Using Neighborhood Beyond One Hop in Disruption-Tolerant Networks

Tiphaine Phe-Neau†, Marcelo Dias de Amorim†, and Vania Conan‡
†CNRS and UPMC Sorbonne Universités
{tiphaine.phe-neau, marcelo.amorim}@lip6.fr vania.conan@fr.thalesgroup.com
‡Thales Communications

Abstract—Most disruption-tolerant networking (DTN) protocols available in the literature have focused on mere contact and intercontact characteristics to make forwarding decisions. Nevertheless, there is a world behind contacts: just because one node is not in contact with some potential destination, it does not mean that this node is alone. There may be interesting end-to-end transmission opportunities through other nearby nodes. Existing protocols miss such possibilities by maintaining a simple contact-based view of the network. In this paper, we investigate how the vicinity of a node evolves through time and whether such information can be useful when routing data. We observe a clear tradeoff between routing performance and the cost for monitoring the neighborhood. Our analyses suggest that limiting a node’s neighborhood view to three or four hops is more than enough to significantly improve forwarding efficiency without incurring prohibitive overhead.

I. INTRODUCTION

As our urban society lives on, the more technologically nomadic its citizens get. During their daily commuting, people carry electronic devices like smartphones, portable game stations, or laptops. Such objects embed wireless interfaces and important storage abilities traveling with their owners. Noticing those advantages, an emerging networking paradigm adapted itself to leverage these opportunistic contacts, namely disruption-tolerant networking [1].

Disruption-tolerant networks lack overall network knowledge. DTN nodes are only aware of what they learned via encounters. To forward information, nodes try to make the most of their local knowledge. Attentive to these difficulties, researchers created ingenious forwarding techniques in DTNs. Some approaches rely on full flooding via encountered nodes like Epidemic forwarding but are quite wasteful in terms of energy and resources optimization [2]. Wiser epidemic protocols emerged as with PRoPHET or Spray-and-Wait, where nodes choose their next hops based on probabilistic likeliness of meeting the destination or through distributed flooding [3], [4].

Some other techniques choose to use the social behavior of the participants. As in a city people tend to cluster into communities around different points of interests, Ott et al. presented a protocol leveraging end-to-end and multihop DTN paths [5]. End-to-end paths occur among connected components whereas DTN ones happen between these temporary components. Sarafijanovic-Djukic et al. made a similar observation in the VANET environment [6]. Later, Heimlicher and Salamatin laid their study over the groundwork that mobile wireless networks tend to have connected crowds [7].

The inherent nature of DTN pushes us to use local information to make routing decisions, but why do we restrain our knowledge to contacts information as featured in Epidemic, PRoPHET or Spray-and-Wait? A node’s current connected component may help us take forwarding decisions. Here, we try to see beyond a node’s mere contacts and dig deeper into its immediate neighborhood.

To detect the impact of neighborhood beyond one hop in DTN protocols, we perform our analysis over the most basic DTN forwarding strategy, namely the WAIT protocol. In this protocol, the source stores the message until it meets the destination. The main criticism on the WAIT approach is that the source may wait for a quite long time before being able to deliver the message or, worst, to completely fail delivering it. In this study, we rely on our previous findings that nodes are frequently nearby even if not in direct contact and therefore, there are non negligible n-hop paths around nodes to use [8]. Localized neighborhood knowledge can be an important asset for DTN nodes. However, the main challenge in providing extended neighborhood knowledge to nodes is to find a good balance between efficiency and probing costs. In fact, we can wonder how far a node should probe its neighborhood for surrounding knowledge. The more information about the network, the better decisions we can make but the higher the costs induced.

We first identify the costs of neighborhood knowledge through simple yet realistic overheads. Then we analyze the gains that neighborhood knowledge brings in the WAIT protocol and mitigate our observations according to the aforementioned overheads. Finally, we issue an empirical scope limitation for neighborhood monitoring that brings both improved efficiency and constrained costs. Performances comparison with other DTN protocols is beyond the scope of our study and will be considered as a further research topic.

As a summary, this paper makes the following key contributions:

- We confirm the power of localized neighborhood knowledge in the design of efficient forwarding algorithms for DTNs.
- We define custom performance metrics for the WAIT protocol and quantify the performance gains brought by neighborhood knowledge.
• We show an empirical limitation of neighborhood knowledge allowing better waiting times and constrained monitoring costs.

In the remainder of this paper, we first formulate the problem and the necessary background knowledge for its definition in Sections II, III, and IV. Then, we evaluate this proposal on a number of synthetic and real-world mobility traces in Section V. We finally conclude the paper and suggest ideas for future work in Section VI.

II. PROBLEM FORMULATION

In the current DTN landscape, protocols derive their transmission decision based on their contact observations. We consider that a contact happens between two nodes whenever they are within each other’s communication range.

The performance of DTN forwarding protocols is related to two main parameters. The first is the time-to-live (TTL), which is used to bound resource utilization or network availability through time; whenever the TTL expires, messages are dropped, which guarantees that messages do not travel indefinitely in the network. The second and most important parameter is the waiting time. It is the duration a node waits before sending a message toward the destination. The lower the waiting time, the more chances we have to respect a message TTL and the better the induced delivery delay. In the case of the WAIT protocol, the waiting time only stops when the source meets the destination (no intermediate relays). Note that the two parameters are closely related. The waiting time has a particular meaning as it is related to the user experience: any user notices the duration before its message gets delivered and benefit from the inbuilt nature of DTNs, offering nodes a consistent performance.

To answer this question, we investigate the impact of the neighborhood knowledge in the WAIT protocol by focusing on the waiting parameter – the time a node waits before being able to send a message straight to the destination.

III. THE n-ARY INTERCONTACT CHARACTERIZATION

We rely on the DTN characterization based on contact and n-ary intercontact to discriminate a node’s connected component [8]. For each node, any neighbor staying at a 1-hop distance is considered to be in contact. In our definition, we say that other nodes are in n-ary intercontact. This intercontact characterization holds two main notions:

Definition 1. Favorable intercontact. Two nodes are considered in “favorable” intercontact with parameter n when there is a contemporaneous shortest path of length n ∈ [2; ∞] separating the two nodes under consideration. These nodes belong to the same connected component.

Note that the parameter n helps represent the limit threshold we want to observe in this study.

Definition 2. Pathless intercontact. In opposition to favorable situations, “pathless” intercontact indicates the lack of end-to-end paths between the two nodes, i.e., n = ∞.

For further details on the n-ary intercontact characterization, please refer to [8]. Using this intercontact characterization, we introduce a threshold T between the favorable and pathless intercontacts to limit the knowledge a node will have about its neighborhood. All nodes in favorable intercontact of parameter n ≤ T are in the node connected component. Otherwise, they are considered outside the node T-neighborhood. The additional favorable intercontact knowledge allows nodes to
know members of their $T$-neighborhood, i.e., all nodes to which they have an end-to-end path of length $l \leq T$. By choosing $T = 1$, we define that a node only knows its contacts. With $T = 2$, it knows all its contacts with his 2-hop neighbors and so on. Note that to be sure to get complete knowledge of the connected component, we must set $T = \infty$. In Fig. 1 we show a node’s knowledge depending on the $T$ threshold.

Now, whenever node $A$ wants to send a message to node $B$, $A$ scans its neighborhood up to $T$ hops. If $B$ arrives within $A$’s $T$-neighborhood, $A$ sends the message to $B$ via a multihop contemporaneous path. To analyze raw performance of waiting times between the mere WAIT protocol and the WAIT protocol with neighborhood awareness, we do not consider any specific message sizes or throughputs that would impact delivery times. We focus on waiting times and on additional data impacting the control overhead.

IV. COST ANALYSIS

To take into account the costs of multihop messaging and neighborhood monitoring, we identified two main sources of overhead. We use the message as the unit of comparison.

1) Data Overhead ($D_o$): represents the total cost to deliver a message. Clearly, any protocol with extended neighborhood knowledge is costlier than its simple version. Whenever the source switches to multihop transmission mode, the message follows a contemporaneous end-to-end path to the destination and has to sustain several store and forward processes. The “extra” cost of such a communication, in terms of additional messages sent, is the number of hops between $A$ and $B$ minus one:

$$D_o = n - 1.$$  

2) Neighborhood Knowledge Overhead ($N_o$): represents the overhead to gather information about the neighborhood. To get all nodes $T$-neighborhood, the basic approach consists in sending epidemic probes with an upper threshold of $T$. Node $A$ broadcasts a discovery message (DM) to its contacts with a TTL set to $T$. All nodes who received the DM rebroadcast this message with a TTL set to $T - 1$, and so on. We assume that each transmission is acknowledged (see Fig. 2 for a detailed example). This leads to a cost of:

$$N_o = 2 \times \text{card}(\text{CC of size } < T) + \text{card}(\text{CC of size } = T) + 1,$$  

where “card” stands for cardinality and “CC” for connected component. $N_o$ does not depend on the path length that DMs have to cross. With little aggregation, $N_o$ only depends on the number of neighbors in a node’s connected component. $N_o$ is responsible for most of the overhead in our analysis as it consists in frequent neighborhood monitoring.

V. EVALUATION

To quantify the performance gains enabled by neighborhood awareness, we simulate the WAIT protocol with and without $T$-neighborhood ($T \geq 2$) knowledge using real-world mobility data and the ONE simulator [9].

A. Datasets

To grasp our issue in different situations, we confront both scenarios to two synthetic mobility models and two realistic datasets. The synthetic models are:

- **RandomTrip** is a well-known mobility model making up known issues of Random Waypoint [10]. We simulated 20 nodes during 9 hours with speeds ranging from 0 to 7 m/s on a 50x60 m$^2$ surface. This scenarios intends to emulate a working day structure.
- **Community** is a mobility model founded on social network theory [11]. This model emphasizes the human tendency to aggregate in a societal way based on relationships. We simulated 50 nodes during 9 hours on a 1,500 x 2,500 m$^2$ surface. This dataset denotes an urban-wise environment.

We also consider two real-world datasets probing contacts between nodes via motes. Motes log the presence of other motes within a 10-meter wireless range.

- **Infocom05** retraces contacts between 41 attendees during a 5-day conference [12]. We focus on a 12-hour period of the second day which presents the highest network activity. Each device performs a scan every 120 seconds.
- **Rollernet** occurs during a 3-hour rollerblading tour in Paris [13]. This trace has a finer beaconing granularity of 15 seconds and represents a highly dynamic sport event.

We investigate in the following how much we can reduce the waiting times using neighborhood awareness.

B. Delay observations

For each mobility trace and each pair of nodes, we randomly generated 10 messages at different time instants. We chose to generate sparse messages for waiting times to better reflect the impact of neighborhood monitoring. The most symptomatic situation arises when a pair of nodes never come in contact, but once and a while they belong to the same connected component. In this situation, the WAIT protocol drops the message whereas the neighborhood-aware variant can manage to forward it correctly.

As scarce as this situation may sound, it happens for 10% of pair of nodes in Infocom05, 53% in Community, and around 55% of Rollernet nodes. If these nodes try to send a message using the WAIT protocol, they will fail. These
fractions of nodes have infinite waiting delays when \( \text{WAIT} \) is in use. Otherwise, with the neighborhood-friendly version, they manage to deliver messages with bounded waiting times.

For these nodes with bounded waiting delays, we analyze to which extent neighborhood knowledge helps lower their waiting times. In Fig. 3, we show the averaged pairwise waiting times for each dataset. Each bar represents the average waiting delay we obtain with \( T \)-neighborhood probing. For every dataset, between the first and second bars, we notice significant reduction in the waiting times: 40% in Infocom05 and Rollernet, 57% in RandomTrip, and around 80% in Community. We observe that, although we keep reducing the waiting delays, the gains for \( T > 4 \) are much smaller. This corroborates our first feeling that localized knowledge should be enough and suggests that we can, in practice, keep \( T \) small (two or three).

On the one hand, when nodes have no contact opportunities and therefore an infinite waiting delay, neighborhood monitoring can bind waiting times and enable forwarding possibilities. On the other hand, when nodes do have contact opportunities, neighborhood monitoring still helps in lowering waiting delays (up to a factor of four in our scenarios). \( T \)-neighborhood monitoring clearly helps reducing waiting delays; but it also ignites some costs, which we investigate in the next subsection.

### C. Overhead study

Supporting neighborhood knowledge monitoring does not come for free. Any node needs to probe its neighborhood and create a flow of messages around.

1. **Impact of neighborhood knowledge overhead:** There are many strategies for connected component gathering, from link state-like solutions to flooding techniques. For our study, we chose to compare two naive behaviors:

   - Nodes monitor their \( T \)-neighborhood when they have a message to send and stop when the message TTL expires (called \( \text{TS} \) hereafter).

   - Nodes monitor their \( T \)-neighborhood when they have a message to send and stop when the message TTL expires (called \( \text{TS} \) hereafter).

   In Fig. 4, we represent \( N_o \) during the whole experiment duration for a node in the Infocom05 with \( \text{CS} \) probing every 30 seconds. Each curve represents the behavior for each threshold \( T \). There are major differences between thresholds. Monitoring only contacts induces fewer overhead than any deeper neighborhood monitoring. Beyond \( T = 4 \), there are no noticeable differences for \( N_o \). Overall behaviors are quite alike and depend on the surrounding density.

In Fig. 5, we plot \( N_o \) of the same source node as before. This time, we use the \( \text{TS} \) method for neighborhood analysis. The reason we have noticeable jumps in all curves is, when the destination comes into the source’s \( T \)-neighborhood, this latter stops monitoring its surroundings. Contact monitoring drops all but one message and is only plotted for the reader’s information. An interesting result is how, for the same number of delivered messages (7 messages), probing the 3-neighborhood and beyond gives better results than probing only the 2-neighborhood in terms of \( N_o \). The reason is that the faster the source finds the destination, the shorter the waiting delay and the lower the \( N_o \).

For the tradeoff analysis on lowering waiting delays and overheads costs, we will consider \( \text{TS} \) as the neighborhood monitoring strategy. For \( \text{WAIT} \), a constant monitoring \( \text{CS} \) is a wasteful strategy in terms of messages or energy use.

2. **Impact of data overhead:** \( N_o \) seems to be the most expensive in terms of messages sent, yet, we also have to take into account the \( D_o \): the amount of messages generated over an end-to-end path transmission. \( D_o \) adds an insignificant amount of messages to \( N_o \). It is important to underline that having a large \( D_o \) (i.e., a long path between the sender and the destination) can lead to undelivered messages. This is why one would prefer shorter \( T \).
its contacts without being able to deliver any messages. An interesting threshold $T$ for the whole dataset duration. We understand that above a certain constrain message overheads and deliver a higher message rate with time. With a simple probing technique, we manage to reduce waiting delays. In Fig. 3, we observe that a larger $T$ leads to lower waiting delays. Yet, neighborhood monitoring is an expensive process. In Fig. 5, we see how $N_c$ evolves with time. With a simple probing technique, we manage to constrain message overheads and deliver a higher message rate than with the WAIT protocol.

For each node, we analyze the average number of neighbors in their $T$-neighborhood. TABLE I shows this value for the whole dataset duration. We understand that above a certain threshold $T_t$, a node’s $T$-neighborhood does not expend much (except for the RandomTrip dataset, which has a random movement pattern and a high density). In Community or Infocom05, a node’s $T$-neighborhood does not grow significantly anymore above $T_t = 4$.

**Arguments.** People do move around with their devices. Each of them may recompute its neighborhood every once and a while after a journey. With the whole network movement, each node is more likely to find its destination even with a shorter threshold. We have seen how observing a node’s 3-neighborhood is interesting enough to lower the waiting delays and constrain messages overheads. Monitoring 3-neighborhood can even reduce overheads by lowering such a delay compared to sensing 2-neighborhood. Moreover, lower $T$ thresholds enable lower $D_o$ and promote quicker message forwarding.

**VI. SUMMARY AND OUTLOOK**

In this paper, we examine the impact of neighborhood awareness on the waiting time in a variation of the simple WAIT protocol. Most DTN techniques only focus on sensing direct contacts and do not inquire about their nearby neighbors. This strategy is the most straightforward and rational. However, we cannot deny the sociological nature of DTNs. People do not wander randomly in a city. They gather around specific persons or locations. To our opinion, ignoring a node’s immediate neighborhood results in a loss of useful information.

Our findings show that neighborhood probing significantly improves the WAIT protocol in terms of waiting delays. When delays used to be infinite, they are now bounded. When delays were high, they are now lowered by a factor up to 4 in our scenarios. Yet, neighborhood monitoring ignites messaging overhead. But, by limiting a node’s neighborhood vision to a threshold $T$ of three or four, we constrain costs and still enhance performances. According to our results on the raw WAIT protocol, we encourage our community to consider potential performance gains that neighborhood knowledge could bring to more elaborated DTN schemes.

**ACKNOWLEDGMENT**

This work is partially supported by the ANR project CROWD under contract ANR-08-VERS-006.

**REFERENCES**

[1] K. Fall, “A delay-tolerant network architecture for challenged internets,” in ACM SIGCOMM, 2003.

[2] A. Vahdat and D. Becker, “Epidemic routing for partially connected ad hoc networks,” Duke University, Tech. Rep., 2000.

[3] A. Lindgren, A. Doria, and O. Schelén, “Probabilistic routing in intermittently connected networks,” SIGMOBILE Mob. Comput. Commun. Rev., vol. 7, July 2003.

[4] T. Spyropoulos, K. Psounis, and C. S. Raghavendra, “Spray and wait: an efficient routing scheme for intermittently connected mobile networks,” in ACM SIGCOMM workshop on Delay-tolerant networking, 2005.

[5] J. Ott, D. Kutscher, and C. Dwertmann, “Integrating dt and monet routing,” in ACM SIGCOMM workshop on Challenged Networks, 2006.

[6] N. Sarafjanovic-Djukic, M. Pidrkowski, and M. Grossglauser, “Island hopping: Efficient mobility-assisted forwarding in partitioned networks,” in IEEE SECON, 2006.

[7] S. Heimlicher and K. Salamatian, “Globs in the primordial soup: the emergence of connected crowds in mobile wireless networks,” in ACM MOBIHOC, 2010.

[8] T. Phe-Neau, M. Dias de Amorim, and V. Condon, “Fine-grained inter-contact characterization in disruption-tolerant networks,” in IEEE ISCC, 2011.

[9] A. Keränen, J. Ott, and T. Kärkkäinen, “The one simulator for dt protocol evaluation,” in SIMUTOOLS, 2009.

[10] S. PalChaudhuri, J.-Y. Le Boudec, and M. Vojnovic, “Perfect simulations for random trip mobility models,” in IEEE INFOCOM, 2005.

[11] M. Musolesi and C. Mascolo, “Designing mobility models based on social network theory,” SIGMOBILE Mob. Comput. Commun. Rev., vol. 11, pp. 59–70, July 2007.

[12] A. Chaintreau, P. Hui, J. Crowcroft, C. Diot, R. Gass, and J. Scott, “Impact of human mobility on opportunistic forwarding algorithms,” TMIC, vol. 6, no. 6, pp. 606–620, 2007.

[13] P. U. Tournoux, J. Leguay, F. Benbadis, V. Conan, M. Dias de Amorim, and J. Whitbeck, “The accordion phenomenon: Analysis, characterization, and impact on dt routing,” in IEEE INFOCOM, 2009.