Adaptive REACT protocol for Emergency Applications in Vehicular Networks

Erwin Van de Velde[1] and Chris Blondia
PATS Research Group
Dept. Mathematics and Computer Science
University of Antwerp - IBBT VZW
Middelheimlaan 1,
B-2020 Antwerpen, Belgium
{firstname.lastname}@ua.ac.be

***Abstract*—Vehicular communication is gaining more and more interest from big consortia and companies in the car industry. Wireless communication in a vehicular environment creates unique opportunities but poses also its own challenges. Therefore we proposed in previous work a new routing protocol, REACT, which uses geographical information and is able to react to a fast changing environment. In this paper, we will look at a few optimizations for this protocol in order to make it more adaptive to the current network conditions. As vehicles move very fast, link breaks occur more often. We attempt to predict this behavior, trying to use a link as long as possible but not longer than it really exists. Another important part of the REACT protocol consists of beaconing as means of exchanging position information with neighbors. The interval between two consecutive beacons is an important parameter in determining the end-to-end delay and we propose an algorithm that adapts this parameter in order to speed up neighbor discovery without generating too much beaconing overhead. The two optimizations were implemented and tested in a simulator and the results show a performance amelioration in the end-to-end delay in scenarios with sparse and dense road traffic.**

I. Introduction

Wireless communication is available in many aspects of our society and people depend more on it every day. Besides the infrastructured networks, there are also the Mobile Ad Hoc Networks (MANET [11]), which have been around for several years now, facilitating a lot of new applications in networks of different scales. Some of these networks consist of mobile computers like laptops and serve as a LAN without infrastructure, but there are also specialized Ad Hoc networks like Body Area Networks (BAN), which are used in e.g. medical applications and are constrained to one person's body. Another group are the Vehicular Ad Hoc Networks (VANETs), which enable a whole new range of exciting applications, ranging from traffic safety applications to infotainment. Communication in this kind of network is challenging however and new protocols have to be developed in order to make these applications work. They form the topic of this paper, in which optimizations for the REACT [5] routing protocol for vehicular communication will be discussed. Several projects and consortia are also working on this topic: e.g. Fleetnet [7], DSRC [6], Car 2 Car Communication Consortium [3].

[1]Funded by grant of the Fund for Scientific Research Flanders (Aspirant)

First of all, it is important to define the type of applications for which the network will be used: emergency applications. An example of this type of applications is a service that alerts other vehicles when a vehicle is involved in an accident or has to break abruptly. Vehicles driving towards the place of the incident would get this information and could react appropriately by automatically slowing down, for example, or with a driver notification. Another application could be the transmission of an alert of an approaching emergency vehicle in order to warn cars by other means than siren and turning light and to provide the emergency vehicle with a path of all green traffic lights. This type of applications has a set of distinct demands:

- The information that should be transmitted consists of a single data packet containing the message.
- The packet should always reach its destination, even when network conditions are bad: e.g. sparse topology, i.e. too few cars to always maintain full network connectivity.
- The delay should be limited.

Another issue that influences the design of the routing protocol is the type of network: a vehicular network is a mobile ad hoc network which differs from other ad hoc networks in several aspects:

- Nodes in the network move fast, up to 250km/h relative speed difference on highways.
- Nodes are geographically constrained as vehicles are limited to driving on the roads. Only at crossroads vehicles can leave a certain path.
- The network is large-scale: it can cover a whole city or even larger areas.
- The network is almost surely partitioned: the further vehicles are located from each other, the more likely it is that there is not always end-to-end connectivity.
- Energy constraints are not as severe as in other ad hoc networks since the vehicle is able to supply virtually unlimited power while the engine is running.

Considering these properties of both the traffic and the network, the challenge is clear: to forward packets in a way that fulfills the demands of the applications while taking the network properties into account.

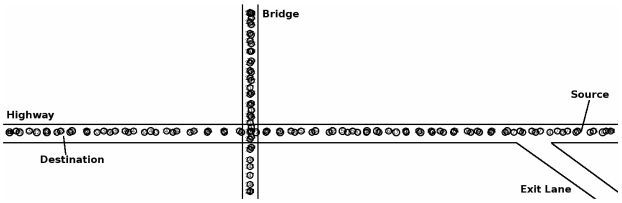

Fig. 1. Scenario Description

Topology based routing protocols, whether they keep a view of the entire network (proactive routing protocols, e.g. OLSR [4]) or of the routes they use (reactive protocols, e.g. AODV [12]), have difficulties coping with the network properties: the fast mobility of the nodes results in many link up and link down events, making it hard for these protocols to maintain a correct topological view of the network. Furthermore they are incapable of overcoming topology gaps to get to another partition of the network.

Position based routing protocols (e.g. GPSR [10], CBF [9]) often use only information of their neighbors, reducing the chance of having outdated information. Moreover, positional information is much more predictive as a vehicle will always be located in the vicinity of the position where it was a couple of seconds ago. With multiple updates of the neighbors, it becomes even possible to estimate their speed, direction and acceleration. Most of these protocols however always try to get closer to the destination in a straight line. In vehicular networks this is not always possible nor desirable as the users that need to be reached drive on roads. If a message generated by an emergency application needs to reach all vehicles driving towards the place of an accident within a certain range, it is of no use to travel in straight lines as by doing this, it is likely that some vehicles following curved roads will not receive the information. Therefore, it is often better to 'keep the information on the road' instead of finding the shortest path in a geographical sense. Besides the necessity to do this in order to reach all the intended vehicles, it also reduces overhead by limiting the network traffic only to the required trajectories along the road.

REACT [5], Routing Protocol for Emergency Applications in Car-to-Car Networks using Trajectories, is a protocol that tries to offer a solution to the challenges mentioned above and in this paper we will propose some optimizations to the protocol in order to make it more adaptive with reference to the traffic conditions.

II. Scenario Description

A. Scenario Description

This paper focuses on a highway scenario with one lane in each direction. The highway is crossed by a bridge on which cars are driving too and there is one exit lane. An incident is simulated in a car on the highway, which will then alert all other cars behind it on an 8 km long part of the highway. Figure 1 shows the highway on the horizontal axis and the bridge on the vertical one. The exit lane can be chosen by cars driving towards the right of the figure. An alert packet is being transmitted at the Source and should reach the destination area as soon as possible. Some scenario parameters are given in table I. The following assumptions were made in this paper:

- All links are bidirectional.
- All vehicles are equipped with GPS receivers.

Description	Value
Highway length	10 km
Bridge position	4 km on highway
Bridge length	3 km
Sender	9.5 km on highway
Destination	1.5 km on highway
Exit position	8.5 km on highway

TABLE I
Scenario parameters

III. Protocol Description

A. Description of the protocol basics

1) Beaconing: Every node informs his neighbors of its presence by sending a beacon every β seconds, adding a small jitter in order to avoid repetitive collisions of beacons from different vehicles due to synchronization. The beacons carry the information shown in table II. A GPS receiver in the car can provide both the position information and the required clock synchronization. On the reception of a beacon, each neighbor will add this node to the neighbor list or update the node's position. In case of an update, the previous stored beacon will be kept as well in order to estimate the speed and direction of the node. A known neighbor times out after $\gamma * \beta$ seconds without having received a beacon (γ is the number of beacons that a node is allowed to miss) and is removed from the neighbor list.

Source ID
Source location
Timestamp

TABLE II
Beacon Packet Information

2) Alert messages: An alert message is generated whenever a vehicle wants to inform other vehicles about an incident. In order to do this, a packet contains the information shown in table III. The source location can be retrieved from the GPS system of the vehicle and by using the navigational system, it is easy to find out where the cars that are driving towards the place of the incident are: it is more or less a backtracking algorithm from the place where the vehicle is located to all possible starting points to get to this place several kilometers away. In the scenario for this paper it is very easy, since it is a highway scenario. The distance can differ depending on the incident or other parameters and the trajectory or trajectories should cover all cars that could be affected by this alert. The message ID field contains a unique number identifying

Source ID
Source location
Trajectory and final destination of the packet
Next forwarder ID
Packet Life Time (time until which the packet is valid in the network, equivalent to the Time To Live in the IP protocol)
Message ID
Message

TABLE III
ALERT PACKET INFORMATION

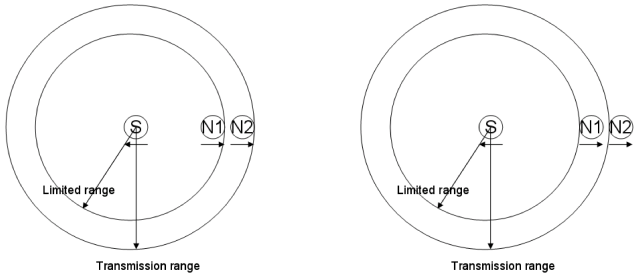

Fig. 2. Range Limiting

the message in the network and the message itself in the packet should inform the other vehicles of the exact type of incident that has happened, but this is abstracted in this paper. The Packet Life Time announces the time until when the information in this packet is valid in the network and is defined as $current\ time + validity\ time$. When the Packet Life Time is exceeded, the packet should be dropped. The next forwarder ID is filled in with a neighbor's ID by the originating hop and each forwarding hop if a next forwarder is found, otherwise an invalid neighbor address is used (special flag address, e.g. the broadcast address).

Whenever a node has to transmit a message, whether it generated the message itself or it was appointed as next forwarder, it will use the following algorithm in order to try to find the best next forwarder:

1) Calculate distance between the current node and the destination along the trajectory, we call it δ_0 and we take $\delta_{cur} = \delta_0$.
2) If $\delta_0 < \Delta$, with $2 * \Delta$ diameter of the destination range, the algorithm stops.
3) For each neighbor in the neighbor list:
   a) Calculate the distance between the neighbor's last known position and the destination and call it δ_i. If $\delta_i < \delta_{cur}$, proceed, else go to the next neighbor.
   b) Calculate the cosine of the angle of the trajectory and the neighbor's direction (if two positions are available for that neighbor). If the cosine of the angle is smaller than α, proceed, else go to the next neighbor. The value of α depends on the maximum angle between the trajectory and the neighbor's direction you want to allow for the next forwarder and is defined as the cosine of this maximally allowed angle.
   c) The neighbor is accepted as best option until now, $\delta_{cur} = \delta_i$
4) If a neighbor that is in a good position to forward the packet has been found, elect him as next forwarder.

If the algorithm is unsuccessful in finding a neighbor, the packet will be broadcasted anyway but with an invalid neighbor address. Every vehicle receiving the packet will check if it is valid along its own trajectory, i.e. if the reverse trajectory of the alert message and the node's trajectory coincide. If this is true, the packet will be delivered to the emergency application which can alert the driver or can take action. If the next forwarder ID in the packet is the vehicle's own ID, it will forward the packet using the algorithm above.

B. Protocol Properties

1) Acknowledgments: The wireless medium is often not reliable and popular MAC protocols like IEEE802.11 do not provide acknowledgments for broadcast packets. In previous work [5], this was discussed and a solution was proposed using the overhearing of consecutive transmissions as a way to solve this problem.

2) Store and Forward: The same paper [5] also proposed a scheme for solving the problem of topology gaps when there are too few cars to give a fully connected topology. This mechanism, store-and-forward ensures that information is delivered when the topology is fully connected over time by storing the information temporarily on an intermediate node until a next hop is found.

C. Protocol Enhancements

1) Adaptive Range Limiting: The technique of range limiting was mentioned in previous work [5] but has been enhanced. The problem that should be solved is the following: Selecting a next forwarder from neighbors in the full transmission range can lead to undesirable side effects. A neighbor that was in the transmission range at the moment of its last beacon transmission, but at the edge, can already have left this range and choosing this neighbor as next forwarder will lead to an acknowledgment timeout. As alerts should travel as fast as possible through the network due to time constraints, this situation should be avoided. Using a limited range in the algorithm can help preventing this situation. By using a range smaller than the transmission range, the probability of the next forwarder leaving the transmission range before receiving the alert can be dramatically reduced. E.g. in figure 2 (left side), if node S wants to transmit a packet to the destination, it can choose between nodes N1 and N2. Suppose N2 is closer to the destination than N1 and both vehicles are on the right trajectory, then naively N2 would be chosen as next forwarder. However, if after receiving a beacon from node N1 and N2, node S has moved away from these nodes, it can happen that N2 is out of the transmission range of S (figure 2 (right side)). Choosing N2 as next forwarder would introduce delay as node S has to wait for a transmission acknowledgment that will

not come. By limiting the range used in the next forwarder selection algorithm, it is more probable that the node will still be in range. Thus if N1 has been chosen, the algorithm ensures a higher probability of the next forwarder still being in range (unforeseen events like acceleration, N1 being switched off etc. can still result in an acknowledgment timeout).

On the other hand, if traffic is very sparse, it is more likely that the current forwarder can find a next forwarder in the full transmission range than in the limited range. Even when there is no next forwarder in range, it will definitely arrive in the transmission range faster than in the limited range. Concluding from this, in different traffic situations, range limiting may be helpful, but it not always is. It is thus clear that an adaptive use of the limited range is required in order to speed up packet delivery in all scenarios. An easy but effective approach is the adaptation of the range limiting algorithm to an algorithm that estimates the neighbors' positions. With two consecutive beacons containing position updates of a neighbor, one can easily estimate the neighbor's position and thus calculate whether the neighbor is still in range or not at the time of the alert transmission with the following formulas:

$$x_c = x_1 + (t_c - t_1)\frac{(x_2 - x_1)}{(t_2 - t1)}$$
$$y_c = y_1 + (t_c - t_1)\frac{(y_2 - y_1)}{(t_2 - t1)}$$

Index 1 in the formulas stands for the one before last beacon that is received and index 2 for the last received beacon. Index c stands for the current location. The formulas calculate an estimate of the x and y coordinates of the neighbor.
On a per neighbor basis, this estimation can be used in order to enhance the results as will be shown in the simulations. An easy extension is the use of three beacons of a neighbor (if already available) in order to take possible accelerations also into account.

2) Adaptive beaconing: Using a fixed beaconing interval is the easiest solution, but sometimes it is better to adapt the beaconing interval. When a node receives a beacon from a new neighbor, it can respond immediately with an own beacon to notify the neighbor of its existence (fast response). After a short time, only a fraction (e.g. 1/10th) of the normal beaconing interval, a second beacon is transmitted (fast repetition) in order to give the neighbor two measurements, enabling it to make accurate estimations of speed and direction. Furthermore, when a node wants to send or forward an alert message and can not find any suitable next forwarder, it will send beacons faster (e.g. ten times faster), hoping to trigger a fast response of a new neighbor, until the node is able to forward the packet or until the message expires. The combination of fast response and fast repetition is called 'Adaptive beaconing'.

There are many examples of other protocols where the beaconing interval is altered. A well-known example is FastOLSR [2] where the HELLO interval is reduced when the node is moving fast. The authors point out that this is used to discover new links and link breaks faster. While the latter is less applicable here due to the position estimation, the discovery of new links is very important. The adaptive beaconing presented in this paper is more reactive to the situation (i.e. the arrival of new neighbors or the presence of an alert message that needs to be sent) instead of being based on node speed, which is always high in a vehicular network.

"Adaptive Position Update in Geographic Routing" [13] is another way of adapting beaconing with position updates, based on position estimations. The closer a vehicle stays to the predicted route broadcasted in the last position update (position, direction and speed), the longer the interval between two consecutive beacons. This however poses a problem for emergency applications. When the beacon interval is longer, it also takes longer to learn about new neighbors, which can be important when no next forwarder can be found at the moment.

"Contention Based Forwarding (CBF)" [8] is radically different since it uses no beacons at all. The forwarder is elected by a contention mechanism in which the neighbor closer to the destination will set a shorter timer than the other neighbors. This mechanism works very well, but occasionally results in packets being duplicated. Solutions for duplication have been proposed in the same paper.

D. Protocol Beaconing Overhead

The beaconing overhead depends on three parameters: the beaconing interval, the number of neighbors that a vehicle has and the size of a beacon. In previous work, it was shown that even in the most dense scenario, distance between two cars is uniformly distributed over $[30m, 60m]$, the protocol overhead remains low: 1155.55... B/s. This is calculated with a fixed beaconing interval (0.25s) instead of an adaptive beaconing interval.

IV. Simulation Study

The simulations were carried out using the NS-2 network simulator [1]

A. Simulation Description

Description	Value
Transmission range	250m
Bandwidth	2 Mbps
Highway speed	30m/s (108km/h)
Bridge speed	20m/s (72km/h)
Distance between cars	between $x * v$ and $2 * x * v$, uniformly distributed
x	between 1 and 15 (used in the distance between cars)
Destination range	50m
Beaconing interval (=τ, =β in the algorithm description)	0.25s, 0.5s or 1.0s and adaptive
Limited range	220m, 235m or 250m and adaptive
Distance source - destination	8000m
γ in algorithm description	3
α in algorithm description	0.5

TABLE IV
Simulation Parameters

In the first simulation results, the influence of adaptive beaconing on the REACT protocol is shown, with and without range limiting. Secondly the effect of adaptive range limiting is discussed and thereafter the combination of the optimizations is compared to the use of no optimization or the use of only one of them. For all simulations there were twenty runs for each value of x between 1 and 15.

The parameter x translates directly to the distance between two consecutive cars (see table IV): if $1 <= x <= 3$, the distance between two consecutive cars will be smaller than 220m (between 30 and 60m for 1 and between 90m and 180m for 3) and there is full connectivity for all limited ranges. If $x = 4$ (distance between two consecutive cars is between 120m and 240m), there is only full connectivity guaranteed when using the full transmission range. For smaller limited ranges and for $x >= 5$ (distance between two consecutive cars is between 150m and 300m) full connectivity cannot be guaranteed as the distance between two consecutive cars can be larger than the transmission range. However, cars driving in the other direction and the network mobility can still provide connectivity over time (i.e. sometimes a node has to wait for a next forwarder, but connectivity is still possible). From $x = 8$ (distance between two consecutive cars is between 240m and 480m) or $x = 9$ (distance between two consecutive cars is between 270m and 540m) on, depending on the limited range used, the distance between two consecutive cars is definitely larger than the limited range and so two consecutive cars will never be able to talk directly to eachother. From than on communication is almost impossible as it depends on cars driving in the opposite direction, which also drive with 240m to 480m in between them (for $x = 8$). In fact in the latter case, the packet will be carried most of the time by a car C driving in the same direction the message has to travel (opposite lane of the highway) and C will pass it to cars driving towards the source when possible. While these cars are not able to forward the packet (distances too big), they will hand the packet back over to C when they cross it. This means that the packet travels only as fast as car C in these cases. Notice that all cars driving towards the incident get warned as soon as possible and that the information does not get lost even though it probably does not reach the initially intended final destination area.

B. Simulation Results

The graphs below have been made using twenty different simulation runs for every tuple of parameters. The error bars in the graphs show the standard deviation over the different runs. This deviation shows that the delay depends on the current situation, which can be more or less favorable, but a certain tendency holds in all cases. The packet life time used in the simulations was 200s, all (%f) graphs show the percentage of runs in which the alert packet did not reach the destination within this life time. Concerning the graph legends, ARNG means adaptive range limiting and ABCN adaptive beaconing (with fast repetition using 1/10th of the beaconing interval).

1) The influence of the beaconing interval: Previous work [5] showed improvements in end-to-end delay when reducing the beaconing interval, where the improvements got smaller every time the beaconing interval was reduced further. Here we study this in some more detail, with special attention to the adaptive beaconing case, which is described in section III-C2.

Figure 3 (no range limiting) shows that when the adaptive beaconing technique is used, we can reduce the difference

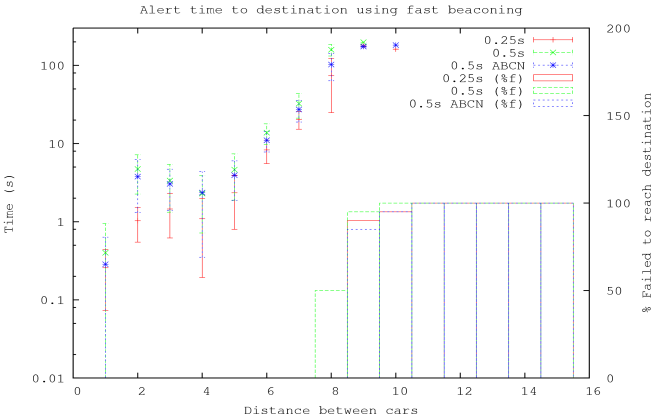

Fig. 3. Influence of using adaptive beaconing on alert time to destination (logarithmic scale)

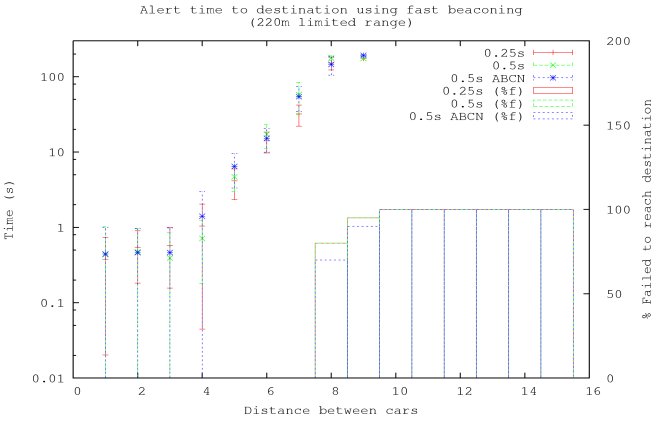

Fig. 4. Influence of using adaptive beaconing on alert time to destination, 220m range limiting (logarithmic scale)

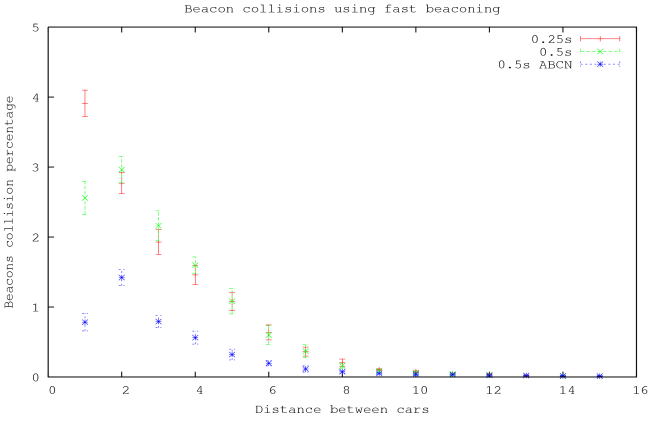

Fig. 5. Influence of using adaptive beaconing on beacon collisions

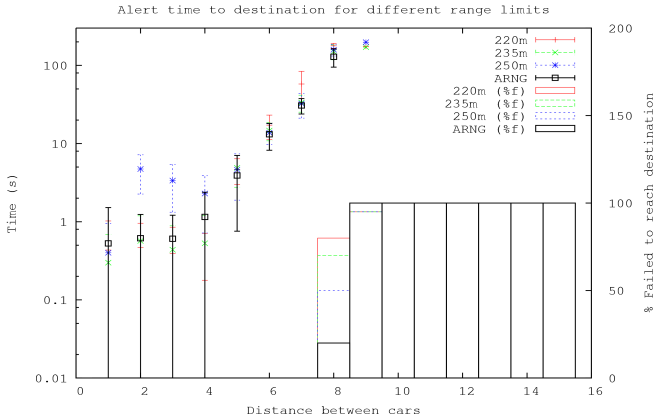

Fig. 6. Influence of the range limiting on alert time to destination (logarithmic scale)

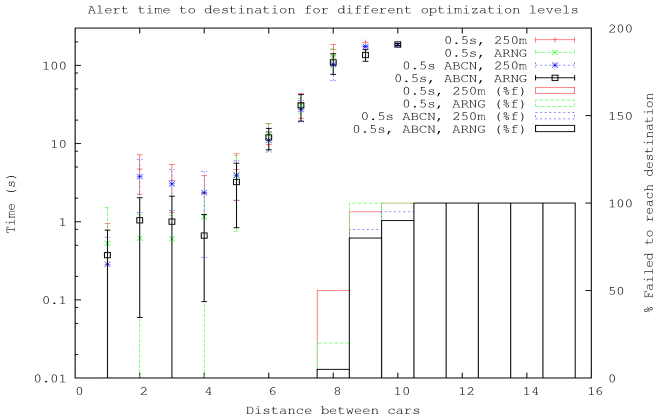

Fig. 7. Influence of optimization levels (logarithmic scale)

between 0.25s and 0.5s in sparse scenarios even further, but not that much in more dense scenarios. In more dense scenarios, the problem of the next hop having left the transmission range has a big negative impact while there are always good next hops, so the adaptive beaconing algorithm for learning the existence of new neighbors faster does not have a real added value here.

Using limited ranges (figure 4), the positive effect in the sparser scenarios has almost completely vanished and the different graphs are closer together: while waiting until a new neighbor enters the limited range, a vehicle will already have received two beacons (since the neighbor is already in transmission range) and thus adaptive beaconing does not help at all.

A positive side effect of the adaptive beaconing is the fact that it reduces collisions of the beacons. Figure 5 shows this: at first it may seem strange that adaptive beaconing reduces the number of collisions, but if one looks back at the fast response of adaptive beaconing, one sees that a node receiving a beacon from an as yet unknown neighbor will send a beacon in reply immediately after the received beacon. Since all nodes in the network do this and since the network has a linear form (road strip), the nodes get synchronized in such a way that they transmit beacons one after another and so this reduces the chance of collisions.

As a last item here, one should take into account that the biggest part of the delay is caused by gaps in the topology and closing these gaps takes a lot more time than will be spent on waiting for beacons. Faster beaconing will not make the cars move any faster, with or without the adaptive beaconing variant.

2) The influence of range limiting: In previous work, it was shown that range limiting is especially helpful in dense scenarios where there are abundant next hops to be chosen, so that one does not need to take a next hop near the edge of the transmission range, risking it to have left this range. In more sparse scenarios, looking for next hops in a limited range can cause extra delays.

The ARNG values in the graph however show that if a more adaptive approach to range limiting is used, as described in section III-C1, both sparse and dense scenarios can be improved. Using the adaptive approach, calculations can be made in order to be sure that the neighbor is still in range. It is not so naive as to believe that neighbors that have left the transmission range recently to be still in range and not too unbiasedly strict in limiting its neighbor set. In the very dense scenarios the performance is very good, comparable with the performance of the limited ranges while in the sparse scenarios the performance of the adaptive approach is better than using the full transmission range.

3) Standard versus different levels of optimization: As a last result, a break down of the optimizations is shown in figure 7. Using a beaconing interval of 0.5s, it is clear that the adaptive range approach is indeed a good optimization for very dense vehicular networks whereas the fast beaconing performs very good in sparse scenarios. Using both optimizations together gives an overall better performance and results in less packets lost when there are fewer vehicles. While one optimization alone sometimes outperforms the use of both, the difference is negligible and the use of both optimizations clearly prepares the vehicles for alert messaging in all traffic situations.

V. Future Work

In future work it is important to look at adaptations of REACT, mostly in the way the next hop is calculated and the trajectory is described, to other scenarios like a Manhattan city model.
Furthermore, better MAC protocols for vehicular networks and ways of using the REACT protocol with them should be investigated, more in particular the possibilities of cross-layering with these MAC protocols. Another track in this research should include the study of other applications with unicast traffic like communication between specific cars and applications where a vehicle queries the network for information (e.g. where is a parking space for this vehicle?).

VI. Conclusions

This paper presents optimizations to the REACT routing protocol for vehicular networks. The protocol takes into ac-

count both the special properties of the vehicular network and the requirements of the emergency alert application using geographical and topology information. On top of this, the proposed optimizations make it possible for the protocol to adapt to situations with more or less traffic. The results show a clear performance gain when using the adaptive beaconing and range limiting, explaining in detail where each optimization reduces the end-to-end delay.

REFERENCES

[1] Ns-2 network simulator.

[2] Mounir Benzaid, Pascale Minet, and Khaldoun Al Agha. Integrating fast mobility in the olsr routing protocol. In *Fourth IEEE Conference in Mobile and Wireless Communications Networks*, September 2002.

[3] Car 2 car communication consortium. http://www.car-to-car.org/.

[4] T. Clause and P. Jacquet. Optimized link state routing protocol (olsr). Technical report, Project Hipercom, INRIA, October 2003. RFC 3626.

[5] Erwin Van de Velde, Chris Blondia, and Luca Campelli. React: Routing protocol for emergency applications in car-to-car networks using trajectories. In *Proceedings of The Fifth Annual Mediterranean Ad Hoc Networking Workshop*, pages 145–151, June 2006.

[6] Dedicated short range communications. http://www.leearmstrong.com/Dsrc/DSRCHome.htm.

[7] Fleetnet. http://www.et2.tu-harburg.de/fleetnet/index.html.

[8] Holger Füssler, Jörg Widmer, Michael Käsemann, Martin Mauve, and Hannes Hartenstein. Contention-based forwarding for mobile ad-hoc networks. In *Elsevier's Ad Hoc Networks*, volume 1, pages 351–369, 2003.

[9] Holger Füssler, Jörg Widmer, Martin Mauve, and Hannes Hartenstein. A novel forwarding paradigm for position-based routing (with implicit addressing). In *Proc. of IEEE 18th Annual Workshop on Computer Communications (CCW 2003)*, pages 194–200, October 2003.

[10] B. Karp and H.T. Kung. Gpsr: Greedy perimeter stateless routing for wireless networks. In *6th Annual ACM/IEEE International Conference on Mobile Computing and Networking*. MobiCom2000, 2000.

[11] Mobile ad hoc networks. http://www.ietf.org/html.charters/manet-charter.html.

[12] Charles E. Perkins, E. Belding-Royer, and S. Das. Ad hoc on-demand distance vector (aodv) routing. Technical report, Nokia Research Center and University of California and University of Cincinnati, February 2003. RFC 3561.

[13] Mahbub Hassan Quan Jun Chen, Salil S. Kanhere. Adaptive position update in geographic routing. In *ICC '06. IEEE International Conference on Communications*, pages 4046–4051, June 2006.