A distributed algorithm for path maintaining in dynamic networks

F. El Ali(1)(2), B. Ducourthial(1)(2)
(1) Université de Technologie de Compiègne
(2) CNRS Heudiasyc UMR6599, BP 20529, 60205 Compigne Cedex, France
(corresponding author: Bertrand.Ducourthial@utc.fr)

***Abstract*— Routing becomes obsolete in ad hoc dynamic networks. Path maintaining becomes necessary in order to maintain a communication that already started between two entities. In this paper we present a path maintaining algorithm.**

***Index Terms*— Ad hoc, dynamic, path maintaining**

I. Introduction and related works

Ad hoc dynamic networks represent a challenging domain of application due to their specifications. Still we find many applications on these networks nowadays, like the vehicular networks, drone networks or even pedestrian networks. The general method to study these networks starts by studying their dynamics. The dynamic changes are closely related to the frequency of neighborhood changes, and the functioning and requirements of the protocols.

Unicast communications in these networks require a continuous communication between the source of the messages and the destination even if they are separated by multiple hops. Many routing protocols exist for mobile networks, like AODV, OLSR and geographic routing among others. But these protocols are not efficient if the dynamic of the network increases to exceed the admitted dynamicity. Not to forget that these protocols require broadcast in general, location services [5] [4] or infrastructure bases, which make them inadequate to highly mobile networks. We should also mention that the need for global knowledge in these protocols increases the control over the network, in order to build or to update the structures needed to maintain routes.

II. Contribution

Due to their specificities, dynamic networks do not require routing as those used in networks with fixed topologies or low dynamics. In fact, in mobile networks, the search for a distant destination might be obsolete since the nodes are frequently moving, meaning that the destination might have changed its location by the time the source receives its response. The search for the destination implies the broadcast of messages in order to identify its position. And in case some geographic protocols admit the knowledge of the position of the destination, this means that a location service has already identified this position using broadcasted messages. As explained in [2], the need for unicast communication is in order to maintain a communication that started when the source and the destination were neighbors, and that is needed to pursued for a certain duration. This problem is called path maintaining. This said, the aim of the path maintaing application is to maintain a communication initiated with a path of length 1 and to avoid the flooding of the network with unnecessary messages.

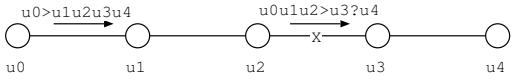

Fig. 1: Local periodic messages

In this paper, we describe the path maintaining algorithm, that we denote PTH. PTH is a path maintaining algorithm that relies on local adjustments of the path. Local adjustment is used in order to avoid having a global knowledge of the network, which implies to flood the network with too much control messages in order to build a global knowledge that might become unusable because of the dynamic. PTH adjusts the path between the source and the destination using neighbor nodes of the broken links in the path. Starting from a direct communication between the source and the destination, PTH interferes when these two entities move apart, and the direct communication between them becomes impossible. At this point, the neighbor nodes of the broken link are used to relay the messages. The same mechanism is used for all the broken links of the path between the source and the destination. PTH relies on three mechanisms: local periodic update for the nodes of the path only, path reduction and path extension. These mechanisms are detailed in the next section.

III. The PTH algorithm

In this section, we describe our algorithm named PTH for solving the path maintaining problem. It is mainly based on three mechanisms, that we describe in this section: local periodic update, path extension and path reduction.

The first mechanism – **local periodic update** – allows to deal with neighborhood changes and failures. It relies on periodic diffusion of messages in the neighborhood of the sender. These messages contain some local information regarding the path. Such information is composed by the list of members of the path (eg. $\mathtt{u}_0\mathtt{u}_1\mathtt{u}_2\mathtt{u}_3\ \ldots$) (see Figure 1), some flags allowing to determine the sender and the willing receiver of the current message ($\mathtt{u}_0\mathtt{u}_1\mathtt{>}\mathtt{u}_2\mathtt{u}_3$ if u_1 is the sender and u_2 the receiver) and an uncertainty flag regarding a member of the path ($\mathtt{u}_0\mathtt{u}_1\mathtt{>}\mathtt{u}_2\mathtt{?}\mathtt{u}_3$ (see Figure 1) if u_1 has a doubt on its successor u_2). Such a message informs both the predecessor and the successor of the local state of the

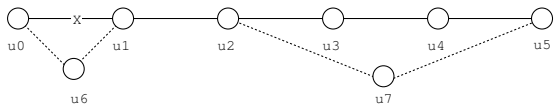

Fig. 2: Extension of the path through u6 and reduction through u7

path, which in turns can then update their own information. Other nodes not belonging to the path may propose to belong to, in order to repair or shorten it.

The second mechanism – **path extension** – allows to deal with link breaking. This can appear when two neighbors go far each other (Consider Figure 2, with $\mathtt{u}_0 = \mathtt{u}_i$, $\mathtt{u}_1 = \mathtt{u}_j$ and $\mathtt{u}_6 = \mathtt{v}$ for instance). Consider that the link $[u_i, u_j]$ in the path breaks. As the node u_i sends periodically some messages containing $\mathtt{u}_i{>}\mathtt{u}_j$, it expects implicit acknowledgment of u_j, that is messages without uncertainty, containing $\mathtt{u}_i\mathtt{u}_j{>}$. This means that $\mathtt{u}_j$ receives the message of $\mathtt{u}_i$ and forwards it with the path $\mathtt{u}_i\mathtt{u}_j{>}$. If it does not receive such messages from u_j, it sets the uncertainty flag on its successor u_j in its next messages: $\mathtt{u}_i{>}\mathtt{u}_j?$. Either u_j receives messages of u_i with an uncertainty flag on itself (communication $u_i \leftarrow u_j$ broken) or it does not receive at all messages from u_i (communication $u_i \rightarrow u_j$ broken). In both cases, it will set the uncertainty flag on its predecessor u_i in its messages; they will contain: $\mathtt{u}_i?\mathtt{u}_j{>}$. Such uncertainty flags allows to neighbors of both u_i and u_j to propose repairing the path. If a neighbor node v notices such exchanges, it counts the received messages from u_i and u_j. When a given threshold is reached, v proposes to be relay of the communication by sending $\mathtt{u}_i?\mathtt{v}{>}\mathtt{u}_j?$. As several neighbors of both u_i and u_j may propose to be a relay, u_i (closer from the source) chooses the first (say v) and sends $\mathtt{u}_i{>}\mathtt{v}\mathtt{u}_j?$. Then v sends $\mathtt{u}_i\mathtt{v}{>}\mathtt{u}_j?$. Then u_j sends $\mathtt{u}_i\mathtt{v}\mathtt{u}_j{>}$ and the path is repaired (to the condition that a node v was present in the neighborhood of the breaking edge extremities).

The third mechanism – **path reduction** – consists in reducing the path when possible. This can be done by a node in the path, or neighbor of the path. Consider the case when the reduction between two given nodes u_i and u_j (where u_i precedes u_j in the path) can be done by a node v, neighbor of the path (Consider Figure 2, with $\mathtt{u}_2 = \mathtt{u}_i$, $\mathtt{u}_5 = \mathtt{u}_j$ and $\mathtt{u}_7 = \mathtt{v}$ for instance) . This node v, by receiving the messages sent by u_i and u_j simultaneously, observes that it can reduce the path by being a relay between u_i and u_j.

The proposition of the reduction is done by placing in the path the shortcut denoted by $\mathtt{u}_i\text{-}?\mathtt{v}\text{-}?\mathtt{u}_j$ (with the uncertainty flag on the links $[u_i, v]$ and $[v, u_j]$). The message sent by v is of the form: $\ldots \mathtt{u}_i\text{-}?\mathtt{v}\text{-}?\mathtt{u}_j \ldots \mathtt{v}{>}\mathtt{u}_j \ldots$. When u_j receives this message, it confirms to v the possibility of reduction, removing the uncertainty on the reduction link $[v, u_j]$. The node v then asks the permission of u_i to be a relay. At this point, u_i sends a validation message $\ldots \mathtt{u}_i\text{-}\mathtt{v}\text{-}\mathtt{u}_j{>} \ldots \mathtt{u}_j \ldots$ to its successor in the path; this message will be relayed until reaching u_j.

Indeed, as several overlapping reductions may occur simultaneously, the reduction $u_i - v - u_j$ has to be validated by all nodes from u_i to u_j in the path. This allows to avoid any split of the path into several smaller disconnected paths. If a node is already engaged with another reduction, it will not forward the validation message containing the reduction. In the converse case, the node forwards the message and becomes engaged. By this way, in case of two reductions in conflict for a common sub-path, the validated reduction is the one that first sent a validation message into the common sub-path. When u_j receives the validation message, it confirms the reduction by relaying from now messages of v instead of messages of its old predecessor u_{j-1}. An engaged node will either leave the path or receive a message without reduction from its predecessor; it stops then being engaged. The same process is used when the reduction is proposed by nodes u_i belonging to the path instead of a neighbor node.

IV. PTH VALIDATION AND PERFORMANCES

Whatever are the performance of a given algorithm, the dynamic of the network can lead to failure because it could compromise any communication. Our protocol is able to run even in highly dynamic network but the following requirement is necessary: a neighbor of the path is present during the path extension. This operation is very short (few local messages). By the way, we have a contract between the dynamic and the specifications of our protocol. This has been formalized in [3] under the term of best effort algorithm. A best effort algorithm does its best regarding the dynamic while still ensuring some useful properties for those (users or other protocols) relying on it. More precisely, whenever a *topological property* is fulfilled, a *continuity property* is ensured. Here, the topological property is the presence of a neighbor node when two nodes belonging to the path move far each other. The continuity property is the existence of the path from source to destination, the two initiators which were in the same neighborhoods at the beginning. Proof is omitted by lack of place.

PTH has been compared to a basic geographic routing with a location service, and to a traditional broadcast in the same scenario under the Airplug emulator [1] (with 31 cars moving from the UTC laboratory to the train station in Compiègne), and starting from the same initial state. Consider now N the number of messages sent by the source node in the network. While the broadcast flooded the network with 8.4×N messages, the geographic routing with its location service sent to the network 5×N messages, and PTH sent 2.4×N messages. Not to mention that when 31 cars were involved in sending messages for the broadcast or geographic routing, only 6 were involved when running the PTH algorithm.

REFERENCES

[1] A. Buisset, B. Ducourthial, F. El Ali, and S. Khalfallah. Vehicular networks emulation. Proceedings of ICCCN 2010, 2010.

[2] B. Ducourthial and Y. Khaled. *Vehicular Networks: Techniques, Standards and Applications*, chapter Routing in Vehicular Networks: User Perspective. CRC Press (Taylor & Francis Group), 2009.

[3] Bertrand Ducourthial, Sofiane Khalfallah, and Franck Petit. Best-effort group service in dynamic networks. In *SPAA 2010, June 13-15, 2010, Thira, Santorini, Greece*, 2010.

[4] Michael Kasemann, Holger Fuler, Hannes Hartenstein, and Martin Mauve. A reactive location service for mobile ad hoc networks. 2002.

[5] Hanan Saleet, Otman Basir Rami Langar and, , and Raouf Boutaba. Proposal and analysis of region-based location service management protocol for vanets. *IEEE "GLOBECOM" 2008 proceedings*, 2008.