Performance Evaluation of Dynamic Networks using an Evolving Graph Combinatorial Model

Julian Monteiro and Alfredo Goldman
Department of Computer Science
University of São Paulo
Rua do Matão 1010, São Paulo, SP, 05508-090, Brazil
Email: {jm,gold}@ime.usp.br

Afonso Ferreira†
CNRS
MASCOTTE Project, INRIA Sophia Antipolis
B.P. 93, F-06902 Sophia Antipolis Cedex, France
Email: afonso.ferreira@sophia.inria.fr

***Abstract*— The highly dynamic behavior of wireless networks make them very difficult to evaluate, e.g. as far as the performance of routing algorithms is concerned. However, some of these networks, such as intermittent wireless sensors networks, periodic or cyclic networks, and low earth orbit (LEO) satellites systems have more predictable dynamics, as the temporal variations in the network topology are somehow deterministic.**

Recently, a graph theoretic model – the *evolving graphs* – was proposed to help capture the dynamic behavior of these networks, in view of the construction of least cost routing and other algorithms. The algorithms and insights obtained through this model are theoretically very efficient and intriguing. However, there is no study on the uses of these theoretical results into practical situations. Therefore, the objective of this work is to analyze the applicability of the *evolving graph* theory in the construction of efficient routing protocols in *realistic* scenarios.

In this paper, we used the *NS2* network simulator to first implement an evolving graph based routing protocol, and then to evaluate such protocol compared to three major ad-hoc protocols (DSDV, DSR, AODV). Interestingly, our experiments showed that *evolving graphs* have all the potentials to be an effective and powerful tool in the development of algorithms for dynamic networks, with predictable dynamics at least. In order to make this model widely applicable, however, some practical issues still have to be addressed and incorporated into the model, like stochastically predictable behavior. We also discuss such issues in this paper, as a result of our experience.

***Index Terms*— Ad hoc wireless networks, sensor networks, evolving graphs, routing protocols, performance analysis**

I. Introduction and Motivation

Wireless communication networks have become increasingly popular in the computing industry and are widely available in our every day life. A promising type of these networks is the Mobile Ad hoc NETwork (MANET), which is a collection of mobile devices that are dynamically connected in an arbitrary manner, without the aid of any established infrastructure or centralized administration [1]. These mobile devices with wireless transmitters are called *nodes*. When two nodes want to communicate, they may not be within each other's range, but they may communicate if other nodes between them also participate in the network, acting as routers, forwarding packets to the other end. These are called multi-hop wireless ad hoc networks.

In several environments these nodes are free to move and they may have nonuniform characteristics, driving the emergence of complex ad hoc networks that may have a highly dynamic behavior [2]. Thus, a large number of routing protocols have been developed for MANETs [3]–[5]. Besides the mobility, such protocols must deal with the typical limitations of these networks, like energy limitations, low processing capacity, low bandwidth, and high error rates [1].

There are different approaches which try to optimize the cost of a routing path, but most of them do not take into account the fact that some networks have a predictable dynamics. This is the case of some networks such as low earth orbit satellite systems (LEO), some wireless sensor networks, periodic or cyclic behavior networks, and others which the network dynamics are somehow deterministic.

LEO satellite networks [6], [7] communicate via inter-satellite links among satellites that are in communication range of each other, and the dynamic topology is fixed. Hence the trajectories of the satellites are known in advance, and it is possible to exploit this determinism in routing optimization.

Wireless Sensor Networks (WSN) [2], [5], [8] is a network composed of hundreds or even thousands of sensor nodes inter-connected and collaborating with each other to perform a specific task, e.g., detection of fire in forests, mapping of bio-complexity of the environment, vehicle tracking and detection, etc. For example, due to energy limitations, network nodes can be scheduled to sleep in given periods, in the aim of energy saving. So, in this case, the network topology changes can also be predicted.

The behavior of these networks can be planned in a deterministic way, henceforth referred to as ***Fixed Schedule Dynamic Networks* (FSDNs)** [9], [10], where the topology dynamics at different time intervals can be predicted (see Fig. 1). Therefore, since the classical graph model hardly apply to these networks, recently, *evolving graphs* (EG) [9]–[14] have been proposed as a formal abstraction for dynamic networks, and can be perfectly suited to the case of FSDNs and the construction of least cost routing and other algorithms. The algorithms and insights obtained through this model are theoretically very efficient and intriguing. However, no study

This work was partially supported by the INRIA-FAPESP project MOBI-DYN

†Currently on leave as Science Officer for ICT at the COST Office, Brussels, BE.

exists on usages of these theoretical results into practical situations.

The purpose of this paper is to analyze the capability of an EG based routing protocol to perform optimally in a FSDN. We have conducted a performance evaluation through extensive simulations using *NS2* within different scenarios. The results are compared using three major ad hoc protocols: DSDV [15], DSR [16] and AODV [17].

The reminder of this paper is organized as follows: In the next section we describe the concept of *evolving graphs*. Section III shows the simulation environment and the decisions made in the implementation. Section IV presents the simulation results and analysis. Finally, we present our conclusions in Section V.

II. Evolving Graph Model

To capture the deterministic behavior of some fixed scheduled dynamic networks (FSDNs), we use the *evolving graph* (EG) model introduced recently. This theory studied in [9]–[11], [14] and aims to represent a formal abstraction of dynamic networks, which formalize a time domain in graphs.

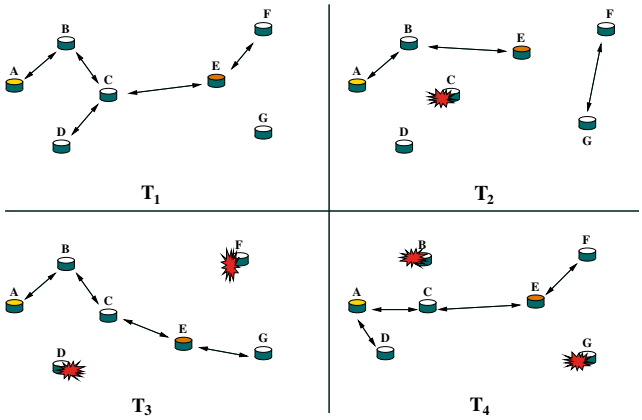


Fig. 1. The evolution of a MANET over time. The indices correspond to successive snapshots

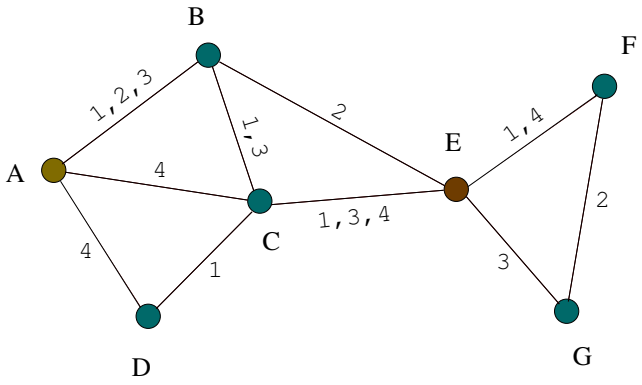


Fig. 2. Evolving graph corresponding to the MANET in Fig. 1. Edges are labeled with corresponding presence time intervals. Observe that $\{E,G,F\}$ is not a valid journey, since the edge $\{G,F\}$ exists only in the past with respect to $\{E,G\}$

As an example, consider the four snapshots taken at different time intervals of a MANET, as despicted in Fig. 1. As one can readily observe, nodes D and G are never connected on a single time interval. Notwithstanding, D can indeed send messages to G, using the *path over time* composed of D, C, E, F, G. Surprisingly, this otherwise trivial fact cannot be directly modeled by usual graphs.

Concisely, an evolving graph is an indexed sequence of τ subgraphs of a given graph, where the subgraph at a given index corresponds to the network connectivity at the time interval indicated by the index number, as shown in Fig. 2.

The time domain is further incorporated into the model by restricting *journeys* (i.e., the equivalent of *paths over time*) to never move into edges which only existed in past subgraphs. A journey in an evolving graph is thus a path in the underlying graph whose edge time-labels are in a non-decreasing order. Now, it is easy to see in Fig. 2 that D, C, E, F, G is a journey, as mentioned above. Further, note that D, C, E, G is also a journey, with less hops, but delivering the message later (in time interval 3 instead of 2), giving raise to different objective functions that may be optimized.

A. Journey Metrics

In the pursue of an optimal journey in FSDN, three metrics have been formalized until now for EG [9]. They are the *foremost*, *shortest*, and *fastest journey*, which find, respectively, the earliest arrival date, the minimum number of hops, and the minimum delay (time span) route. These three parameters can be individually optimized in polynomial time [9].

We use in this paper the *Foremost Journey* algorithm, which computes from a source node s the journeys that arrive the earliest possible on all other nodes. The algorithm to compute such journeys can be seen as an adaptation of *Dijkstra*'s shortest paths algorithm [18], and is detailed below.

B. Foremost Journey Algorithm

Remind that, in order to compute shortest paths, the usual Dijkstra's algorithm proceeds by building a set C of *closed* vertices, for which the shortest paths have already been computed, then choosing a vertex u not in C whose shortest path estimate, $d(u)$, is minimum, and adding u to C, i.e., closing u. At this point, all arcs from u to $V - C$ are *opened*, i.e., they are examined and the respective shortest path estimate, d, is updated for all end-points. In order to have quick access to the best shortest path estimate, the algorithm keeps a min-heap priority queue Q with all vertices in $V - C$, with key d. Note that d is initialized with ∞ for all vertices but for s, which has $d = 0$.

The main observation in Dijkstra's method is that prefix paths of shortest paths are shortest paths themselves. Unfortunately, the prefix journey of a foremost journey is not necessarily a foremost journey (e.g., considering the EG in Fig. 2, a message sent at time interval 1 from A to G can use the journey A, B, E, G, with the packet reaching G at time interval 3. The prefix journey A, B, E will reach E at time

interval 2, although this is not a foremost journey from A to E, which is in fact A, B, C, E, arriving at moment 1).

Notwithstanding, it was proven that there exists at least one foremost journey with such a property in an evolving graph [9], [10].

Therefore, to compute the *Foremost journey* from s to all other nodes, we use a direct adaptation of *Dijkstra*, sketched below, as detailed in [9], [10]:

1) Set $d(s) = 0$, and $d(u) = \infty$ for all other nodes.
2) Initialize *min-heap* Q, sorted by d, with only s in the root.
3) While Q $\neq \emptyset$ do
   a) $x \leftarrow$ root of heap Q.
   b) Delete the root of heap Q.
   c) For each open neighbor v of x do
      i) Compute first valid edge schedule time greater or equal to current time step
      ii) Insert v in the heap Q if it was not there already.
   d) If needed, update $d(v)$ and its key.
   e) Update the heap Q.
   f) Close x. Insert it in the foremost journeys tree.

At the end, we have a tree with the *Foremost journey* traversal path from s to all other nodes.

Note that the computation of the first valid edge schedule done at the inner loop may take into account the traversal time for the edge, i.e., the duration of the transmition can be considered.

The routing protocol originated by this algorithm will be henceforth referred to as EG_{Proto} and is detailed in Section III-A

C. Related Work

The work that has been done until now for the evolving graph theory is restricted to the formalization of a FSDN, and to the exploitation of the model to decrease the complexity of standard algorithms, and to the construction of reference models. The concept of EG and some least cost journeys were detailed in past articles, but, to the best of our knowledge, none experimental simulations was done until now.

III. Simulation Environment

We have conducted our performance analysis using the $NS2$ [19] simulator version 2.29, with the mobile extensions by CMU Monarch [20] which provides IEEE 802.11 Medium Access Control (MAC) protocol [21], and realistic radio and physical layer. The radio model uses characteristics similar to the commercial radio interface, Lucent WaveLAN, modeled as a shared-media radio with a nominal bit rate of 2 MB/s and nominal propagation range of 250m.

In all simulations, **50 nodes** are randomly placed in a 1500m x 500m area. The simulation time is **600 seconds**. A number of **10 constant bit rate (CBR) UDP traffic flows**, are randomly chosen between node pairs. The average **traffic rate is 2 packets/sec** with 256 bytes packet length each. Similar models were used in [1], [22]–[24]. We do not use TCP for the simulations, as we did not want to investigate TCP particularities, which uses flow control, retransmit features and so on. We are looking for a general view of how the routing protocols behaves.

We divided the mobility model in two separated kind of scenarios, one using the popular *random waypoint* model [25], and another even more suitable for FSDNs, called *intermittent model*, as proposed below.

In the ***random waypoint*** scenario, a mobile node alternately pauses a constant PAUSE TIME and moves to a randomly chosen location, with a constant speed, but uniformly randomly chosen from 1 to $19m/s$. The simulation was run with values of PAUSE TIME varying from 0 (continuous motion) to 500 seconds (very low mobility). To avoid known problems of the *random waypoint* model shown in [26], we are using only non-zero values of minimum speed. The use of this classical scenario, yet with its known problems, is important to compare the results with other performance studies.

On the other scenario we constructed a mobility model based on fixed nodes that remain uninterruptedly turning themselves on and off (awake and sleep) in given periods. Here, the parameters we change are the SLEEP PROBABILITY (ranging from 0 to 50%), and the SLEEP TIME (ranging from 6s to 180s). In the beginning of simulation each node is awake and for the entire simulation has a SLEEP PROBABILITY to go to sleep (turning itself off), and remaining in this state for an uniformly randomly chosen SLEEP TIME. Once this time expires, the node is turned on for another randomly chosen time from the same range of SLEEP TIME, and after that starts the process again. We called this scenario ***intermittent model***.

For each evaluated parameter we created 5 random scenarios with different random seeds. Therefore we ran the simulation 40 times for the first model, i.e., 5 times for each value of PAUSE TIME: $0, 5, 25, 50, 125, 250, 375, 500$ seconds, and 180 times for the *intermittent model*, i.e., 5 times for each combination of SLEEP PROBABILITY: $0, 10, 20, 30, 40, 50\%$ and SLEEP TIME: $6, 15, 30, 60, 120, 180$ seconds. Note that in the second scenario we change two parameters, and the value of 0% of SLEEP PROBABILITY means that no one go to sleep, hence the scenario remains static from the beginning to the end of the simulation, and this is relevant to measure the limits of the network. All four routing protocols (AODV, DSDV, DSR, EG_{Proto}) were run on the same 220 scenarios. Thus, identical mobility and traffic scenarios are used across protocols.

The implementations used to evaluate all protocols, with the exception of the EG-based protocol, are the ones provided in the *NS2* package. All of them were implemented by the CMU Monarch group [20]. We used the default parameters and constants. AODV implementation, as of *NS2* version 2.29, contains some further optimization code from [24], [27].

Each simulation in *NS2* generates a trace file, containing all communications that have been done between nodes, including the MAC layer. Afterwards this file is analyzed to consolidate the results.

A. EG_{Proto} - Evolving Graph Based Routing Protocol

Mobility in *NS2* is usually represented by a script in a separated file used by the simulation. Therefore, we made a program that reads the mobility model used by *NS2* to capture the node movements, and to generate a corresponding EG, which is used as input for the foremost journey *evolving graph* based protocol (EG_{Proto}).

Before the simulation begins, this EG of the system is distributed among the nodes. So, each node in the simulation knows exactly what is going to happen in the network topology. It knows exactly when some other node will be available or not for communication.

At first sight, this important assumption may not appear to be realistic. However, there are many situations, e.g. those shown in [2], [5], [6], in which an EG can be built before the routing phase.

B. Implementation of EG_{Proto}

If all nodes – this can be changed for transmitting nodes only – have the knowledge of the EG, the implementation of the routing algorithm is straightforward.

Let *edge schedules* be a set of time intervals representing the existence of link-connectivity between two nodes. An *edge* exists when two nodes are in the range of each other. The *Evolving Graph* (EG) of a dynamic network can be represented by a list of *edge schedules* for each pair of nodes. Thus, each node has a list of its neighbors at a given time.

When a packet arrives at node u, the node computes the *foremost journey* (as shown in Section II-B) from packet source to the destination. If the next node v in the journey is available (i.e, there is an edge in *edge schedule* at that moment), then the packet is forwarded to v. If v is not reachable, the node u schedules the transmission of the packet to the earliest feasible edge present in *edge schedules* of (u, v).

TABLE I
EDGE SCHEDULES FOR THE EG IN FIG. I.

Node pair	Edge schedules
A – B	1, 2, 3
A – C	4
A – D	4
B – C	1, 3
B – E	2
C – D	1
C – E	1, 3, 4
E – F	1, 4
E – G	3
F – G	2

In Table I we show the corresponding *edge schedules* as shown in the example of Fig. 2. Note that the *edge schedules* are the presence time intervals at the labels.

As an example, the foremost journey for a message sent at time index 1 from D to G will be D, C, E, F, G. The packet will reach F at the same time index 1, then afterwards F will schedule to send the packet to G at the earliest edge presence in the *edge schedules* of $F - G$. Hence the packet will be sent by F at time index 2.

If two routes have the same time length when computing the *foremost journey*, the one with less number of hops will be chosen for routing.

C. Other Routing Protocols

A great deal of work has been produced comparing the performance of the three major ad hoc protocols, DSDV, DSR, and AODV [1], [22]–[24]. Consequently, we chose these three to be compared to EG_{Proto} .

The first one, Destination-Sequenced Distance Vector (DSDV) described in [15], is a *proactive* table-driven protocol based on the distributed Bellman-Ford algorithm, with the loop-freedom improvement. Each node has a routing table for all reachable nodes, which stores the next-hop and number of hops to destination. DSDV requires periodical broadcasts to update the routing table.

Dynamic Source Routing (DSR) [16] is a *reactive* protocol, allowing nodes to dynamically discover a route to destination. These routes are stored in a route cache. Source routing means that each packet carries in its header the complete ordered list of nodes to the destination (path).

The last protocol, Ad-hoc On-Demand Distance Vector Routing (AODV) [17] is based on DSDV and DSR. AODV is also a *reactive* protocol, which requests a route when needed, and maintain a traditional routing table to destinations in use. A routing table entry is *expired* if not used recently.

There are many other routing protocols [4], [5], [7], [8] with specialized characteristics. We are not going to compare EG_{Proto} to them here, because this first experiment aims to compare EG_{Proto} with massively tested and analyzed protocols, as are the three above. Unfortunately, we could not include comparisons to Optimized Link State Routing Protocol (OLSR) [28] due to space ant time limitations. Such a comparison will be provided in the journal version of this paper.

D. NS2 Implementation Details

In *NS2*, the act of sleep and wake a node used by the *intermittent model* scenarios are done using the commands *on* and *off* already implemented in the mobile agent. But the last version of *NS2* did not work correctly with this command. To do so, we remove the line that do a `reset-state` in the command `off` in the `mobilenode.cc` file.

We have not yet tested the new energy model extension proposed in [29] that adds full control to the radio state, providing sleep and turn on commands to the mobile node.

When using the *intermittent scenario*, some problems appeared with DSDV protocol, which stays in infinite loop when managing some internal queues. This is a well known misbehavior of DSDV, and the workaround was to uncomment a block of code that dequeue all packets in the queue in the $dsdv.cc$ file.

IV. SIMULATION RESULTS

Our results are based in a simulation of an ad hoc network composed by wireless mobile nodes moving around, going to sleep a while, and communicating with each other.

As well as mobility models, the results shown here are separated in two parts, one using the *random waypoint* mobility model, and another using *intermittent model*.

We focused our analysis in four main metrics:

- *Average throughput*: The average number of packets received per amount of time by *all nodes*;
- *Average end-to-end delay*: The average time between sending and receiving time for packets *received*;
- *Ratio of dropped packets by no route (NRTE)*: Fraction of dropped packets by *no available route* per total number of sent packets;
- *Ratio of dropped packets by interface link queue overflow (IFQ)*: Fraction of dropped packets by *link queue overflow* per total number of sent packets;

The EG_{Proto} used in these experiments do not have any protocol overhead (there are no control messages since all nodes already have the EG of the whole network), hence we did not investigate the overhead metric.

A. Random Waypoint Mobility Model

As mentioned earlier, in the random waypoint scenario the parameter we are changing is the PAUSE TIME. Low values of PAUSE TIME means *high* mobility and high values of PAUSE TIME means *low* mobility. As shown in Fig. 3, the EG_{Proto} performance has better values of throughput for all pause times, besides the fact that most protocols got very close values in this metric (we omitted the DSDV due to its very low values of up to 13000b/s). Actually, as expected, EG_{Proto} produced the best results in this metric in all simulations scenarios.

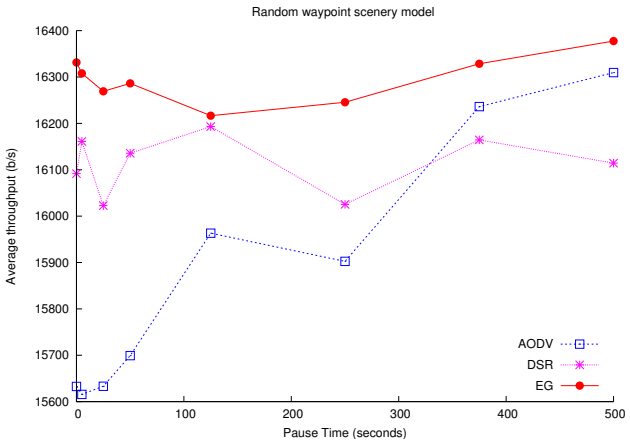


Fig. 3. Average throughput as a function of PAUSE TIME (mobility)

In Fig. 4, the number of dropped packets for the EG_{Proto} is almost zero for all PAUSE TIMES, that is, less than 0.5% of packet loss. The ratio of dropped packets for DSR is pretty good too, an average of 2.5% packet loss. It is surprising the fact of AODV did not perform well at high mobility values, opposed as shown by Perkins, Royer, Das, and Marina in [24]. We attributed this behavior to the very low network load of our simulation (2 pkts/s with 10 traffic sources) or to unknown adjusts in algorithm parameters.

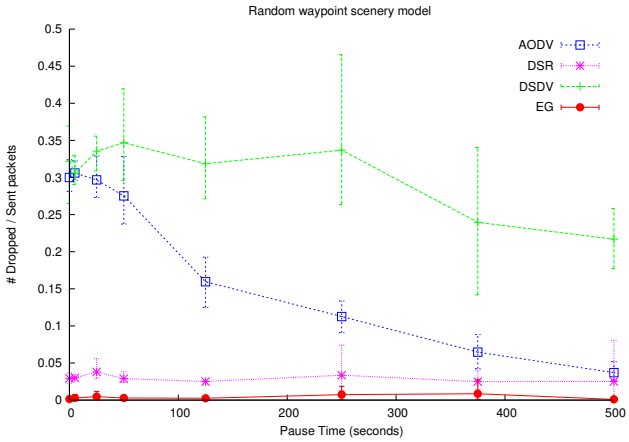


Fig. 4. Drop ratio as a function of PAUSE TIME (mobility)

B. Intermittent Mobility Model

The intermittent mobility scenario is more realistic in the sense of FSDN networks, on which the nodes have fixed position and their on/off dynamics can be more easily predicted.

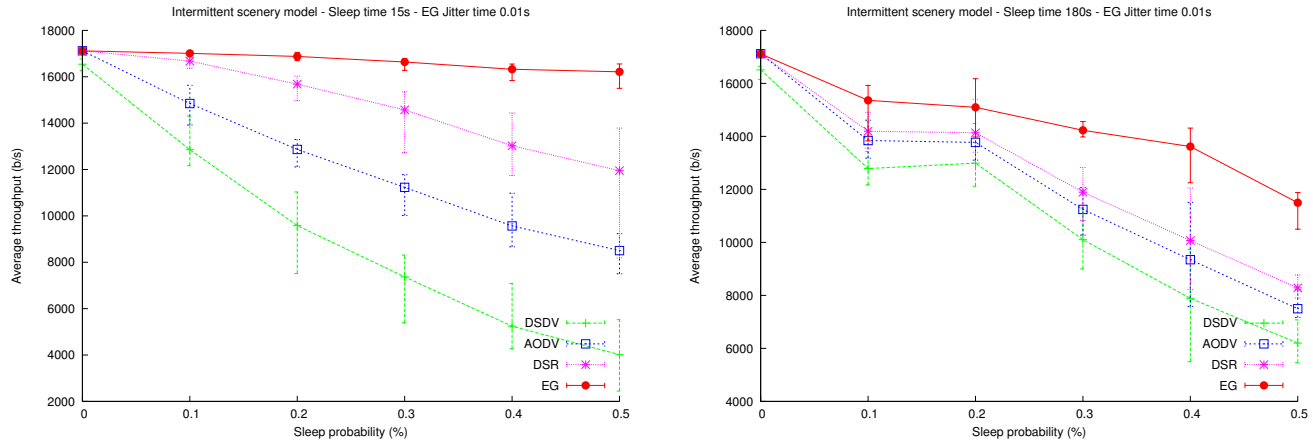
In the next results we change the values of SLEEP PROBABILITY from 0 to 50%. High probability to sleep means that the network has a low connectivity, i.e., a lot of nodes are disconnected from each other because many of them are in sleep state. The values of SLEEP TIME means dynamism, i.e., how often the connections among nodes changes, low values of SLEEP TIME means high dynamics and vice versa.

In Fig. 5, we show again that EG_{Proto} has better values of throughput. But, in the case of high dynamics (SLEEP TIME 15s), the throughput of EG_{Proto} is almost constant for all connectivity scenarios. On the other hand, in the case of low dynamics, the values of throughput for EG decreases, with the others protocols, as the connectivity decreases (from 0 to 50% SLEEP PROBABILITY).

This EG_{Proto} decline is due primary to a high number of inexistent routes, as shown in the graph of Fig. 6(b). The high values of dropped packets by no available route (NRTE) means that a high number of nodes are disconnected, therefore the throughput decreases.

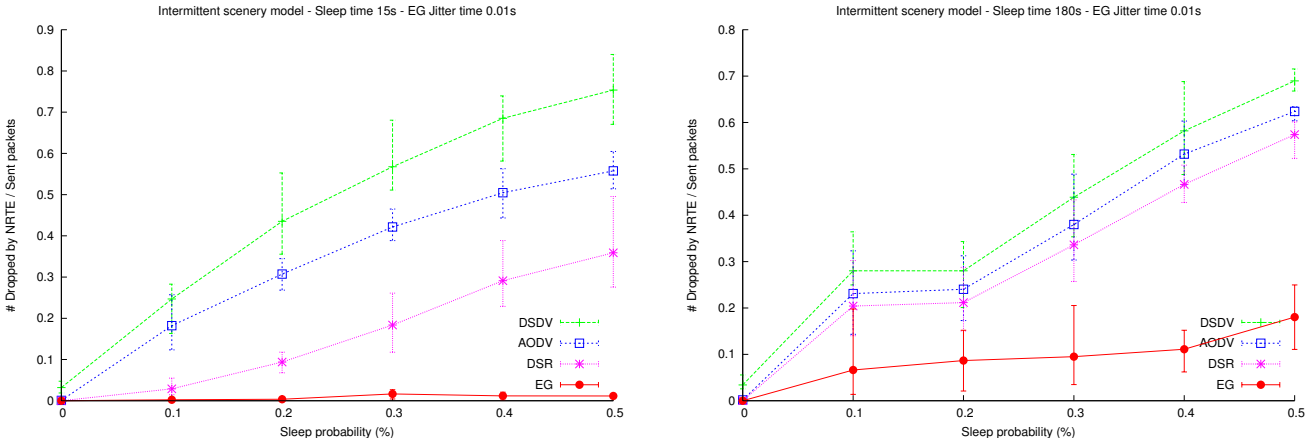
The intrinsic behavior of EG_{Proto} to schedule packets to be sent when some node awakes arises the problem of *bottlenecks*, on which a lot of packets are scheduled to be sent in the same moment, and the link interface queue (IFQ) does not hold that incoming traffic. To minimize this problem we increased the default length of IFQ from 50 to 500 packets. The simulation done using the *random waypoint model* does not suffer from this effect, due to the high mobility of the system, the *bottlenecks* do not exist at all. This characteristic appeared in the low connectivity and low dynamics scenarios of the *intermittent model*, on which the nodes in the evolving graph remain disconnected for long time periods.

In Fig. 7(a) we see the high values of dropped packets by IFQ overflow on a low connectivity scenario. These high values negatively affect the throughput of EG_{Proto} as already



(a) SLEEP TIME 15s (b) SLEEP TIME 180s

Fig. 5. Average throughput as a function of SLEEP PROBABILITY (connectivity).



(a) SLEEP TIME 15s (b) SLEEP TIME 180s

Fig. 6. Drop packets by NRTE ratio as a function of SLEEP PROBABILITY (connectivity).

shown in Fig. 6(b).

The problem of IFQ overflow can be minimized with the use of randomly chosen jitter time when sending packets to the network. But, in order to be able to perceive the improvement, the values of jitter time must be at least 0.5 seconds, and this is not feasible for regular routing applications. In Fig. 7 we show that the drop rate by IFQ decreases at least 6 times when we add a randomly chosen jitter on sent time (ranging from 0 to 0.5s).

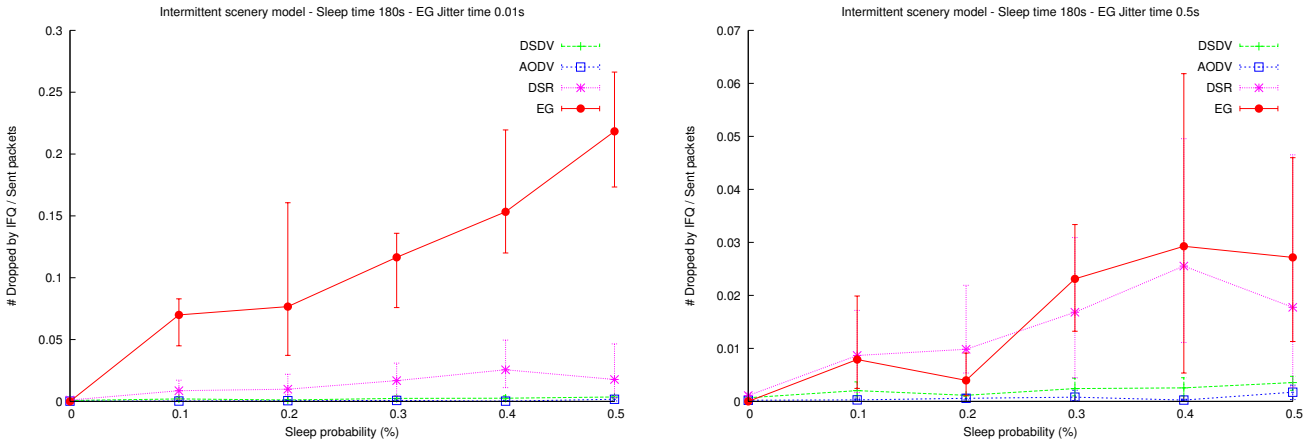
Another characteristic of EG_{Proto} is the high values of the average end-to-end delay, as shown in Fig. 8. Again, due to the behavior of EG_{Proto} scheduler, some packets may await for a long time upon an edge exists, and this wasted time is computed by the metric. In another words, even using the *Foremost Journey* EG based routing algorithm, the packets could take long in time to reach the destination. As it is proven in [9], using EG the packets will reach the destination as soon as possible if the journey exists in any moment. Moreover, on the other protocols the packets would just be dropped. Hence, there is a tradeoff between low drop rates and average end-to-end delay values.

The values of EG_{Proto} on the Fig. 6 shows that the number of dropped packets by NRTE using EG_{Proto} is a lower bound value for all protocols, i.e., when EG_{Proto} drop a packet by NRTE, means that the requested path does not exist in any moment. Therefore, EG_{Proto} can be used as a benchmark to measure how good the other protocols are performing.

V. CONCLUSION

Our contribution in this paper is to show that an EG based routing protocol is well suited for networks with predictable dynamics, and the theory as a whole is a powerful tool for the development of ad hoc networking protocols.

The EG based protocol has been formalized to provide optimal routing according to its metrics. We implemented the *Foremost Journey* and performed extensive simulations using *NS2*. We have compared the performance of the new EG_{Proto} with three major ad hoc protocols: DSDV, DSR, and AODV.



(a) Jitter time 0.01s

(b) Jitter time 0.5s

Fig. 7. Drop by IFQ overflow on a SLEEP TIME 180s as a function of SLEEP PROBABILITY (connectivity). Observe that the right graph (jitter 0.5s) has a number of dropped packets at least 6 times less than left graph values.

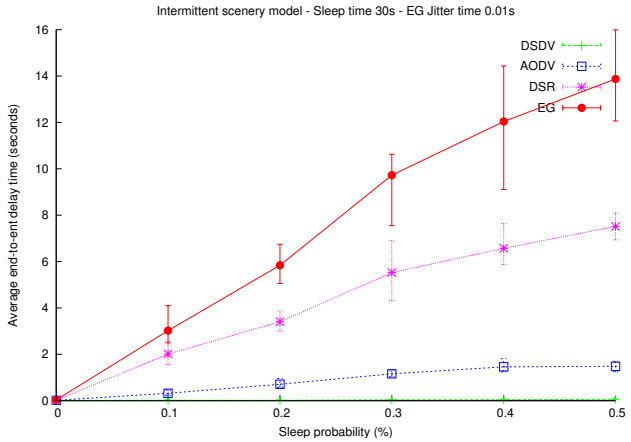


Fig. 8. Average end-to-end delay on SLEEP TIME 30s as a function of SLEEP PROBABILITY (connectivity)

The results showed that the throughput of the network using EG was the highest, compared to all protocols in all metrics. For the number of dropped packets by no available route (NRTE), EG_{Proto} got the best values as expected. We have shown that EG values can be used as a lower bound in this last metric.

This first implementation of an EG based protocol pointed out some topics to be studied in more detail. For instance, a *bottleneck* problem happens when an important node become unavailable for long time, causing a huge overhead when it comes back, and packets are dropped due to collision and queue overflows. The use of high values of jitter time when sending packets can minimize the drop rate, but is not feasible in regular protocols. The development of a good EG adaptative algorithm could manage that, perceiving these congested nodes to find out alternative routing paths.

It should be noted that the high values of average end-to-end delay is an inherent characteristic of the communication network dynamics. In the case of EG based protocols, on which the *foremost journey* metric are studied, the end-to-end delay is in any case the minimum arrival date for a packet. If long delays need to be managed, then one policy could be to drop packets that are aged in the network, or – even better – use a *fastest delay* approach instead of the *foremost journey* as done here.

Future work includes implementation of other EG based protocols with different metrics, like *shortest path* and *fastest delay*. A natural extension to this work is related to the deviations in the predicted network dynamics, on which the actual EG used by the nodes is not accurate anymore. This engenders the utilization of a model with stochastically predictable behavior to better address such variations.

ACKNOWLEDGMENT

The authors would like to thank Aubin Jarry for his help in the development of this paper.

REFERENCES

[1] S. Corson and J. Macker, "Mobile ad hoc networking (MANET): Routing protocol performance issues and evaluation considerations," IETF, RFC 2501, January 1999. [Online]. Available: http://www.ietf.org/rfc/rfc2501.txt

[2] D. Estrin, R. Govindan, J. S. Heidemann, and S. Kumar, "Next century challenges: Scalable coordination in sensor networks," in *Proceedings of the 5th international conference on Mobile Computing and Networking (MobiCom)*, 1999, pp. 263–270. [Online]. Available: citeseer.ist.psu.edu/estrin99next.html

[3] D. Lang, "A comprehensive overview about selected ad hoc networking routing protocols," Technische Universitaet Munchen, Department of Computer Science, Tech. Rep. TUM-I0311, March 2003. [Online]. Available: http://wwwbib.informatik.tu-muenchen.de/infberichte/2003/TUM-I0311.pdf

[4] E. M. Royer and C.-K. Toh, "A review of current routing protocols for ad-hoc mobile wireless networks," *IEEE Personal Communications*, pp. 46–55, Apr 1999. [Online]. Available: citeseer.ist.psu.edu/royer99review.html

[5] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "Wireless sensor networks: a survey," *Computer Networks*, vol. 38, pp. 393–422, 2002. [Online]. Available: citeseer.ist.psu.edu/akyildiz02survey.html

[6] A. Ferreira, J. Galtier, and P. Penna, "Topological design, routing and hand-over in satellite networks," in *Handbook of Wireless Networks and Mobile Computing*, I. Stojmenovic, Ed. John Wiley and Sons, 2002, pp. 473–507.

[7] M. Werner and G. Maral, "Traffic flows and dynamic routing in leo intersatellite link networks," in *Proceedings of the International Mobile Satellite Conference (IMSC)*, June 1997, pp. 283–288.

[8] D. Estrin, L. Girod, G. Pottie, and M. Srivastava, "Instrumenting the world with wireless sensor networks," in *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing (ICASSP), Salt Lake City, Utah*, May 2001. [Online]. Available: citeseer.ist.psu.edu/estrin01instrumenting.html

[9] B. Bui-Xuan, A. Ferreira, and A. Jarry, "Computing shortest, fastest, and foremost journeys in dynamic networks," *International Journal of Foundations of Computer Science*, vol. 14, no. 2, pp. 267–285, April 2003. [Online]. Available: http://www.inria.fr/rrrt/rr-4589.html

[10] A. Ferreira, "Building a reference combinatorial model for dynamic networks: Initial results in evolving graphs," INRIA, Research Report 5041, Dec 2003. [Online]. Available: http://www.inria.fr/rrrt/rr-5041.html

[11] B. Bui-Xuan, A. Ferreira, and A. Jarry, "Evolving graphs and least cost journeys in dynamic networks," in *Proceedings of WiOpt – Modeling and Optimization in Mobile, Ad-Hoc, and Wireless Networks*. INRIA Press, March 2003, pp. 141–150.

[12] S. Bhadra and A. Ferreira, "Computing multicast trees in dynamic networks using evolving graphs," INRIA, Research Report 4531, Ago 2002. [Online]. Available: http://www.inria.fr/rrrt/rr-4531.html

[13] ——, "Complexity of connected components in evolving graphs and the computation of multicast trees in dynamic networks," in *Proceedings of the International Conference on AD-HOC Networks and Wireless (Adhoc-Now)*, ser. Lecture Notes in Computer Science, vol. 2865. Springer Verlag, Oct 2003, pp. 259–270.

[14] A. Ferreira and A. Jarry, "Complexity of minimum spanning tree in evolving graphs and the minimum-energy broadcast routing problem," in *Proceedings of WiOpt – Modeling and Optimization in Mobile, Ad-Hoc and Wireless Networks*, mar 2004.

[15] C. Perkins and P. Bhagwat, "Highly dynamic destination-sequenced distance-vector routing (DSDV) for mobile computers," in *Proceedings of ACM SIGCOMM Conference on Communications Architectures, Protocols and Applications*, 1994, pp. 234–244. [Online]. Available: citeseer.ist.psu.edu/perkins94highly.html

[16] D. B. Johnson and D. A. Maltz, "Dynamic source routing in ad hoc wireless networks," in *Mobile Computing*, Imielinski and Korth, Eds. Kluwer Academic Publishers, 1996, vol. 353, pp. 153–181. [Online]. Available: citeseer.ist.psu.edu/johnson96dynamic.html

[17] C. E. Perkins and E. M. Royer, "Ad-hoc on-demand distance vector routing," in *Proceedings of the 2nd IEEE Workshop on Mobile Computing Systems and Applications (WMCSA)*, February 1999, pp. 90–100. [Online]. Available: http://www.research.att.com/conf/wmcsa99/papers/perkins.ps.gz

[18] T. H. Cormen, C. E. Leiserson, and R. L. Rivest, *Introduction to Algorithms*. Cambridge, Massachusetts: The MIT press, 1990.

[19] The VINT Project, "Network simulator – ns2," http://www.isi.edu/nsnam/ns/, Page accessed on Dec 2005.

[20] Rice University Monarch Project, "The CMU monarch wireless and mobility extensions to NS," http://www.monarch.cs.rice.edu/, Page accessed on Dec 2005.

[21] "IEEE standard for wireless LAN medium access control (MAC) and physical layer (PHY) specifications," http://grouper.ieee.org/groups/802/11/main.html, Page accessed on Dec 2005.

[22] J. Broch, D. A. Maltz, D. B. Johnson, Y.-C. Hu, and J. Jetcheva, "A performance comparison of multi-hop wireless ad hoc network routing protocols," in *Proceedings of the 4th international conference on Mobile Computing and Networking (MobiCom)*, 1998, pp. 85–97. [Online]. Available: citeseer.ist.psu.edu/broch98performance.html

[23] P. Johansson, T. Larsson, N. Hedman, B. Mielczarek, and M. Degermark, "Scenario-based performance analysis of routing protocols for mobile ad-hoc networks," in *Proceedings of the 5th international conference on Mobile Computing and Networking (MobiCom)*, 1999, pp. 195–206.

[24] C. E. Perkins, E. E. Royer, S. R. Das, and M. K. Marina, "Performance comparison of two on-demand routing protocols for ad hoc networks," in *IEEE Personal Communications*, Feb 2001, vol. 8, pp. 16–28. [Online]. Available: citeseer.ist.psu.edu/article/perkins01performance.html

[25] T. Camp, J. Boleng, and V. Davies, "A survey of mobility models for ad hoc network research," *Wireless Communications and Mobile Computing (WCMC): Special issue on Mobile Ad Hoc Networking: Research, Trends and Applications*, vol. 2, no. 5, pp. 483–502, 2002. [Online]. Available: citeseer.ist.psu.edu/camp02survey.html

[26] J. Yoon, M. Liu, and B. Noble, "Random waypoint considered harmful," in *Proceedings of IEEE INFOCOM*, 2003, pp. 1312–1321. [Online]. Available: citeseer.ist.psu.edu/yoon03random.html

[27] "AODV code for CMU wireless and mobility extensions to NS-2," http://www.cs.sunysb.edu/ mahesh/aodv/, Page accessed on Dec 2005.

[28] P. Jacquet, P. Mühlethaler, T. Clausen, A. Laouiti, A. Qayyum, and L. Viennot, "Optimized link state routing protocol," in *Proceedings of the 5th IEEE International Multitopic Conference (INMIC'01)*. IEEE, December 2001, pp. 62–68. [Online]. Available: http://hipercom.inria.fr/olsr/inmic2001.ps

[29] V. Kakadia and W. Ye, "Energy model update in ns-2," http://www.isi.edu/ilense/software/smac/ns2_energy.html, Page accessed on Dec 2005.