A Light Protocol for Distance Estimation in Bluetooth Mobile Ad-Hoc Networks

F. Gil-Castiñeira and F. J. González-Castaño

Departamento de Ingeniería Telemática, Universidad de Vigo, ETSI Telecomunicación, Campus, 36200 Vigo, Spain
{javier,xil}@det.uvigo.es

Abstract. According to the results in this paper, Bluetooth MANET route hop count depends linearly on the distance between origin and destination, and therefore hop count may be a valid metric for MANET guidance services. However, scatternet-based or on-demand route formation algorithms for ideal MANETs are not well suited to Bluetooth technology, since route lifespan is too short even in case of moderate user walking speeds. As a consequence, we propose a feasible light protocol to estimate route lengths, based on Bluetooth inquiry/inquiry scan states. This protocol works properly for walking speeds, and it can be used to find persons in large spaces.

1 Introduction

1.1 Motivation

Nowadays, many commercial terminals have Bluetooth modems embedded. As a consequence, Bluetooth has been considered a suitable MANET-supporting technology [**?**]. However, as we will see in section 2, scatternet-based or on-demand route formation algorithms for ideal MANETs are not well suited to Bluetooth technology, since route lifespan is too short even in case of moderate user walking speeds.

This paper focuses on the problem of estimating distances between MANET terminals moving at walking speeds ($\sim$ 0.8 meters per second) as a function of route hop count. Our results suggest that Bluetooth MANET route hop count is a linear function of distance, and therefore hop count may be a valid metric for MANET guidance services. We propose a feasible light protocol to estimate route lengths, based on Bluetooth inquiry/inquiry scan states. This protocol works properly for walking speeds, and it can be used to find persons in large spaces.

2 Initial approaches

All approaches in this paper are based on the same idea: a requesting device sends a "search request" with the Bluetooth address of the target device. Every node in

the scenario must participate as an intermediate node if required, depending on ongoing requests. An underlying protocol routes the request to the target device, which generates a response. Once the response arrives to the origin (requesting device), it is possible to estimate the distance to the destination (target device) from the number of route hops. The destination can send multiple responses, so that the origin can find the destination by following those directions that lead to a hop decrease.

2.1 Scatternet-oriented approach

The first approach consists of creating an scatternet comprising as many nodes in the scenario as possible and implementing a routing protocol over it. In a mobile scenario, a suitable routing choice is the well-known DSR algorithm [**?**]. **Simulation results**: In order to test this approach, we modified Blueware [**?**] to support a modified DSR version. Blueware is a Bluetooth simulator designed to test the TSF scatternet formation algorithm [**?**]. In our experiment, DSR was used to create routes to estimate distances on top of a TSF scatternet. Simulation results indicate that this approach is not valid in mobile environments, because the scatternet gets easily unconnected due to user mobility. We conclude that the scatternet-oriented approach is only valid in case of static or extremely slow nodes.

2.2 On-demand approach

The second approach does not assume an underlying topology of Bluetooth connections. When a node issues a distance estimation request, it initiates a route generation process that creates on-demand Bluetooth connections along the route as needed.

Assuming that all nodes scan their surroundings to find their neighbors, the normal operation of the system is as follows:

- *Source node*:
  - On user demand, it builds a *search packet* containing the target address and a zero-hop value.
  - Sets temporal connections with its neighbors and sends the search packet to them.
- *Intermediate node search packet procedure*:
  - Gets incoming search packets and decides to delete (hop limit reached) or forward them.
  - In the latter case, it increases the hop counter and attaches its address to the search packet.
  - Sets temporal connections with its neighbors and sends the search packet to them (but to those already belonging to the incoming route).
- *Target node*:
  - Gets incoming search packets and checks if previous copies have already arrived.

    - If not, it reads the route in the search packet and builds a *response packet*.
    - Sets up a connection with the last node in the route and sends the response packet.
    - Resends the response packet a number of times. This is *i*) to increase the probability of a distance estimation packet reaching the source node and *ii*) to let the source node track target location changes for a while.
- *Intermediate node response packet procedure*:
    - Gets the response packet and reads the next route hop.
    - Sets up a connection with the corresponding route node and passes the response packet to it.
    - If the connection is not possible, the intermediate node broadcasts the response packet to its neighbors.

Simulation results: We implemented a test-bed based on Bluehoc_ex [?] to evaluate the on-demand approach. We considered a scenario with 50 nodes in a 50×50 m^2 room, with initial random uniform positioning. Once the simulation starts, the nodes move along straight lines at constant speeds (with random direction changes). We observed the following:

- The variation of the distance between source and target and the variation of the number of hops in between have the same sign in most cases. For a given source and destination separated by more than 30 m, we generated a bundle of 32 source displacements (angular increments of 45^0 and lengths in $\{5, 10, 15, 20\}$ m). The signs of both variations (distance and hops) did not differ in 83% of the cases.
- We observed that the first packet received by the target followed a near-shortest-path route. Figure 1 shows average number of hops versus distance in meters. Note that the number of hops grows linearly with distance after a 5-m initial value.
- The density of nodes does not influence the relationship in Figure 1 significantly for more than $\sim$20 Bluetooth class 2 nodes. To arrive to this conclusion, we performed a series of simulations incrementing the number of nodes in (0,200] in 5-node steps. In all cases, initial node location was random and uniformly distributed. It could be expected that the number of hops should grow with the density of nodes. However, Bluetooth nodes do not set connections depending on signal strength (as IEEE 802.11 does) and, consequently, the average number of hops in the shortest routes is similar regardless of the number of nodes, once a certain density threshold is reached.

Nevertheless, we found two important limitations of the on-demand approach:

- A long delay is due to the lack of synchronization among nodes. If the distance between source and target is $\sim$10 hops, the time to receive a response ranges from 40 to 150 seconds. Although the distance covered by 10 hops can be up to 100 meters with class 2 modems, this response time is unacceptable compared to context change rates due to walking speed.

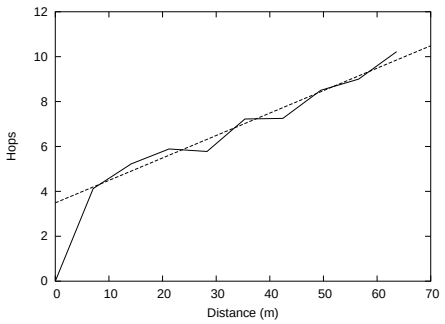

Fig. 1. Number of hops vs. distance

– For moderate to high walking speeds (i.e above 0.5 m/s) the response routes always fail. This compromises the feasibility of the system described in [?] (a system to route data in an ad-hoc Bluetooth scatternet) in mobile environments.

3 Inquiry-oriented approach

The long delay in the on-demand approach is due to state transitions and queue managing overload. In ideal conditions, hop delay would be the time to set a connection (1.804 seconds according to [?]). In other words, 10-hop route establishment would be $\sim$ 36 seconds (the minimum time observed in our realistic simulations).

A Bluetooth device looking for neighbors sends `BT_ID` inquiry packets [?]. All devices in inquiry scan state within emitter range and listening in the same frequency will receive those packets, and they will answer with a `FHS` packet [?]. A `BT_ID` packet does not contain information on the source. It carries a 64-bit *inquiry access code* to indicate the kind of device that must answer (there is a generic code (GIAC) and a group of specific codes (DIAC). The remaining codes are reserved for future use).

3.1 Distance Estimation Protocol

In order to overcome the limitations of Bluetooth technology, we do not establish routes. Our distance estimation protocol, based on the inquiry procedure, is described next. Its packets carry a *target address* (48 bits), a *source address* (48 bits) and a *hop counter*.

The source node simply submits a search packet to its neighbors with hop counter set to 0. Any intermediate node receiving a search packet increases the hop counter and resubmits the packet, unless a hop limit is reached (in that case, the intermediate node discards the packet). Eventually, if there are no isolated regions due to node ranges, the search packet reaches the target. The target sets

the hop counter to 0 and starts the process again with another search packet swapping previous source and target addresses. When the new search packet arrives to the source node, the source node recognizes its address and obtains an estimation of the distance to the target.

Although this protocol seems extremely simple, its implementation is not obvious: evidently, it can be programmed with `DM` packets at application level [?]. However, `DM` packets require an existing connection, and we have seen in section 2 that connection setting times are unacceptable.

Alternatively, we reserve a `BT_ID` inquiry code for a "distance estimation inquiry". Every participating node must alternate inquiry and inquiry scan states. While in inquiry state, the nodes send distance estimation inquiries. All devices nearby in inquiry scan state with a pending transmission (initial source transmission, target transmission or intermediate node re-transmission) must answer with a "distance estimation `FHS` packet" - the search packet. `FHS` packets have a 144-bit payload, enough for source address, target address, hop counter, sequence number and control information (hop limit, etc.). Figure 2 shows the format of the `FHS` search packet.

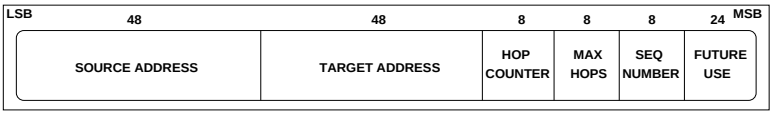

Fig. 2. Distance estimation `FHS` packet

3.2 Simulation results

We implemented the inquiry-oriented approach as a modification of Bluehoc_ex. The simulations indicate that source-to-target transmission delay is basically the aggregation of the inquiry times of all nodes involved. We must remind the reader that the minimum inquiry time is 1.25 ms, its recommended maximum value is 10.24 s and its average value in our scenario is 3-5 s. Figure 3 shows elapsed times to obtain a distance estimation. Compared with the results in section 2.2, the delay of the inquiry-oriented approach is one order of magnitude lower.

4 Conclusions

This paper has studied the problem of estimating distances in Bluetooth MANETs as a function of route hop counts. In our scenarios, route hop count is a linear function of distance, and therefore it may be a valid metric for guidance services.

Due to technological reasons, scatternet-based or on-demand route formation algorithms for ideal MANETs are not well suited to Bluetooth technology, since route lifespan is too short even in case of moderate user walking speeds. As a

Fig. 3. Inquiry-oriented approach elapsed times

solution, we have proposed a light protocol for distance estimation requiring a minimum modification of the inquiry procedure. It can be easily implemented by modifying the baseband firmware of Bluetooth devices. Simulation results indicate that distance estimation delay is acceptable for walking speeds (i.e. $\sim$15 s for $\sim$100 m). Forthcoming work will study the impact of Bluetooth 1.2 on distance estimation protocol performance.