \cup \{v_2^3\}$ $V \leftarrow V \cup \{v_2^3\}$ $A \leftarrow A \cup \{(v_1^1, v_2^3)\}$ $path_2^3 = (1)$

$c(v_1^1, v_2^3) = 2$ $te_2^3 = (1, 1, 11, 1)$

Iteration 3

$v_a^b = v_1^2$ $Q \leftarrow Q - \{v_1^2\}$

$te_1^2(1) = 1$, $te_1^2(2) = 2$ and $rec_1^2 = 0 \rightarrow SF$

Only $(v_1, v_3), (v_1, v_5) \in A_2$

moreover, $v_3^4 \in Q$ and $v_5^4 \in Q$. As $te_3^4(1) = 3$ and $te_5^4(1) = 5$

with $te_3^4(2) = 4$, $te_5^4(2) = 4$ and with $rec_3^4 = rec_5^4 = 0 \rightarrow$

$ste_3^4 = (1, 2, 11, 2) \qquad rec_3^4 = 1$

$ste_5^4 = (1, 2, 11, 2) \qquad rec_5^4 = 1$

$A \leftarrow A \cup \{(v_1^2, v_3^4), (v_1^2, v_5^4)\} \qquad path_3^4 = (1) = path_5^4 \qquad c(v_1^2, v_3^4) = c(v_1^2, v_5^4) = 2$

Iteration 4

$v_a^b = v_2^2 \qquad Q \leftarrow Q - \{v_2^2\}$

$te_2^2(1) \neq 2$, $etiq_2^2 = 0$, $te_2^2(3) \neq 2$

$|\{v_2^l \in V \mid l < 2 \text{ and } te_2^l = te_2^2\}| = 0 < g_2$ and $2 \notin path_2^2 = (1) \rightarrow SF$

Only $(v_2, v_3) \in A_2$ and $v_3^4 \in Q$

As $te_3^4(1) = 3$, $te_3^4(2) = 4$ and $rec_3^4 = 1 \rightarrow$

$rec_3^4 = 2 \qquad path_3^4 = \emptyset \qquad A \leftarrow A - \{(v_1^2, v_3^4)\}$

Iteration 5

$v_a^b = v_2^3 \qquad Q \leftarrow Q - \{v_2^3\}$

$te_2^3(1) \neq 2$, $etiq_2^3 = 0$, $te_2^3(3) \neq 2$

$|\{v_2^l \in V \mid l < 3 \text{ and } te_2^l = te_2^3\}| = 0 < g_2$ and $2 \notin path_2^3 = (1) \rightarrow SF$

Only $(v_2, v_3) \in A_3$ and $v_3^5 \notin Q \rightarrow$

$Q \leftarrow Q \cup \{v_3^5\} \qquad V \leftarrow V \cup \{v_3^5\} \qquad A \leftarrow A \cup \{(v_2^3, v_3^5)\} \qquad path_3^5 = (1, 2)$
$c(v_2^3, v_3^5) = 2 \qquad te_3^5 = (1, 1, 11, 1)$

Iteration 6

$v_a^b = v_3^3 \qquad Q \leftarrow Q - \{v_3^3\}$

$te_3^3(1) = 3$, $te_3^3(2) = 3$ and $rec_3^3 = 0 \rightarrow SF$

Only $(v_3, v_4), (v_3, v_7) \in A_3$,

moreover $v_4^5 \notin Q$ and $v_7^5 \notin Q \rightarrow$

$Q \leftarrow Q \cup \{v_4^5, v_7^5\}$ $V \leftarrow V \cup \{v_4^5, v_7^5\}$ $A \leftarrow A \cup \{(v_3^3, v_4^5), (v_3^3, v_7^5)\}$
$path_4^5 = (3) = path_7^5$ $c(v_3^3, v_4^5) = c(v_3^3, v_7^5) = 2$ $te_4^5 = te_7^5 = (3, 3, 8, 3)$

Iteration 7

$v_a^b = v_5^3$ $Q \leftarrow Q - \{v_5^3\}$

$te_5^3(1) = 5$, $te_5^3(2) = 3$ and $rec_5^3 = 0 \rightarrow SF$

Only $(v_5, v_4), (v_5, v_6) \in A_3$,

moreover $v_4^5 \in Q$ and $v_6^5 \notin Q$

$Q \leftarrow Q \cup \{v_6^5\}$ $V \leftarrow V \cup \{v_6^5\}$ $A \leftarrow A \cup \{(v_5^3, v_6^5)\}$ $path_6^5 = (5)$
$c(v_5^3, v_6^5) = 2$ $te_6^5 = (5, 3, 10, 3)$

And as $te_4^5(1) \neq 4 \rightarrow$

$A \leftarrow A \cup \{(v_5^3, v_4^5)\}$ $etiq_4^5 = 1$

Iteration 8

$v_a^b = v_3^4$ $Q \leftarrow Q - \{v_3^4\}$

$te_3^4(1) = 3$, $te_3^4(2) = 4$ and $rec_3^4 = 2 \neq 1 \rightarrow SF$

Only $(v_3, v_4), (v_3, v_7) \in A_4$,

moreover $v_4^6 \notin Q$ and $v_7^6 \notin Q \rightarrow$

$Q \leftarrow Q \cup \{v_4^6, v_7^6\}$ $V \leftarrow V \cup \{v_4^6, v_7^6\}$ $A \leftarrow A \cup \{(v_3^4, v_4^6), (v_3^4, v_7^6)\}$
$path_4^6 = (3) = path_7^6$ $c(v_3^4, v_4^6) = c(v_3^4, v_7^6) = 2$ $te_4^6 = te_7^6 = (3, 4, 8, 4)$

Iteration 9

$v_a^b = v_5^4$ $Q \leftarrow Q - \{v_5^4\}$

$te_5^4(1) = 5$, $te_5^4(2) = 4$, $rec_5^4 = 1$, $ste_5^4(3) \neq 5$, $ste_5^4(1) \neq 5$

$|\{v_5^l \in V \mid l < 4 \text{ and } te_5^l = te_5^4\}| = 0 < g_5$ and $5 \notin path_5^4 = (1) \rightarrow$

$tte_5^4 = te_5^4$ and $te_5^4 = ste_5^4 = (1, 2, 11, 2)$ and SF

Only $(v_5, v_4), (v_5, v_6) \in A_4$,

moreover $v_4^6 \in Q$ and $v_6^6 \notin Q$

$Q \leftarrow Q \cup \{v_6^6\}$ $V \leftarrow V \cup \{v_6^6\}$ $A \leftarrow A \cup \{(v_5^4, v_6^6)\}$ $path_6^6 = (1, 5)$
$c(v_5^4, v_6^6) = 2$ $te_6^6 = (1, 2, 11, 2)$

And as $te_4^6(1) \neq 4 \rightarrow$

$A \leftarrow A \cup \{(v_5^4, v_4^6)\}$ $etiq_4^6 = 1$

By other hand, as $v_5^5 \notin Q$, $Q \leftarrow Q \cup \{v_5^5\}$ $V \leftarrow V \cup \{v_5^5\}$

$rec_5^5 = 0$ $te_5^5 = (5, 5, tte_5^4(3), tte_5^4(4)) = (5, 5, 10, 4)$

Iteration 10

$v_a^b = v_3^5$ $Q \leftarrow Q - \{v_3^5\}$

$te_3^5(1) \neq 3$, $etiq_3^5 = 0$, $te_3^5(3) \neq 3$

$|\{v_3^l \in V \mid l < 5 \text{ and } te_3^l = te_3^5\}| = 0 < g_3)$ and $3 \notin path_3^5 = (1, 2) \rightarrow SF$

Only $(v_3, v_7) \in A_5$ and $v_7^7 \notin Q \rightarrow$

$Q \leftarrow Q \cup \{v_7^7\}$ $V \leftarrow V \cup \{v_7^7\}$ $A \leftarrow A \cup \{(v_3^5, v_7^7)\}$ $path_7^7 = (1, 2, 3)$
$c(v_3^5, v_7^7) = 2$ $te_7^7 = (1, 1, 11, 1)$

Iteration 11

$v_a^b = v_4^5$ $Q \leftarrow Q - \{v_4^5\}$

$te_4^5(1) \neq 4$ and $etiq_4^5 = 1 \rightarrow$ $V \leftarrow V - \{v_4^5\}$ $A \leftarrow A - \{(v_3^3, v_4^5), (v_5^3, v_4^5)\}$

Iteration 12

$v_a^b = v_5^5$ $Q \leftarrow Q - \{v_5^5\}$

$te_5^5(1) = 5$, $te_5^5(2) = 5$ and $rec_5^5 = 0 \rightarrow SF$

Only $(v_5, v_6), (v_5, v_9) \in A_5$,

moreover $v_6^7 \notin Q$ and $v_9^7 \notin Q \rightarrow$

$Q \leftarrow Q \cup \{v_6^7, v_9^7\}$ $V \leftarrow V \cup \{v_6^7, v_9^7\}$ $A \leftarrow A \cup \{(v_5^5, v_6^7), (v_5^5, v_9^7)\}$

$path_6^7 = (5) = path_9^7$ $c(v_5^5, v_6^7) = c(v_5^5, v_9^7) = 2$ $te_6^7 = te_9^7 = (5, 5, 10, 4)$

Iteration 13

$v_a^b = v_6^5$ $Q \leftarrow Q - \{v_6^5\}$

$te_6^5(1) \neq 6$, $etiq_6^5 = 0$, $te_6^5(3) \neq 6$

$|\{v_6^l \in V \mid l < 5 \text{ and } te_6^l = te_6^5\}| = 0 < g_6$ and $6 \notin path_6^5 = (5) \rightarrow SF$

Only $(v_6, v_9) \in A_5$ and $v_9^7 \in Q$. As $te_9^7(1) \neq 9 \rightarrow$

$A \leftarrow A \cup \{(v_6^5, v_9^7)\}$ $etiq_9^7 = 1$

Iteration 14

$v_a^b = v_7^5$ $Q \leftarrow Q - \{v_7^5\}$

$te_7^5(1) \neq 7$, $etiq_7^5 = 0$, $te_7^5(3) \neq 7$

$|\{v_7^l \in V \mid l < 5 \text{ and } te_7^l = te_7^5\}| = 0 < g_7$ and $7 \notin path_7^5 = (3) \rightarrow SF$

Only $(v_7, v_8) \in A_5$ and $v_8^7 \notin Q \rightarrow$

$Q \leftarrow Q \cup \{v_8^7\}$ $V \leftarrow V \cup \{v_8^7\}$ $A \leftarrow A \cup \{(v_7^5, v_8^7)\}$ $path_8^7 = (3, 7)$

$c(v_7^5, v_8^7) = 2$ $te_8^7 = (3, 3, 8, 3)$

Iteration 15

$v_a^b = v_4^6$ $Q \leftarrow Q - \{v_4^6\}$

$te_4^6(1) \neq 4$ and $etiq_4^6 = 1 \rightarrow$ $V \leftarrow V - \{v_4^6\}$ $A \leftarrow A - \{(v_3^4, v_4^6), (v_5^4, v_4^6)\}$

Iteration 16

$v_a^b = v_6^6$ $Q \leftarrow Q - \{v_6^6\}$

$te_6^6(1) \neq 6$, $etiq_6^6 = 0$, $te_6^6(3) \neq 6$

$|\{v_6^l \in V \mid l < 6 \text{ and } te_6^l = te_6^6\}| = 0 < g_6$ and $6 \notin path_6^6 = (1, 5) \rightarrow SF$

Only $(v_6, v_9), (v_6, v_{10}) \in A_6$,

moreover $v_9^8 \notin Q$ and $v_{10}^8 \notin Q \rightarrow$

$Q \leftarrow Q \cup \{v_9^8, v_{10}^8\} \qquad V \leftarrow V \cup \{v_9^8, v_{10}^8\} \qquad A \leftarrow A \cup \{(v_6^6, v_9^8), (v_6^6, v_{10}^8)\}$
$path_9^8 = (1,5,6) = path_{10}^8 \qquad c(v_6^6, v_9^8) = c(v_6^6, v_{10}^8) = 2 \qquad te_9^8 = te_{10}^8 =$
$(1,2,11,2)$

Iteration 17

$v_a^b = v_7^6 \qquad Q \leftarrow Q - \{v_7^6\}$

$te_7^6(1) \neq 7$, $etiq_7^6 = 0$, $te_7^6(3) \neq 7$

$|\{v_7^l \in V \ \mid l < 6 \text{ and } te_7^l = te_7^6\}| = 0 < g_7$ and $7 \notin path_7^6 = (3) \rightarrow SF$

Only $(v_7, v_8) \in A_6$ and $v_8^8 \notin Q \rightarrow$

$Q \leftarrow Q \cup \{v_8^8\} \qquad V \leftarrow V \cup \{v_8^8\} \qquad A \leftarrow A \cup \{(v_7^6, v_8^8)\} \qquad path_8^8 = (3,7)$
$c(v_7^6, v_8^8) = 2 \qquad te_8^8 = (3,4,8,4)$

Iteration 18

$v_a^b = v_6^7 \qquad Q \leftarrow Q - \{v_6^7\}$

$te_6^7(1) \neq 6$, $etiq_6^7 = 0$, $te_6^7(3) \neq 6$

$|\{v_6^l \in V \ \mid l < 7 \text{ and } te_6^l = te_6^7\}| = 0 < g_6$ and $6 \notin path_6^7 = (5) \rightarrow SF$

Only $(v_6, v_9), (v_6, v_{10}) \in A_7$,

moreover $v_9^9 \notin Q$ and $v_{10}^9 \notin Q \rightarrow$

$Q \leftarrow Q \cup \{v_9^9, v_{10}^9\} \qquad V \leftarrow V \cup \{v_9^9, v_{10}^9\} \qquad A \leftarrow A \cup \{(v_6^7, v_9^9), (v_6^7, v_{10}^9)\}$
$path_9^9 = (5,6) = path_{10}^9 \qquad c(v_6^7, v_9^9) = c(v_6^7, v_{10}^9) = 2 \qquad te_9^9 = te_{10}^9 =$
$(5,5,10,4)$

Iteration 19

$v_a^b = v_7^7 \qquad Q \leftarrow Q - \{v_7^7\}$

$te_7^7(1) \neq 7$, $etiq_7^7 = 0$, $te_7^7(3) \neq 7$

$|\{v_7^l \in V \ \mid l < 7 \text{ and } te_7^l = te_7^7\}| = 0 < g_7$ and $7 \notin path_7^7 = (1,2,3) \rightarrow SF$

Only $(v_7, v_8) \in A_6$ and $v_8^9 \notin Q \rightarrow$

$Q \leftarrow Q \cup \{v_8^9\}$ $V \leftarrow V \cup \{v_8^9\}$ $A \leftarrow A \cup \{(v_7^7, v_8^9)\}$ $path_8^9 = (1,2,3,7)$ $c(v_7^7, v_8^9) = 2$ $te_8^9 = (1,1,11,1)$

Iteration 20

$v_a^b = v_8^7$ $Q \leftarrow Q - \{v_8^7\}$

$te_8^7(1) \neq 8$, $etiq_8^7 = 0$ but $te_8^7(3) = 8$ (arrival to its destination)

Iteration 21

$v_a^b = v_9^7$ $Q \leftarrow Q - \{v_9^7\}$

$te_9^7(1) \neq 9$ and $etiq_9^7 = 1 \rightarrow$ $V \leftarrow V - \{v_9^7\}$ $A \leftarrow A - \{(v_5^5, v_9^7), (v_6^5, v_9^7)\}$

Iteration 22

$v_a^b = v_8^8$ $Q \leftarrow Q - \{v_8^8\}$

$te_8^8(1) \neq 8$, $etiq_8^8 = 0$, but $te_8^8(3) = 8$ (arrival to its destination)

Iteration 23

$v_a^b = v_9^8$ $Q \leftarrow Q - \{v_9^8\}$

$te_9^8(1) \neq 9$, $etiq_9^8 = 0$, $te_9^8(3) \neq 9$

$|\{v_9^l \in V \mid l < 8 \text{ and } te_9^l = te_9^8\}| = 0 < g_9$ and $9 \notin path_9^8 = (1,5,6) \rightarrow SF$

Only $(v_9, v_{10}) \in A_8$ and $v_{10}^{10} \notin Q \rightarrow$

$Q \leftarrow Q \cup \{v_{10}^{10}\}$ $V \leftarrow V \cup \{v_{10}^{10}\}$ $A \leftarrow A \cup \{(v_9^8, v_{10}^{10})\}$ $path_{10}^{10} = (1,5,6,9)$ $c(v_9^8, v_{10}^{10}) = 2$ $te_{10}^{10} = (1,2,11,2)$

Iteration 24

$v_a^b = v_{10}^8$ $Q \leftarrow Q - \{v_{10}^8\}$

$te_{10}^8(1) \neq 10$, $etiq_{10}^8 = 0$, $te_{10}^8(3) \neq 10$

$|\{v_{10}^l \in V \mid l < 8 \text{ and } te_{10}^l = te_{10}^8\}| = 0 < g_{10}$ and $10 \notin path_{10}^8 = (1,5,6)$ $\rightarrow SF$

Only $(v_{10}, v_9) \in A_8$ and $v_9^{10} \notin Q \rightarrow$

$Q \leftarrow Q \cup \{v_9^{10}\}$ $V \leftarrow V \cup \{v_9^{10}\}$ $A \leftarrow A \cup \{(v_{10}^8, v_9^{10})\}$ $path_9^{10} = (1, 5, 6, 10)$ $c(v_{10}^8, v_9^{10}) = 2$ $te_9^{10} = (1, 2, 11, 2)$

Iteration 25

$v_a^b = v_8^9$ $Q \leftarrow Q - \{v_8^9\}$

$te_8^9(1) \neq 8$, $etiq_8^9 = 0$, $te_8^9(3) \neq 8$

$|\{v_8^l \in V \mid l < 9 \text{ and } te_8^l = te_8^9\}| = 0 < g_8$ and $8 \notin path_8^9 = (1, 2, 3, 7) \rightarrow$ *SF*

Only $(v_8, v_{11}) \in A_9$ and $v_{11}^{11} \notin Q \rightarrow$

$Q \leftarrow Q \cup \{v_{11}^{11}\}$ $V \leftarrow V \cup \{v_{11}^{11}\}$ $A \leftarrow A \cup \{(v_8^9, v_{11}^{11})\}$ $path_{11}^{11} = (1, 2, 3, 7, 8)$ $c(v_8^9, v_{11}^{11}) = 2$ $te_{11}^{11} = (1, 1, 11, 1)$

Iteration 26

$v_a^b = v_9^9$ $Q \leftarrow Q - \{v_9^9\}$

$te_9^9(1) \neq 9$, $etiq_9^9 = 0$, $te_9^9(3) \neq 9$

$|\{v_9^l \in V \mid l < 9 \text{ and } te_9^l = te_9^9\}| = 0 < g_9$ and $9 \notin path_9^9 = (5, 6) \rightarrow$ *SF*

Only $(v_9, v_{10}) \in A_9$ and $v_{10}^{11} \notin Q \rightarrow$

$Q \leftarrow Q \cup \{v_{10}^{11}\}$ $V \leftarrow V \cup \{v_{10}^{11}\}$ $A \leftarrow A \cup \{(v_9^9, v_{10}^{11})\}$ $path_{10}^{11} = (5, 6, 9)$ $c(v_9^9, v_{10}^{11}) = 2$ $te_{10}^{11} = (5, 5, 10, 4)$

Iteration 27

$v_a^b = v_{10}^9$ $Q \leftarrow Q - \{v_{10}^9\}$

$te_{10}^9(1) \neq 10$, $etiq_{10}^9 = 0$ but $te_{10}^9(3) = 10$ (arrival to its destination)

Iteration 28

$v_a^b = v_9^{10}$ $Q \leftarrow Q - \{v_9^{10}\}$

$te_9^{10}(1) \neq 9$, $etiq_9^{10} = 0$, $te_9^{10}(3) \neq 9$

$|\{v_9^l \in V \mid l < 10 \text{ and } te_9^l = te_9^{10}\}| = 1 < g_9$ and $9 \notin path_9^{10} = (1, 5, 6, 10)$ $\rightarrow$ *SF*

Only $(v_9, v_{10}) \in A_{10}$ and $v_{10}^{12} \notin Q \rightarrow$

$Q \leftarrow Q \cup \{v_{10}^{12}\}$ $V \leftarrow V \cup \{v_{10}^{12}\}$ $A \leftarrow A \cup \{(v_9^{10}, v_{10}^{12})\}$ $path_{10}^{12} = (1, 5, 6, 10, 9)$ $c(v_9^{10}, v_{10}^{12}) = 2$ $te_{10}^{12} = (1, 2, 11, 2)$

Iteration 29

$v_a^b = v_{10}^{10}$ $Q \leftarrow Q - \{v_{10}^{10}\}$

$te_{10}^{10}(1) \neq 10$, $etiq_{10}^{10} = 0$, $te_{10}^{10}(3) \neq 10$

$|\{v_{10}^l \in V \mid l < 10 \text{ and } te_{10}^l = te_{10}^{10}\}| = 1 < g_{10}$ and $10 \notin path_{10}^{10} = (1, 5, 6, 9) \rightarrow SF$

Only $(v_{10}, v_{11}) \in A_{10}$ and $v_{11}^{12} \notin Q \rightarrow$

$Q \leftarrow Q \cup \{v_{11}^{12}\}$ $V \leftarrow V \cup \{v_{11}^{12}\}$ $A \leftarrow A \cup \{(v_{10}^{10}, v_{11}^{12})\}$ $path_{11}^{12} = (1, 5, 6, 9, 10)$ $c(v_{10}^{10}, v_{11}^{12}) = 2$ $te_{11}^{12} = (1, 2, 11, 2)$

Iteration 30

$v_a^b = v_{10}^{11}$ $Q \leftarrow Q - \{v_{10}^{11}\}$

$te_{10}^{11}(1) \neq 10$, $etiq_{10}^{11} = 0$ but $te_{10}^{11}(3) = 10$ (arrival to its destination)

Iteration 31

$v_a^b = v_{11}^{11}$ $Q \leftarrow Q - \{v_{11}^{11}\}$

$te_{11}^{11}(1) \neq 11$, $etiq_{11}^{11} = 0$ but $te_{11}^{11}(3) = 11$ (arrival to its destination)

Iteration 32

$v_a^b = v_{10}^{12}$ $Q \leftarrow Q - \{v_{10}^{12}\}$

$te_{10}^{12}(1) \neq 10$, $te_{10}^{12}(3) \neq 10$ and $|\{v_{10}^l \in V \mid l < 12 \text{ and } te_{10}^l = te_{10}^{12}\}| = 2 = g_{10} \rightarrow V \leftarrow V - \{v_{10}^{12}\}$ $A \leftarrow A - \{(v_9^{10}, v_{10}^{12})\}$

Note that in this case $10 \in path_{10}^{12} = (1, 5, 6, 10, 9)$, so even if node 10 would not forwarded two times the corresponding packet before, to avoid loops it would not forwarded the packet anyway.

Iteration 33

$v_a^b = v_{11}^{12} \qquad Q \leftarrow Q - \{v_{11}^{12}\} = \emptyset$

$te_{11}^{12}(1) \neq 11$, $etiq_{11}^{12} = 0$ but $te_{11}^{12}(3) = 11$ (arrival to its destination)

$Q = \emptyset \leftarrow$ end of the first phase.

Second Phase

$P := \{v_9^{10}, v_6^5, v_2^2\}$

Iteration 1

$v_a^b = v_9^{10} \qquad P \leftarrow P - \{v_9^{10}\} \qquad V \leftarrow V - \{v_9^{10}\} \qquad A \leftarrow A - \{(v_{10}^8, v_9^{10})\}$

$P \leftarrow P \cup \{v_{10}^8\}$

Iteration 2

$v_a^b = v_{10}^8 \qquad P \leftarrow P - \{v_{10}^8\} \qquad V \leftarrow V - \{v_{10}^8\} \qquad A \leftarrow A - \{(v_6^6, v_{10}^8)\}$

Iteration 3

$v_a^b = v_6^5 \qquad P \leftarrow P - \{v_6^5\} \qquad V \leftarrow V - \{v_6^5\} \qquad A \leftarrow A - \{(v_5^3, v_6^5)\}$

Iteration 4

$v_a^b = v_2^2 \qquad P \leftarrow P - \{v_2^2\} = \emptyset \qquad V \leftarrow V - \{v_2^2\} \qquad A \leftarrow A - \{(v_1^0, v_2^2)\}$

$P = \emptyset \leftarrow$ **END**.

Some comments about the obtained results

Regarding the packets sent we observe that:

- v_3 has to send a packet at $t = 4$ and receives 2 packets simultaneously at $t = 4 - r_3$ without being the destination. Due to interferences it will not forward at $t = 4$ any of these packets, sending its own (see iterations 3, 4 and 8).

- v_4 receives 2 packets at $t = 5 - r_4$ without being the destination. Since it had no local packet to be sent at $t = 5$, it will not forward any packet at $t = 5$ (see iterations 6, 7 and 11).

- v_5 must send a packet at $t = 4$, but receives a packet without being the destination at $t = 4 - r_4$; it did not forward this packet previously. It forwards the packet received at $t = 4$ (which was send from v_1 at $t = 2$),

delaying the sending of its own packet at $t = 5$, that is the first available instant of time (see iterations 3 and 9).

- v_4 receives 2 packets at $t = 6 - r_4$ without being the destination. Since it had no local packet to be sent at $t = 6$, it will not forward any packet at $t = 6$ (see iterations 8, 9 and 15).

- v_9 receives 2 packets at $t = 7 - r_9$ without being the destination. Since it had no local packet to be sent at $t = 7$, it will not forward any packet at $t = 7$ (see iterations 12, 13 and 21).

- v_9 receives twice the same packet (sent from v_1 at $t = 2$) at $t = 8$ and $t = 10$ (see iterations 16, 23, 24 and 28).

- v_{10} receives twice the same packet, being the destination (sent from v_5 at $t = 5$) at instants $t = 9 - r_{10}$ and $t = 11 - r_{10}$ (see iterations 18, 26, 27 and 30).

- v_{10} receives three times and at three different instants of time the same packet (sent from v_1 at $t = 2$). It forwards it at $t = 8$ and $t = 10$ but not at $t = 12$ since $g_{10} = 2$. At the same time at $t = 12$ the loop avoidance condition for not forwarding a received packet holds too, since v_{10} exist in a component of vector $path_{10}^{12}$. (see iterations 16, 23, 24, 28, 29 and 32).

References

Buchegger, S., & Le Boudec, J.-Y. (2002). Performance analysis of the confidant protocol. In *Proceedings of the 3rd ACM international symposium on Mobile ad hoc networking & computing* MobiHoc '02, 226–236. New York, NY, USA: ACM.

Cheng, H., Yang, S., & Cao, J. (2012). Dynamic genetic algorithms for the dynamic load balanced clustering problem in mobile ad hoc networks. *Expert Systems with Applications*, *http://dx.doi.org/10.1016/j.eswa.2012.08.050. In Press*.

Dijkstra, E.W. (1959). A note on two problems in connexion with graphs. *Numerische Mathematik, 1*, 269–271.

Ferreira, A. (2004). Building a reference combinatorial model for manets. *IEEE Network*, 24–29.

Ferreira, A., Goldman, A., & Monteiro, J. (2007). On the evaluation of shortest journeys in dynamic networks. *Network Computing and Applications, IEEE International Symposium on*, *1*, 3–10.

Frantti, T. (2012). Expert system for open-loop power control of wireless local area networks. *Expert Systems with Applications*, *39*, 10191–10201.

Frantti, T., & Koivula, M. (2011). Fuzzy packet size control for delay sensitive traffic in ad hoc networks. *Expert Systems with Applications*, *38*, 10188 – 10198.

Gutierrez-Reina, D., Marin, S. T., Johnson, P., & Barrero, F. (2012). An evolutionary computation approach for designing mobile ad hoc networks. *Expert Systems with Applications*, *39*, 6838 – 6845.

Liao, W.-H., Kao, Y., & Wu, R.-T. (2011). Ant colony optimization based sensor deployment protocol for wireless sensor networks. *Expert Systems with Applications*, *38*, 6599 – 6605.

Mengual, L., Marban, O., Eibe, S., & Menasalvas, E. (2012). Multi-agent location system in wireless networks. *Expert Systems with Applications*, *http://dx.doi.org/10.1016/j.eswa.2012.10.037. In Press*.

Monteiro, J., Goldman, A., & Ferreira, A. (2006). Performance evaluation of dynamic networks using an evolving graph combinatorial models. In *IEEE International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob'2006). Montreal, Que. 19-21 June 2006*, 173–180.

Oliveira, R. R., Noguez, F. C., Costa, C. A., Barbosa, J. L., & Prado, M. P. (2012). Swtrack: An intelligent model for cargo tracking

based on off-the-shelf mobile devices. *Expert Systems with Applications*, *http://dx.doi.org/10.1016/j.eswa.2012.10.021. In Press.*

Quintas, D., & Friderikos, V. (2012). Energy efficient spatial tdma scheduling in wireless networks. *Computers & Operations Research*, *39*, 2091–2099.

Santiago, F. M., Lopez, F. A., Montejo-Raez, A., & Lopez, A. U. (2012). Geoasis: A knowledge-based geo-referenced tourist assistant. *Expert Systems with Applications*, *39*, 11737 – 11745.

Tian, J., Haehner, J., Becker, C., Stepanov, I., & Rothermel, K. (2002). Graph-based mobility model for mobile ad hoc network simulation. In *35th Annual Simulation Symposium, San Diego, California*, 337–344.

Torres, R., Mengual, L., Marban, O., Eibe, S., Menasalvas, E., & Maza, B. (2012). A management ad hoc networks model for rescue and emergency scenarios. *Expert Systems with Applications*, *39*, 9554 – 9563.

Yoon, J.-W., & Cho, S.-B. (2012). An intelligent synthetic character for smartphone with bayesian networks and behavior selection networks. *Expert Systems with Applications*, *39*, 11284 – 11292.

Zhou, J., Ji, Z., Takai, M., & Bagrodia, R. (2004). Maya: Integrating hybrid network modeling to the physical world. *ACM Trans. Model. Comput. Simul.*, *14*, 149–169.

***Highlights (for review)**

- An analytical model based on evolving graphs of smartphones based wireless networks is proposed.
- We present an algorithm that provides an exhaustive evaluation of routing conditions when dealing with ON-OFF behavior.
- The presented algorithm can help routing protocol designers to determine the best recurring strategies and parameters.
- Computational results are given, both on static and dynamic scenarios.