Proc. of the Second Intl. Conf. on Advances in Information Processing and Communication Technology - IPCT 2015

ISBN: 978-1-63248-044-6 doi: 10.15224/ 978-1-63248-044-6-125

Netlogo, Agent-based tool for Modeling and Simulation of Routing Problem in Ad-hoc Networks

Akram Kout Said Labed Salim Chikhi

***Abstract*— Mobile Ad hoc network (MANET) is an autonomous system of mobile hosts (nodes) connected by wireless link forming a temporary network without the aid of any established infrastructure or centralized administration. Typical applications of MANETs are: emergency and rescue operations, disaster relief efforts, military operations and exploration mission where cellular infrastructure is unavailable. The main problem of mobile ad hoc networks is to design routing protocols allowing for communication between the hosts. The dynamic nature of ad hoc networks makes this problem especially challenging. Communication in MANET is multi-hop due to limited transmission range; this decentralized operation relies on the cooperative participations of all nodes. MANETs are considered as complex system characterized by high dynamic topology, local interactions, auto-organization and emergence. Modeling and simulation are very important in the design and development of distributed interacting system because of their particular stochastic nature. This article seeks to use agent-based tools for modeling ad hoc network. We focus on Netlogo, an important tool in the modeling and simulation domain of complex system. We have successfully implemented distributed Dijkstra's shortest path algorithm to solve the routing problem. Obtained Results show the quick convergence of Dijkstra's Algorithm to shortest paths relating a source node with all accessible destinations.**

***Keywords*— Mobile Ad-hoc Networks, Routing, Modeling and simulation, Distributed Dijkstra's Algorithm.**

I. Introduction

Mobile ad hoc network (MANET) is a collection of wireless mobile nodes dynamically forming a temporary network without the use of any centralized administration. The challenge in MANETs is to find a path between communicating nodes. Such type of networks is characterized by the absence of centralized infrastructure, dynamic topology, the constraint of energy, the heterogeneity of nodes, multi-hop communication and limited bandwidth.

Akram Kout
MISC Laboratory
Constantine University 2, Algeria

Said Labed
MISC Laboratory
Constantine University 2, Algeria

Salim Chikhi
MISC Laboratory
Constantine University 2, Algeria

Routing in MANET is extremely challenging. In order to efficiently transmit data to destinations, the applicable routing algorithms must be implemented in mobile ad-hoc networks. Thus we can increase the efficiency of the routing by satisfying the Quality of Service (QoS) parameters by developing routing algorithms for MANETs.

A complex system consists of a number of subunits that can be described in terms of relatively simple rules: the interaction between these units led to the emergence of sophisticated features which cannot be explained in terms of individual properties. In the case of a wireless ad-hoc network, the properties of the individual units (nodes) are relatively simple, but a collection of these nodes do not necessarily realize an efficient functional network. It is therefore desirable to induce self-organization using protocols that control the overall behavior of the system through the knowledge of the information flow between its parts. Existing algorithms in conventional networks are based on this principle, but most algorithms rely on the knowledge of the global information (eg. routing tables that describe all possible routes between nodes), by contrast, algorithms that take advantage of self-organization in ad hoc networks are based on paradigms that tend to reduce the dependence of the global information: they get the desired effect, based on local information or probabilistic methods. As paradigms evolve centralized control to a purely self-organized system, the degree of determinism in the algorithms decreases, and at the same time the evolution of the network increases. An ad-hoc network can be considered as a complex system consisting of a number of mobile agents coupled via radio links, so many collective properties associated with such systems are presented or can be induced. Moreover, the properties of other types of complex networks such as social networks, or certain biological networks can be designed in ad-hoc networks. Since MANET could be viewed as a distributed, decentralized and self-organized system, we use Netlogo simulator to demonstrate the global behavior emergence in such networks. To this end, we implemented a distributed version of Dijkstra's shortest path algorithm.

The rest of the paper is organized as follows: Section II reviews related works. In Section III, we discuss the Routing in Mobile Ad-hoc Networks while Section IV outlines network models using graph theory. Next, in Section V, we present the distributed Dijkstra's shortest path algorithm and its modifications for finding stable paths. Section VI shows the simulation, outcomes and discusses the obtained results. Finally, Section VII concludes the paper with some future directions.

Proc. of the Second Intl. Conf. on Advances in Information Processing and Communication Technology - IPCT 2015

ISBN: 978-1-63248-044-6 doi: 10.15224/ 978-1-63248-044-6-125

II. Related Works

The first idea of using multi-agent programming language for modelling and simulating mobile ad hoc networks is inspired by work presented in [2]. [2] pinpoints shortcomings of traditional network simulators, when used for simulating complex computer networks such as pervasive computing, large scale peer-to-peer networks involving considerable environment. Author proposes the use of NetLogo for modelling and simulating complex computer networks, describes NetLogo programming language and its highlights features. In order to demonstrate effectiveness of NetLogo, the author has created four elementary models of complex computer networks. However, it is well known that creating computer models of mobile ad hoc network with all details about network-state remains impossible and generally undesirable. Models of real world are therefore created with certain level of granularity. Paper [3] demonstrated that poorly selected level of granularity, which neglects some important details, has very serious impact on the accuracy of simulation results. Therefore, authors in [3] recommend independently analysing, designing and optimising needed level of mobile ad hoc network model granularity according to intention of created model and according to results which model should provide.

Main critical factor affecting the results of mobile ad hoc network simulation is mobility model of nodes. Mobility model of nodes, as defined in [4], "is a set of rules used to generate trajectories for mobile entities." This paper demonstrates critical impact of chosen model of mobility on results obtained from simulation of mobile ad hoc network. Badly chosen mobility model of nodes leads to incorrect simulation results. Standard mobility models are stochastic. However, papers [5][6] have shown the harmful impact of stochastic mobility models to overall simulation analysed and demonstrated.

Currently, the new trend is to use more realistic mobility models as done in [7].

Recently, authors in [8] have demonstrated the appropriateness of NetLogo for modelling and simulating high level aspects of mobile ad hoc networks. Specifically, they used NetLogo for simulating and evaluating security criteria of various public key infrastructure approaches in mobile ad hoc network.

III. Routing in Mobile Ad-hoc Networks

Routing in ad hoc networks is the most important problem. Because of nodes mobility, locating a destination at a given moment becomes complicated. So, a routing protocol is used in order to discover paths between communicating nodes (see Fig. 1). Routing protocols for MANETs can be separated into three main categories, namely: proactive, reactive and hybrid protocols that combine proactive and reactive operation.

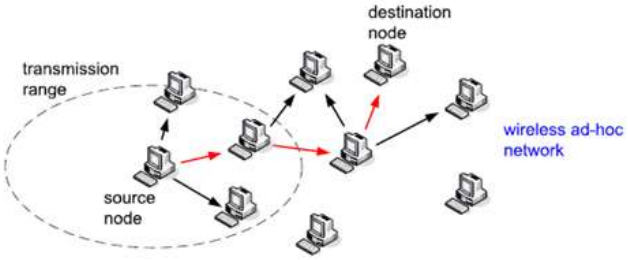


Fig 1. Mobile Ad-hoc Network

The routing strategy is fundamental for mobile ad hoc networks. It must be done in a rational manner, i.e. with minimal control and good conservation of bandwidth. Furthermore, the changes in topology must be taken into account. For this, a hundred routing protocols have been proposed

- Proactive protocols: control packets are constantly broadcast on the network to maintain the state of the link between each pair of nodes. At each node a table is constructed where each entry indicates the next hop to a certain destination. The most important routing protocols of this class are: Destination Sequence Distance Vector (DSDV) and Optimized Link State Routing (OLSR). The main limit of this category is its weakness against path conservation.
- Reactive routing protocols, (or on demand), that create and maintain the paths as needed. When a node needs a route, a global discovery procedure of paths is launched to obtain a valid path to the destination. The most important routing protocols of this class are: Ad hoc On demand Distance Vector (AODV) and Dynamic Source Routing (DSR). The main limits of this category are: Delay caused by a search before transmission, Path expiration after a certain time.
- Hybrid protocols combine the two approaches. They use a proactive protocol, to learn the close neighborhood (on two or three hops), and they have the paths in the immediate neighborhood. Beyond the predefined area, the hybrid protocol uses the techniques of reactive protocols to look for routes between non-neighboring nodes (more than two or three hops). The most important routing protocol of this class is Zone Routing Protocol (ZRP).

IV. Modeling of Mobile Ad-hoc Network

A MANET can be modeled as a graph G (V, E) where V is the set of vertices and E is the set of edges. Two vertices (nodes) of a graph are connected only if there is a communication link between them. Once a MANET is represented as a graph, the next question that arises is whether a graph property has implications for MANET. Table 1. represents the different network models :

Proc. of the Second Intl. Conf. on Advances in Information Processing and Communication Technology - IPCT 2015

ISBN: 978-1-63248-044-6 doi: 10.15224/ 978-1-63248-044-6-125

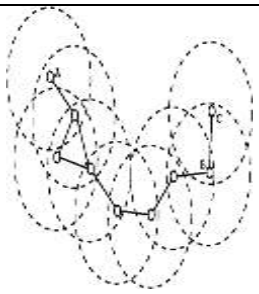
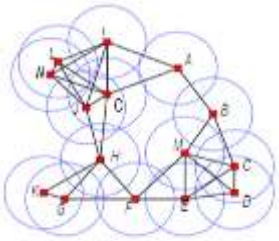
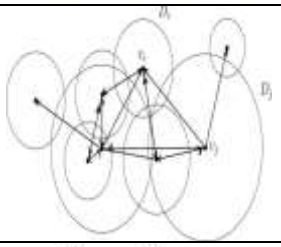
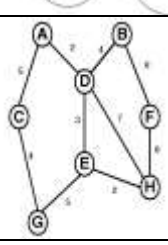
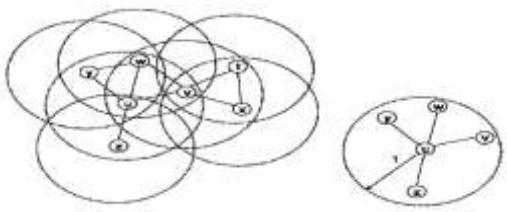
Model	Description	Example
Unit Disk Graph (UDG)	• In the graphs model each node is identified by a single disk of radius r = 1. • there is an edge between two nodes u and v if and only if the distance between u and v is less than 1. • it does not model the weights on the edges and nodes.	
Undirected Graph (UG)	• This Graph is described as: G = (V, E) where V is the set of vertices or nodes and E is the set of non-oriented edges. • Lack Of modeling weights on the edges and nodes.	
Directed Graph (DG)	• The same description for the undirected graph except that E in this model is the set of directed edges.	
Directed Weighted Graph *(DWG)*	• The Same description for the undirected graph but this model models the weight of nodes and edges.	

Table 1. Network Models

UDG is a particular instance of a graph in which each node is identified by a unit circle of radius r = 1, and there is an edge between two nodes u and v if and only if the distance between u and v is at most 1 [10, 11]. The model is shown in Fig 2.a, the transmission range of each node is drawn as a dashed circle. The edges that connect the nodes, are drawn as straight lines. The neighbors of node u is v, w, y and z, which are shown in the simplified graph in Fig 2.b. Because of that the most adequate network model for Mobile ad-hoc networks is UDG.



-a- -b-

Fig 2. « Unit Disk Graph » Model

v. Dijkstra Shortest Path Algorithm

In this section we describe the centralized approach of Disjktra's shortest path, after that we present the distributed Dijkstra's shortest path algorithm and its modifications for finding stable paths;

A. *Centralized Approach of Dijkstra's Algorithm :*

This algorithm is often used to solve the routing problem in telecommunication networks. This algorithm determines the shortest path between the source and all other nodes in the graph .However, the algorithm can be used to calculate the shortest path between a source node and a destination node. Dijkstra algorithm need to have the link costs of all links.

Notations:

- *S*: Source node.
- visited node: unreturned node.
- *d*: distance of a node from the source, this distance is estimated minimum.
- *V*: set of all vertices.
- *V-C*: the nodes that are in V and are not in C.

Principle:

- Construction of a set *C* of visited vertices, for which the shortest paths have already been calculated.
- Then by choosing a vertex *u* not in *C*, the shortest path estimated *d(u)* is minimum.
- Add *u* to *C* and say that *u* is visited. At this point, all *u* arcs *V-C* are open, i.e., they are reviewed and the shortest path to the respective estimated, it is updated for all endpoints.
- Note that *d* is initialized to ∞ for all vertices except vertex *u* where *d* = 0 (in terms of routing protocols, *d* must be initialized with the current time *t*).

Algorithm: Dijkstra (Centralized Approach)

```
1   dist[s] ←0
2   for  all v ∈ V–{s}
3        do  dist[v] ←∞
4   S←∅
5   Q←V
6   while Q ≠∅
7   do   u ← mindistance(Q,dist)
8        S←S∪{u}
9        for all v ∈ neighbors[u]
10            do  if   dist[v] > dist[u] + w(u, v)
11                     then     d[v] ←d[u] + w(u, v)
12  return dist
```

Proc. of the Second Intl. Conf. on Advances in Information Processing and Communication Technology - IPCT 2015

ISBN: 978-1-63248-044-6 doi: 10.15224/ 978-1-63248-044-6-125

B. *Decentralized Approach of Dijkstra's Algorithm :*

Actually similar to centralized algorithm. The computation of the shortest path by each node is done separately based on link cost information received and available at a node. Link cost seen by node *i* at time *t* regarding a link is : *d[v](t)*.

The main idea of the decentralized approach of Dijkstra Algorithm is that it starts with a flooding operation in order to discover the topology of the network i.e. each router sees only local information.

✓ *Flooding Function:*

- Each node acts as both a transmitter and a receiver.
- Each node tries to forward every message to every one of its neighbours except the source node.

The idea of flooding is very simple:

- Each node had a bit (implemented as boolean) *sent* initialized to *FALSE*.
- Initially the source node sends a packet *pck* to all its neighbors and set *sent* to *TRUE*.
- Every neighbor: When receiving *pck* and *sent* is *FALSE*: it sends a *pck* to all neighbors,
- Update the routing table in each node traversed by *pck* and set *sent* to *TRUE*, until all network nodes are traversed by *pck*.

As result of flooding, we have a global view of network. Now, we can apply Dijkstra's shortest path algorithm as follows:

Algorithm: Dijkstra (Decentralized Approach)

```
1   Flooding()
2   dist[s](t) ←0
3   for  all v ∈ V–{s}
4         do  dist[v](t) ←∞
5   S←∅
6   Q←V
7   while Q ≠∅
8   do   u ← mindistance(Q,dist(t))
9        S←S∪{u}
10       for all v ∈ neighbors[u]
11            do  if   dist[v](t) > dist[u](t) + w(u, v)(t)
12                 then     d[v](t) ←d[u](t) + w(u, v)(t)
13  return dist(t)
```

- Time Complexity : is equivalent to the diameter of the Graph

VI. Implementation and Simulation

✓ *Review of Simulators*

Here we give a review of the most used simulators in this area (Modeling and Simulation) :

- GraphStream: is a Java library that focuses on aspects of dynamic graphs. Its main objective is network modeling of dynamic interactions of different sizes. The purpose of the library is to provide a means of representing static and dynamic graphs, for this GraphStream offers several classes of graphs can model directed graphs and undirected. GraphStream can store any type of data on chart elements: numbers, strings or any other type of object. Furthermore, in addition, GraphStream provides a way to manage the evolution of the graph in time. This means manipulating the way the nodes and edges are added and / or deleted, and how the data attributes can appear, disappear and evolve [12].
- Grph: is a Java library for manipulating graphs. According to its original motivation: useful for experimentation graphs and network simulation [13]. Grph also has the feature to come with tools such as a calculation engine of evolution, a bridge linear solvers, a Framework for distributed computing.
- NS-2: This is an event-based simulator and discreet evolution of the modeled environment. The monolithic kernel is implemented in C ++ and additional modules are integrated. Its configuration is done using a scripting language (otcl derived from TCL). This is a free project. Initially the simulator running on a single CPU and suffered from poor performance with the inability to simulate more than a few hundred stations [14]. This issue is now resolved, a module responsible for distributing the simulation starts on different processors instances of the simulator by partitioning the simulated network.
- NetLogo: is particularly appropriate to simulate very complex systems that develop over time. Modelers can give instructions to hundreds of NetLogo agents who are independent and that compete. This provides the opportunity to explore the connections between the behavior of individuals (micro level) to the emergence of a global behavior (macroscopic level).
- Madhoc: It is a mobile network simulator for the design and analysis of distributed algorithms. It is developed in parallel by the University of Luxembourg and the University of Le Havre by L. Hogie [15]. It is used especially for generating dynamic graphs, incorporating the constraints of MANETs (stations behavioral patterns, mobility models, environmental models, etc.).
- OPNET : has very good interfaces and inherits advantages from C++ language but its extendibility is far beyond from a researchers needs. The developed simulator in this study is tried to merge such properties in a single platform and exploits Java advantages.
- OMNET++ : is well object-oriented and has modular and hierarchical structure, but is difficult to deploy due for its highly platform dependent programming language.

A. *Netlogo*

NetLogo is particularly appropriate to simulate complex systems that develop over time. Modelers can give instructions to hundreds of NetLogo agents who are independent and that compete. This provides the opportunity to explore the

Proc. of the Second Intl. Conf. on Advances in Information Processing and Communication Technology - IPCT 2015

ISBN: 978-1-63248-044-6 doi: 10.15224/ 978-1-63248-044-6-125

connections between the behavior of individuals (micro level) to the emergence of a global behavior (macroscopic level).
The major characteristics of NetLogo are:

- a fully programmable simulator,
- it uses mobile agents moving in a grid of stationary agents,
- It uses links that can be created between the agents to allow the establishment of a network,
- an unlimited number of agents and variables can be used in the simulation,
- a wide vocabulary for a programmable primitive language,
- NetLogo is a simulator for complex dynamic systems.

With the agent-based approach, you can program the behavior of individual agents and NetLogo shows what can emerge from their interactions. With a dynamic system modeler, he don't program the behavior of different agents. Thus it is necessary to program the population of just how agents behave as a whole. There are four types of agents

- Patches: are the spatial components of the "world" that is modeled; the agents move patches and patches can store variables.
- Turtles: these are our individual agents, they have features, they move into the world, they are born and dead.
- Links: a particular type of agent that connects two agents and is represented as a line drawn between these agents. This link can be oriented or not.
- The Observer has no specific position, it can imagine looking from above the world of turtles and patches.

B. *Simulation Results and discussion*

There exists a variety of mobility models which are currently used within the mobile ad hoc networking community. We used the most popular one, the random waypoint mobility model (RWP). As described in [9], each node randomly chooses a destination location (in terms of its x, y coordinates) in the simulation area and moves towards this destination with a randomly chosen velocity. When the destination is reached, the station remains a tthe same place for a while. Once this time expires, the node chooses a random destination in the simulation area and a speed that is uniformly distributed in [minspeed, maxspeed] [1].

The experiments were carried out on a PC of Intel Pentium Core2 Duo processor with 2.4 GHz CPU and 3 GB of RAM. As we said before, we used Netlogo (version 5.0.5)Simulator, fig. 3 shows the simulation environment :

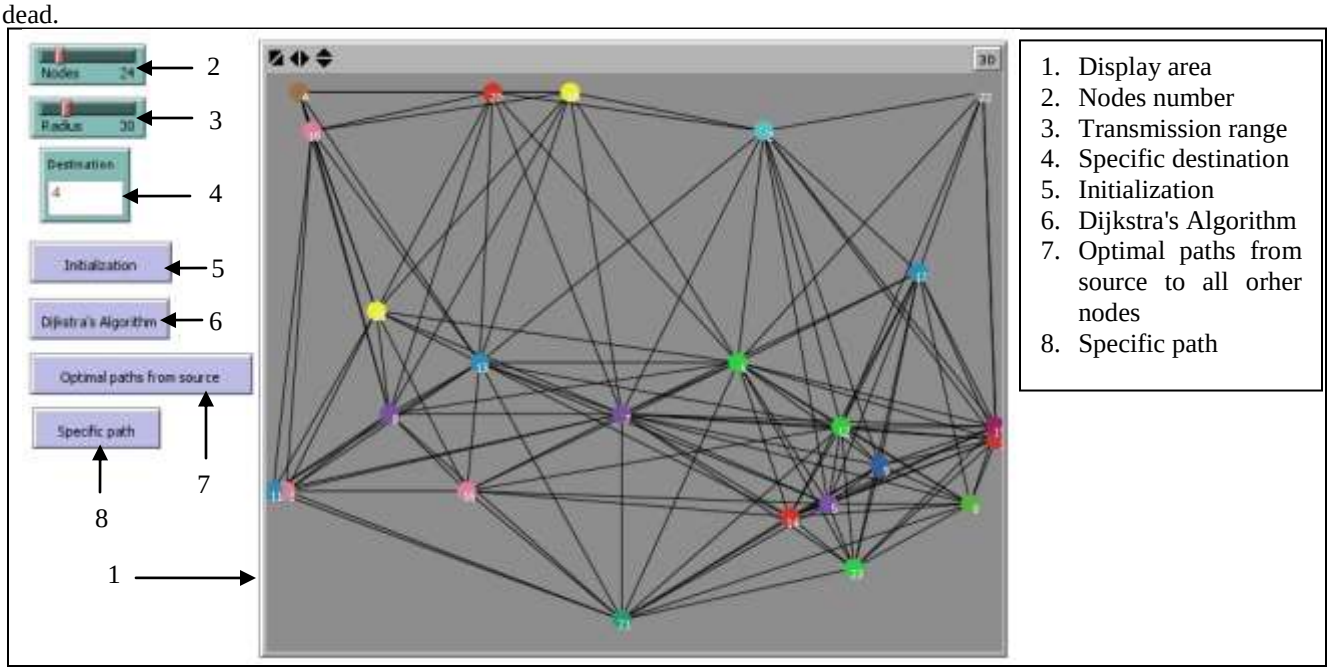


Fig 3. Simulation environment

Here we use the following configuration (Initialization) :

- 24 nodes.
- 30m as transmission range.
- Node number 4 as specific destination for the study case.

In the proposed (decentralized) approach, we have introduced 3 scenarios : dynamic topology, nodes addition and nodes remove, and in every scenario we trigger the flooding process. We discuss each scenario separately.

Proc. of the Second Intl. Conf. on Advances in Information Processing and Communication Technology - IPCT 2015

ISBN: 978-1-63248-044-6 doi: 10.15224/ 978-1-63248-044-6-125

a) Dynamic topology

In dynamic topology scenario, nodes move randomly in the simulation area according to RWP mobility model. Obtained results are shown in Fig. 6 and Fig. 7.

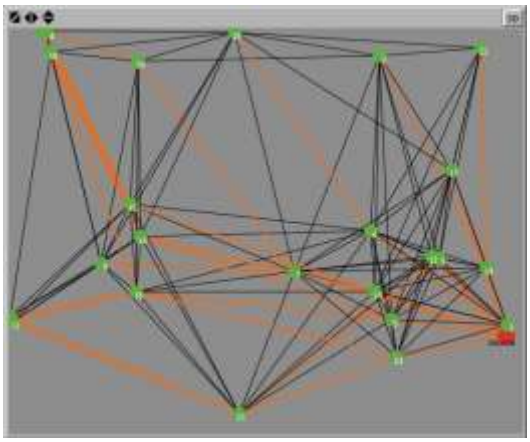


Fig. 6 Dijkstra's Algorithm
(dynamic topology scenario)

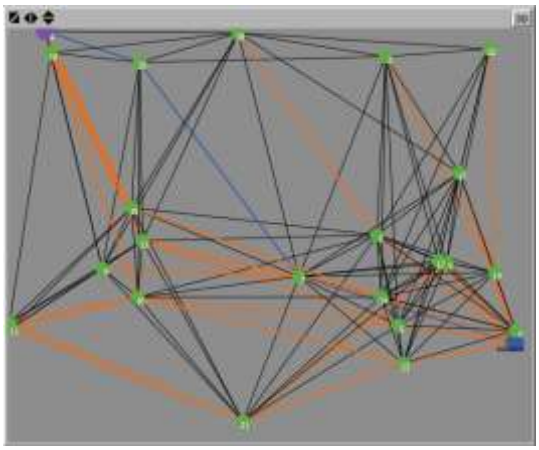


Fig. 7 Specific path (to node 4)
(dynamic topology scenario)

Fig. 6 shows the result of Dijkstra's Algorithm (dynamic topology scenario) that is colored on orange color, it converges quickly. The routing table in this case is (source = node 0):
{{table: [[0 [0]] [3 [0 3]] [19 [0 19]] [9 [0 9]] [12 [0 12]] [23 [0 23]] [5 [0 5]] [14 [0 14]] [6 [0 6]] [17 [0 17]] [7 [0 7]] [21 [0 23 21]] [22 [0 19 22]] [2 [0 19 2]] [16 [0 5 16]] [13 [0 7 13]] [15 [0 14 15]] [18 [0 6 18]] [8 [0 7 8]] [20 [0 7 20]] [11 [0 5 16 11]] [1 [0 5 16 1]] [10 [0 14 15 10]] [4 [0 7 20 4]]]}}.
Fig. 5 demonstrates the specific path, it is mentionned in blue color. Its routing table is : [0 7 20 4].

b) Nodes addition

Here, we apply the scenario of node addition: we add 3 nodes randomly in the network, after that we apply Dijkstra's algorithm (see Fig. 8, orange color), also we observe our specific path (see Fig. 9, blue color).

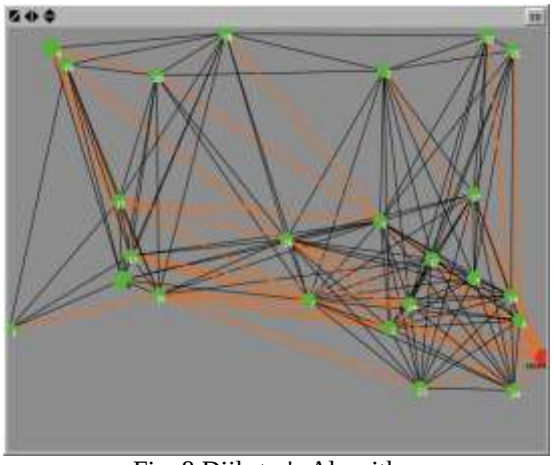


Fig. 8 Dijkstra's Algorithm
(nodes addition scenario)

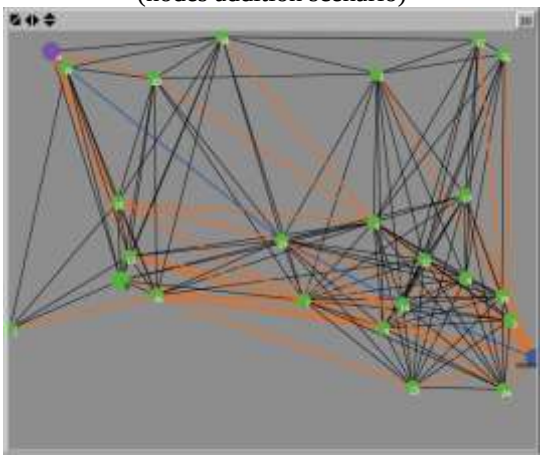


Fig. 9 Specific path (to node 4)
(nodes addition scenario)

The routing table, in this scenario, becomes as follow :
{{table: [[0 [0]] [24 [0 24]] [3 [0 3]] [19 [0 19]] [9 [0 9]] [23 [0 23]] [14 [0 14]] [12 [0 12]] [5 [0 5]] [17 [0 17]] [6 [0 6]] [7 [0 7]] [26 [0 14 26]] [25 [0 3 25]] [2 [0 19 2]] [22 [0 19 22]] [16 [0 5 16]] [13 [0 7 13]] [8 [0 5 8]] [15 [0 14 26 15]] [18 [0 6 18]] [20 [0 6 20]] [1 [0 5 16 1]] [10 [0 14 26 10]] [4 [0 14 26 10 4]]]}}, The routing table of our specific destination becomes : [0 14 26 10 4].

c) Nodes remove:

The last scenario in our study is remove nodes. here we remove 2 nodes randomly (except source and destination nodes), then we apply Dijkstra's algorithm as depicted in Fig. 10 with the orange color, also we observe our specific path (the blue path, see Fig. 11).

Proc. of the Second Intl. Conf. on Advances in Information Processing and Communication Technology - IPCT 2015

ISBN: 978-1-63248-044-6 doi: 10.15224/ 978-1-63248-044-6-125

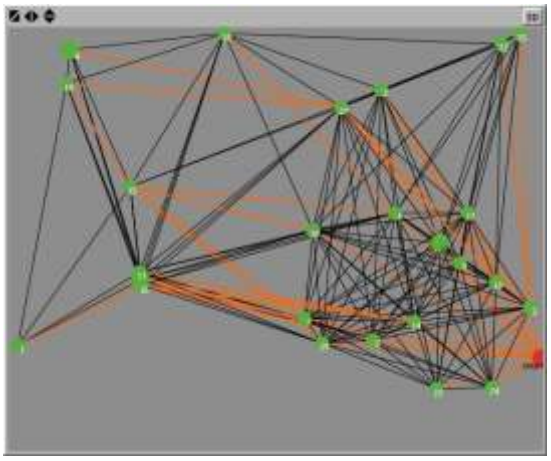


Fig. 10 Dijkstra's Algorithm
(nodes remove scenario)

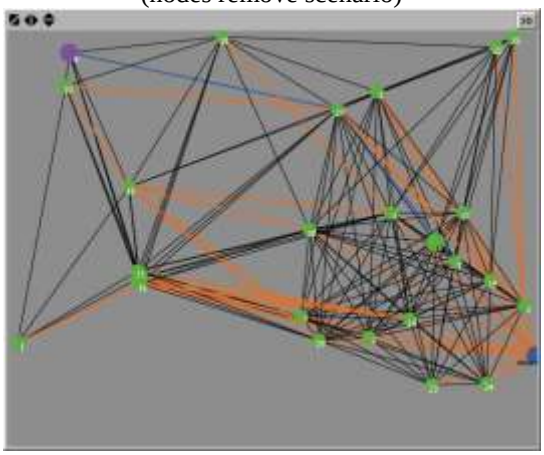


Fig. 11 Specific path (to node 4)
(nodes remove scenario)

In this scenario, The routing table is as follow : {{table: [[0 [0]] [3 [0 3]] [24 [0 24]] [19 [0 19]] [23 [0 23]] [9 [0 9]] [14 [0 14]] [12 [0 12]] [17 [0 17]] [5 [0 5]] [6 [0 6]] [29 [0 29]] [7 [0 7]] [28 [0 28]] [2 [0 19 2]] [22 [0 3 22]] [27 [0 9 27]] [25 [0 3 25]] [16 [0 7 16]] [8 [0 7 8]] [13 [0 14 13]] [15 [0 28 15]] [18 [0 6 18]] [1 [0 7 16 1]] [10 [0 28 15 10]] [4 [0 9 27 4]]]}}, The routing table of our specific destination becomes : [0 9 27 4].

VII. Conclusion

We have successfully implemented distributed Dijkstra's shortest path algorithm to solve the routing problem. Results show that Dijkstra's Algorithm converges quickly finding the shortest path between a source node and the other accessible nodes of network. The utility of graph theory algorithms is to bring solutions to routing problem (network overload) in MANET (enhance routing protocols). Netlogo aim to model a distributed, decentralized approach for searching the shortest path (routing) in the network. As future work, we plan, for this system, to use approaches based ants, bees and Cuckou search.

References

[1] Hogie L., Mobile Ad Hoc Networks: Modelling, Simulation and Broadcast-based Applications", France: University of Le Havre, 165 p., Doctorate dissertation, (April 2007).

[2] Niazi M., Hussain A.Agent based Tools for Modeling and Simulation of Self-Organization in Peer-to-Peer, Ad-Hoc and other Complex Networks", IEEE Communications Magazine, Vol.47 No.3, (March 2009), pp. 163 – 173.

[3] Heidemann J., Bulusu N., Elson J., Intanagonwiwat C., Lan K., Xu Y., Ye W., Estrin D., Govindan R..: "Effects of detail in wireless network simulation", In Proceedings of the SCS Multiconference on Distributed Simulation, (2001), pp. 3 – 11.

[4] Tuduce C., Gross T., A mobility model based on WLAN traces and its validation", In Proc. of the 24th IEEE International Conference on Computer Communications (INFOCOM), Miami, United States of America, (March 2005), pp. 664 – 674.

[5] Yoon J., Liu M., Noble B., Random waypoint considered harmful" In INFOCOM: Twenty-Second Annual Joint Conference of the IEEE Computer and Communications Societies, 2003, pp. 1312 –1321.

[6] Bettstetter Ch., Resta G, Santi P., The node distribution of the random waypoint mobility model for wireless ad hoc networks", In IEEE Transactions on Mobile Computing, 2(3), (2003), pp. 257–269.

[7] Jardosh A., Belding-Royer E. M., K. C. Almeroth, Suri S.: „Towards Realistic Mobility Models For Mobile Ad hoc Networks", United States of America, Department of Computer Science, University of California, (2003).

[8] Miroslav Babiš, Peter Magula; "NetLogo – an alternative way of simulating mobile ad hoc networks"; Wireless and Mobile Networking Conference (WMNC), Bratislava, 122 – 125, 2012

[9] Christian Bettstetter, Giovanni Resta, and Paolo Santi. The node distribution of the random waypoint mobility model for wireless ad hoc networks. IEEE Transactions on Mobile Computing, 2003.

[10] B. N. Clark, C. J. Colbourn, and D. S. Johnson, "Unit disk graphs," Discrete Math., vol. 86, pp. 165-177, 1990.

[11] F. Kuhn, T. Moscibroda, and R. Wattenhofer, "Unit disk graph approximation," presented at the Proceedings of the 2004 joint workshop on Foundations of mobile computing, Philadelphia, PA, USA, 2004.

[12] A. Dutot, F. Guinand, D. Olivier and Y. Pigné, 2007: GraphStream: A tool for bridging the gap between complex systems and dynamic graphs , in Emergent Properties in Natural and Artificial Complex Systems (EPNACS'07), Workshop of the 4th European Conference on Complex Systems (ECCS'07), Dresden, Germany

[13] http://www.i3s.unice.fr/~hogie/grph/

[14] http://nsnam.isi.edu/nsnam/index.php/Main_Page

[15] Luc Hogie. Madhoc, a mobile ad hoc network simulator http://agamemnon.uni.lu/~lhogie/madhoc/, 2003-2008.

About Author (s):

Image	Akram Kout is a PhD Student in MISC Laboratory at the Constantine University, Algeria. His research areas include routing algorithms for mobile ad-hoc networks, graph theory.
Image	Said Labed received his PhD in Computer Science from the University of Constantine, in 2013. He is member of MISC Laboratory. His research areas include soft computing, artificial life, distributed systems.
Image	Salim Chikhi is currently a Professor at the Constantine 2 University, Constantine, Algeria. He is the leader of the SCAL team of MISC Laboratory. His research areas include soft computing and artificial life techniques and their application.