DDR-BASED MULTICAST PROTOCOL WITH DYNAMIC CORE (DMPDC)

Shiyi Wu and Christian Bonnet
Institut Eurécom *
2229, Route des Crêtes
B.P. 193
06904 Sophia Antipolis, France
{Firstname.Name}@eurecom.fr

Abstract— **Mobile ad-hoc network (MANET) needs special multicast routing protocols to adapt its characteristics, including local broadcast capacity, arbitrary topology change, bandwidth constraint and power limitation. This paper proposes a new multicast routing protocol called DDR-based Multicast Protocol using Dynamic Core (DMPDC) for MANET. The aim of this protocol is to find a tradeoff between routing overhead and data transmission for an efficient use of bandwidth and power. DMPDC benefits from a logical infrastructure offered by Distributed Dynamic Routing algorithm (DDR) and constructs a group-shared multicast tree with a dynamically selected core only when group traffic is present. DMPDC attempts to react more quickly to broken tree edges by detecting link failures during data forwarding.**

Keywords—**Mobile ad hoc networks, ad hoc routing, multicast routing.**

I. INTRODUCTION

A Mobile Ad-hoc NETwork (MANET) is a collection of wireless mobile nodes forming a dynamical temporary network without the use of any existing network infrastructure or centralized administration. Different from traditional wireless networks, nodes have to use other mobile nodes in the network instead of some fixed router or an infrastructure as relay if the destination of data is not in their coverage area. The natures of mobile nodes decide the features of MANET such as broadcast capacity, dynamic topology, bandwidth and power constraints, etc. This type of networks is suitable for supporting the applications like virtual classroom, emergency search and rescue operation, data exchanging during conference, meeting, etc. Eventually, MANET needs multicast routing protocols to establish many-to-many communication for efficient delivery data among a group. The properties of MANET make multicast routing protocols for wired networks (ex. CBT [1], DVMRP [4], MOSPF [10] and PIM [3]) not suitable for MANET because they can not face group membership dynamics and topology changes at the same time with limited bandwidth.

Lots of multicast routing protocols have been proposed such as AMRoute [2], AMRIS [14], CAMP [6], LAM [7], MAODV [13], MZR [5], NSMP [8] and ODMRP [9]. These protocols can be classified as *tree-based* and *mesh-based* approach, according to **routing structure**. The tree-based approach consists in creating and maintaining a multicast routing tree to deliver data. Once a tree is established, packets or messages are sent to all the routers in the tree only once. For a tree containing N nodes, N-1 links are needed to forward multicast packets with point-to-point links. In the case that the network has broadcast links using a single channel, only internal tree nodes forward multicast packets. Benefiting from broadcast capability of MANETs, the latter approach proposes to use *mesh* - a connected graph for multicast traffic forwarding. Multicast packets are broadcast to node's neighborhood. Only the neighbors who are also mesh members will react to non-duplicated multicast packets. Mesh structure offers redundant routes for data delivery which gives rich connectivity but involves more nodes for forwarding than tree structure. Therefore, tree-based approach is more efficient in data transmission than mesh in long term data delivery. Here, data transmission efficiency means the number of data packets transmitted in the network for delivering a data packet.

Tree is sensitive to node mobility since it provides minimal connectivity among multicast group members. Routing messages are needed to repair tree branches every time topology changes touch them. In the contrary, mesh offers more connectivity and, as a result, is robust against topology changes. Mesh can tolerate link failure between two mesh members if they can get data from other mesh members. Hole will disappear in the next period of mesh refresh. So, in terms of routing overhead, mesh-based approach is more efficient than tree-based approach in high mobility network.

Multicast routing protocols can also be categorized into *source-oriented* and *group-shared*, according to **construction mechanism**. Group-shared mechanism gives one structure per group and all sources use this structure to distribute their data. On the other hand, source-oriented mechanism can construct a structure for each source according to various criteria (ex: shortest path, various QoS requirements). Therefore, it is more efficient in terms of data transmission than group-shared mechanism when a group has multiple sources. However, this mechanism can easily suffer from scalability problem as the number of groups and sources per group increases. Moreover, source-oriented mechanism should pay much more routing overhead to maintain all structures for a group than group-shared mechanism, in which only one structure should be maintained per group. These issues make group-shared mechanism more suitable for the MANET environment.

This paper proposes a multicast routing protocol, DDR-based Multicast Protocol with Dynamic Core (DMPDC) for MANETs, which constructs a group-shared tree for a multicast session on demand based on the Distributed Dynamic Routing

---

* This work is supported in part by HITACHI EUROPE

algorithm (DDR) [11]. In the rest of this paper, Section II gives an overview of DMPDC that includes basic ideas of DDR and general description of this multicast protocol. Then, Section III describes in detail the creation and maintenance of multicast tree (MTree). Our conclusion is given in Section IV.

II. PROTOCOL OVERVIEW

A. *DDR algorithm*

Distribute Dynamic Routing algorithm (DDR)[11],[12] is designed to offer a flexible infrastructure on which several routing protocols may be defined according to specific application requirements.

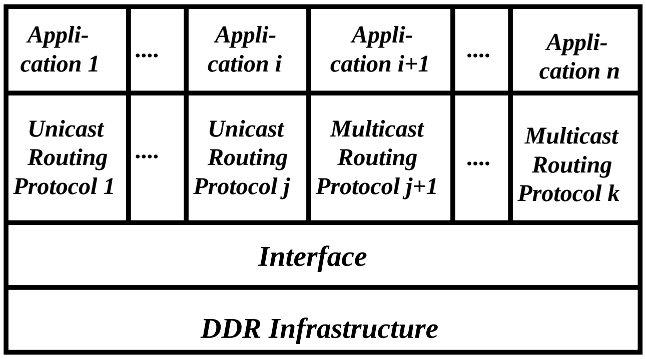
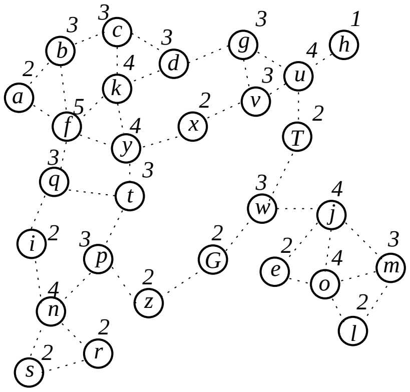


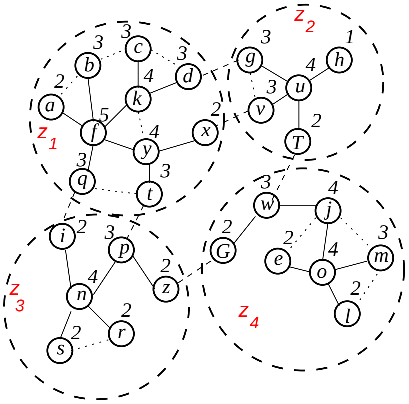
Fig. 1. DDR Infrastructure

Fig. 1 represents the *infrastructure* provided by DDR that can interact with both unicast routing protocols and multicast routing protocols via an *interface*. DDR is a multi-mode routing infrastructure: it can be degraded to reactive approach if the zone size is too small, and can be expanded to proactive approach if the zone size becomes too large.[1] DDR can be simply extended to a routing protocol as a routing infrastructure. For this purpose, DDR could be tuned via the interface in order to satisfy both application requirements and network properties (for further information refer to [11]).

The main idea of DDR is to construct a forest from a network topology, where each tree of the constructed forest has to be optimal. Then, each tree forms a **zone**. After that, the network is logically partitioned into a set of non overlapping dynamic zones. Each node computes periodically its **zone ID** independently so that zones are named. Each zone is connected via the **gateway nodes** that are not in the same tree but are in the direct transmission range of each other. The connection between two neighbor zones is called **bridge**. So, the whole network can be seen as a set of zones connected by bridges. Thus, each node from zone z_i can communicate with another node from zone z_j. Each node is assumed to maintain routing information only to those nodes that are within its zone, and information regarding only its neighboring zones.



(a) Network Topology



(b) Network Topology under DDR

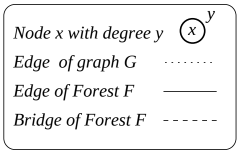


Fig. 2. DDR Infrastructure

Fig 2(a) represents an arbitrary network topology. Once DDR algorithm is executed on each mobile node, the network is partitioned into a set of non overlapping dynamic zones, as it is illustrated in Fig 2(b).

B. *General Description of DMPDC*

The aim of DDR-based Multicast Routing protocol using Dynamic Core (DMPDC) is trying to find a tradeoff between routing overhead and data transmission efficiency in MANET.

The interest of working on the top of DDR infrastructure is that it reduces the cost of multicast tree maintenance but gains data transmission efficiency from tree structure. In DDR, a node knows routes to all other nodes of the same zone. This information is periodically refreshed against topology changes. DMPDC uses a tree structure to connect all group members distributed in different zones. Therefore, the multicast tree (MTree)

---

[1] The average size of zone should be a function on some network parameters like average node degree, number of nodes in the network and some other parameters. Node degree is the number of nodes can be heard by a node.

[2]construction and maintenance can be decomposed into two parts: DDR deals with intra-zone routes, DMPDC copes with inter zone connections. Therefore, the routing message for MTree maintenance is greatly reduced.

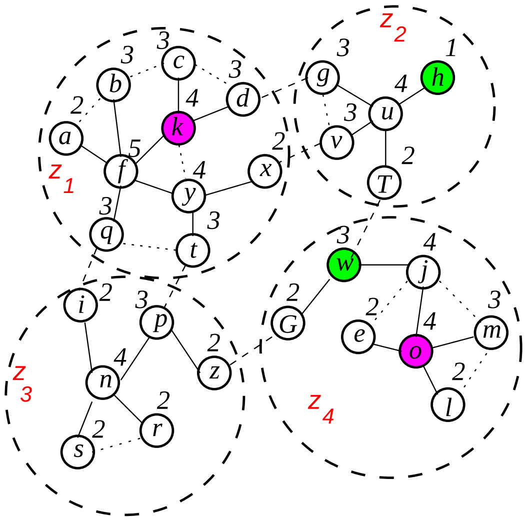
Fig. 3(a) shows four multicast group members distributed in different DDR zones. DMPDC constructs a MTree to connect them, as it is illustrated in Fig. 3(b). Fig. 3(c) represents the MTree in an abstract way. It shows that the vertices of DMPDC MTree are group members and gateway nodes. A MTree edge is either an intra zone route or a bridge.

DMPDC differs from other group-shared tree-based multicast routing protocols that also use the concept of *core* ([2], [14], [6], [8] and [13]). In our protocol, *core* is the first source of a multicast session. This choice guarantees that core is interested in participating in the multicast group and transmitting multicast traffic. If no core is present in the network, it is not necessary to construct and maintain MTree and all receivers remain silent. This property can be named as **on demand** compared to other group-shared tree-based multicast routing protocols. In those protocols, MTree should be maintained even when no multicast traffic is present. Another advantage of this choice is that DMPDC reduces to source-oriented in the case of a single source, and is group-shared in the case of multiple sources. Single point failure caused by core power exhaustion can also be overcome. When core finds that it has not enough power to rely the traffic from other sources of the same group, it can ask another source to take the role of core.

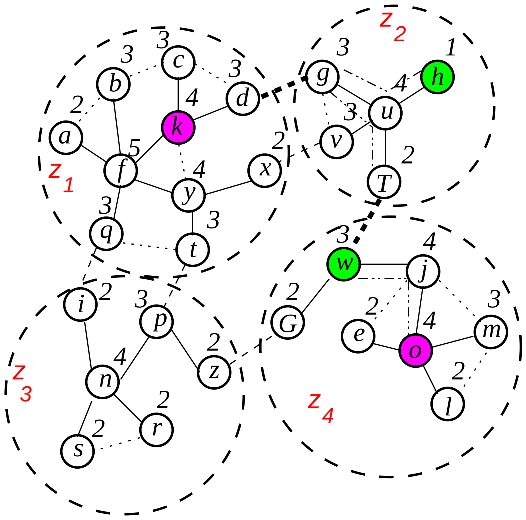
All tree-based multicast protocols use *timeout-based path monitoring* to detect link failures. A link failure is detected when a node has not heard its MTree neighbor during the last period. If a link failure is detected, downstream (the direction from root to leaf nodes) node rejoins MTree to maintain MTree connection. This mechanism reacts slowly to link failure according to the time period value. Consequently, data transmission is blocked until the branch is repaired. DMPDC proposes to use *reactive path monitoring* in which a MTree member supervises links when it forwards data to other next MTree member(s). Hence it can immediately discover a broken branch and react to this failure if necessary. This recovery allows downstream group members to hear and be heard until the next period arrives. We need periodical MTree refresh to construct better MTree branches than recovered ones. But the aim of periodical refresh in mesh-based protocols is for overcoming link failure. So, the period of tree refresh can consequently be much longer than that of mesh. Therefore, routing overhead caused by tree maintenance (topology change) is reduced because of on-demand local MTree branch recovery and a greater period of tree refresh. On the other hand, data delivery efficiency is improved by periodical tree refresh.

III. PROTOCOL DESCRIPTION

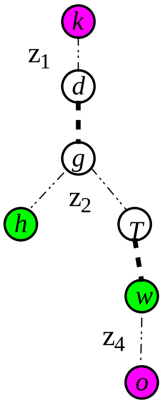
After forest construction, DDR establishes two tables for each node in the network: **intra-zone table** and **inter-zone table**. Intra-zone table keeps the information within a zone. By con-



(a) Multicast group members



(b) MTree constructed by DMPDC



(c) Abstract MTree

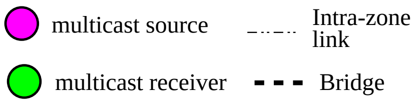


Fig. 3. DMPDC constructs a MTree in a multicast session

---

[2] In the rest of this paper, we use MTree to stand for multicast tree.

sulting this table, a node can know through which node it can get to another node in the same zone. For example, in Fig. 2(b), **intra-zone table** tells node a that the next hop to node d is node f. In contrast to intra-zone table, inter-zone table keeps the information concerning neighboring zones. One entry of this table represents a bridge to a neighbor zone (for more details, see [11]). A routing protocol works on the top of DDR would decide which bridge should be taken if node u wants to send data to node f in Fig. 2(b).

The control part of DMPDC consists of two aspects: **MTree construction** and **MTree maintenance**. MTree construction is the aspect by which a core is selected and advertised to the network. Nodes that are interested in the multicast session join the MTree. MTree maintenance is the aspect where MTree members detect broken branches and repair the failure to continue multicast traffic delivery in MTree. MTree maintenance takes also care of group members leaving. **Data forwarding** in DMPDC is somewhat different from that in other protocols. It needs traffic packets taking some extra information in their head to help routing.

A. MTree Construction

Each node in MANET possesses a *multicast routing table* (MRTable) which stores multicast routing information. The existence of an entry in the table, which corresponds to a multicast session tree, is the indication of the traffic presence in the group. A source can know whether it should act as core of this multicast session or just participate to the multicast tree as a normal source by checking if there is an entry. Similarly, a receiver can decide that it should join the tree or remain silent by examining MRTable. Routing entry has two states: active and inactive. A node that is tree member has an active entry and the entries of other nodes are inactive.

MTree construction is based on the following mechanism: a core broadcasts *Core Advertisement* (CA) message to the network which constructs reverse path to core at the same time. A node that is interested in the multicast group sends a *Route Active Request* (RAR) message towards core and waits for a *Route Active Acknowledge* (RAA). This procedure of joining is called **RAR/RAA** procedure.

A CA message contains multicast group address, core address and reference to identify CA. All other nodes in the network react only once to CA and discard all duplicates. They create an inactive route entry in their MRTable corresponding to the multicast session. Then, they construct a reverse path to core so that they can send or forward routing messages towards core. CA has a field to help reverse path establishment. Initially, this field is the node ID of core. The nodes in the same zone as core need just store core ID since they already have route to core in their intra-zone table. The gateway nodes that react to CA from a bridge modify this field by replacing it with the bridge information. The rest of the nodes in the zone know through this bridge that the core can be found. The advantage of using bridge instead of only a gateway node ID is that in case one bridge end is inaccessible, nodes can still try to contact the other bridge end or the corresponding neighbor zone to construct a path. This method can reduce the possibility of broadcast. Consequently, after all nodes in the network receive CA, they know the creation of a multicast session and own a route to the core of this session. The reverse paths constructed by this mechanism contain only gateway nodes and core.

For example, in Fig. 3(a), nodes h, k, o, and w are members of a multicast group. Among them, node k and o want to send data to the group and node h and w are the receivers of the group. Suppose node k is the creator of this multicast session. Node k broadcasts a CA(k) to the network. Nodes in z_1 store k into reverse path field of the multicast routing entry. When this CA goes through the bridge from node d to node g, node g changes the content of CA into (g, z_1, d). Upon receiving this modified CA, all the other nodes in z_2 know the core can be reached through the bridge between node g and node d. Therefore, node g becomes an upstream node in the reverse path to core of z_2. In the same way, node w forwards CA(w, z_2, T) to its zone (z_4) when it hears CA from node T. If node w moves out of z_4, the nodes in z_4 can still try finding a way to node T or z_2 to repair reverse path locally.

A RAR/RAA procedure begins by a group member sending a RAR towards core. This node becomes a potential MTree member. A RAR message is addressed to the upstream node to give the information of the potential downstream MTree member. The upstream node is either a gateway node or the core itself. The content of RAR is changed each time it passes a gateway node. Then RAR is forwarded to this upstream node along an intra-zone route or through a bridge. The addressed node becomes potential MTree member. This node should store potential downstream member information into its multicast route entry, update the content of RAR and then continue to forward this message towards the core by forwarding this message to its upstream node. A potential MTree member will not send another RAR when it receives any other RARs after having sent a RAR. But it records the information of these potential downstream members for confirming their membership as soon as RAA arrives. The first MTree member receiving RAR breaks off forwarding and replies RAA to active route entries of potential MTree members in the branch. RAA message gives the exact upstream MTree member. A RAR/RAA procedure is finished when the leaf node of the branch receives RAA.

So, in Fig. 3(a), if node h in z_2 wants to take part into the mulitcast session created by node k, it should send a RAR(h) towards core. This RAR is first sent to node g along intra-zone route. Upon receiving this RAR, node g becomes a potential MTree member. It registers node h into its MRTable as a potential downstream node of MTree member. It modifies downstream information by inserting its node ID and zone ID, which results in the content of RAR becoming (g, z_2, h). Then node g forwards this updated RAR to node d through the bridge. At last, node d sends RAR(d, g, z_2) to core. Core replies to node d with a RAA. This RAA is transmitted through node d and g until reaching node h so that the branch is added to MTree.

B. Tree Maintenance

During data transmission, a MTree member finds a branch broken when a MTree neighbor is in neither its intra-zone table nor its neighborhood. There are two cases for a broken branch: the branch is towards upstream MTree member or the branch is towards downstream MTree member. In the former case, the node sends a local broadcast RAR to find the upstream node and rejoin the MTree. This type of RAR propagates in the zone of the initiator and its neighbor zones. Only the MTree member specialized in RAR can reply RAA to avoid response coming from downstream MTree members of the initiator. In the latter case, the node sends a *Join Invitation* (JI) to invite the downstream MTree member to rejoin the tree. Upon receiving JI, the destination takes part in the tree by running a RAR/RAA procedure.

Core periodically computes a new reference and broadcasts CA to refresh MTree. These CAs also update reverse paths to core. All other group members run a RAR/RAA procedure to re-construct the MTree once they receive a CA.

Periodical MTree refresh gives multicast group members the possibility to leave the group implicitly. These members test first whether the next period of CA arrives soon. Then they do not run a RAR/RAA procedure so that the branches will be pruned silently. Otherwise, these members should explicitly leave MTree by sending a message to their upstream nodes. The procedure for a core leaving the group after finishing the transmission of data to multicast group is different from normal group member. It checks whether there is another source in MTree that can become new core. If this core is the only source, it dismisses the MTree. Otherwise, the new core is in charge of sending periodical CA.

Because of network partition or any other reason, a source may make a decision to become a core without hearing a CA from core. So there may be more than one core existing in the network. After the network converges, a core can hear the CAs of other cores. These cores use core competition algorithm to decide which source is the winner and it should continue acting as core. Cores can use their identification or other informations to compete. The loosers will stop sending periodical CA and not react to group member join packets so that theirs tree will be dismissed automatically after the multicast route entries time out.

C. Data Forwarding

Once the MTree is constructed or the branch is successfully added into MTree, a source begins to send traffic to the multicast group. This protocol needs traffic packets to carry two additional fields in their head: *last_member* and *next_member*. *Last_member* is the MTree member that forwards this traffic packet. This field tells a MTree member from which branch the packet comes, so that this member can forward it to all other branches. *Next_member* indicates to which MTree vertex this packet should be forwarded. If the MTree edge between *last_member* and *next_member* is intra-zone route, the intermediate nodes in the route use intra-zone tables to relay this packet to *next_member*. When a MTree member receives non-duplicate traffic packet from a MTree edge, first it records packet identification then it updates packet head. Finally, it forwards the packet to the other branch(es) of the MTree if the other end of the branch is reachable. Otherwise it will mark a broken branch. There is a great possibility that a packet should be sent to more than one MTree edge. The node duplicates the packet and sends them to corresponding directions.

IV. CONCLUSION AND FUTURE WORK

We have proposed a new multicast routing protocol, called DMPDC, for MANET. DMPDC is designed to find a tradeoff between data transmission and routing overhead so that multicast routing protocol can be scalable for long term and short term data delivery in both low mobility and high mobility situations. To reach this goal, DMPDC constructs and maintains a group-shared MTree on the top of an infrastructure offered by DDR and using a dynamic core. Core is the indication of traffic and a MTree exists only when there is traffic in a group. This protocol discovers MTree edge broken during data transmission. This allows MTree member to repair the failure immediately without blocking data delivery. All these mechanisms allow DMPDC to give a better utilization of radio resources and mobile node's power. A performance analysis will be carried out to make a study of DMPDC for different traffic loads (group size, number of sources per group, etc.) and mobility (speed, pause time, etc.).

REFERENCES

[1] T. Ballardie, P. Francis, and J. Crowcroft. Core based trees (cbt) - an architecture for scalable inter-domain multicast routing. In *ACM SIGCOMM'93*, pages 85–95, Oct. 1993.

[2] E. Bommaia, M. Liu, A. McAuley, and R. Talpade. Amroute: Ad hoc multicast routing protocol. Internet-Draft, Aug. 1998.

[3] S. Deering, D. L. Estrin, D. Farinacci, V. Jacobson, C.-G. Liu, and L. Wei. The pim architecture for wide-area multicast routing. *IEEE/ACM Transactions on Networking*, 4(2):153–162, Apr. 1996.

[4] S. E. Deering and D. R. Cheriton. Multicast routing in datagram internetworks and extended lans. *ACM Transactions on Computer Systems*, 8(2):85–110, May 1990.

[5] V. Devarapalli, A. A. Selcuk, and D. Sidhu. Mzr: A multicast protocol for mobile ad hoc networks. Internet-Draft, Nov. 2000.

[6] J. J. Garcia-Luna-Aceves and E. L. Madruga. A multicast routing protocol for ad-hoc networks. In *IEEE INFOCOM'99*, pages 784–792, New York, NY, Mar. 1999.

[7] L. Ji and M.S. Corson. A lightweight adaptive multicast algorithm. In *IEEE GLOBECOM'98*, pages 1036–1042, Sydney, Nov. 1998 1998.

[8] S. Lee and C. Kim. Mobile and ad hoc networking and computing. In *the First Annual Workshop on Mobile Ad Hoc Networking & Computing, MobiHOC 2000*, Boston, August 2000.

[9] S. J. Lee, M. Gerla, and C. C. Chiang. On-demand multicast routing protocol. In *IEEE WCNC'99*, pages 1298–1304, New Orleans, LA, Sep. 1999.

[10] J. Moy. Multicast routing extensions for ospf. *Communications of the ACM*, 37(8):61–66, 114, Aug. 1994.

[11] Navid Nikaein, Houda Labiod, and Christian Bonnet. Ddr - distributed dynamic routing algorithm for mobile ad hoc networks. In *the First Annual Workshop on Mobile Ad Hoc Networking & Computing, MobiHOC 2000*, pages 19–28, Boston, August 2000.

[12] Navid Nikaein, Houda Labiod, and Christian Bonnet. Synthèse sur les problèmes de routage dans les réseaux mobiles ad hoc. In *DNAC 2000*, Paris, Nov. 2000.

[13] E. M. Royer and C. E. Perkin. Multicast ad hoc on-demand distance vector (maodv) routing. Internet-Draft, Jul. 2000.

[14] C. W. Wu, Y. C. Tay, and C. K. Toh. Ad hoc multicast routing protocol utilizing increasing id-numbers (amris) functional specification. Internet-Draft, Nov. 1998.