A Modification of DYMO Routing Protocol with Knowledge of Nodes' Position: Proposal and Evaluation

Enrica Zola, Francisco Barcelo-Arroyo, Israel Martin-Escalona

A modification of DYMO Routing Protocol with Knowledge of Nodes' Position: Proposal and Evaluation

Enrica Zola, Francisco Barcelo-Arroyo, Israel Martin-Escalona

Universitat Politècnica de Catalunya
c/ Jordi Girona 1-3, Mòdul C3, 08034 Barcelona, Spain
{enrica,barcelo,imartin}@entel.upc.edu

Abstract. Knowledge of the physical location of the nodes is known to improve performance in wireless networks. This is especially true in MANETs, where routing protocols face a continuously changing topology. In the past, routing protocols such as Beacon-Less Routing (BLR) used the location information of the nodes to build the forwarding path in a distributed manner. In this work, we borrow the forwarding approach in BLR and apply it in the route discovery process of DYMO. Under the assumption of nodes knowing their own location, the receiving nodes will compute a delay. The node with lower delay will resend the RREQ first. The rest of forwarding nodes will drop the RREQ once they receive this first RREQ. Thus the best forwarding node is selected in a distributed manner. This modification is expected to reduce the amount of RREQs circulating in the network, lessening the routing overhead.

Keywords: MANET; routing protocol; DYMO; DFD; localization.

1 Introduction

Routing has been one of the hot topics in Mobile Ad-hoc Networks (MANETs) for years. The frequent changes in the network topology became a challenge for researchers, used to deal with networks in which routes were stable. Several solutions to MANET routing have been proposed, but they brought a new issue: protocol scalability. A very well-known solution for MANET routing is the Dynamic MANET On-demand (DYMO) routing protocol [1]. As a reactive protocol, DYMO establishes a route from a source node to a destination node before sending data. What makes DYMO different from classic approaches, such as AODV, is that intermediate nodes are able to learn the route to all the predecessor nodes in the path, thus lessening the number of RREQs generated in the network.

Nowadays, mobile network receivers are commonly integrated with positioning technologies. The nodes' position information can be exploited to improve the efficiency of routing in MANETs. Several proposals use positioning to minimize the search space for route discovery towards the destination node [2] or to apply source routing in order to establish the geographical path that a packet must follow towards its destination [3]. The approach followed by Beacon-Less Routing Algorithm (BLR) [4] is different. In this case, the source node broadcasts data packets without the need to discover the path to the destination beforehand. Knowing the nodes' positions is a requirement in the BLR approach. On the other hand, beacon-based protocols periodically share location information among neighbouring nodes to maintain routing tables up to date. However, the more accurate positioning information the more protocol overhead (i.e. location data is flooded more frequently). Thus, the authors in [5] proposed the Dynamic Route Maintenance algorithm for self-configuring the node's beacon interval according to the mobility pattern of the neighbouring nodes (i.e., shorter intervals for higher mobility nodes).

This paper is focused in assessing the benefits of applying the forwarding strategy of BLR to the route discovery process of DYMO. This approach is named DYMOselfwd (DYMO with selective forwarding) in this paper. It relies on two assumptions: 1) nodes always know their own position (e.g., through GPS) and 2) nodes always know the destination node's position. The specific procedure followed to discover destination node's position is out of the scope of this work. Anyway, the reader can assume that positioning information is provided by an external location management system, as commonly assumed in the literature [4,6]. In next generation wireless networks (i.e., 5G), the ad-hoc network may be used in order to reach a fixed router (i.e., LTE eNodeB or public 802.11 access point in malls, sport centers, libraries, etc.) that provides fixed connection to the Internet; in such scenario, the position of the destination node may be easily assumed to be known. The use of positioning information in DYMOselfwd is expected to reduce the routing overhead, which means 1) fewer nodes contending the access for re-broadcasting the RREQs (i.e. lower probability of collision) and 2) less traffic in the network (i.e. longer network lifetime and less congestion issues). The reduction in collisions is awaited to compensate for the increased delay of DYMOselfwd and therefore the actual delay of the route set-up will be almost the same as provided by DYMO.

There are several works that propose using location information to improve the route-discovery stage in AODV. For instance, authors in [7] propose that only nodes in the *forwarding region* forward RREQs, so that the overhead is reduced. A similar approach is followed in [8], but forwarding nodes are selected according to nodes' mobility information. To the best of our knowledge, the methodology described here, which has been proposed for flooding approaches [4], has never been applied to the route discovery process of AODV or DYMO protocols. The modified version of the DYMO with location information was already proposed in [9], but no evaluation was provided. The simulation results shown in this paper prove the improvement expected by the proposed algorithm.

2 Using Position Information in DYMO

Route discovery and route management can be considered the basic operations of the DYMO protocol [1]. The route discovery starts when a source node (A) has to send data to a target node (O) and there is no established route between nodes A and O yet. In that case, the source node A broadcasts a special message called Route Request (RREQ). Every neighbouring node of A receiving the RREQ (i.e., K_i) will add A as next hop in its own routing table (i.e., direct transmission). At this point, each node K_i will check whether it knows the path to O. If so, K_i will send a Route Reply (RREP) back to A; otherwise, K_i has to broadcast the RREQ again after adding its own address in the path accumulation field. This means that the RREQ size grows with the number of hops required. The benefit of this path accumulation is twofold: any intermediate node is able to learn its next hop 1) towards A and 2) towards all the predecessor nodes in the accumulated path. Once the RREQ has reached 1) a node with established route to O or 2) the destination node O, a RREP is unicasted back to A. This last message is the responsible for setting up the path (i.e., next hop) at all the intermediate nodes.

2.1 Modified Route Discovery

As described before, A will broadcast a RREQ packet if it has not yet a route to reach its destination O. DYMOselfwd will include the transmitter's (TX_pos) and the destination's position (DST_pos) in the RREQ packet. Any intermediate node that will rebroadcasts the RREQ will substitute the TX_pos field with its own position. In the case of no RREP being heard, we propose that only one node broadcasts the RREQ again. The selection of this node is explained in Section 2.2. With this approach the amount of RREQ circulating in the network is expected to reduce if compared to DYMO, thus lessening the routing overhead.

DYMO and AODV require the same number of RREQs for a given path. This amount is equal to the number of nodes (N) in the network, since every node receiving a RREQ is expected to rebroadcast it. On the other hand, in DYMOselfwd the number of RREQs linearly depends on the number of hops between source and destination (num_hops). DYMOselfwd is expected to significantly reduce the amount of routing overhead even though RREQ packets are actually larger (due to the embedded positioning data). In the worst-case scenario, three neighbours of the source node will rebroadcast its own RREQ since they do not see each other (i.e., hidden node problem). Accordingly, the amount of required RREQ packets will be, at most, 3 times the num_hops, which is always less than in the figure required by original DYMO algorithm.

2.2 *Best Node* Selection

The procedure followed to select the *best node* (i.e., the intermediate node in charge of rebroadcasting a RREQ first) is inspired in the BLR algorithm [4]. In BLR, data packets are broadcasted through the network until they reach the destination, without the need to set-up or maintain routing tables. The *best node* is selected in a distributed fashion, by applying different delays to each candidate node for broadcasting the packet. BLR will set the shortest delay to the best forwarding node, which is defined as the node with the best position in the *forwarding area*. Accordingly, the *best node* is the first to rebroadcast the data packet. The same idea is applied to broadcast the RREQ in DYMOselfwd. Any node receiving a RREQ regarding an unknown route follows this procedure (see Fig. 1):

1. It computes the distance between the transmitter and the destination nodes ($dist_{tx\_node}$). Remind that TX_pos and DST_pos data are included in each RREQ packet.
2. It computes its distance to the destination (own_dist).
3. If it has an own_dist shorter than $dist_{tx\_node}$, it becomes a candidate for rebroadcasting the RREQ. Otherwise, the node discards the RREQ. This prevents nodes in the opposite direction from rebroadcasting the RREQ (i.e., node E in Fig. 1).

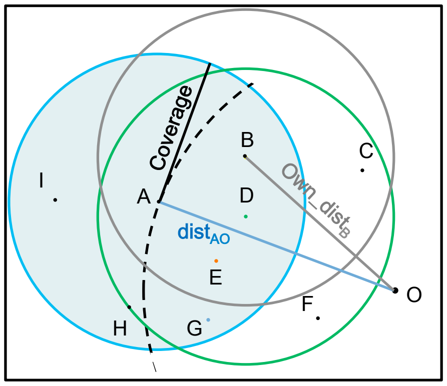


Fig. 1. *Best node* selection for rebroadcast the RREQ sent by A. D has the shortest own_dist.

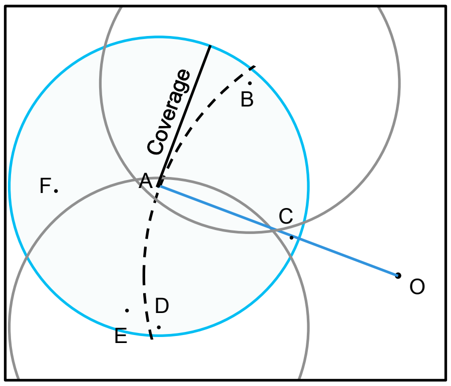


Fig. 2. Up to three nodes may rebroadcast the RREQ due to the hidden node problem.

At this point, the Dynamic Forward Delay (DFD) [4] approach is used to select the *best node* in a distributed way. This approach consists of each candidate node delaying the rebroadcast of the RREQ according to the values previously computed. Hence, the *best node* is the node with the lowest forward delay. A complete discussion on the best DFD function is out of the scope of this paper. However, it is likely to think that the DFD depends on the node's own_dist, so that the node with the best position in the *forwarding area* is pushed to be the first rebroadcasting the RREQ.

Candidate nodes hearing a RREQ defer from rebroadcasting any pending RREQ. In case of multiple RREQs reaching the destination node (i.e., because of the hidden node problem), the first RREQ received at destination is selected. The sequence number in the RREQ preserves from looping. In the case that noise or other unexpected events prevent a node from hearing the RREQ sent by the *best node*, more than one node will be rebroadcasting the RREQ, thus ending up with a behaviour similar to the original DYMO protocol.

Fig. 1 illustrates the algorithm of the *best node* selection. In this example, source node A is looking for a route towards destination node O. The transmission range (i.e. the maximum distance at which the signal transmitted is heard, under ideal radio conditions) is labeled as *Coverage*. The RREQ sent by A is received by nodes B, D, E, G, H, and I, which are the nodes that lay inside the blue coverage area of A. The dashed line represents the points at $dist_{tx\_node}$ (i.e., $dist_{AO}$ in Fig. 1), highlighting the area in which candidate forwarding nodes can be. Thus, only nodes B, D, E and G are candidates for rebroadcasting the RREQ since their own_dist to O is shorter than $dist_{tx\_node}$. Among all of them, D has the shortest own_dist. Therefore, it is the first node to rebroadcast the RREQ, i.e., D is the *best node* at this hop. The other candidates inside the coverage range of D (i.e., green circle in Fig. 1) defer from rebroadcasting the pending RREQ once they hear the new RREQ from D (i.e., using a higher sequence number).

According to the geometry depicted in the scenario of Fig. 2, up to three nodes may rebroadcast a RREQ due to the hidden node problem. In this scenario, nodes B, C and D do not hear each other and consequently they all rebroadcast the RREQ. In scenarios like Fig. 2, the RREQ rebroadcasted by C will first reach the destination node O, since C has shorter own_dist if compared with B and D.

The advantage expected after applying the DYMOselfwd definition is twofold. First, the routing overhead will decrease if compared with the original DYMO algorithm, even in the worst-case scenario. This aspect has been studied through simulation in Section 4. Second, the establishment of a route before sending data packets will guarantee higher throughputs, since DYMOselfwd confines broadcasting to RREQ packets while data packets can be unicasted at higher rates (i.e., IEEE 802.11 nodes must use basic transmission rates for broadcasted frames). This approach can be considered as an improvement over the DFD as used in BLR. This second conjecture has not been tested in this work and requires further research.

3 Simulation tool and scenarios

Some tests have been run using simulation. The tool used is Omnet++ 4.3 [10]. The INET framework has been used to build the network models used in this work. INET provides a complete implementation of several network protocols (e.g. IP, UDP, TCP, ARP, etc.), wireless technologies (e.g. IEEE 802.11) and network approaches (e.g. ad hoc). At the same time, this framework provides a complete set of radio models and mobility patterns. The DYMOselfwd was implemented starting from the module described in [11]. After testing other modules it proved to be the most adequate to carry out the modifications successfully. Several upgrades have been implemented to the DYMO module in order to properly work as described in the draft and to allow the achievement of statistics.

All simulations have been run along 1,250 s; the transmitter sends a UDP segment every 250 ms, allowing 5,000 segments. Independent means have been obtained by averaging over five repetitions of each experiment. In all cases, the confidence interval is better than 95%. Two scenarios have been designed for testing purposes with 9 and 13 nodes respectively. For each scenario, simulations have been run for static and dynamic (i.e. moving) cases. The mobility model assumed for dynamic has been the Random Waypoint without pause times. Different pedestrian speeds have been considered in order to test the ability to recover from changing topologies. The area is 150 x 150 m. The coverage has been set to 71 meters so, even in the case of 9 nodes there are always two nodes at sight at least. Other parameters of the configuration are summarized in Table 1 along with other parameters that have been configured as usual in other similar works [11].

Fig. 3 shows the scenarios. These layouts are used for the static cases and as the starting layout for the dynamic cases. The scenario with 9 nodes (3x3) represents the case of low density and is considered in order to check this constraint situation. In this scenario, each node has no more than 2 neighbouring nodes in the direction of the sink. The scenario with 13 nodes represents a medium density. In the remaining of the paper we refer to the 9 and 13 scenarios as

Table 1. Configuration parameters used in simulation

Speed	0 | 1 | 2 m/s
Protocol	DYMO | DYMOselfwd
Propagation model	Two Ray Ground Model
Shadowing	0 dB
Frame size	512 Bytes
MAC Protocol	802.11g
Bit Rate	54 Mbps
Carrier frequency	2.4 GHz

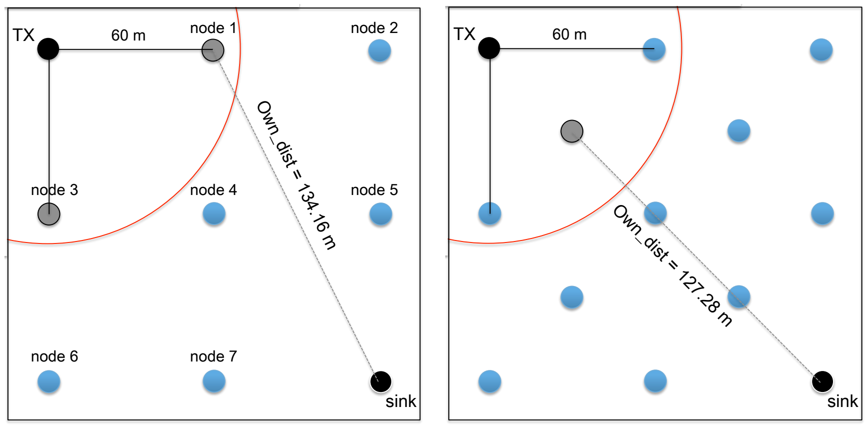


Fig. 3. Nodes' locations in the simulation area for the 9 (low density, on the left) and 13 (normal density, on the right) nodes scenarios. *TX* is the node sending UDP segments to the *sink*. Its coverage area is represented in red.

low and normal density respectively. In all scenarios, there is one node sending UDP segments to a *sink*. They are both represented by black circles in Fig. 3 and labelled as *TX* and *sink* respectively. The coverage range of *TX* is represented by a red circle. The *best node* for retransmitting the RREQ is represented by a grey circle; *own_ dist* is also displayed in each layout.

The metrics obtained from the simulations in order to evaluate the performance of DYMOselfwd and compare to the standard protocol are listed below:

– Troute: Time to establish the route (when the source receives the first RREQ from the *sink*).

Table 2. Results with static nodes in the two density scenarios.

	Low density		Normal density	
	DYMO	DYMOselfwd	DYMO	DYMOselfwd
Troute [ms]	5.21	5.72	5.71	5.79
ColMac	15.8	20.2	30.2	0.0
MACPdu	20,253.0	20,262.4	20,351.6	20,176.0
REQtot	180.2	178.4	265.6	84.0

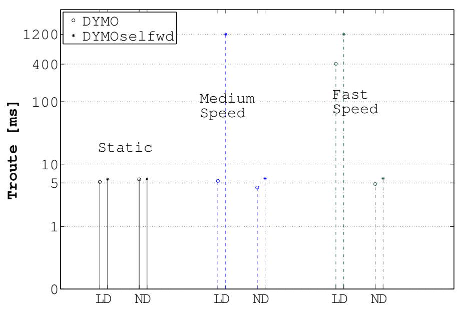
- ColMAC: number of collisions at MAC level.
- MACPdu: Total number of MAC frames sent by all nodes.
- REQtot: Total number of requests sent by all nodes.

4 Results

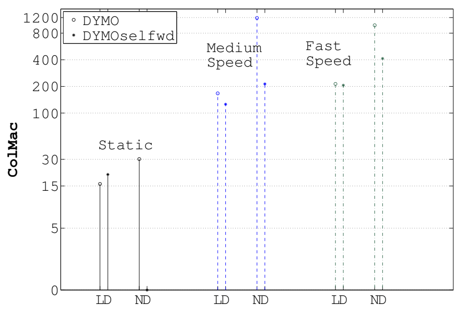
4.1 Static nodes

This scenario suits the case of a sensor network; the results are displayed in Table 2. For low density, DYMO performs better than DYMOselfwd: the three first metrics in the table are better while the last one is slightly worse. This is due to the symmetry: since there are two nodes placed at the same distance from the *sink* (node 1 and node 3 in Fig. 3), they re-send the RREQ from *TX* at the same time and they do not listen to each other. Thus the number of nodes sending the RREQ is the same in both protocols and they achieve similar results. However, due to the modified algorithm, the establishment of the first route takes longer in DYMOselfwd due to the extra delay in the retransmission of the RREQ. Moreover, this extra delay causes a larger number of collisions, mainly occurring in the node located in the middle of the layout, and consequently a higher number of MACPdu sent out in the medium.

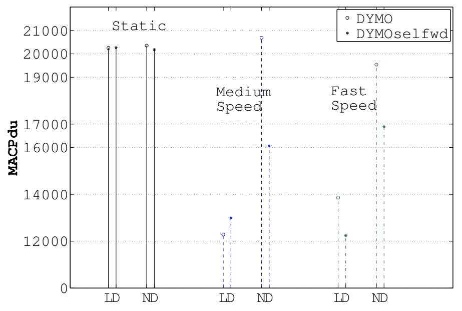
In order to confirm that the worst results obtained here are due to the symmetry in the layout, we have slightly changed the location of the nodes in the low density scenario as follows: node 1, node 2, node 5 and *sink* have been moved 5 meters left along the horizontal axis, and node 5 and *sink* have been also moved 5 meters up along the vertical axis. As expected, after breaking the symmetry the number of collisions decreases for both protocols (i.e., ColMac



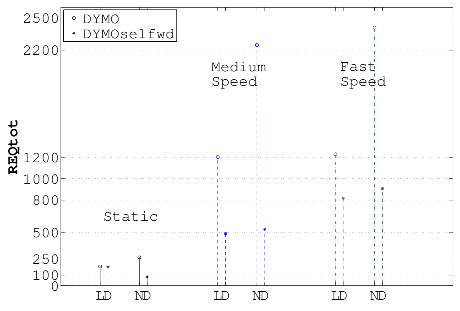
(a) Troute



(b) ColMAC



(c) MACPdu



(d) REQtot

Fig. 4. Results for DYMO (o) and DYMOselfwd (*) for the static (solid black line), medium speed (dashed blue line) and fast speed (dash-dot green line) scenarios. For each, the low density (LD) and the normal density (ND) cases are shown.

is 7.0 for both) and the performance still remains similar for both protocols. However, now DYMOselfwd performs slightly better than DYMO (i.e., Troute, MACPdu and REQtot are slightly smaller in DYMOselfwd).

For normal density, the metrics are better for DYMOselfwd with the only exception of the time to establish the first route. The conclusion of this experiment is that DYMOselfwd does not respond so well to the situation of low connectivity but achieves an obvious improvement in other cases. It is important to notice that the number of REQtot is always smaller in DYMOselfwd.

In the next sections results with different mobility settings will be shown. In order to facilitate comparison, results are shown in Figure 4. Figure 4(a) shows the results for Troute, Figure 4(b), 4(c) and

4(d) those for ColMAC, MACPdu and REQtot, respectively. Values for the static case are shown with a solid black line, those for the medium speed case with a dashed blue line and those for the fast speed with a dash-dot green line. Results for the DYMO protocol are marked with a circle while those for the DYMOselfwd with an asterisk.

4.2 Medium speed

This scenario represents the case of MANET with medium pedestrian speed (1 m/s). The results are represented in blue in Figure 4 and the conclusions are quite similar to those obtained for static nodes: DYMOselfwd improves the performance only after a certain degree of connectivity is present in the network. In this case, the consequence of the low connectivity is worse than in Section 4.1 and the longer time to finish the first route must be highlighted. This is because in the static case the node at the upper corner was always at sight of two others, while now because the random movement this amount of two can decrease to one or none. A diffusion-like strategy is more efficient. However, it must be stressed that the number of RREQ sent by all nodes is always smaller for DYMOselfwd.

4.3 Fast speed

Now all results represented in green in Figure 4 are worse than for medium speed. This could be expected as connected to the well-known concept of "mobility cost" in mobile networks: movement and speed show always consequences in the performance, network management and signalling needs. This said, the conclusions are not far from the ones drawn in the former cases: DYMOselfwd gets noticeable improvements when a reasonable degree of connectivity is present. Moreover, again the number of RREQ is always smaller in DYMOselfwd. Here it is relevant to stress that the upgrade to DYMOselfwd is more noticeable for fast speed: the collisions are reduced from 14,581 to 1,671 at fast speed while the reduction is from 1,082 to 158 at slow speed (also the total number of MACPdu shows a better improvement).

5 Conclusions

Routing in MANETs is still a challenging topic for researchers, who aim at finding a solution that is scalable but also flexible enough to adapt to the changing mobile scenario. The algorithm proposed in this paper (DYMOselfwd) takes advantage of the location information that many devices have at hand and increases the scalability of the DYMO routing protocol by dramatically reducing the number of route requests (RREQs) sent by the nodes in the network. DYMOselfwd is inspired by the forwarding strategy of BLR [4] for which a node retransmits a frame (i.e., the RREQ here) after delaying its transmission for a given time interval. Since nodes closer to the destination node will apply a smaller delay, the shortest route will be established and less RREQs will circulate in the network. Simulations run in scenarios with different density and speed show that the overall throughput always improve with DYMOselfwd if compared to DYMO. Moreover, the extra delay introduced does not have an impact on the throughput. We can conclude that, in common scenarios in MANETs, the benefits of the DYMOselfwd are evident. In case of more sparse scenarios, the protocol requires larger time to establish the first route and MAC collisions may increase due to the hidden node problem; however, the overall throughput in the network remains high.

The error in the destination node's position may have a negative impact on the performance of DYMOselfw. In the future, we aim at analyzing this impact. Moreover, we aim at extending the present evaluation by comparing our model with another location-based routing protocol and by assessing the tradeoff between the protocol simplicity and its scalability.

Acknowledgment

We would like to express our gratitude to Jordi Ruiz Paños and Raúl Ruiz Díaz who were in charge of implementing the simulations.

This research was partially supported by the Spanish Government and ERDF through CICYT project TEC2013-48099-C2-1-P.

References

1. C. Perkins, I. Chakeres, Dynamic MANET On-demand (AODVv2) Routing, IETF Internet Draft (Standards Tracks, work in progress), 2012.<http://tools.ietf.org/html/draft-ietf-manet-dymo-22#page-33>.
2. Y. Ko, N.H. Vaidya, Location-Aided Routing (LAR) in mobile ad hoc networks, in: Proc. ACM MobiCom, 1998, pp. 66-75.
3. V. Giruka, M. Singhal, A self-healing on-demand geographic path routing protocol for mobile ad-hoc networks, Elsevier Ad Hoc Networks, 5 (7), (2007) 1113-1128.
4. M. Heissenbüttel, T. Braun, T. Bernoulli, M. Wälchli, BLR: Beacon-Less Routing Algorithm for Mobile Ad-Hoc Networks, Elsevier Computer Communications Journal (Special Issue), 27, (2003) 1076-1086.
5. C. H. Chou, K.F. Ssu, H.C. Jiau, Dynamic route maintenance for geographic forwarding in mobile ad hoc networks, Elsevier Computer Networks, 52(2), (2008) 418-431.
6. K. Endo, Y. Inoue, Y. Takahashi, Performance modeling of beaconless forwarding strategies in multi-hop wireless networks, Elsevier Computer Communications, 35 (1), (2012) 120-128.
7. R. Reno, Enhanced AODV for directional flooding using coordinate system, in: Proc. International Conference on Networking and Information Technology (ICNIT), 2010, pp. 329-332.
8. Y. Khamayseh, O.M. Darwish, S.A. Wedian, MA-AODV: Mobility Aware Routing Protocols for Mobile Ad Hoc Networks, in: Proc. Fourth International Conference on Systems and Networks Communications, 2009, pp. 25-29.
9. E. Zola, F. Barcelo-Arroyo, DYMO Routing Protocol with Knowledge of Nodes' Position, in: Proc. Joint ERCIM eMobility and MobiSense Workshop, 2012, pp. 9-14.
10. Omnet++ release 4.3. <http://www.omnetpp.org/>.
11. C. Sommer, I. Dietrich, F. Dressler, Simulation of Ad Hoc Routing Protocols using OMNeT++, Springer Mob. Netw. Appl, 15 (6), (2010) 786-801.