CARPOOL: Connectivity Plan Routing Protocol

Ioannis Komnios and Vassilis Tsaoussidis
Space Internetworking Center, Office 1, Building A, Panepistimioupoli Kimmeria,
Department of Electrical and Computer Engineering, Democritus University of Thrace,
67100, Xanthi, Greece
{ikomnios, vtsaousi}@ee.duth.gr

Abstract. Basic Internet access is considered a human right, however geographical, technological and socio-economic reasons set barriers to universal Internet access. To address this challenge, we have proposed an access method based on message ferrying that enables free delay-tolerant Internet access to all, and developed Connectivity plAn Routing PrOtOcOL (CARPOOL), a reference routing protocol for the proposed access method. In this paper, we describe CARPOOL in depth and evaluate its performance for increasing traffic load. Focusing on an urban scenario, where means of public transport, such as buses, follow predefined routes and schedules, CARPOOL utilises *a priori* knowledge about their current location to extend Internet access provided by hotspots to users and areas that are not typically covered. Our simulation results show that CARPOOL effectively exploits the existing connectivity plan of public transportation, achieving high delivery ratio with minimum overhead. This paper also discusses possible enhancements of the proposed routing protocol.

Keywords: Delay Tolerant Networking, DTN routing, Message ferries

1 Introduction

The majority of people living in the developed world are already experiencing how access to the Internet is transforming their way of living. Internet has now become a critical infrastructure for the society with its availability levels increasing and its traffic volume constantly growing. Based on this consensus, in 2011 the United Nations declared Internet access itself a human right [1]. In a constantly evolving and expanding digital world, however, geographical isolation and socio-economic restrictions pose barriers to the invasion of the Internet to all parts of the society: remote regions demand significantly higher cost for Internet deployments, while economic challenges exclude the under-privileged from accessing the Internet even in well-connected environments.

Delay/Disruption Tolerant Networking (DTN) architecture [2, 3] and its supporting Bundle Protocol (BP) [4] is an emerging technology to support the new era in interoperable communications by providing delay-tolerant access even when traditional continuous end-to-end connectivity fails. DTN has been frequently coupled with the concept of message ferrying, especially as far as remote areas are concerned, to facilitate data transfers through cars, buses, trams, trains etc.

In this paper, we focus on metropolitan environments with an ultimate goal to extend free delay-tolerant Internet access to the under-privileged society that is currently excluded from today's digital world. To achieve that, we extend the existing free Internet access provided by public hotspots that are usually scattered around a city. Actually, we broaden connectivity options by deploying DTN nodes both on typical means of public transport (ferries), such as buses and trams, and their corresponding stops. Offline DTN gateways located near ferry stops collect Internet access requests from end-users in that area and DTN ferries act as relays between offline gateways or designated gateways that have access to the Internet and are capable of handling such requests. Through simulations we have identified that existing DTN routing solutions underperform in such dense environments, due to their associated high overhead and their excessive energy needs.

Our novelty lies in the utilization of *a priori* knowledge of contacts between gateways and ferries, in an effort to achieve high delivery ratio with minimum overhead. First, we investigate the potential of existing DTN routing schemes to support free Internet in high traffic load conditions and we observe that existing protocols are insufficient primarily because they fail to guarantee some level of service. Further, we describe and evaluate CARPOOL, a DTN routing protocol that utilises the connectivity plan between ferries and gateways (i.e. ferry stops) to compute routes to online gateways. CARPOOL was briefly described in [5]; here we describe and evaluate CARPOOL in detail, we discuss issues that may arise and we propose possible solutions. We also note that geographical extension of the Internet is here confined only within a metropolitan area: we do not include here ferries to reach isolated regions. However, this is our ultimate target and does not cancel the advantages of this standalone proposal. These are: (i) an easy-to-deploy access method that exploits information regarding the schedule of the ferries, which is already available and well-known in all major cities worldwide, (ii) free delay-tolerant access to the Internet for everyone, and (iii) energy-efficient design that delegates all expensive computing operations to gateways with increased computing and power capacity.

The remainder of this paper is structured as follows: in Section 2, we discuss related work in the field of DTN and free Internet access. In Section 3, we describe in detail the proposed access method along with CARPOOL routing protocol, while in Section 4 we present our experimental results. In Section 5 we discuss CARPOOL and we propose mechanisms to enhance its performance. We conclude this paper in Section 6.

2. Related Work

In an effort to provide Internet access to all members of the society, several economic models, such as providing restricted Internet access during night at a lower price, have been proposed in the recent past. Nonetheless, these models are not affordable to all, leaving certain members of the society with the only alternative of using random hotspots when available. Several governments and local administrations have undertaken the initiative to deploy hotspots in points of interests, however cost-efficiency is a critical factor that hinders extended deployments [6].

User-provided networks, where an Internet connection is shared freely and transparently among end-users in a way that is technically and legally independent of access or infrastructure providers, have been proposed as a solution [7]. Among others, the authors of [8] explore incentives and algorithms for broadband access sharing to support nomadic users. On the practical side, PAWS project [9] aims at providing free Internet for all by making the existing broadband connections in homes and public buildings publicly available. Similarly, BT FON initiative [10] encourages FON members to share their home broadband connection and get in return free access at millions of other FON hotspots, worldwide. Even though the aforementioned solutions can provide free Internet access in specific areas, they fail to provide extended coverage.

In recent years, Delay Tolerant Networking architecture has attracted the attention of the research community in an effort to provide Internet access to remote and disconnected regions. Routing has been one of the key challenges for DTNs, since an end-to-end path from the source to the destination might not exist in time. Epidemic routing [11] was one of the first proposals in this area by employing pair-wise exchanges of messages among all mobile hosts that connect to each other, maximizing delivery rate and minimizing overall latency. Naturally, the main disadvantage of epidemic routing is the extreme overhead it creates. Extending the idea of epidemic routing, the authors of [12] proposed a routing algorithm for Delay Tolerant Networks (PRoPHET) that exploits the non-randomness of real-world encounters by maintaining a set of probabilities for successful delivery to known destinations and replicating messages during opportunistic encounters only if the node that does not have the message appears to have a better chance of delivering it. In an effort to reduce the transmission overhead of epidemic routing while keeping delivery probability high, the authors of [13] proposed Spray-and-Wait routing protocol. In Spray-and-Wait, for each message originating at the source node, L copies are forwarded to the network. If the destination does not receive a copy of the message, each node that has received the message performs direct delivery to the destination.

As far as vehicular DTNs are concerned, MaxProp routing [14] is one of the most promising solutions based on prioritizing the schedules of packets to be transmitted and to be dropped. These priorities are built on path likelihood to peers according to historical data and some enhancement mechanisms. The concept of exploiting DTN ferries or data mules has been popular for data collection from sensors [15]. A few papers that consider message ferries for data transmission in DTN have also been presented in bibliography. KioskNet [16] was one of the first proposals on the field with a main goal to employ buses and cars as "mechanical backhaul" devices to carry data to and from a remote village and an Internet gateway. KioskNet was proposed at a period when DTN research was at its infancy, thus no clear routing solution was provided. ALARMS routing protocol [17] was later introduced to deliver bundles through message ferries. Ferries connect to gateways and pass information regarding their path for the next two rounds. Based on this information, gateways calculate the routing path that achieves earliest delivery. This work, however, does not consider global knowledge of the network and a path to the destination can only be found if a ferry exists to directly connect the source and the destination.

Our protocol differs from the aforementioned solutions, since CARPOOL utilizes *a priori* knowledge of future contacts between DTN ferries and stationary DTN gateways. Utilizing such knowledge, CARPOOL identifies all possible routes between two nodes and selects the one that achieves earliest delivery. Our aim is not to compare CARPOOL with DTN routing solutions that have no or partial knowledge of the network, but to highlight the inefficiency of these protocols in dense urban environments and propose a reference routing protocol that achieves high delivery ratio with minimum overhead. Our approach shares the philosophy of Contact Graph Routing [18], which is the most prominent routing solution in space internetworking. Similar to prescheduled contacts between ferries and gateways, Contact Graph Routing extracts a path for space data transmission utilizing *a priori* knowledge of contacts between space assets, which may include dynamic aspects as well [19]. Applications that can benefit from the proposed architecture include E-mail [20], fbDTN [21], Twitter [22] etc.

3 Architecture Overview

The proposed approach to free delay-tolerant Internet access aims at extending the existing access provided by public hotspots. Section 3.1 presents the access model we propose in order to achieve this goal, while Section 3.2 illustrates a specific realisation of the model within a DTN routing protocol that takes advantage of known contacts.

3.1 Model

Our access model consists of two major components: *DTN gateways* that are responsible for handling requests from end-users within their radius and *DTN ferries* that are responsible for transferring messages across the gateways. While both components are crucial for our access model, we intentionally delegated all computational tasks to the gateways, since we assume that DTN ferries have restricted energy and computational capabilities. Typically, the travel plan of buses, trams and trains is predefined and only minor delays can occur. Therefore, in our model we assume that all gateways have global knowledge of the connectivity plan. Of course, in case of a major delay, the updated traffic schedule is flooded into the network through a central administrative node.

DTN gateways are resource-capable fixed nodes located near ferry stops. We assume that certain gateways have access to the Internet through a hotspot that exists in the area (*online DTN gateways*), while the majority is offline. All gateways have effectively enough buffer to store messages from several end-users and are equipped with network interfaces for data exchange with the mobile devices of the end-users and the DTN ferries. Once an end-user device discovers a DTN gateway in its radius, a request to/from the Internet is transferred from/to the relevant application.

When a bundle is received by an offline gateway, valid paths between this gateway and online gateways are calculated based on the connectivity plan and a path that achieves earliest bundle delivery is selected. Once a path is selected, the gateway extracts the ID of the next gateway on this path, the ID of the ferry that will transfer the bundle and the estimated forwarding time and stores the bundle in its buffer until a contact to the ferry becomes available, when it forwards the bundle. The procedure of selecting the next gateway is detailed in the next subsection.

In essence, instead of storing the end-to-end path through the network, we only store the next gateway on the path. This approach ensures that our model takes into consideration and proactively handles changes to the initial connectivity plan. If the full path to an online gateway was stored, the time-shift of an intermediate contact, even for a few seconds, would lead to a significant delay, let alone bundle expiration. The proposed method reacts to changes in the state of the network by re-evaluating the best route for a bundle at every gateway.

Unlike upload operations, downloading data from the Internet requires an additional publish/subscribe session layer (e.g. similar to the one presented in [23]), in order to allow for applications such as RSS content distribution and web access over DTN. The proposed routing protocol can function efficiently in both cases. We also note that the proposed model can be extended to support data transfers between end-users that are located near gateways, as well. Fig. 1 contains a sample topology corresponding to our model. We highlight that the majority of gateways do not have access to the Internet.

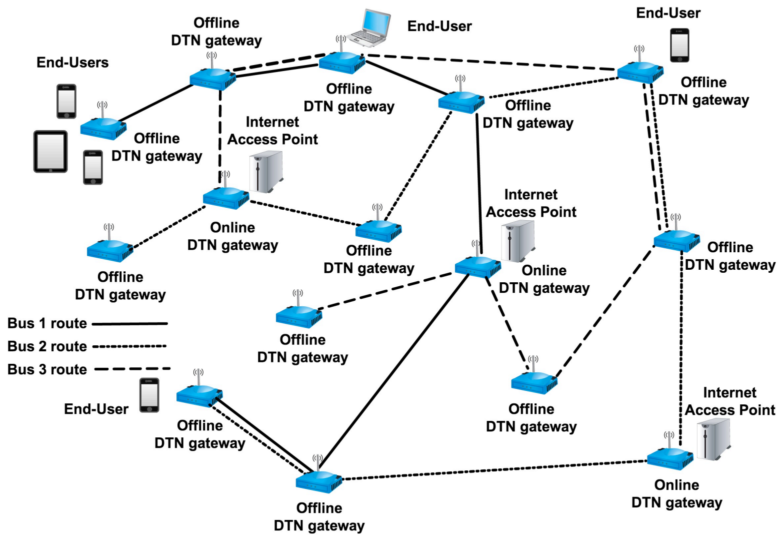

Fig. 1. Sample topology

3.2 CARPOOL Protocol

In order to support the proposed access model, we have designed and implemented CARPOOL, a DTN routing protocol that utilises *a priori* knowledge of the connectivity plan to deliver bundles among DTN gateways. The proposed routing protocol works as follows: all gateways hold the list of online gateways and the overall connectivity plan, which includes all contacts between gateways and ferries along with the scheduled start of each contact. In particular, the entries of the connectivity table for each ferry have the following 3-tuple structure (GatewayID, FerryID, ContactTime). When an offline gateway receives a new bundle from an end-user, CARPOOL identifies the most suitable next gateway for this bundle, in terms of estimated bundle delivery time to an online gateway. CARPOOL identifies paths between an offline gateway and an online gateway, starting from the destination, and moving towards the source in a hop-by-hop manner. The values required as input to the algorithm are PreviousGateway and NewArrivalTime. Initially, PreviousGateway is set to the ID of an online gateway and NewArrivalTime equals to bundle creation time plus TTL. The current gateway first identifies all contacts in the overall connectivity table that satisfy the following requirements:

- GatewayID equals PreviousGateway and
- ContactTime is greater than CurrentTime and less than the latest arrival time (NewArrivalTime).

For each of the aforementioned contacts, we store a set of 3-Tuples: the contact itself and the exact previous contact (in terms of time) between the same ferry and another gateway. When the previous gateway that this ferry has traversed becomes the current gateway, we have identified a direct contact, where the current gateway is only one hop away from an online gateway. Otherwise, the algorithm re-executes using as input the GatewayID and the ContactTime of the previous contact. Thus, we now search for valid contacts that are two hops away from an online gateway. This process is continued until a path is found.

In order to reduce the complexity and the associated computational overhead of our algorithm, instead of identifying all possible paths and selecting one that achieves earliest delivery, we first sort valid contacts to an online gateway starting from the earliest, prior to applying our selection algorithm. This way, we need not calculate all paths from the current gateway to all online gateways; instead we simply select the first plausible path to an online gateway, which is also a path that guarantees earliest delivery. Once a path has been discovered, the GatewayID of the next gateway on the path (NextGateway), the FerryID of the ferry that will transfer the bundle (NextFerry) and the TimeToForward that corresponds to the time that the bundle will be forwarded are added to the header of the bundle; the bundle is stored in the gateway buffer, until a connection between the gateway and *this* ferry exists. The CARPOOL Algorithm is presented in Algorithm 1.

When a connection is up, the gateway uploads bundles waiting to be forwarded through that ferry and downloads bundles from the ferry that are destined to that gateway. When an online gateway receives a new bundle, the bundle is forwarded to the receiving application through the Internet. When an offline gateway receives a bundle, the algorithm is re-executed and the corresponding fields in the header of the bundle are updated. Our access model faces two limitations: the finite buffer size of gateways and ferries, as well as the small window of communication opportunities between gateways and ferries. In the event that this window does not suffice for all bundles to be delivered at the gateway or the ferry respectively, the path for the unserviced bundles is re-calculated. Similarly, CARPOOL re-calculates the path for all bundles in the network with TimeToForward greater than CurrentTime aligned within a fixed threshold, set according to the arrival time of the next ferry to this gateway, in order not to miss this contact opportunity. This allows to cancel the impact of the schedule deviation and typically suffices to accommodate minor schedule drifts. In essence, CARPOOL recalculates routes each time a message misses its expected contact due to high load in the network and short connectivity time between gateways and ferries.

It should be noted that, in contrast to most solutions proposed in literature, CARPOOL is not a replication scheme. Only a single copy of each bundle exists in the network at any given time, keeping overhead to minimum.

```
For each Ferry F do
        For Contact C of ConnectivityTable_F do
                If (C.GatewayID = PreviousGateway) AND
                   (C.ContactTime ≥ CurrentTime) AND
                   (C.ContactTime ≤ NewArrivalTime) then
                     //A valid contact has been found. We store a set of 3-tuples: this
                //contact and the previous contact the ferry has with another gateway
                        Add Set(C_prev,C) to ValidContacts
                Endif
        Endfor
Endfor
//Sort valid contacts starting from the earliest contact
C Sort ValidContacts
//Identify a path from source to destination
For Set(S_prev,S) of ValidContacts do
        If (S_prev.Gateway = CurrentGateway) then
        //Path found. Store header fields and exit algorithm
                NextGateway = S.GatewayID
                NextFerry = S.FerryID
                TimeToForward = S_prev.ContactTime
                Exit
        Else
        //We have not found a path from current gateway to the online gateway.
        //Re-run the algorithm moving one hop further from the online gateway
                PreviousGateway = S_prev.GatewayID
                NewArrivalTime = S_prev.ContactTime
                Algorithm (PreviousGateway, NewArrivalTime)
        Endif
Endfor
```

Algorithm 1. CARPOOL Algorithm

4 Evaluation

Through extensive simulations, we evaluate the performance of CARPOOL in a dense urban environment and study the impact of increased traffic load on its performance comparatively with four widely-used routing protocols, namely, Epidemic [11], PRoPHET [12], binary Spray-and-Wait with 10 message copies [13] and MaxProp [14].

4.1 Evaluation Methodology

The CARPOOL protocol has been implemented and evaluated using the Opportunistic Network Environment (ONE) simulator [24]. Initially, we created the connectivity plan for the entire simulation using as input:

(1) The ID and the coordinates of each gateway,

(2) The ID and the speed of each ferry, along with the gateways on the path of the ferry in the order it transverses them,

(3) The waiting time at each stop and

(4) The start times of each ferry.

We assume that all ferries follow the reverse path once they reach their destination. All gateways become aware of the connectivity plan.

We selected a topology for our simulations that corresponds approximately to an abstraction of the transport service of Thessaloniki, Greece, that includes both the city center and the suburbs. In total, our simulation environment covers an area of approximately 100 km^2 that includes 106 offline gateways (i.e. bus stops) and 15 online gateways. Our scenarios follow 60 ferries (i.e. buses) travelling on 20 routes. The speed of the ferries ranges from 5m/s to 14m/s. All gateways and ferries are equipped with 2GB storage size and wireless network cards at 10Mbps data rate and 50m communication radius. The overall duration of all simulations is 48 hours, including a sufficient training period for protocols to initialize themselves. The traffic load varies from 2500 to 50000 messages per 12 hours. Bundle size ranges from 500kB to 2MB. Given the delay-tolerant nature of the applications, bundle TTL is set to 20h, sufficiently large to accommodate all communication attempts by all protocols. The simulation topology is depicted in Fig. 2.

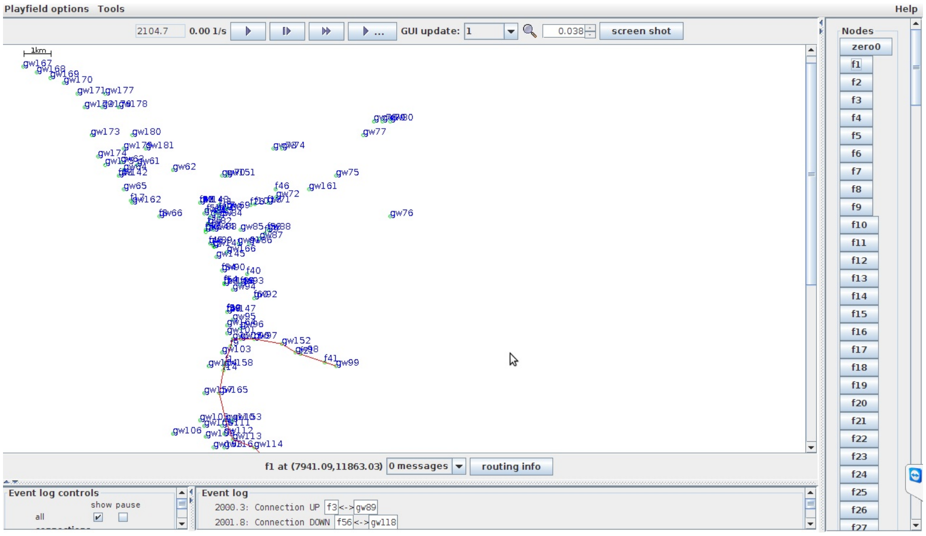

Fig. 2. Simulation topology

4.2 Evaluation Metrics

We evaluate performance using the following metrics:

1. Delivery ratio expresses the fraction of the total generated messages that are successfully delivered.

$$Delivery\ Ratio = \frac{\text{Number of messages successfully delivered}}{\text{Number of messages generated}}$$

2. Overhead ratio is calculated as the number of messages relayed minus the number of messages delivered to the number of messages delivered.

$$Overhead\ Ratio = \frac{\text{Number of messages relayed-Number of messages delivered}}{\text{Number of messages delivered}}$$

3. Median latency is computed as the numerical value separating the higher half of all message latencies from the lower half.

$$Median\ Latency = \frac{\sum_{i=1}^{\text{Number of messages successfully delivered}} \text{Latency}_i}{\text{Number of messages successfully delivered}}$$

4.3 Evaluation Results

Results in Fig. 3 illustrate the delivery ratio of the five routing schemes for increasing traffic load. We notice that in low traffic load conditions (less than 20000 messages in 12h) only CARPOOL and MaxProp manage to deliver all messages; the three other protocols fail to achieve maximum delivery ratio. CARPOOL achieves increased delivery ratio, since contacts between gateways and ferries are known *a priori* and in the event of unexpected delays, new paths to online gateways are being re-discovered. It should be noted that in contrast to other protocols, our current version of CARPOOL does not exploit short contacts between ferries. Through its scheduling tactics at the gateways and the ferries, MaxProp achieves high delivery ratio. Epidemic routing suffers from its excessive overhead and experiences worst performance. In high traffic load conditions (more than 25000 messages in 12h) the delivery ratio of all protocols decreases when traffic load increases. Even in the worst scenario, CARPOOL performs significantly better than all other protocols, managing to successfully deliver 82% of the created bundles despite heavy congestion. Unlike other protocols, the delivery ratio of CARPOOL, even in worst-case scenarios, suffices in its own right to guarantee some level of service. A user may feel confident that even if one attempt fails, most likely this will not be repeated. Several approaches to increase the delivery ratio of CARPOOL are proposed in Section 5.

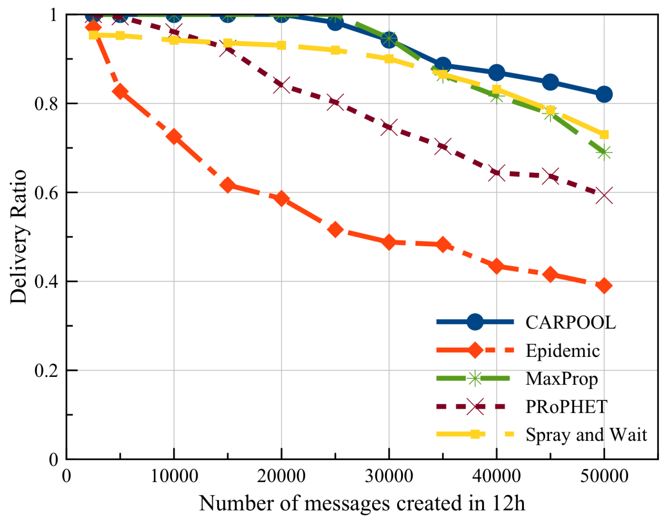

Fig. 3. Delivery ratio for increasing number of messages

In Fig. 4 we show the overhead ratio observed for each routing protocol for increasing traffic load. Given the density of the ferries and the gateways, along with the limited energy capacity of the ferries, overhead becomes important. As expected, CARPOOL presents minimum overhead, since there exists only one copy of each message in the network at any given time. Since CARPOOL keeps a single copy per message, it minimises energy consumption of battery-powered devices, but also allows for better bandwidth utilisation, which practically means our network can accommodate more users. Spray-and-Wait also keeps overhead low (as defined in its simulation settings), while the other three protocols suffer from increased overhead. As shown in Fig. 4, overhead decreases for the rest of the protocols when traffic load increases. This is justified by the protocols' failure to operate in regular mode, since they cannot create their typical number of copies. The result of this overhead reduction is their functional blackout as it appears in the corresponding heavy-traffic delivery ratio results in Fig. 3.

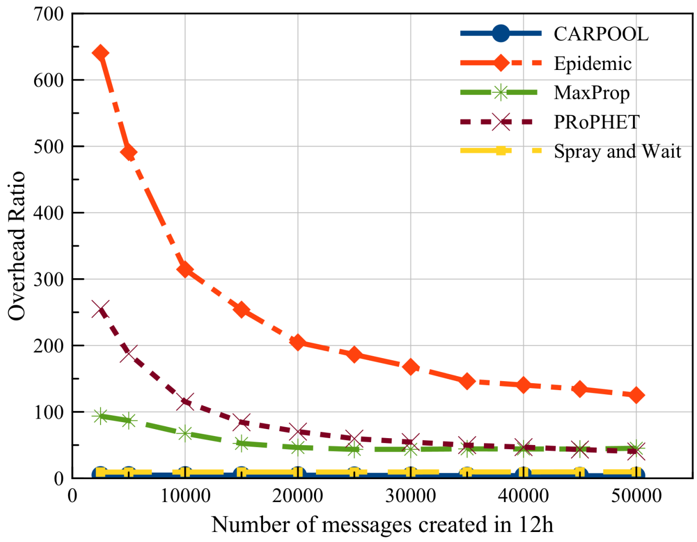

Fig. 4. Overhead ratio for increasing number of messages

In Fig. 5 we show the median latency of each protocol for increasing traffic load. The median latency of all protocols presents a steady increase for increasing traffic load with the exception of MaxProp, which presents a rapid increase in latency as traffic load increases. CARPOOL performs sufficiently well. Spray-and-wait outperforms all other protocols at the expense of higher overhead and less delivery ratio. CARPOOL achieves exactly the same median latency with Spray-and-Wait in highly congested networks and manages to deliver 10% more packets with 1/10 of the overhead of Spray-and-Wait. When the network is not congested, Spray-and-Wait achieves lower median latency than CARPOOL by exploiting opportunistic contacts between ferries as well. We are confident that CARPOOL will outperform Spray-and-Wait even in median latency, when opportunistic contacts between ferries are also exploited. Several approaches to achieve that are described in the following Section.

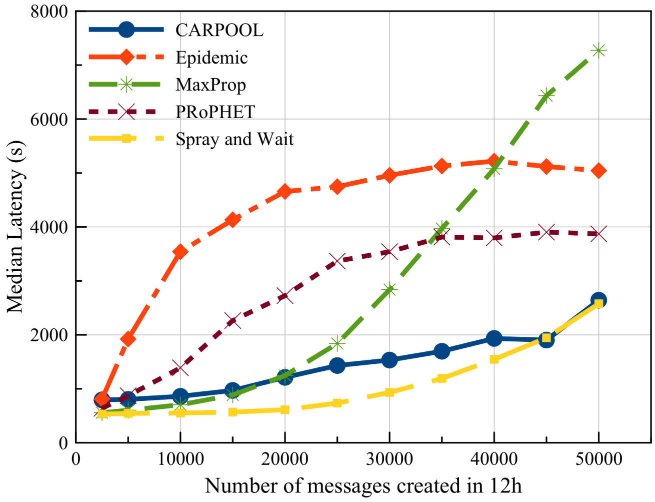

Fig. 5. Median latency for increasing number of messages

5 Discussion

The motivation behind this work was to design a platform for delay-tolerant Internet access that extends the existing coverage of free WiFi access points in an urban environment. Based on the simulation results in Section 4, we show that CARPOOL successfully exploits existing knowledge on the connectivity plan of typical means of public transport, in order to identify a route from the source node to a WiFi access point. From our evaluation results we observe that Spray-and-Wait with 10 copies achieves delivery ratio that approaches CARPOOL and smaller median latency when the network is not congested. This argues towards allowing a few copies to be sent over different paths in order to achieve increased delivery ratio (assuming that a node may fail to deliver its messages) and reduced delay. Towards this direction, we plan to incorporate a mechanism that injects two replicas of each message in the network that follow the two fastest paths, as calculated by the source gateway.

In order to further enhance the performance of CARPOOL, we plan to exploit opportunistic contacts between ferries. A simple approach would be for two ferries to exchange all messages when in range. However, this would lead to significant overhead and would not function properly given the small contact duration window. As a more sophisticated solution, during a contact between two ferries, both ferries can recalculate the estimated delivery time of all messages they hold through the other ferry; if the estimated delivery time of a message through the other ferry is smaller, then the message is forwarded to the other ferry.

The proposed system sustains minor delays by scanning for transmission opportunities for a time period that ranges for a few seconds before and after the estimated arrival of each ferry at a gateway. In order for the system to remain sustainable when a significant bias is introduced in the connectivity table or major delays occur due to traffic or road accidents and the new connectivity plan cannot be flooded to all nodes, we plan to enhance CARPOOL by allowing ferries to recalculate the estimated delivery time whenever a significant delay occurs. In particular, when a ferry reaches a gateway later than expected, it can download all messages destined to this gateway and, at the same time, recalculate the fastest route to an online gateway for all messages the ferry carries and are not destined to this gateway. This way, the overall latency will be significantly reduced. Major delays in ferry schedule can also lead in loops; to solve this problem we plan to include two new fields in message header: Last Ferry and Last Gateway. By holding and checking the values of the last ferry and gateway visited by the message, typical loops can be avoiding.

6 Conclusions

In this paper, we have described in detail an access model for urban environments suitable to extend Internet access opportunities to free users. To achieve this, we have employed delay-tolerant networking properties into CARPOOL routing protocol and have shown that an acceptable level of service can be provided. This was justified by our simulation results of delivery ratio (i.e. service probability) and overhead (i.e. potential to accommodate more users). Our work constitutes a first attempt to promote free Internet to all, relied on the potential to exploit the pre-known transportation schedules of large cities and was further enhanced with dynamic decisions to avoid schedule deviations. Several enhancements of CARPOOL were also proposed. As future work, we plan to incorporate these mechanisms into CARPOOL and investigate its performance when large deviations to the predefined contact plan occur.

References

1. La Rue, F. 2011. Report of the Special Rapporteur on the promotion and protection of the right to freedom of opinion and expression. *Human Rights Council, Seventeenth session Agenda item 3*. United Nations General Assembly
2. Cerf, V., Burleigh, S., Hooke, A., Torgerson, L., Durst, L., Scott, K., Fall, K., and Weiss, H. 2007. Delay–Tolerant Networking Architecture. *Internet RFC4838*.
3. Fall, K. 2003. A Delay-Tolerant Network Architecture for Challenged Internets.In *Proc. of ACM SIGCOMM*.
4. Scott, K., Burleigh, S. 2007. Bundle protocol specification. *RFC 5050*.
5. Komnios, I. and Tsaoussidis, V. 2013. CARPOOL: extending free internet access over DTN in urban environment. In *Proc. of the 2013 ACM MobiCom workshop on Lowest cost denominator networking for universal access* (LCDNet '13). ACM, New York, NY, USA, 21-24.
6. Thraves, G., Urueta, A., Vidales, P., and Solarski, M. 2008. Driving the deployment of citywide ubiquitous WiFi access. In *Proc. of the Proceedings of First International Conference on Simulation Tools and Techniques for Communications,* pp. 1-8.

7. Sofia, R., and Mendes, P. 2008. User-provided networks: consumer as provider. *Communications Magazine,* IEEE, vol. 46, no. 12, pp. 86–91.
8. Psaras, I., and Mamatas, L. 2011. On demand connectivity sharing: Queuing management and load balancing for user-provided networks, *Computer Networks,* vol. 55, no. 2, pp. 399–414.
9. Sathiaseelan, A., et al., "Public Access WiFi Service (PAWS)," Digital Economy All Hands Meeting, Aberdeen, October 2012.
10. FON WIRELESS. http://www.fon.com
11. Vahdat, A., and Becker, D. 2000. Epidemic routing for partially connected ad hoc networks. *Technical Report CS-200006*, Duke University.
12. Lindgren, A., Doria, A., Davies, E., and Grasic, S. 2011. Probabilistic routing protocol for intermittently connected networks. *Internet Draft*.
13. Spyropoulos, T., Psounis, K., and Raghavendra, C. S. 2005. Spray and wait: An efficient routing scheme for intermittently connected mobile networks. *In Proc. ACM SIGCOMM Workshop on Delay Tolerant Networking* (WDTN).
14. Burgess, J., Gallagher B., and Levine, B.N., 2006. MaxProp: Routing for Vehicle-Based Disruption-Tolerant Networks. In *Proceedings of IEEE Infocom*.
15. Ho, M., and Fall, K. 2004. Poster: Delay tolerant networking for sensor networks. In *The First IEEE Conference on Sensor and Ad Hoc Communications and Networks (SECON 2004) (invited)*
16. Seth, A., Kroeker, D., Zaharia, M., Guo, S., and Keshav, S. 2006. Low-cost communication for rural Internet kiosks using mechanical backhaul. In *MobiCom '06: Proceedings of the 12th annual international conference on Mobile computing and networking*, pp. 334–345, New York, NY, USA.
17. Xian, Y., Huang, C., and Cobb, J. 2010. Look-ahead routing and message scheduling in delay-tolerant networks. In *Proc. IEEE Conference on Local Computer Networks (LCN)*.
18. Burleigh, S., 2010. Contact Graph Routing. *Internet Draft*.
19. Bezirgiannidis, N., Burleigh, S., and Tsaoussidis, V. 2013. Delivery Time Estimation for Space Bundles. In IEEE Transactions on Aerospace and Electronic Systems, vol. 49, no. 3, pp.1897-1910.
20. Hyyrylainen, T., Karkkainen, T., Luo, C., Jaspertas, V., Karvo, J. and Ott, J. 2007. Opportunistic Email Distribution and Access in Challenged Heterogeneous Environments, *CHANTS Workshop*.
21. Lindgren, A. 2011. Social Networking in a Disconnected Network: fbDTN: Facebook over DTN. In *6th ACM Workshop on Challenged Networks*. pp. 69–70.
22. Zaragoza, K., Thai, N., and Christensen, T., 2011. An implementation for accessing Twitter across challenged networks. In *Proceedings of the 6th ACM Workshop on Challenged Networks (CHANTS '11)*. New York, NY, USA: ACM, pp. 71–72.
23. Demmer, M., and Fall, K. 2009. The design and implementation of a session layer for delay-tolerant networks. *Computer Communications, Vol. 32, Issue 16*, pp. 1724-1730.
24. Keranen, A., Ott, J., and Karkkainen, T. 2009. The ONE simulator for DTN protocol evaluation. In *SIMUTools '09: Proceedings of the 2nd International Conference on Simulation Tools and Techniques*. New York.