Hierarchical Time-Vector-Routing for Mobile Ad Hoc Networks

Joerg Habetha
Philips Research, Weisshausstrasse 2, D-52066 Aachen, Germany

Diego Calvo de No
Aachen University of Technology, Chair of Communication Networks, Kopernikusstr. 16, D-52074 Aachen, Germany

Abstract — **A new routing algorithm called *Hierarchical Time-Vector-Routing* is presented. The algorithm presumes that the network is divided into logical clusters. In each cluster a so-called Central Controller stores the routing information and exchanges it with its neighbouring Central Controllers. Each routing entry contains a time vector which is used to react to dynamic topology changes and to minimise the amount of exchanged routing information.**
The presented algorithm is well suited for mobile ad hoc networks. The algorithm has been especially designed for a centralised ad hoc network based on the HIPERLAN/2 standard but may be used in any ad hoc network configuration.
The performance of the algorithm is analytically compared to the performance of the Dynamic Source Routing protocol.

I. Introduction

Adaptive routing strategies can be divided into centralized, isolated and distributed algorithms. Distributed routing strategies are the most appropriate solution for the multihop ad hoc network environment. These algorithms are distributed in the sense that each node locally stores routing information, which is updated by message exchanges between the nodes. This is a more robust technique than the centralised algorithms and does not use as much network capacity as the isolated algorithms.
Within the group of distributed routing strategies two main ideas can be distinguished: proactive and reactive routing algorithms. Proactive algorithms continuously evaluate the routes within the network, so that the route is already known to the nodes and can be immediately used, when a packet needs to be forwarded. Reactive strategies invoke a discovery procedure only if a route to a destination is required (*on demand routing*).
A well-suited reactive algorithm for highly mobile, large ad hoc networks is the Dynamic Source Routing (DSR) algorithm [1]. DSR includes source routes in the packet headers, which can lead to large headers degrading system throughput especially in case of short packets. A scheme that eliminates the inclusion of the routes in the packet headers is the Ad Hoc On-Demand Distance Vector (AODV) routing [2]. This is achieved by maintaining routing tables at the nodes while keeping the on-demand route discovery feature of the DSR.
Another way to classify routing algorithms is to distinguish between flat and hierarchical routing techniques. With the flat routing technique all nodes in the network take over the same functionality. The advantage of the flat logical configuration is the existence of multiple paths between source and destination. The traffic can be spread out among multiple routes reducing congestion and eliminating possible traffic bottlenecks in the network. Routes can be chosen to better match the specific requirements of a traffic stream. For example, low delay and low capacity paths could be used for voice traffic, while file transfer could be carried out over high capacity and longer delay routes.
Hierarchical routing techniques divide the network nodes into logical zones or clusters. The aim is to reduce the complexity of the routing process and the storage requirements, as a node may only know the exact routes to destinations inside its own cluster. One node inside a cluster may carry out special tasks. This node will then be called the Clusterhead or Central Controller (CC). Hierarchical routing is well suited for hybrid protocols like the Zone Routing Protocol (ZRP) [3], in which intra-zone routing is proactive and inter-zone routing is reactive. Other examples of hierarchical routing schemes for dynamic networks are treated in [4].
In general the hierarchical organisation can consist of several layers.

II. Routing Criteria

To select a route between all the possible paths a selection criteria is needed. The most common way to select a route is to assign a metric value (number of hops, delay time, etc.) to each possible route and then choose the route with the lowest (or highest) value. This metric value is called cost of the path and can be based on a number of parameters. A simple and very commonly used value in radio transmission environments is the number of hops, because in such environments each hop in a path requires e. g. additional back-off times, acknowledgment messages and leads to possible collisions. The metric values associated with each link can also be related to the capacity or the transmission delays of the links. A link with a higher capacity or lower transmission delay would have a lower cost. The assignment of different parameters to the cost of a link makes QoS routing possible.

III. System Overview

A wireless ad hoc network is formed by mobile hosts without the use of any pre-existing infrastructure. Communication between hosts will in general involve other hosts which have to forward the data (multihop connections). Traditional ad hoc networks have a decentralised Medium Access Control (MAC) as well as decentralised higher layer functions like routing. An example of such a network is the IEEE 802.11 system.

In [5] a concept of a centralised multihop ad hoc network has been presented. The proposed system is based on the HIPERLAN/2 standard. HIPERLAN/2 is a wireless LAN, standardised by the European Telecommunications Standarisation Institute (ETSI) for operation in the unlicensed 5 GHz band. An overview of HIPERLAN/2 and a performance analysis of the system is presented in [6]. The basic HIPERLAN/2 standard defines an infrastructure-supported, base-station-oriented LAN. Nevertheless an extension of the basic standard has been specified, which defines an ad hoc mode of operation. In the ad hoc mode one device is selected to become the so-called Central Controller (CC) of the network. The CC generates the MAC frames and grants the other terminals access to the air interface. The current system, as it is standardised by ETSI, consists of only one cluster. In [5] it has been shown how to extent the coverage area of the system by clustering the network into several clusters and interconnecting them by so-called forwarding stations. The concept is very similar to the one presented in [7]. As terminals move, clusters may split or merge, altering cluster membership of the nodes. The formation of clusters is e. g. treated in [8] and [9]. Considering the clustered network structure on MAC layer, a straight-forward routing approach also uses this clustered structure on network layer. As a CC knows every terminal inside its cluster, routing tables should be stored and updated by the CCs.
The new routing scheme, we develop in the following, is very appropriate for this specific system. Nevertheless it can also be used for any other centralised or decentralised ad hoc network.

IV. Hierarchical Time-Vector-Routing (HTVR)

The routing algorithm presented in this paper is adaptive, distributed, proactive and hierarchical. It takes into account dynamic topology and traffic conditions.
Each Central Controller (CC) performs routing decisions, shares routing information and continuously evaluates the routes within the network. The routing information is stored locally by each CC. For each path only the Id of the forwarder to the next cluster is stored and not the whole path. This has the advantage that if a link of the path changes, this change will only have to be stored at one CC.

A. The local routing tables

Fig. 1 shows the routing table as it is stored at a CC. The local routing tables are composed of N fields (one for each terminal in the network) and a Table Update Time (t_{up}) that gives the time at which the last change(s) were applied to the table.

In addition to the Id of the forwarder to the next cluster, each of the fields **F** on the table has four other entries: A Path Length (PL), a Maximum Transmission Rate (MTR), a Field Generation Time (t_{gen}) and a Field Registration Time (t_{reg}).
The *Path Length* is measured in links and denotes the distance between the CC where the field is stored and the terminal corresponding to this field. The *Maximum Transmission Rate* is the maximum possible transmission rate through this path. It is

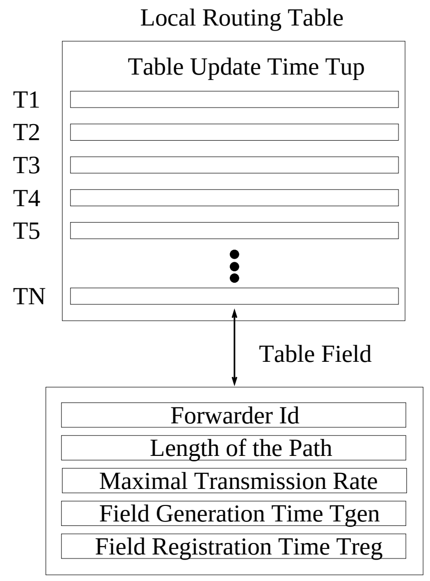

Fig. 1: HTVR routing table

equal to the lowest single link transmission rate of all links that compose the path. Every time a terminal changes its cluster, the change is registered at the routing table of its new CC. The time at which this happens is called the *Field Generation Time*. It indicates how recent the information on this field is. If the information concerning this terminal is transmitted to another CC, the Field Generation Time is then copied onto the routing table of the receiving CC. The *Field Registration Time* of a field gives the time at which the field was last changed.
The different time values, that form a time vector and give the routing algorithm its name, are of crucial importance for the routing procedure.

B. The update procedure

The routing tables are updated at a regular frequency. The update processes are not synchronised. They are performed in each cluster at different times. To update its routing table, the updating CC (UCC) sends an UPDATE-REQUEST containing its t_{up} to its neighbours. The CCs that receive the update-request compare the t_{up} they received with t_{reg} over the whole routing table. Subsequently, they send back an UPDATE-RESPONSE message containing all the fields at which $t_{reg}>t_{up}$. As it receives the UPDATE-RESPONSE, the UCC compares the fields it has received ($\mathbf{F_i^{new}}$) with the fields at its routing table ($\mathbf{F_i^{ucc}}$). If the compared fields meet the update criteria the fields are updated. The update criteria are:

- $t_{gen}^{new}>t_{gen}^{ucc}$
- OR (($t_{gen}^{new}=t_{gen}^{ucc}$) AND ($PL^{new}$+2<$PL^{ucc}$))
- OR (($t_{gen}^{new}=t_{gen}^{ucc}$) AND ($PL^{new}$+2=$PL^{ucc}$) AND ($min(MTR^{new}, MTR^{nch-ucc}$ [1])=MTR^{ucc}))

[1]MTR$^{ucc-ncc}$ is the maximum transmission rate between the neighbouring CC and the updating CC

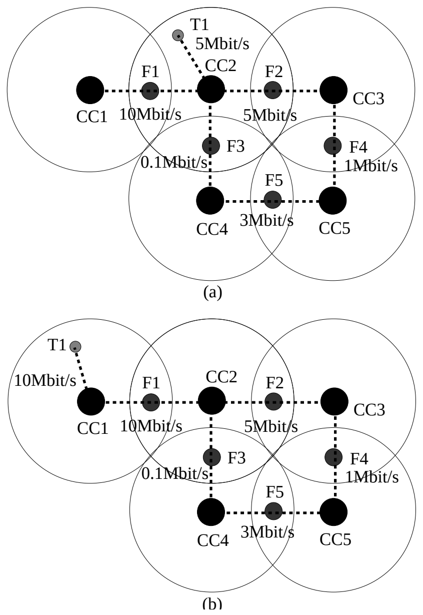

Fig. 2: A routing example

	CC1	CC2	CC3	CC4	CC5
	F1	-	F2	F3	F5
	3	1	3	3	5
t_0	5Mbit/s	5Mbit/s	5Mbit/s	0.1Mbit/s	1Mbit/s
	0	0	0	0	0
	0	0	0	0	0
	-	-	F2	F3	F5
	1	-	3	3	5
t_1	10Mbit/s	-	5Mbit/s	0.1Mbit/s	1Mbit/s
	1	-	0	0	0
	1	-	0	0	0
	-	F1	F2	F3	F5
	1	3	3	3	5
t_2	10Mbit/s	10Mbit/s	5Mbit/s	0.1Mbit/s	1Mbit/s
	1	1	0	0	0
	1	2	0	0	0
	-	F1	F2	F3	F5
	1	3	5	5	5
t_3	10Mbit/s	10Mbit/s	5Mbit/s	0.1Mbit/s	1Mbit/s
	1	1	1	1	0
	1	2	3	3	0
	-	F1	F2	F3	F5
	1	3	5	5	7
t_4	10Mbit/s	10Mbit/s	5Mbit/s	0.1Mbit/s	1Mbit/s
	1	1	1	1	1
	1	2	3	3	4

Table 1: A routing example

If the criteria are met the following changes are applied:

- The new ForwarderId is set to the Id of the Forwarder that connects the UCC with the CC that sent the UPDATE-RESPONSE
- t_{gen}^{ucc}=t_{gen}^{new}
- PL^{ucc}=PL^{new}+2
- MTR^{ucc}=$min(MTR^{new}, MTR^{ncc-ucc})$

Note that in contrast to traditional routing algorithms like Distance-Vector-Routing updating of routing table entries is first of all determined by the age of the received information. This results in the property that bad news propagates as fast as good news through the network thereby eliminating known weaknesses of other protocols like the *count-to-infinity* problem. Another main advantage of the algorithm resides in the fact that a CC does not distribute its whole routing table upon request of another CC but only those entries that have changed since the last update at the requesting terminal.

V. A Routing Example

Fig. 2 and table V illustrate an example of how the routing tables change each time a terminal moves from one cluster to a neighbouring cluster. At t_1 the terminal T1 changes its position from C2 (2a) to C1 (2b). Table V shows the T1 field at every routing table for the period t_0 to t_4.

A remarkable property of this algorithm can be observed with the help of this example. As T1 changes its position, the routing tables are updated all over the network. These updates are not done simultaneously at all the clusters. The new information propagates over the network. The clusters which are closer to C1 receive the information earlier than the clusters that are further away. However, looking at the tables we notice that the most important entry, the forwarder Id, only changes in the routing tables of CC1 and CC2. This is a sign of the robustness of the algorithm. This enables (in most cases) data to be sent through a changing path without having to wait until the information concerning the changes arrives to the sending CC. As the information finally arrives, the CC uses it to ensure the QoS for this path.

VI. Analysis of the Efficiency of the HTVR algorithm

In order to analyse the efficiency of the HTVR algorithm presented above, we will compare it with an on-demand routing algorithm (ODRA). The ODRA searches for a path only if the route is needed. The route is then determined by broadcasting a Route Request message (RREQ). As the RREQ reaches its destination the discovered route is made available by unicasting a Route Reply message (RREP) back to the source node of the RREQ. The chosen scenario for this analysis is represented in figure 3. The network is composed by a square of n^2 clusters. There is always a forwarder between two clusters that overlap. Except for the clusters situated at the borders of the square, every cluster has a direct connection to four neighbouring clusters.

The parameters selected to compare the efficiency of the algorithms are the average routing time (t_{av}) and the average routing information (I_{av}). t_{av} is for the HTVR the time needed to modify a path. For the ODRA it is the time required to find a new path. I_{av} is the average number of information units that are sent through the network for routing pourposes in one second. An information unit is defined here as the routing information of one path. The ODRA packages and the RREQ

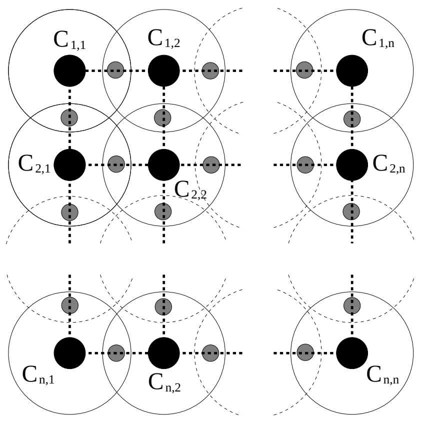

Fig. 3: A routing scenario

packages of the HTVR algorithm are always considered as 1 unit of information. The size of the HTVR RREP packages is equal to its number of fields.

A. t_{av} and I_{av} for the HTVR

For the HTVR t_{av} is equal to:

$$t_{av} = \frac{1}{2f_{up}} + 4t_h = \frac{t_{up}}{2} + 4t_h \quad (1)$$

f_{up} being the frequency with which the routing procedure is repeated and t_h being the time needed for a package to be sent over one hop. As can be depicted from Fig. 3 two hops separate a cluster from each of its neighbouring clusters. To obtain I_{av} we first need to calculate I'_{av}. I'_{av} is the amount of information units sent on average every time the routing procedure is performed by one CC.

$$I_{av} = I'_{av}\frac{n^2}{t_{up}} \quad (2)$$

I'_{av} is composed by a RREQ part and a RREP part:

$$I'_{av} = I_{RREQ} + I_{RREP} = 2(4 - \frac{4}{n}) + (\frac{8n^2 - 8n}{t_r}t_{up}) \quad (3)$$

Each cluster on average has $4 - \frac{4}{n}$ neighbouring clusters. t_r is the average time between two terminal cluster handovers in the network. I_{av} can then be expressed as:

$$I_{av} = 8\frac{n^2}{t_{up}}(\frac{n^2 - n}{t_r}t_{up} + 1 - \frac{1}{n}) \quad (4)$$

B. t_{av} and I_{av} for the ODRA

An important variable needed in order to obtain t_{av} and I_{av} for the ODRA is the average distance between two randomly chosen clusters (d_{av}). The indices i and j give the position of the center of a cluster in the network. The distance between two clusters with indices i, j and k, l respectively in number of hops (two times the distance in clusters) is then equal to:

$$d_{i,j,k,l} = 2(|i - k| + |j - l|) \quad (5)$$

To obtain d_{av} we have to sum $d_{i,j,k,l}$ for i,j,k,l<n and divide it by the number of possible links $n^2(n^2 - 1)$.

$$d_{av} = \frac{\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n}\sum_{l=1}^{n} d_{i,j,k,l}}{n^2(n^2-1)} \quad (6)$$

This can be simplified to the following function:

$$d_{av} = \frac{4}{3}n \quad (7)$$

t_{av} is $2d_{av}t_h$ for the ODRA, because it is the average time it takes a package to travel from a source cluster to a destination cluster and come back.

$$t_{av} = 2d_{av}t_h = \frac{8nt_h}{3} \quad (8)$$

I_{av} can be expressed as:

$$I_{av} = n^2(n^2 - 1)f_{con}I'_{av} \quad (9)$$

f_{con} being the frequency at which any connection between two clusters is asked.
I'_{av} is composed of a broadcasting part I_{br} and a unicasting part I_{uni}.

$$I'_{av} = I_{br} + I_{uni} \quad (10)$$

I_{br} is the information sent during the broadcasting period. Every cluster sends the RREQ to all its neighbouring CCs, except to the CCs from which the message came. Some of the CCs receive the RREQ message from one neighbouring cluster, some receive it from two neighbouring clusters.

Analysing the scenario in Fig. 3 it can be calculated that $2(2n^2 - 2n)$ information units are sent in total if the RREQ message is initiated by one of the $(n-2)(n-2)$ clusters with four neighbouring clusters. $2(2n^2 - 2n - 2)$ information units are sent if the RREQ message is initiated by one of the $4(n-2)$ clusters on the border and $2(2n^2 - 2n + 2)$ information units result if the RREQ message is initiated by one of the four corner clusters. The lower order differences are neglected, which results in the following formula for the number of information units in broadcasting phase:

$$I_{br} = 2(2n^2 - 2n) \quad (11)$$

I_{uni} gives the number of times the RREP is sent and is equal to the average number of links d_{av}. So, I_{av} can be expressed as following:

$$I_{av} = n^2(n^2 - 1)f_{con}(4n^2 - \frac{8n}{3}) \quad (12)$$

In Fig. 4 the average routing time t_{av} is illustrated in units of t_h for the HTVR and the ODRA. The performance of the HTVR is analysed for different values of t_{up}. It can be seen

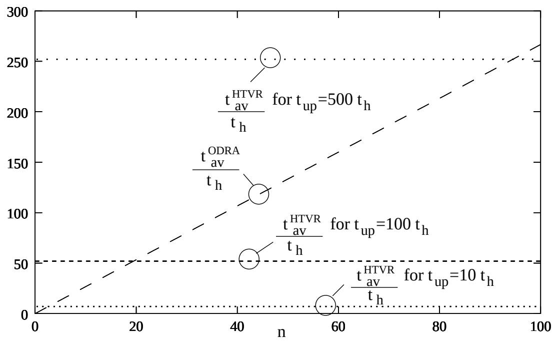

Fig. 4: Average routing time for ODRA and HTVR

that the average routing time is constant in case of the HTVR whereas it is linearly increasing with the number of nodes n in case of the ODRA. This property makes the HTVR especially suited for large networks.

The ratio of the average amount of routing information of ODRA compared to the HTVR $I_{av}^{ODRA}/I_{av}^{HTVR}$ is calculated in Fig. 5 for different values of f_{con}/f_{up}. t_r is assumed to be equal to t_{up} in this example. It can be depicted that the HTVR performs better than the ODRA when n exceeds a value of about 10 (provided that f_{up} being smaller than $100 f_{con}$). It has to be noted that the average time between two terminal handovers t_r is decreasing with the number of terminals in the network:

$$t_r = \frac{t_r'}{N} \tag{13}$$

where t_r' denotes the average time between two cluster changes of the same terminal. The total number of terminals in the network N might be proportional to the number of clusters: $N = a \cdot n^2$, in which case the ratio $I_{av}^{ODRA}/I_{av}^{HTVR}$ becomes constant for large n (namely equal to $\frac{1}{2a} f_{con} t_r'$). It can be concluded that the routing overhead ratio mainly depends on the connection set-up frequency and terminal mobility.

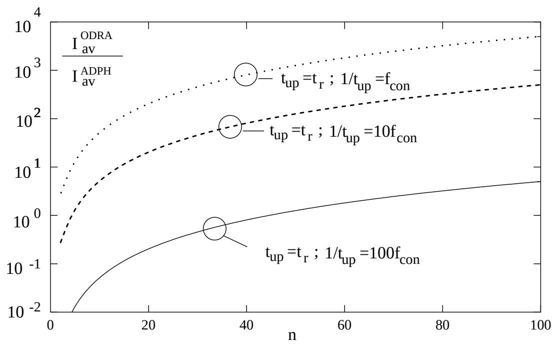

Fig. 5: Ratio of the amount of routing information of ODRA and HTVR

VII. Conclusions

In this paper an adaptive, distributed, proactive and hierarchical routing algorithm called *Hierarchical Time-Vector-Routing* has been presented. The main idea of the algorithm has been to foresee on-demand updates of routing tables between neighbouring clusters. A time vector has been introduced to limit the routing overhead and to quickly adapt to topolocical changes of the network.
The efficiency of the algorithm has been analysed and compared to the efficiency of an on-demand routing algorithm. The HTVR performed better than an ODRA in big scenarios. However, the final choice of the algorithm does mainly depend on the connection set-up frequency and terminal mobility. The HTVR is well suited for scenarios with relatively high connection set-up frequency and moderate terminal mobility. The presented routing algorithm is especially suited for networks, in which a clustering approach has not only been adopted on network layer but also on MAC layer. Nevertheless it can also be applied to decentralised ad hoc networks.

VIII. References

[1] D. Johnson and D. A. Maltz, “Dynamic Source Routing in Ad Hoc Wireless Networks,” in *Mobile Computing* (T. Imielinski and H. Korth, eds.), Kluwer Academic Publishers, 1996.

[2] C. E. Perkins and E. M. Royer, “Ad-hoc on demand distance vector routing,” in *IEEE Workshop on Mobile Computing Systems and Applications (WMCSA)*, Feb. 1999.

[3] Z. J. Haas and M. R. Pearlman, “Zone Routing Protocol for Ad Hoc Networks,” tech. rep., Internet Engineering Taks Force (IETF), Nov. 1997.

[4] F. Kamoun and L. Kleinrock, “Hierarchical routing for large networks: Performance evaluation and optimization,” *Comput. Networks*, vol. 1, pp. 155 – 174, 1977.

[5] J. Habetha and M. Nadler, “Concept of a wireless centralised multihop ad hoc network,” in *Proc. European Wireless 2000*, Sept. 2000.

[6] B. Walke, N. Esseling, J. Habetha, A. Hettich, A. Kadelka, S. Mangold, and J. Peetz, “IP over Wireless Mobile ATM – Guaranteed Wireless QoS by HiperLAN/2,” *Proc. of the IEEE*, vol. 89, Jan. 2001.

[7] M. Gerla and J. Tzu-Chieh Tsai, “Multicluster, mobile, multimedia radio network,” *Wireless Networks*, vol. 1, pp. 255 – 265, Oct. 1995.

[8] J. Habetha, A. Hettich, J. Peetz, and Y. Du, “Central Controller Handover Procedure for ETSI-BRAN HiperLAN/2 Ad Hoc Networks and Clustering with Quality of Service Guarantees,” in *IEEE Annual Workshop on Mobile Ad Hoc Networking & Computing (MobiHOC)*, Aug. 2000.

[9] C. R. Lin and M. Gerla, “Adaptive Clustering for Mobile Wireless Networks,” *IEEE J. Select. Areas Commun.*, vol. 15, pp. 1265 – 1275, Sept. 1997.