Assessing the Effectiveness of DTN Techniques Under Realistic Urban Environments

Sergio Martínez Tornell, Carlos T. Calafate, Juan-Carlos Cano and Pietro Manzoni
Universitat Politècnica de València
Camino de Vera, s/n, 46022 Valencia, Spain
sermarto@upv.es, {calafate, jucano, pmanzoni}@disca.upv.es

***Abstract*—Intelligent Transportation Systems (ITS) require collecting and distributing as much relevant information as possible to provide their services. Such information could also offer new possibilities to various service providers in the wider Smart City context. The distribution of this intelligence is carried out through various vehicular networking strategies, the most flexible of all being Delay Tolerant Networking (DTN). DTN protocols can cope with the problems derived from high mobility and the possibility of high node sparsity. Nevertheless, achieving a fair comparison of DTN solutions in an urban environment is a hard task. In this paper we present a generic DTN model that we use to compare various representative DTN solutions in a metropolitan scenario. We highlight the weak and strong points of each evaluated proposal by also taking into consideration different sending strategies adopted to improve the performance of DTN protocols.**

***Index Terms*—Delay tolerant networks, Vehicular networks, VANET, Modeling.**

I. Introduction

Intelligent Transportation Systems (ITS) require collecting and distributing as much relevant information as possible to provide their services. Recently, the number of sensors available in our vehicles increased notably. Currently, vehicles carry sensors not only related to engine status or speed, but also related to weather (rain and temperature sensor) or the state of the road (ESP sensors). If the authorities in charge of traffic management and safety were able to harvest this information, the required investments in infrastructure and road deployed sensors could be significantly reduced. Smart City service providers could also take advantage of this huge amount of data.

The distribution of this information is carried out through various vehicular networking strategies, the most flexible of all being Delay Tolerant Networking (DTN) . DTN protocols can cope with the problems derived from high mobility and the possibility of high node sparsity. Nevertheless, achieving a fair comparison of DTN solutions in an urban environment is a hard task.

When comparing and evaluating the performance of different Delay Tolerant Network (DTN) protocols, the criteria used to select the next forwarding node, also called the "routing metric", is not the unique element that can make the difference. Several low-level sending mechanisms for one-hop communications such as: (a) the use of ACK packets, (b) the existence of flow control mechanisms, or (c) whether the protocol uses unicast or broadcast messages, heavily affect its performance.

Despite this, various proposals in the literature obviate these issues and do not properly specify which low-level mechanisms are used, which can lead to unfair comparisons. For example, protocols which tend to select the furthest node as the next forwarding node, which are faced with higher transmission losses; the use, or not, of ACK packets will impact their delivery ratio significantly.

The classical existing DTN protocols, e.g., [1], [2], [3] were not specifically designed for vehicular contexts. More recently, DTN *geographic protocols* have been presented for this specific purpose. The most simple geographic protocol is the *Greedy* protocol, which is a DTN variation of another non-DTN geographic protocol, called GPSR [4]. In [5] the authors present *GeOpps*, one of the most advanced DTN protocols for Vehicular Ad-Hoc Networks (VANETs), and compare it against Greedy and MoVe [6], the latter originally designed for Mobile Ad-hoc NETworks (MANETs) and not considering geographic information. In the comparison, authors does not specify whether they used ACK messages or any other mechanism to confirm one-hop transmissions. Moreover, they do not specify which propagation model was used, leading to biased results, as was stated in [7]. In [8] GeoDTN+Nav is presented and compared to GPCR [9], which is a non-DTN protocol. From our point of view, any DTN protocol performs better than a non-DTN protocol in a sparse network, so it would have been more valuable to compare GeoDTN+Nav against others DTN protocols such as GeOpps or GeoSpray [10]. In this same, no specific one-hop communication strategy was defined, and a very simple propagation model was used for the comparison.

In this paper we present a generic DTN model called Generic One-Copy DTN Model (GOD) with the objective of fairly comparing various DTN solutions in a metropolitan scenario. Through this model, and using simulations, we highlight the weak and strong points of the various proposals studied by taking into consideration different sending strategies, adopted to improve the performance of the DTN protocols.

In our study, we compare a protocol we developed, called Map-based Sensor-data Delivery Protocol (MSDP)[11], against the GeOpps and the Greedy protocols. Using accurate mobility and propagation models, we also explore the effects of using ACK messages, and broadcast or unicast transmission

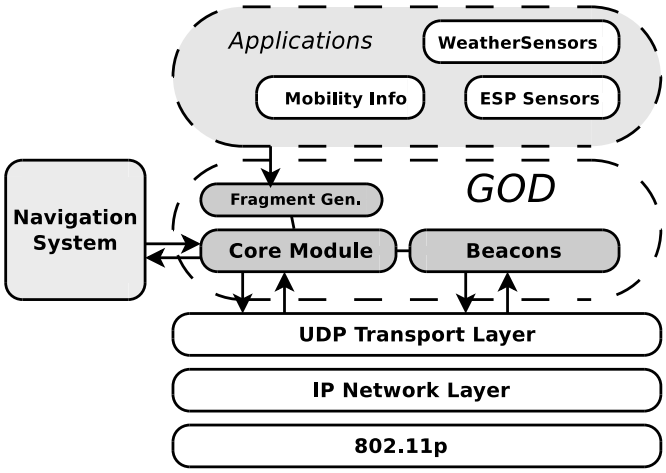


Fig. 1: Generic One-Copy DTN protocol architecture.

in the performance of DTN protocols.

This paper is organized as follows: in Section II we present our Generic One-Copy DTN Model (GOD). The compared protocols are introduced in Section III. The simulation environment and settings are presented in Section IV. Section V presents our results and findings. Finally, section VI concludes this paper and provides details about future work.

II. The Generic One-Copy DTN Model

A DTN protocol can be subdivided into various components, the most important one being the forwarding criteria. The forwarding criteria, also known as the routing metric, represents the criteria the protocol uses to chose the next forwarding node. Other minor and more generic components or mechanisms, such as the use of ACK packets or the adoption of flow control mechanisms, can anyway heavily influence the performance of the DTN protocol as well.

In order to achieve a fair comparison between different DTN protocols we have developed a Generic One-Copy DTN Model (GOD), which can be considered as a super-class of every DTN protocol. This approach not only simplifies the comparison of different DTN solutions, but could also speed up the implementation of new protocols.

Figure 1 shows how our models integrates within the TCP/IP protocols architecture. It is represented as a new layer between the application and the transport layers, allowing our DTN protocol to work independently of the IP routing protocol.

Our GOD model implements the following configurable generic mechanisms:

- ACK Messages: ACK messages are used to confirm every one-hop transmission, to ensure that no fragment is lost during a one-hop transmission;
- Unconfirmed messages: maximum number of unconfirmed data messages. This is specially useful when the communication channel is unstable.
- Redundancy: redundancy fragments can be added to reduce the impact of fragment losses.
- Location: an interface to the Navigation System (NS) to allow the use of geographic data, such as location, direction, programmed route, etc.

The GOD structure is based around three different modules which work coupled: the beacon module, the fragment generator module, and the core module.

A. The beacon module

The beacon module implements the announcements mechanism. It periodically broadcasts the node information, and it also manages the collected information in order to construct a list of the current neighbors. Every beacon packet contains the source address as well as its location, its velocity, its direction, and the node type obtained from the Navigation System. Moreover, extra information may be included within the beacon packet payload if required by a specific DTN protocol; Figure 2 shows the beacons message format.

Beacon Msg	Msg ID	Source ID	Pos	Vel	Node Type	Payload Extra Info
	4 Bytes	8 Bytes	8 Bytes	8 Bytes	1 Byte	Up to MTU

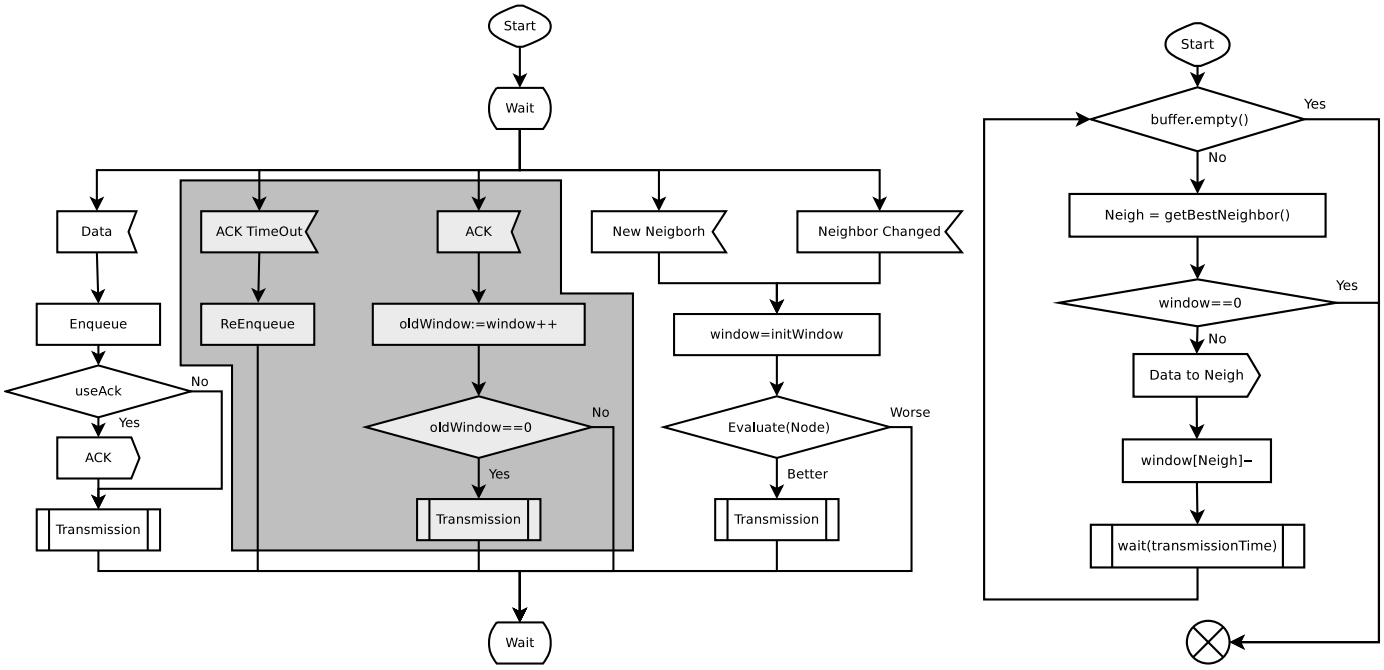
Fig. 2: Beacon messages format.

A *New Neighbor Event* is notified to the **core module** every time a new neighbor is detected. When a certain number of consecutive beacons from the same neighbor are lost, a *Neighbor Disconnected Event* is notified to the **core module**. This module also notifies the **core module** when the information about a neighbor has been updated based on a new received beacon.

Inside this module, the inter-beacon time can be defined to meet protocol requirements. Currently, we only support static inter-beacon times, but in the future we plan to implement dynamic inter-beacon times that may depend, for example, on mobility, or on network density parameters.

B. The fragments generator module

This module is directly connected to the application layer and it is in charge, if required, of dividing large messages into fragments smaller than the MTU. Since the GOD uses UDP packets for one-hop communication, large messages must be split-up to avoid IP fragmentation, which would interfere with routing decisions. Figure 3 shows the relationship between the information messages and their fragments. Then, the fragments are sent to the core module. The fragments generator module can also create **redundancy** fragments to increase reliability. When redundancy is enabled, a percentage of extra fragments are created using Foward Error Correction (FEC) techniques. That is, if a message is divided into N fragments, $N * \alpha$ total fragments will be generated, where redundancy factor α is greater than 1 and depends on the configuration. The basic hypothesis is that the original message can be reassembled with whatever subset of size N of the sent fragments. This redundancy allows reducing the impact of possible fragment losses.



(a) Dtn module flowchart

(b) Transmission process flowchart

Fig. 4: Generic One-copy DTN Model flowchart.

C. The core module

This module is connected to the beacon module, and it is in charge of managing transmission opportunities between nodes. The core module has a buffer where fragments are stored. Fragments are enqueued and dequeued based on their timestamp: older fragments receive higher priority. The events notified by the beacon module are used to keep a sorted list of neighbors according to the chosen routing metric. A schematic overview of this module behavior is described in Figure 4a, where the shadowed parts of the chart are only executed when the ACK mechanism is active. This module is activated by five different types of event:

- *Data Messages*. A Data message can arrive from the fragments generator module or from the network, i.e., from a neighbor. When a Data message arrives, the fragments contained in it are enqueued in the local buffer and, in case it is enabled, its reception is confirmed through an ACK message. This ACK allows the previous source node to remove the confirmed fragments from its buffer.
- *ACK messages*. An ACK message confirms the reception of a Data message, and the fragments contained inside the confirmed Data message can now be definitely deleted from the buffer.
- *ACK Time Out*. When an ACK Time Out occurs, the data fragments contained in the unconfirmed data message are re-enqueued in the sending buffer.
- *New Neighbor and Neighbor Changed*. Whenever one of these events is notified, the value of the transmission window is restarted. The neighbor is evaluated using the metric defined by the specific DTN protocol, and, in case the neighbor is evaluated as a "better" node than the current carrier, a new transmission process is started. Obviously, if the neighbor is detected as an Road Side Unit (RSU), it is always evaluated as the best possible neighbor and a new transmission process is immediately started.

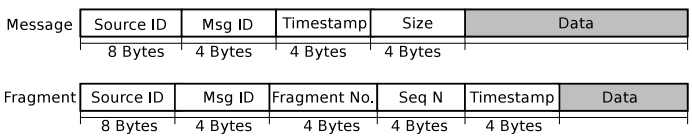


Fig. 3: Information message format and fragment format.

The transmission process tries to transmit as much fragments as possible to the best neighbor; its behavior, represented in Figure 4b, is as follows:

1) The module checks if there are fragments in the buffer to be transmitted; if there are no fragments, the process is over.
2) It obtains the best next node from the node list according to the protocol-specific metric.
3) It checks if the best node's location estimation is inside the defined transmission range.
4) It checks if the transmission window of the best node is equal to 0; in this case, the process is over.
5) It sends a data packet containing as many fragments as possible. This action dequeues the fragments from the buffer.
6) If the ACK mechanism is enabled, it schedules an ACK

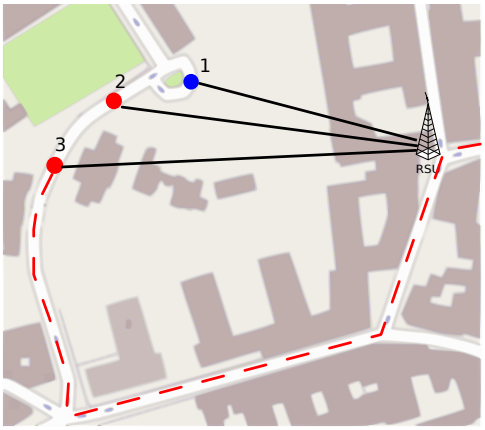


Fig. 5: Example of a local minimum: messages get blocked at node 1.

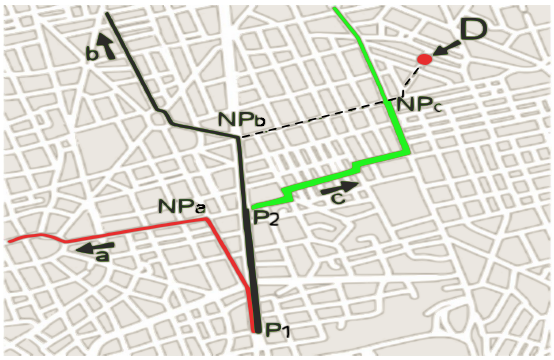


Fig. 6: Example of calculation of the closest point [5].

Time Out for the previously sent data message. If it is not running, it removes the sent fragments.

7) It waits for the time required to send a data packet and its corresponding ACK packets.
8) It starts a new transmission process.

It is worth noticing that, since all the fragments have the same destination (any of the RSUs), the next forwarding node selection is based only on the state of the current carrier and its neighbors. Moreover, we consider that all the RSUs behave as *sinks*. They simply send beacons to announce their presence to vehicles. When vehicles send Data messages to the RSUs, they may confirm their reception through an ACK message if required, and the fragments encapsulated in it are forwarded to the control center through the backbone network.

With GOD, when the ACK mechanism is active, it is ensured that no fragment is removed from the buffer until a neighbor have been confirmed as the new carrier. ACK messages are also used to limit the number of data messages pending confirmation. The use of ACK, as well as the communication type, *i.e.* broadcast or unicast, can be configured in order to model different DTN protocol variants.

III. Overview of the Evaluated DTN Protocols

We used our *Generic DTN One-Copy protocol* to implement and compare three different DTN protocols, namely: *Greedy*, *GeOpps*, and *MSDP*. A common assumption of all these protocols is the presence of a Navigation System (NS) installed in the vehicles. This way, each vehicle is aware of its geographical location and its route. This information is used to increase the packet delivery ratio in DTNs.

A. Greedy

The *Greedy* protocol, or a slightly modified version of it, has been widely used in order to evaluate other proposals, e.g., [5], [8]. It is a DTN variation of existing *location-based* greedy algorithms [12], [9], where the fragments are forwarded to the neighbor that is the closest one to the destination (if closer than the current carrier). Therefore, it uses the distance to the destination as its routing metric. This metric may lead to message losses or high delays when a geographic local minimum exists. Figure 5 shows an example of a local minimum where messages get blocked for a long time on the position of node number 1, since it is the closest node to the RSU.

B. GeOpps

The *GeOpps* protocol was presented in [5]. The next forwarding node selection is based on the Estimated Time of Arrival (ETA) to the destination. GeOpps assumes that each node is aware of the programed route in their Navigation System (NS). Using this information, nodes calculate the closest point to the destination along their route. Then, the closest point is used together with the map information to calculate the ETA from the current position to this closest point. The routing metric, which is called Minimum Estimated Time of Delivery (METD), is the sum of the ETA from the current position to the closest point plus the ETA from the closest point to the destination. Figure 6 shows an example of calculation of the closest point where the solid lines indicate the programmed routes while the dashed lines indicate the estimated route from the closest point to the destination. Finally, the value of the METD is attached to beacon messages, which allows nodes to be aware of the METD of their neighbors, and chose the minimum METD neighbor as the next forwarding node.

C. MSDP

In this article we used a slightly modified version of our MSDP protocol presented in [11]. In order to make routing decisions, MSDP uses the value of a function, called UtilityIndex, to determine which is the best neighbor to forward fragments to. In this work we add a second term to the UtilityIndex, $1/D$, which relates to the distance between the node and the destination of the message. The UtilityIndex, which depends on four parameters, is calculated locally and attached to beacons. The higher the UtilityIndex is, the better the candidate, see Equation 1.

$$UtilityIndex = \frac{P^2}{T} * Q + 1/D \quad (1)$$

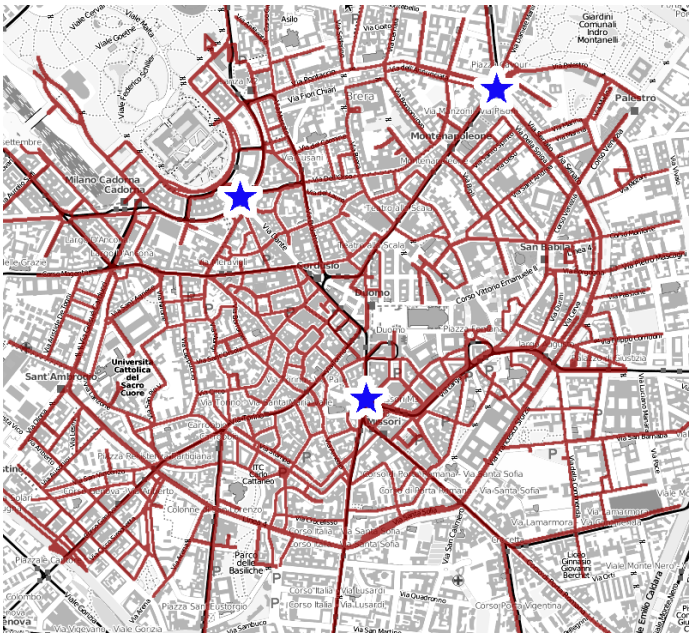


Fig. 7: Map used in our simulations, stars indicate the location of the RSUs (2.6 x 2.6 km).

Parameters P, T, Q, and D refer to *Trustworthy factor*, *Time to reach an RSU*, *Transmission availability*, and *Distance to an RSU*, respectively. In this paper we will only explain in detail this new term, and refer to our previous publication for the rest of the parameters. The previous version of MSDP suffered of inactivity when the carrier node and all its neighbors experienced a value of 0 for the first term of the equation 1. Under this circumstance, no messages were forwarded. By the parameter D we ensure that, at least, messages will be forwarded to the closest neighbor, which increases the probability of eventually finding a node whose first term of the equation is bigger than 0.

IV. Simulation Environment

In this Section we use our *Generic On-Copy Dtn Model* to compare our MSDP against the *Greedy* and *GeOpps* protocols through simulations.

We consider as the reference scenario the one selected in [11]. In that work vehicles used a DTN protocol to collect information from a vehicular sensor network and to deliver it to a remote control center. The information is obtained from in-vehicle sensors, which retrieve it using an On Board Diagnostic 2 (OBD-II) unit [13]. The information messages are then fragmented and routed. A fragment is considered to be delivered when any of the RSUs correctly receive it. Once an RSU receives a fragment, the fragment is sent over the backbone network to the control center. The control center will then reassemble the fragments into the original message and process the content. RSUs are supposed to be placed in strategical places by entities interested in collecting the information, like city councils, or road administrators. The locations of the RSUs are available to vehicles' NSs through a dynamic updating service.

We implemented all the models using the Inet framework for the Omnet++ event-driven simulator [14]. The Inet framework includes detailed implementations of the 802.11 physical and MAC layers.

One of the most important issues in DTN simulations is the mobility of nodes. In our simulations we have generated the node mobility by using VaCaMobil [15]. VACaMobil introduces a configured number of vehicles in the network and keeps this number inside the user defined upper and lower bounds. When a node finishes its trip, it is removed from the network. When using VACaMobil, the network simulator can influence the mobility of the nodes, which is simulated using Simulation of Urban MObility (SUMO) [16]. For the layout we used a 6.7 km^2-area map of the city center of Milan, which has a typical European old city structure. Figure 7 shows the portion of the map used, and the locations of the RSUs are indicated by stars.

We consider that the use of a very simplistic propagation model is one of the main drawbacks of previous studies in this topic. To accurately model real world conditions, we used the propagation model presented in [17], which combines the Nakagami fading model [18] with a visibility model which deals with power losses due to the effect of obstacles.

Every node in our network scenario generates a 2 kBytes message every 10 seconds. The size of the fragments is 450 Bytes. The simulation lasts 3600 seconds, and nodes will generate traffic throughout the entire simulation. Concerning the communication interfaces, each node has two 5.9 GHz 802.11p interfaces tuned at different channels, where the first one is used for beacon broadcasting, while the second one is used for one-hop transmissions. This double-interface model mimics the multi-channel communications scheme of the Dedicated short-range communications (DSRC) [19]. The transmission power of both interfaces is configured to 63 mW, while the gain of the antenna is 5 dBi.

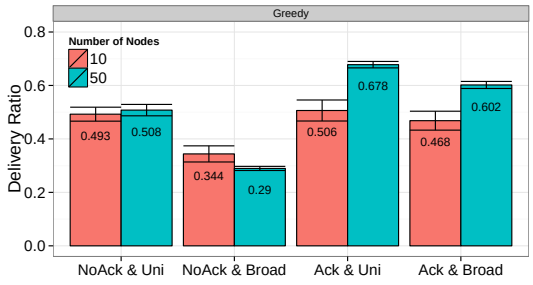
To achieve reasonably conclusive results, we have simulated every scenario 10 times, varying the seed of the simulation. Measures are represented with a 95% confidence interval.

V. Results

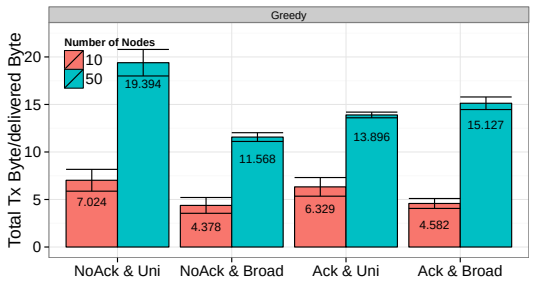
To determine the most relevant reference scenario, we first analyzed the impact of: (a) using ACK messages, and (b) using broadcast or unicast communications. The best-performing transmission mode is then used to compare the performance of Greedy, GeOpps and MSDP as a function of the network density. We evaluate the delivery ratio, the overhead, and the end-to-end delay for each configuration scheme.

A. Evaluating the impact of ACK messages

As explained in previous sections, even simple mechanisms such as the use of one-hop ACK messages to confirm that another node will carry the forwarded fragment, or whether the implemented protocol uses broadcast or unicast communication, can heavily impact the performance of a DTN



(a) Delivery Ratio



(b) Overhead.

Fig. 8: Comparison between different ACK and Broadcast/Unicast combinations.

protocol. In order to evaluate this impact we have simulated four different cases: 1) without ACK messages and using unicast communication, 2) without ACK messages and using broadcast communication, 3) using ACK messages and unicast communication, 4) using ACK messages and broadcast communication. To take scalability into consideration, we varied the number of nodes introduced in our network from 10 to 50, therefore varying the node density from 1.47 to 7.4 nodes/km^2.

The first metric we used was the *Delivery Ratio*, *i.e.* the number of messages successfully received by an RSU divided by the total number of messages sent. Figure 8a shows the result we obtained for the Greedy protocol. Case 2 presents the lowest delivery ratio for both shown densities. A similar performance is shown when only 10 nodes are deployed in the network, but, when we increase the amount of nodes in the network up to 50, case 3 clearly stands up as the best option. Concerning the introduced overhead of the four cases, see Figure 8b, it would be easy to conclude that case 2 is the best option, since it requires fewer bytes per delivered byte; however this result is heavily biased by its low delivery ratio. Eventually, since the three other cases introduce a similar overhead, we conclude that the best option, *our reference transmission mode*, is case 3, *i.e.* using ACK messages and unicast communication, since it is the one that experiences the best delivery ratio at a similar cost in terms of resources.

B. Evaluating the impact of network density

We now use the reference best performing transmission mode, determined in the previous subsection, to compare the three evaluated protocols according to: the delivery ratio, the

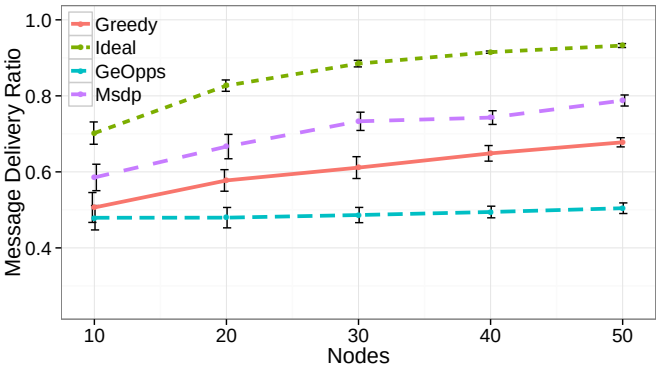


Fig. 9: Delivery Ratio

end to end delay of delivered messages, and the consumed resources (overhead). To evaluate the impact of node density we vary the number of nodes in the network from 10 to 50, thus varying the node density from 1.47 to 7.4 nodes/km^2.

1) Delivery Ratio: The delivery ratio is the ratio between the effectively delivered and the sent messages. Due to mobility-related effects, it is typically impossible to achieve a perfect delivery ratio. To determine a reference upper bound for this metric we have implemented an *Ideal Protocol*. The *Ideal Protocol* is an implementation of the Epidemic protocol [1] that runs on the top of a collision-free MAC layer which only considers propagation and transmission delays, and defines a limited transmission range. Since a copy of every fragment is sent to all neighbors every time a contact occurs, the first copy of a fragment arriving to any of the sinks must have traversed the best possible path in terms of delay.

Figure 9 shows the delivery ratio of the different protocols; it is worth noticing that not even the Ideal protocol reaches a delivery ratio of 1. When there are very few nodes in the network, the delivery ratio depends a lot on the mobility patterns of nodes, which explains the high confidence intervals obtained. On the other side, as we increase the number of nodes in the network, the delivery ratio also increases, because more transmission opportunities become available.

As can be seen, MSDP performs better than any other of the compared protocols, and GeOpps performs worse than

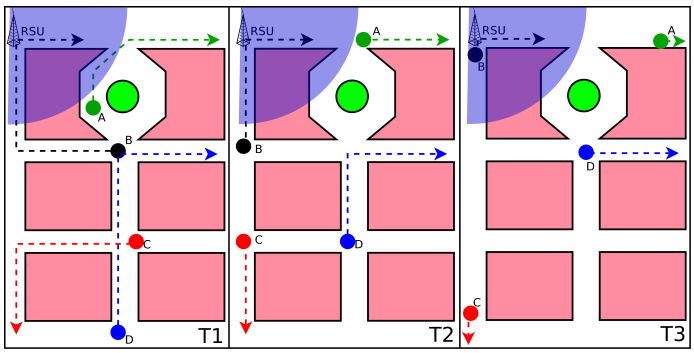
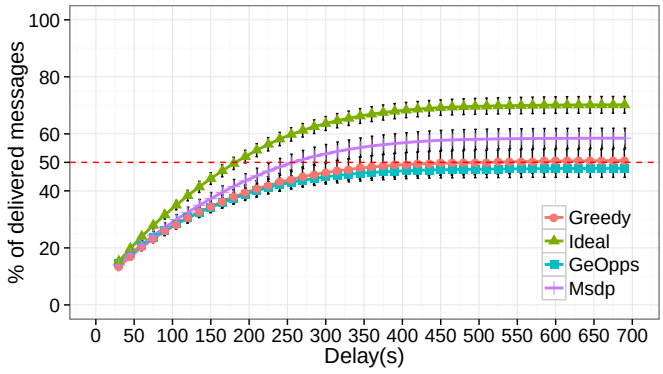


Fig. 10: Node D has a packet to be delivered to the RSU.

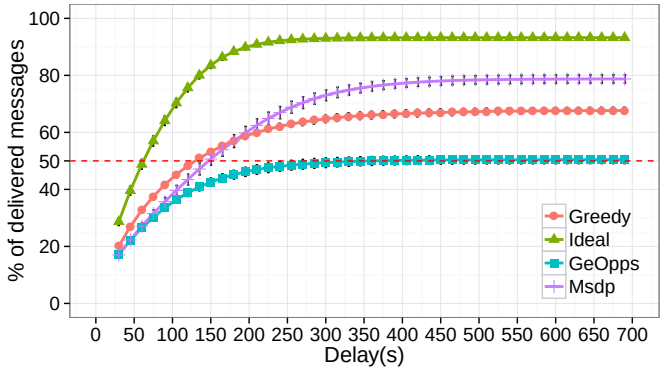
Greedy. One of the reasons why MSDP outperforms the other proposals is shown in Figure 10. When there is only a fragment stored in node D's buffer to be delivered, and assuming that the communication range is approximately equal to the width of a block of buildings (pink rectangles), the considered protocols would behave as follows:

- When using the *Greedy DTN* protocol, the fragment would be immediately forwarded to node A, which is the node closest to the RSU. Then, in $T2$ and $T3$, node A would move away from the RSU with no chance of forwarding the message to any other node.
- When using GeOpps, node D would keep the message in its buffer, since in $T1$ and $T2$ its Minimum Estimated Time of Delivery (METD) is better than the one of node C. However, at $T3$, when node D arrives to its closest point, it would not find any neighbor to forward the message.
- When using MSDP, in $T1$ the UtilityIndex of node C would be bigger than the UtilityIndex of node D due to the second term of the equation 1. Therefore, the fragment is forwarded to node C. Then, according to figure 4a, node C would immediately start a new transmission, forwarding the fragment to node B, which would keep the fragment in its buffer, since its UtilityIndex is bigger than the ones of nodes A and C. In $T3$ node B would be able to deliver the fragment to the RSU. When using the old version of MSDP (without the second term of equation 1), the UtilityIndex of both nodes, D and C, would be 0. In that case, node D would instead keep the packet in its buffer.

2) Delay: The delay is the time elapsed since a message is generated until it is received by an RSU. To evaluate the delay suffered by messages when using each of the protocols, we decided to represent the delay cumulative distribution, which shows the percentage of sent messages that experience a delay smaller than the one represented in the *X* axis. Figures 11a and 11b show the results we obtained for two different vehicle densities. When only 10 nodes are introduced in the network, which results in a very low node density (1.48 nodes/km^2), our MSDP delivers messages faster than any of the other protocols, for example, MSDP delivers 50% of the messages in less than 250s, while the Greedy protocol delivers less than the 45% of the messages during that same period. When the number of nodes is incremented up to 50 nodes (7.4 nodes/km^2), the connectivity of the network increases, and, therefore, the Greedy protocol is able to deliver more messages quickly. In fact, the Greedy protocol delivers the 50% of the packets in less time than our MSDP; however, MSDP is able to achieve a better final delivery ratio on the long term. Since MSDP makes an intensive use of node mobility, the delay experienced by messages, compared to the Greedy protocol, is slightly incremented when a bigger number of direct multihop routes are available. This fact also explains why the difference between the Ideal model and our MSDP is bigger as the density of nodes is incremented.



(a) 10 Nodes



(b) 50 Nodes

Fig. 11: Cumulative distribution of data message delay

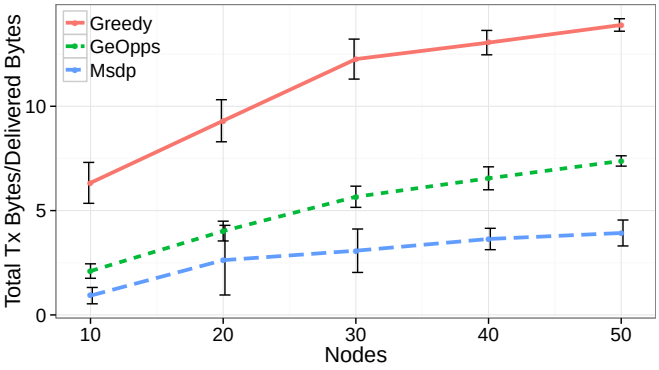


Fig. 12: Total Tx Bytes per Delivered Bytes

3) Overhead: Last but not the least, we evaluate the overhead introduced by each protocol. To obtain the overhead ratio of the evaluated protocols, we divide the total number of transmitted bytes by the number of delivered bytes. This measurement is closely related with the number of hops a fragment must traverse before it arrives to an RSU. Since it does not take into account the amount of data interchanged between nodes and RSU, its value may be smaller than one when most of the delivered fragments are directly transmitted to an RSU.

Figure 12 shows the results we obtained for the different

protocols; The first thing that we appreciate is that our MSDP is the one experiencing the smallest overhead ratio. Both GeOpps and MSDP perform better than the Greedy protocol, which consumes significantly more resources. This difference is explained by the presence of loops; when two nodes move in opposite directions, one node moving away from the RSU; and another one moving close to it, meet, the fragments are sent firstly to the closest one and then, after they cross, are sent back to the original carrier, which is moving closer to the RSU, the same situation occurs when a vehicle overtakes another vehicle. The UtilityIndex of MSDP, as well as the METD of GeOpps, are more stable and, therefore, most loops are avoided.

VI. Conclusions

In this paper we tackled the problems that arise when comparing different DTN protocols. To solve some of them, we have presented the Generic One-Copy DTN Model (GOD). This model allows us to fairly compare different DTN protocols, making it easy to define and apply common low-level sending mechanisms in one-hop communications.

We used GOD to fairly compare our previously presented proposal, called MSDP, against the Greedy and GeOpps protocols. To find the best one-hop communication strategy we first carried out a study to evaluate the impact of the use of i) one-hop ACK messages to confirm that other node will carry the forwarded fragment, or ii) whether the implemented protocol uses broadcast or unicast communication. The results obtained show that MSDP outperforms both Greedy and GeOpps protocols in terms of both delivery ratio and introduced overhead. On the contrary, when the node density of the network increases, the Greedy protocol delivers more messages during an initial short-time period. Nevertheless, our MSDP is able to deliver more messages on the long term. We will analyze the nature of this behavior and use this analysis to improve MSDP in future works.

In this paper we also presented an Ideal DTN model that represents the best possible case where all the forwarding decisions are optimal. The inclusion of this model pointed out that MSDP still has room for improvements. In future works, we will analyze the optimal routes used by the Ideal DTN model in order to mimic them and make new improvements to our MSDP scheme.

Acknowledgments

This work was partially supported by the *Ministerio de Ciencia e Innovación*, Spain, under Grants TIN2011-27543-C03-01 and BES-2012-052673.

References

[1] A. Vahdat and D. Becker, "Epidemic Routing for Partially Connected Ad Hoc Networks," in *Technical Report CS-200006*, vol. CS-200006, no. CS-200006. Duke University, Apr. 2000, pp. CS–2000–06.

[2] T. Spyropoulos, K. Psounis, and C. S. Raghavendra, "Spray and wait," in *Proceeding of the 2005 ACM SIGCOMM workshop on Delay-tolerant networking - WDTN '05*, ser. WDTN '05. New York, New York, USA: ACM Press, 2005, pp. 252–259.

[3] T.-K. Huang, C.-K. Lee, and L.-J. Chen, "PRoPHET+: An Adaptive PRoPHET-Based Routing Protocol for Opportunistic Network," in *2010 24th IEEE International Conference on Advanced Information Networking and Applications*. Ieee, Apr. 2010, pp. 112–119.

[4] B. Karp and H. T. Kung, "GPSR: greedy perimeter stateless routing for wireless networks," in *Proceedings of the 6th annual international conference on Mobile computing and networking*, ser. MobiCom '00, no. MobiCom, ACM. New York, NY, USA: ACM, 2000, pp. 243–254.

[5] I. Leontiadis and C. Mascolo, "GeOpps: Geographical Opportunistic Routing for Vehicular Networks," in *World of Wireless, Mobile and Multimedia Networks, 2007. WoWMoM 2007. IEEE International Symposium on a*. IEEE, Jun. 2007, pp. 1–6.

[6] J. LeBrun, C.-N. Chuah, D. Ghosal, and M. Zhang, "Knowledge-based opportunistic forwarding in vehicular wireless ad hoc networks," in *Vehicular Technology Conference, 2005. VTC 2005-Spring. 2005 IEEE 61st*, vol. 4, 2005, pp. 2289–2293 Vol. 4.

[7] M. Torrent-Moreno, S. Corroy, F. Schmidt-Eisenlohr, and H. Hartenstein, "IEEE 802.11-based one-hop broadcast communications: understanding transmission success and failure under different radio propagation environments," in *Proceedings of the 9th ACM international symposium on Modeling analysis and simulation of wireless and mobile systems - MSWiM '06*, ser. MSWiM '06. New York, New York, USA: ACM Press, 2006, pp. 68–77.

[8] P.-C. Cheng, K. C. Lee, M. Gerla, and J. Härri, "GeoDTN+Nav: Geographic DTN Routing with Navigator Prediction for Urban Vehicular Environments," *Mobile Networks and Applications*, vol. 15, no. 1, pp. 61–82, Jun. 2009.

[9] C. Lochert, M. Mauve, H. Füß ler, and H. Hartenstein, "Geographic routing in city scenarios," *ACM SIGMOBILE Mobile Computing and Communications Review*, vol. 9, no. 1, p. 69, 2005.

[10] V. N. G. J. Soares, J. J. P. C. Rodrigues, and F. Farahmand, "GeoSpray: A Geographic Routing Protocol for Vehicular Delay-Tolerant Networks," *Information Fusion*, Nov. 2011.

[11] S. Martinez Tornell, C. T. Calafate, J.-C. Cano, and P. Manzoni, "A Map-based Sensor data Delivery Protocol for vehicular networks," in *2012 The 11th Annual Mediterranean Ad Hoc Networking Workshop (Med-Hoc-Net)*. Ayia Napa, Cyprus: IEEE, Jun. 2012, pp. 1–8.

[12] C. Lochert, H. Hartenstein, J. Tian, H. Fussler, D. Hermann, and M. Mauve, "A routing strategy for vehicular ad hoc networks in city environments," in *Intelligent Vehicles Symposium, 2003. Proceedings. IEEE*, vol. 2000, no. 01, ACM. Ieee, 2003, pp. 156–161.

[13] International Organization for Standardization, "ISO 15765: Road vehicles, Diagnostics on Controller Area Networks (CAN)," 2004.

[14] "Omnet++," http://www.omnetpp.org/, last visit march 2013.

[15] M. Báguena, S. M. Tornell, A. Torres, C. T. Calafate, J.-C. Cano, and P. Manzoni, "VACaMobil: VANET Car Mobility Manager for OMNeT++," in *3rd IEEE International Workshop on Smart Communication Protocols and Algorithms (SCPA 2013) (ICC'13 - IEEE ICC'13 - Workshop SCPA)*, Budapest, Hungary, Jun. 2013.

[16] M. Behrisch, L. Bieker, J. Erdmann, and D. Krajzewicz, "SUMO - Simulation of Urban MObility: An Overview," in *SIMUL 2011, The Third International Conference on Advances in System Simulation*, Barcelona, Spain, Oct. 2011, pp. 63–68.

[17] M. Baguena, C. T. Calafate, J.-C. Cano, and P. Manzoni, "Towards realistic vehicular network simulation models," in *2012 IFIP Wireless Days*. IEEE, Nov. 2012, pp. 1–3.

[18] T. Rappaport, *Wireless Communications-principles and practice 2002, edition second*, 2nd ed. Prentice Hall PTR, Jan. 2002.

[19] D. Jiang and L. Delgrossi, "IEEE 802.11p: Towards an International Standard for Wireless Access in Vehicular Environments," *VTC Spring 2008 IEEE Vehicular Technology Conference*, pp. 2036–2040, 2008.