Computer Networks 54 (2010) 41–56

Contents lists available at ScienceDirect

Computer Networks

journal homepage: www.elsevier.com/locate/comnet

HURP/HURBA: Zero-configuration hierarchical Up/Down routing and bridging architecture for Ethernet backbones and campus networks

Guillermo Ibáñez [a,*], Alberto García-Martínez [b], Juan A. Carral [a], Pedro A. González [a], Arturo Azcorra [b,c], José M. Arco [a]

[a] Universidad de Alcalá, Escuela Politécnica Superior. Campus Universitario, N-II Km 33,6. Alcalá de Henares, 28871, Spain
[b] Universidad Carlos III de Madrid, Avda, de la Universidad, 30. E-28911 Leganés (Madrid), Spain
[c] IMDEA Networks, Avenida del Mar Mediterráneo, 22 28918 Leganés (Madrid), Spain

ARTICLE INFO

Article history:
Received 22 July 2008
Accepted 11 August 2009
Available online 15 August 2009
Responsible Editor: I.F. Akyildiz

Keywords:
Routing
Computer networks
Protocols
Up/Down routing
Turn prohibition
Cycle-breaking
Ethernet

ABSTRACT

Ethernet switched networks do not scale appropriately due to limitations inherent to the spanning tree protocol. Ethernet architectures based on routing over a virtual topology in which turns are prohibited offer improved performance over spanning tree, although in some cases suffer from excessive computational complexity. Up/Down routing is a turn prohibition algorithm with low computational complexity. In this paper we propose HURBA, a new layer-two architecture that improves Up/Down routing performance due to an optimization based on the use of hierarchical addressing, while preserving the computational complexity of Up/Down. The resulting architecture requires zero-configuration, uses the same frame format as Ethernet, allows upgrades by software update, and is compatible with 802.1D bridges by means of encapsulation. HURP protocol builds automatically a core with the interconnected HURP routing bridges and the standard bridges get connected to the edges in standard spanning trees. Simulations show that the performance of HURP, evaluated over various combinations of network topology and size, is close to the one of shortest path, is consistently better than that of Up/Down, and is equal or better than Turn Prohibition, with the advantage of having a lower complexity.

© 2009 Elsevier B.V. All rights reserved.

1. Introduction

Ethernet is expanding in backbone and campus networks due to its excellent price/performance ratio, configuration convenience and backward compatibility. The use of transparent learning bridges with spanning tree protocols allows loop-free operation with zero configuration, without requiring complex routing information, IP address and segment planning and configuration or a hop-by-hop modification of the header of the frame (as occurs with the Time-to-Live field of the IP packet).

However, spanning tree protocols also limit severely the scalability and performance of Ethernet networks because they block all links exceeding the number of network nodes minus one. The scalability limitation of the spanning tree protocol derives from two factors: low link utilization and vulnerability of the bridged domain to network failures and configuration errors. Network infrastructure utilization is low because the loop prevention mechanism of the spanning tree protocol relies on the activation of just a subset of the available links. As a consequence, the resulting routes along the spanning tree between two arbitrary hosts are not pair-wise shortest paths due to the low connectivity. Vulnerability exists because of hardware failures or configuration errors, that may produce broadcast storms and even network meltdown of the switched domain. To prevent this, IP routers are deployed to segment the network and limit the size of bridged domains. The drawbacks of the use of IP are the requirement of proper IP address

* Corresponding author. Tel.: +34 918856927; fax: +34 918856641.
E-mail address: guillermo.ibanez@uah.es (G. Ibáñez).

and segment configuration, the restrictions imposed to host mobility inside the network, and the need for IP routing configuration.

It would be desirable to develop a new Ethernet architecture that could combine the features of standard bridges and routers, while avoiding the limitations of the spanning tree protocol. Such architecture should fulfill a number of requirements: it should scale to large networks (up to 20,000 bridges or more) with adequate network segmentation to limit damages, should require low configuration (ideally, zero-configuration), and it should preserve transparent operation regarding to hosts and routers.

Two basic approaches are being proposed in the literature to overcome these limitations: routing bridges [8] and VLAN-based multiple spanning tree approaches [20]. On one hand, routing bridges, that are bridges that perform routing at layer-two, suffer from the limitations of the use of a flat address space for hosts and bridges. Furthermore, RBridges [7,8], one particular instance of the routing bridge proposals, uses a non-Ethernet additional encapsulation with a TTL field to prevent loops. On the other hand, the main problem with the use of VLAN-based multiple spanning trees is the high configuration complexity and limited number of spanning tree instances.

A third approach is the prohibition of certain *turns* in the topology, instead of the link prohibition that occurs for typical spanning trees. Therefore, the decision of which links forward frames that must been broadcasted depends on the label associated to the turns defined among two links connected to a node. A $turn(a, b, c)$ around a node b is defined as the pair of links that join b with other two nodes like a and c. If the turn (a, b, c) is prohibited, packets arriving at node b from link $a - b$ cannot be forwarded to link $b - c$. When some turns are prohibited in a network topology, it can be assured that loops will never occur. Compared to link prohibition strategies, turn prohibition provides increased link utilization. It has been proven for some algorithms that the restriction of less than a fraction of 0.33 of total turns guarantees loop-free topologies without blocked network links [2]. Note that the spanning tree protocol prohibits typically in the range of 0.7–0.9 of turns. The prohibition of turns was first proposed in Autonet [1], through the approach known as Up/Down routing, and since then other proposals have appeared, such as Turn Prohibition [2] and Tree Based Turn Prohibition (TBTP) [10]. The main drawbacks of the Up/Down routing proposal are the requirement for additional hardware and software in hosts and the non miscibility with standard bridges. The main drawback of TP and TBTP is its limited scalability due to the computational complexity of the conceptual approach, based on a complete knowledge of topology to select a near optimum set of prohibited turns that prevent loops. Up/Down is selected as a base due to its inherent simplicity that permits local election of prohibited turns once the spaning tree has been built.

In this work we aim to improve the performance of Up/Down without resigning to self-configuration, full distributed operation and sufficient compatibility with standard bridges. The improvement in performance results from the ability to permit some extra turns that would be prohibited in Up/Down, by considering the topological information embodied into the hierarchical address. Additionally, we want to preserve the standard Ethernet frame format and transparency to hosts and routers. Keeping the functionality simple would permit legacy bridges to be upgraded with just software updates, The architecture defined to achieve these goals is named HURBA (Hierarchical Up/Down Routing and Bridging Architecture), in which we propose three novel components.

The first component is a mechanism to assign fixed-size hierarchical addresses, usable as local MAC addresses, to HURBA bridges. The design presented in this paper modifies the proposal previously made by the authors [3] to define fixed length addresses that fit in the standard Ethernet frame format. These “private” MAC addresses, Hierarchical Local MAC addresses (HLMAC), are distinguishable from global MAC addresses by their local/global bit set to the local value. This split of the addressing space allows the coexistence of standard, globally unique MAC addresses with HLMAC addresses. The Combined Spanning Tree Protocol (CSTP) is a variation of RSTP used to assign hierarchically the HLMACs to the bridges. HLMAC addresses are used as ordered node identifiers by the improved Up/Down mechanisms to break cycles [18]. They are also used by the distance vector routing protocol for cross link routing and optionally for direct (i.e. neither broadcast nor MAC address learning-based) forwarding through the spanning tree links through destination address decoding.

The second component is HURP (Hierarchical Up/Down Routing Protocol), an enhanced Up/Down routing algorithm and protocol that establishes shortest path routes that respect turn restrictions based on distance vectors with topologically significant node identifiers. Compared to Up/Down, HURP enables turns that would be prohibited by Up/Down, but are known to end up in the branch of the spanning tree to which the destination belongs.

The third component is an extension of the Rapid Spanning Tree Protocol (RSTP) as a core building protocol that auto configures a core of HURP bridges, where the spanning trees of 802.1D bridges get attached. The result is a zero-configuration architecture, compatible (in island mode) with standard 802.1D bridges.

The remainder of the paper is organized as follows: Section 2 introduces the fundamentals of the turn prohibition paradigm, with special attention to Up/Down routing. Section 3 describes the HURBA architecture. First, the structure of the hierarchical address and its assignment is discussed, then the Combined Spanning Tree Protocol (CSTP) used to actually assign the address is discussed. Next, the HURP protocol is presented from a control plane/user plane perspective and the computational complexity of the combined operation of the HURBA components is analyzed. We finish the description of the HURBA architecture discussing the mechanisms that enable compatible operation with legacy 802.1D devices. Section 6 contains the simulation results that characterize the performance of the routing configuration resulting from HURBA operation, in terms of number of turns prohibited, throughput, and path length. The next section presents related work, and Section 8 the conclusions.

2. Up/Down routing in Ethernet networks

In this section we describe the basis of UP/Down routing algorithm based on prohibiting the down-up turns using the node identifiers, which is seminal for the definition of HURP.

Fig. 1a and b show, respectively a complete network and its spanning tree active topology. Fig. 1c shows the Up/Down principle.

Consider a network modelled as a directed graph G, composed of nodes and bidirectional links [2]. A $(n1, n2)$ pair describes a link from node $n1$ to node $n2$. A *path* is a sequence of nodes successively connected by links so that each two subsequent nodes are connected by a link. In opposition to graph theory, a *cycle* in a path occurs when the first *link* and the last *link* of the path are the same, instead of requiring the first and the last *nodes* to be the same. Therefore, a node may be visited repeatedly without creating a cycle. The *degree* of a node is the number of links connecting the node to neighbour nodes.

In Fig. 1, the path *4-3-1-2-4-6* does not contain a cycle: although node 4 is visited twice, no link is traversed twice. A *turn* is defined as a pair of input–output links around a node. The tuple (a, b, c) represents the turn at node b from link $a - b$ to link $b - c$. In Fig. 1 the turn $(3, 4, 2)$ is the turn from node 2 around node 3 to node. Unless otherwise stated, the turns are symmetrical by default, so the turn (a, b, c) is identical to the turn (c, b, a). Note that the total number of possible turns around a node of degree d is $d(d - 1)/2$.

Suppose a spanning tree $T(G) = (V_T; E_T)$ being V the vertices and E the links, is built providing connectivity to a graph G. Links belonging to this tree are referred to as *tree links*. Other links are referred to as *cross links*. The goal of an Up/Down or turn prohibition algorithm is the construction of a cycle-breaking set of prohibited turns $S_T(G)$ that breaks all cycles in the graph.

Up/Down is the simplest approach for the construction of such a cycle-breaking set $S_T(G)$. The spanning tree $T(G)$ is used to assign an identifier that increases with the distance to the root node. Nodes at the same level are ordered according to bridge ID. Once the identifiers are assigned to the nodes, a link (a, b) is considered to go "up" if $a > b$. Conversely, if $a < b$, it is said that link goes "down". A turn (a, b, c) is referred to as an *up/down* turn if node if $a > b$ and $b < c$. If $a < b$ and $b > c$ the turn is *up/down*. For a cycle to occur, it must involve at least one up/down turn and one down/up turn. By prohibiting all the down/up turns, cycles are broken in the topology and all nodes can be reached. An illustration of the Up/Down algorithm is provided in Fig. 1c, being node 1 the root bridge.

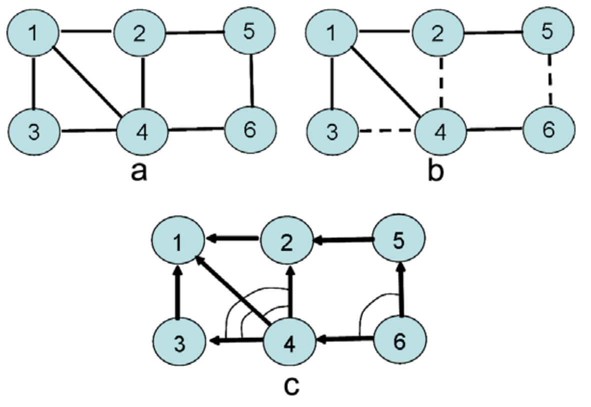


Fig. 1. (a) Original graph, (b) spanning tree, and (c) Up/Down showing down-up (prohibited) turns according to node identifier values.

At Fig. 1c the prohibited turns by Up/Down protocol using the down-up criteria will be (3-4-2), (3-4-1), (2-4-1) and (4-6-5) because the identifier of the node in the middle of tuple is higher than the other two. With the identifier assignment criteria, the "up" part of a link is the end that is closer to the root bridge, so this assignment forbids in particular paths that descend in the spanning tree and use another branch of the spanning tree to ascend. Note that the identifiers used by Up/Down routing are flat and ordered, but do not convey other tree related topological information.

Other algorithms like Turn Prohibition (TP) and Tree Based Turn Prohibition (TBTP) process the complete network topology by selecting iteratively the candidate nodes to prohibit turns around them. The criteria to assign the turn prohibitions can vary, being one example the minimum node degree (TP). To show the effectiveness of selective prohibition of turns to prevent cycles, just prohibiting the turn 2-4-3, prevents cycles in 1-2-3-4. To prevent all cycles in Fig. 1, many combinations exist. One possibility is to prohibit all turns around node 1, namely turns (3,1,4), (4,1,2), turn (3,1,2), and around node 5, the turn (4,5,6). Note that each prohibited turn prevents one or several cycles. Prohibited turn around node 5 breaks two cycles: 1-2-5-6-4-3 and 2-5-6-4. Care must be taken, however [2], to avoid disconnection of the graph. TP and TBTP are complex because they require the knowledge and visitation of the complete topology to minimize the set of prohibited turns. Therefore, we choose Up/Down as a starting point for HURBA by its low complexity and because it only requires the construction of a spanning tree. Once the identifiers are assigned, each node may enforce at once down up turn prohibition just by comparing its identifier with the identifiers of its neighbors.

Besides prohibiting turns, routing information (excluding routes forbidden by the turn prohibition mechanism), can be exchanged over the links to minimize path length between nodes, obtaining the shortest cycle-free routes.

3. Hierarchical Up/Down routing and bridging architecture

3.1. Bridge address structure and assignment

We have seen that Up/Down routing assigns identifiers to nodes according to distance to the root bridge to assign direction to links. HURP assigns hierarchical local MAC addresses to bridges and uses them as node identifiers for up/down routing. We will discuss below that the use of hierarchical addresses enables improvements in performance and operation compared to the Up/Down proposal. Unless otherwise stated, we will consider that these addresses are only assigned to bridges, although the addressing method is applicable to the hosts and routers connected to bridges as well. We now describe a mechanism to automatically assign Hierarchical Local MACs (HLMAC) to bridges by means of a variation of RSTP that assign identifiers to bridges according to distance to the root bridge. We will show later that it is not required to involve the hosts in the hierarchical addresses assignment to assure the prevention of cycles. HLMACs are also used by the distance vector routing protocol for cross link routing and for direct forwarding through the spanning tree links through destination address decoding when routing tables are not available.

Fig. 2 shows the address assignment principle. Root port is the origin, has no hierarchical address. R assigns address 1 to bridge 1 because designated port 1 is connected to that bridge (point to point are links assumed, as with RSTP). Bridge 1 assigns address 1.3 to that bridge because it is connected at designated port of bridge 1, and so on.

In order to ease coexistence of these addresses with legacy equipment, HLMACs are defined as a subset of the existing MAC address space. HLMACs are defined as 48-bit standard format MAC addresses in which the bit 1 of byte 0 is set to 1 to indicate that the address is local. The rest of addresses with the bit 1 of byte 0 set to 0 are the usual MAC addresses, flat and globally unique. Bit 0 of byte 1 indicates that address is individual or (multicast) group address. The remaining 46 bits are structured into 6 hierarchical levels in which the first level has 6 bits, and the other 5 have 8 bits. These levels correspond to the position of the bridge in the spanning tree. HLMAC addresses are interpreted as variable-length address conveyed in the standard 6 byte length field. The first level (from left to right) containing all zeroes indicates the end of address.

To stress the hierarchical nature of the HLMAC addresses we use for their notation a different separator, '—', between each 8 bit block of the address, coded in hexadecimal. HLMAC addresses are assigned hierarchically following the path defined by a spanning tree, from the root to the edges. The root bridge is assigned with address 40—00—00—00—00—00 (all zeroes except for the local bit). In the rest of the paper we represent only the 46 bit of HLMAC address and omit bits 0 and 1 for the sake of clarity. As explained above, at each stage, the address of a B bridge connected to a bridge A that precedes B in the spanning tree is obtained by inserting the number of the port at A that connects to B in the first hierarchical level with null value. In other words, an address a—b—c, expresses the chain of designated port IDs a, b, c, traversed in the descending path from the Root Bridge till the root port of that bridge. Please remember that an address represents both the address of bridge and the subtree rooted at that bridge. Fig. 3 illustrates the HLMAC address assignment: The root bridge assigns to D1 the HLMAC address 32 by appending 32 (port ID of designate port) to the null value, D1 appends 7 and assigns 32—7 to D2 and so on. Port numbering starts at value 1, not 0, to distinguish the end of the address.

The maximum depth (number of levels) of the address is 6 for the default (implicit) format with 8 bits (up to 255 active ports per switch with the exception of up to 63 ports for the first level). When a switch is assigned an address of maximum depth it will not assign further addresses through its designated ports and will act as a HURP edge switch, thus setting the limit of the HURP core mesh. In section E below we describe the process in more detail.

3.2. Combined Spanning Tree Protocol (CSTP)

The Combined Spanning Tree protocol is an extension of the RSTP protocol to benefit from its advantages and preserve compatibility with STP and RSTP. It builds a spanning tree with all the interconnected HURP capable bridges and assigns to each one a HLMAC addresses. The assignment of HLMAC addresses to the bridges is piggybacked on the exchange of information required to build the spanning tree by the RSTP Protocol from root node to designated nodes. Therefore, the CSTP BPDU is a RSTP BPDU extended with a field containing the HLMAC (6 byte) address assigned to the bridge connected to the designated port that is emitting that BPDU. These BPDUs, containing RSTP information plus the HLMAC address assigned to the receiving bridge, are exchanged periodically, every *Hello_Time* by every bridge with its neighbors. A bridge HLMAC address emitted in a BPDU by a designated port is considered stable and assigned when the root port connected to the designated

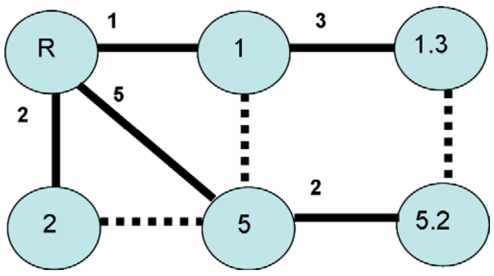


Fig. 2. Hierarchical address assignment principle.

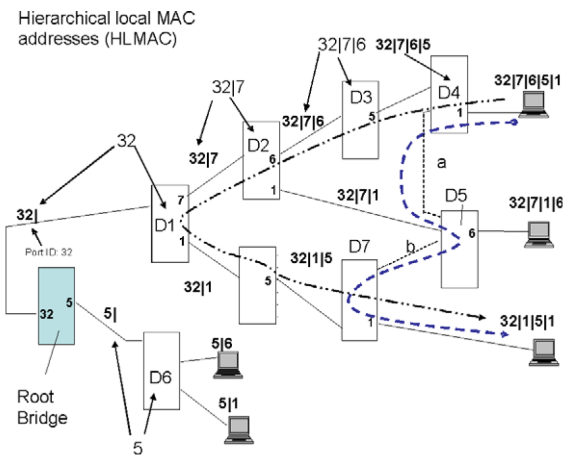


Fig. 3. Assignment of Hierarchical Local MAC addresses based on spanning tree. -..Tree forwarding. —Transversal forwarding.

port is enabled. As with RSTP, this occurs when the port transits to forwarding state. This situation is known at once by the root port bridge of the HURP bridge getting its HLMAC assigned.

In the Protocol Version Identifier field, a new value is used to distinguish whether the bridge is executing the standard STP, RSTP, or the new CSTP protocol. Using a similar mechanism to the STP/RSTP protocol migration dialog between 802.1D standard bridges [4], HURP capable bridges listen to the protocol version identifier at the other end at each port and execute accordingly to the highest functionality protocol common to them: STP, RSTP or CSTP. Bridges that have no connection to the CSTP spanning tree default to standard bridge (STP or RSTP) as described below.

4. HURP protocol

The HURP (Hierarchical Up/down Routing Protocol) is a hierarchical distance vector routing and forwarding protocol. HURP exchanges distance vector information over a virtual topology built using enhanced Up/Down for bridges that have been assigned HLMAC identifiers. HURP allows routing through cross links, (as long as turns are permitted), otherwise blocked by the spanning tree protocol, while preventing frame loops just by turn prohibition of the down-up turns. HLMAC addresses are also used to forward frames along the branch of the spanning tree in which the destination is located by direct decoding of the ports to select, that are coded in the destination address, without neither routing tables nor MAC address learning, when no routes are available.

4.1. HURP Control plane

HURP exchanges routes between bridges using an enhanced distance vector protocol. Messages and operation are similar to those of the RIP protocol. The metric can be the hop count as in RIP or, as in 802.1D, inversely proportional to the link speed. At each node, the HURP protocol selects routes through cross links when their cost is equal or better than cost via tree links. Every bridge transmits to its neighbours the known shortest distance routes that do not use a prohibited turn (down-up), i.e. routes that would contain a down-up turn at the announcing node are filtered. Each bridge builds a distance vector with the best available routes obtained from neighbors using the Bellman–Ford algorithm. The routes learnt from each neighbor are also excluded from the announcement to that neighbor (split horizon). Note that the usage of a loop-free underlying virtual topology below the distance vector information exchange precludes the occurrence of the count-to-infinity problem.

As shown in Fig. 4 above, HURP provides a significant enhancement in performance over the Up/Down mechanism for prohibiting turns because it can take advantage from the topological information contained in HLMAC addresses in the following way: a turn that should be prohibited, can be permitted if the turn ends up at a bridge belonging to the *destination branch*. A bridge belongs to

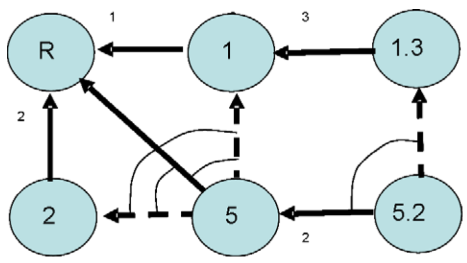


Fig. 4. HURP: enhanced Up/Down with hierarchical addresses. Prohibited turns shown will be permitted for routes that reach its destination branch in that turn (have common prefix).

the *destination branch* of a frame traversing it if the Bridge HLMAC address is a prefix of the frame's destination HLMAC address, or vice versa (the HLMAC is a prefix of the Bridge HLMAC). In other words, the destination bridge HLMAC and the next hop bridge belong to the same branch (i.e. have a common prefix). This mechanism is illustrated with an example: Suppose node 5.2 belongs to branch of node 5. In Fig. 4 frames with destination node 1.3 may execute turn at node 5 toward node 1 because node 1 belongs to destination branch. The same occurs for the turn 5-5.1–1.3. Both of them are down-up turns and would be prohibited by Up/Down without topological identifiers.

The turns that reach the destination branch can be permitted because, once a frame arrives at any point of the destination branch, it is guaranteed that the frame will reach the destination bridge, without cycles, just descending or ascending through the destination branch. This is so because the remaining path to the destination is a shortest path because both transited and destination bridges are located in a branch of the spanning tree. Since the spanning tree is built selecting shortest paths to root bridge, fractions of these paths are also shortest paths.

It is worth to note that an additional advantage of using hierarchical identifiers is that routes may be aggregated by nodes and a single identifier announced when routing information is exchanged. Therefore, the number of advertisements can be greatly reduced (at the cost of loss of routing accuracy), so that a higher number of nodes could be served.

A link or bridge failure may cause reconfiguration in the spanning tree and result in changes in active topology. Reconfiguration in HURP protocol follows similar rules than RSTP [6] and produces the same effects regarding port states. The main difference is that in RSTP the learnt MAC addresses are flushed while in HURP the assigned HLMACs are deleted as bridge addresses and routing table at every HURP bridges is erased after receiving the topology change notification. Forwarding of frames with HLMACs is immediately stopped at ports that lose their valid HLMAC address until new HLMAC addresses are assigned (i.e. until spanning tree branches are reconfigured). HLMAC addresses might appear volatile due to their dependency of the spanning tree. However, it must be taken into account that, unless a root bridge failure occurs, only a fraction of addresses will be affected in case of reconfiguration. Besides this, the fast reconfiguration capabilities of RSTP [13–15] minimize unavailability of HLMAC routing.

4.2. HURP user plane

Forwarding with HURP can operate in two modes:

– with shortest paths via both tree and non-tree (cross) links using routing tables built by the vector distance protocol, or
– by using exclusively the information encoded in the hierarchical destination address.

In this latter case, the frame is sent in the ascending direction trough the spanning tree via root ports until the frame reaches the common node to destination and source tree sub-branches and descending by destination address decoding, without routing tables. If origin and destination addresses are in separate branches, the frame ascends until the root bridge and descends afterwards through destination branch. This forwarding mode does not use routing tables, only needs stable HLMAC addresses, and seems suitable also for very high speeds switches by avoiding table look up delays. The preferred forwarding mode is one based in distance vector shortest paths, and the alternative through the spanning tree should only be configured when performance requirements or other reasons preclude the use of routing tables.

The complete and detailed HURP forwarding algorithm is shown at Fig. 5 below and forwarding examples at Fig. 3 above. It operates as follows: if the HLMAC of the bridge being traversed is included into the destination HLMAC (i.e. the destination HLMAC is longer), the frame is forwarded downwards the tree using the output port identified by the first octet exceeding this bridge HLMAC. If the destination HLMAC is included into the bridge HLMAC, the frame is forwarded through the root port. The root port is also the default route if there are no alternative routes. This is shown in Fig. 3 by dotted line.

Forwarding in shortest path mode through cross links makes use of the routing tables constructed with the interchange of distance vectors. Routing may be performed on an exact match of destination address or on a prefix matching basis. This allows the aggregation of routes and shortens routing tables. Fig. 3 shows, with a discontinuous line, the route followed by a frame from an originating terminal with HLMAC address 32.7.6.5.1 until destination host with address 32.7.1.5.0. The first leaf bridge 32.7.6.5 has a shortest path route through intermediate bridge 32.7.6.1 and forwards the frame through cross links *a* and then *b* until destination F 32.1.5

4.3. Computational complexity of HURBA mechanisms

The worst-case computational complexity of HURP is polynomial in the number of nodes N, more precisely $O(Nd^2)$, where d represents the maximal degree of any node in the network, as it is reasoned next:

1. Building the spanning tree with CSTP: CSTP (and in general any spanning tree protocol) is basically a distance vector protocol in which the distance is computed to one node, the root bridge. Distance vectors to the root are interchanged between neighbour nodes. The complexity is $O(Nd)$.
2. Address assignment: Addresses are assigned by CSTP based on the spanning tree ports. Each node assigns d addresses in the worst case, so the overall complexity is $O(Nd)$.
3. Turn prohibition: The prohibition of turns is computed comparing values of the hierarchical node addresses. Since at each node, $d*(d-1)/2$ possible turns are evaluated for prohibition, the complexity is $O(d^2)$ per node, and $O(Nd^2)$ for the computation of prohibited turn at all nodes.
4. Shortest path route computation: Routes are computed with a distance vector protocol. At each node a distance vector is computed and transmitted to neighbour nodes, eliminating from the announcements the routes

HURP forwarding algorithm

```
 # Check if destination HLMAC and this bridge HLMAC have some prefix in common (i.e belong to same branch)
CommonAddressPrefix= CommonPrefix(DestinatAddressHLMAC,  ThisBridge.HLMAC)
IF CommonAddressPrefix  ≠  null ; destination host is connected at same tree branch.
AddressSuffix =  DestinatAddress.HLMAC  XOR ThisBridge.HLMAC
IF length (DestinatAddress.HLMAC) < length (ThisBridge.HLMAC); destination host is up in the tree
output interface = root port   ; root port is selected, to forward frame  up in the spanning tree
FI
 IF length (DestinatAddress.HLMAC) > length (ThisBridge.HLMAC); destination host is down  the tree
forwarding port = (DestinatAddress.HLMAC XOR ThisBridge.HLMAC) AND OctectMask  ;
forward frame via the designated port, coded in 1st octet of suffix
 FI
IF length (DestinationAddress.HLMAC) = length (ThisBridge.HLMAC); destination is this bridge
 Extract frame;tear off HLMAC header and CRC to  decapsulate original frame  (replace HLMAC by GMAC if MAC NAT
used)
 Deliver frame ; deliver 802.1D frame to standard bridges.
FI
Else ;  destination host is in a different branch
IF  ForwardingTable (destination) =/  null ; there is a route (via cross links or tree links)
output interface = ForwardingTable (DestinationAddress.HLMAC) ; route frame via table
Else ; there is no HURP route, frame is routed via spanning tree.
output interface  = root port   ;  forward frame via root port of this bridge, up in the spanning tree
FI
```

Fig. 5. HURP forwarding algorithm.

that would result in a prohibited turn around the announcing node. A maximum of d vectors are received at each node. Complexity is the same as with Bellman–Ford, i.e. $O(Nd)$.

Note that steps 1 and 2 are triggered by a change in the spanning tree changes, i.e. the addition or deletion of a node, or the addition or deletion of a link belonging to the spanning tree. Turn prohibition computation is performed on each node when its own address changes, a new neighbour appears, or any neighbour changes its address due to RSTP reconfiguration. Finally, shortest path route computation is performed each time the distance vector information exchanged periodically changes.

5. Compatibility with standard bridges

Two mechanisms for interoperability with standard 802.1D bridges and auto configuration have been devised for HURBA: the automatic construction of the HURP core by all directly connected HURP bridges surrounded by separate and standard spanning trees rooted at the edge HURP bridges, and the encapsulation of frames entering the HURP core from 802.1D sub networks.

5.1. Core and trees construction

Fig. 6 illustrates the process of core construction by CSTP protocol in a mixed network environment with HURP and standard bridges. In a) the initial topology, consisting of HURP capable bridges interconnected with 802.1D bridges is shown. CSTP operates as RSTP building the HURP spanning tree, but only the ports connected to other HURP bridges connect to the HURP core tree. The ports connected to standard bridges execute the standard RSTP protocol and behave as edge bridges. These HURP bridges participate in the standard spanning tree and not in the HURP core spanning tree. They announce at their 802.1D ports a high priority value to be best candidates to become elected as root. If more than one edge HURP capable bridge (like R and B3 in Fig. 6) the one with lower bridge ID is elected as root of spanning tree of the standard bridges connected to them. Additional HURP capable bridges not being elected root will block their links to that tree. Isolated HURP capable bridges, disconnected from the HURP root bridge, like B4, default to standard 802.1D operation.

5.2. Encapsulation

The default operation of the protocol requires the encapsulation of the Ethernet frame entering the HURP bridges core. In this case the ingress frame with global MACs from hosts as source and destination addresses gets encapsulated with an outer frame header containing the HLMAC addresses of source and destination edge HURP bridges close to the destination, to allow HURP frame forwarding in the core mesh. When the source address is a global MAC, the external header includes as source address the HLMAC of the HURBA bridge through which the frame has accessed to the HURBA core, to allow the destination to relate for response frames the remote source MAC address with the remote source HLMAC address of the closest bridge. In any other case, the HLMAC appears in both the internal and external header. Frames arriving to the HURBA core that are addressed to global MACs for which the bridge does not have a mapping to the corresponding HLMAC are encapsulated with its original global MAC and broadcasted through the spanning tree of the core. In the core bridges, forwarding of frames containing global addresses at HURP bridges is performed according to standard 802.1D rules.

Consider an example in the scenario depicted in Fig. 7: H1 and H2 are global MACs, and B1 and B2 HLMACs. Host H1 sends a unicast frame to B1 with origin global MAC of H1 and destination global MAC H2. B1 receives the frame and looks for a HLMAC associated to the destination global MAC but it does not find it. Then, it encapsulates the frame

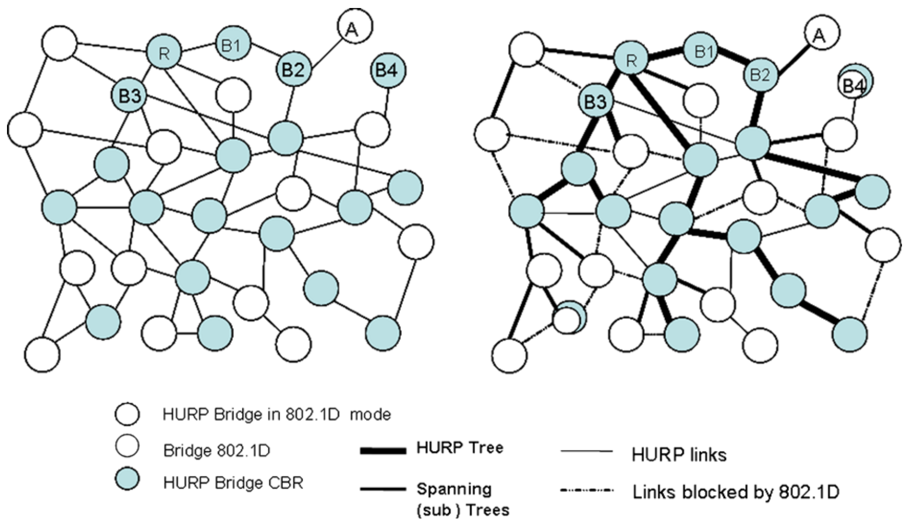


Fig. 6. Autoconfiguration in combined topologies: (a) Arbitrary topology (HURP and legacy bridges), and (b) topology after autoconfiguration.

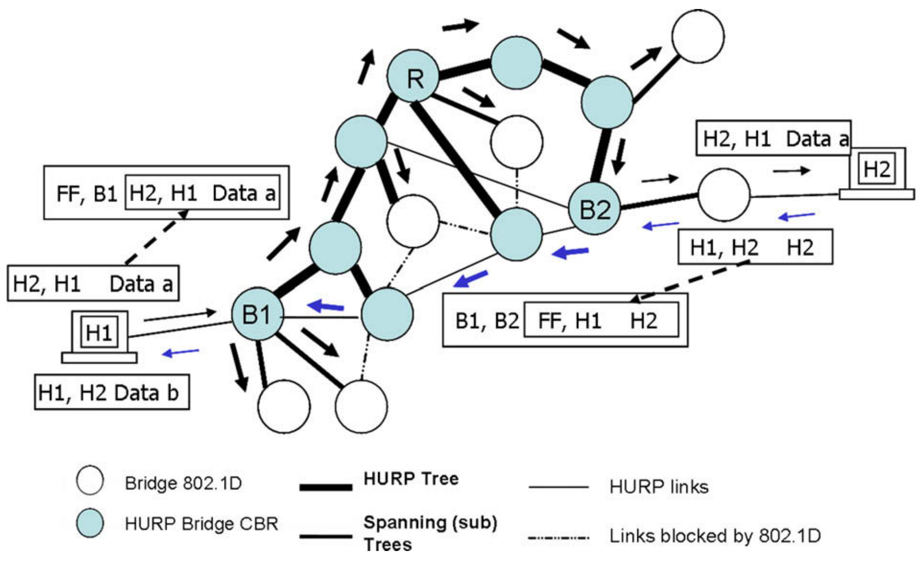


Fig. 7. Encapsulation and dual (edge bridge and host) MAC learning.

on a header with the broadcast address as destination, and with the HLMAC of B1 as source address. The frame is forwarded through the spanning tree built by CSTP. The frame arrives at all edge nodes, including B2, and B2 learns the B1-H1 association, it decapsulates frame and forwards the original frame to H2. The destination node H2 responds to H1 with a unicast frame and destination address H1 and source address H2. B2 encapsulates the frame with destination address B1, and origin B2. B2 reads the route table for finding the next hop to B1 destination and selects shortest path through a cross link. The same process is performed at each HURP bridge in the core till B1 is reached. Finally, B1 receives the frame, removes HURP header and delivers frame to H1.

6. Performance evaluation

In this section we perform simulations to compare the performance of HURBA with other protocols such as shortest path routing, spanning tree and the de facto reference for turn prohibition algorithms, the Up/Down algorithm. However, conditions similar to papers presenting performance results for TP and TBTP are also simulated to allow a comparison with these protocols. The main performance parameters to evaluate are throughput and path length. The fraction of prohibited turns to total turns is also evaluated, in order to provide an insight on throughput and path length results.

6.1. Methodology

We use flow level simulations to compare the throughput of the algorithms in relation to properties of network topologies like network size, average degree of nodes and degree distribution function. Random topologies of varying size with fixed node degree, average node degree (Waxman) and Barabasi–Albert distribution (*scale free*) are evaluated [12]. All links are assumed to have equal capacity, and the cost is 1 per hop. As said before, we include topologies and conditions similar to those in [3,6] to facilitate comparison with the performance evaluation of the TP and TBTP protocols presented in those references. The high variability in performance of the classic Up/Down protocol depending on specific topologies [16], or on the root bridge elected, has been mentioned before [2]. To evaluate the impact of the topology, we evaluate HURBA and the rest of the protocols with very different topologies, regular and random, with fixed and different distributions of node degree. To evaluate the effect of the bridge elected as root, we repeat each simulated scenario varying the elected root so that every bridge of the topology is elected, and we obtain the mean of the results.

For each graph generated the spanning tree is computed and hierarchical identifiers are assigned. Prohibited turns are computed for Spanning Tree, Up/Down and HURP. Routes are computed with the corresponding STP, Up/Down and HURP protocol restrictions, and without prohibited turns, to be able to compare with shortest path. The metrics evaluated are: Fraction of prohibited turns, average path length and maximum throughput. The fraction of prohibited turns and the path length are obtained from the resulting graph, while for the throughput estimation we use a flow model that is described in section C.

6.2. Fraction of prohibited turns

We have evaluated the number of prohibited turns for different types of random topologies. Parameters like number of nodes, average node degree, and node degree distribution, have a relevant impact in the performance of turn prohibition algorithms. To evaluate these effects we first simulate a variety of random 120 node topologies with a random (Waxman) distribution of node degrees and another set of 120 node topologies with fixed node degree.

Then we simulated topologies of varying sizes (16, 32, 64, 128 nodes) with Waxman, power law (Barabasi–Albert) and fixed node degree. Due to the low performance of the spanning tree protocol, it is not included in all comparisons to enhance the precision of graphic comparisons of HURP with Up/Down.

6.2.1. 120 node Waxman topologies

We evaluated a series of 120 node Waxman model random topologies with different values of average node degree with random degree distribution. For each average node degree, 40 topologies were generated with the BRITE tool [12] (intra AS level, Waxman model and default parameter values) and evaluated. Fig. 8 shows the absolute maximum and minimum and average fraction of prohibited turns for spanning tree, Up/Down and HURP. HURP performs between 5% and 10% better than Up/Down in all cases.

6.2.2. Fixed node degree topologies

Random topologies of 120 nodes with fixed node degree (all nodes equal degree), were generated and simulated, with 40 topologies per degree, with the results shown in Fig. 9. The ratio of prohibited turns is worse than Waxman random topologies increasing significantly for all protocols: Up/Down, HURP and spanning tree. The ratio of prohibited turns is worse also than regular network topologies like three dimensional meshes. We consider random fixed node degree topologies difficult to find in practice and less representative.

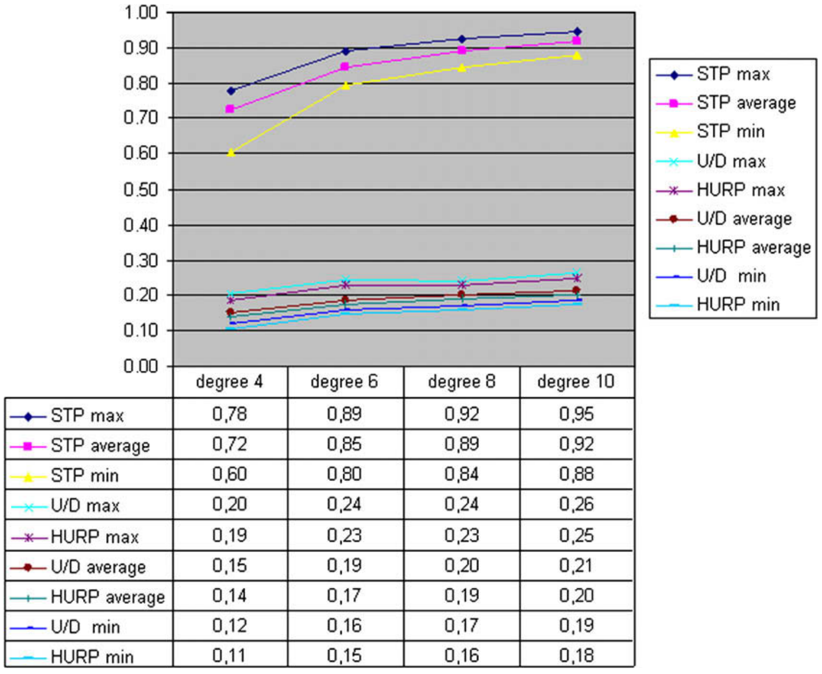
The HURP maximums are close to the average of Up/Down and stay below in all cases. The advantage of HURP over Up/Down diminishes as the network size increases, as occurred in other topologies.

6.2.3. Barabasi varying size topologies

We simulated 30 power law (scale free) topologies of 32, 64, 128 and 256 nodes. Topologies were generated with BRITE selecting Barabasi–Albert node distribution model. The results for absolute maximum, minimum and average values for Up/Down and HURP are shown in Fig. 10. Results are grouped by topology size and ordered by degree inside each topology. Turn prohibition decays with increasing network sizes. The graphs show that the maximum and minimum values of HURP are significantly below those of Up/Down. The values increase with average node degree, and decrease when the number of nodes is increased. Up/Down performance stays bounded regardless of the bridge selected as root.

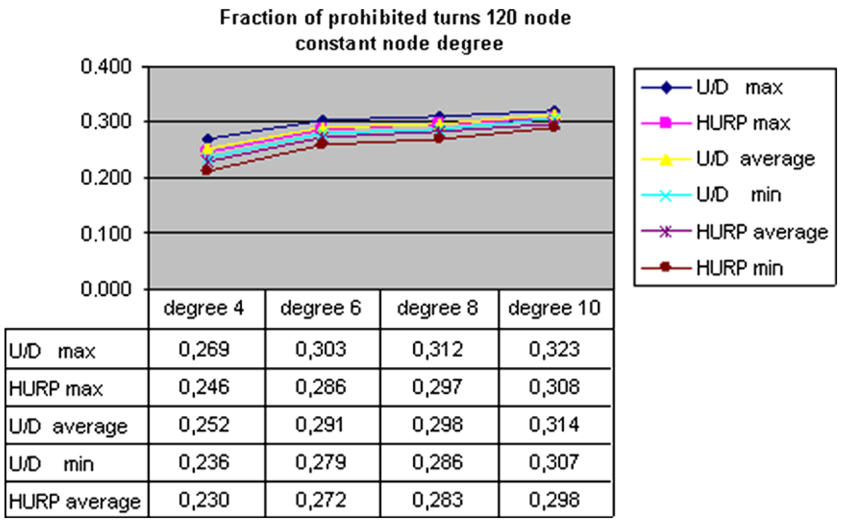
6.2.4. Summary performance comparison with fixed node degree topologies

We now combine, for comparison purposes, the results for Spanning Tree, Up/Down, HURP above with those for Turn Prohibition (TP) and Tree Based Turn Prohibition (TBTP) obtained with similar conditions of fixed node degree topologies. Results for Spanning Tree, and Up/Down obtained as described above in 2.b coincide with those obtained at [2] for TP and at [10] for TBTP, and suggest a fair comparison. We have included in Table 1 the results from TP and TBTP, respectively to obtain a global view of the five protocols for the fixed node degree topologies. HURP nearly coincides in performance with TBTP (although it is decentralized and much simpler than both TBTP and TP) and performs better than Up/Down. As said



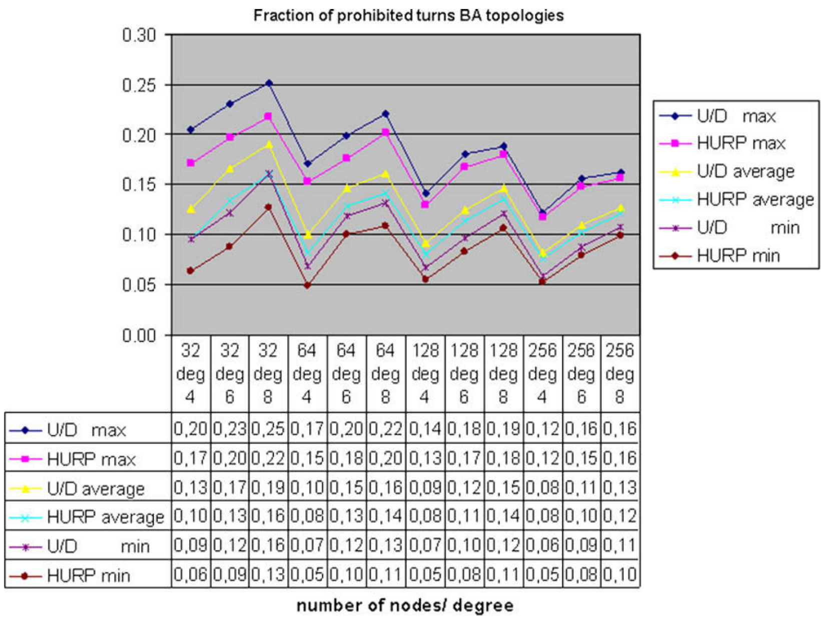
	degree 4	degree 6	degree 8	degree 10
STP max	0,78	0,89	0,92	0,95
STP average	0,72	0,85	0,89	0,92
STP min	0,60	0,80	0,84	0,88
U/D max	0,20	0,24	0,24	0,26
HURP max	0,19	0,23	0,23	0,25
U/D average	0,15	0,19	0,20	0,21
HURP average	0,14	0,17	0,19	0,20
U/D min	0,12	0,16	0,17	0,19
HURP min	0,11	0,15	0,16	0,18

Fig. 8. Fraction of prohibited turns of Waxman topologies vs. average node degrees.



	degree 4	degree 6	degree 8	degree 10
U/D max	0,269	0,303	0,312	0,323
HURP max	0,246	0,286	0,297	0,308
U/D average	0,252	0,291	0,298	0,314
U/D min	0,236	0,279	0,286	0,307
HURP average	0,230	0,272	0,283	0,298

Fig. 9. Fraction of prohibited turns as a function of node degree. All nodes at each topology have same degree.



	32 deg 4	32 deg 6	32 deg 8	64 deg 4	64 deg 6	64 deg 8	128 deg 4	128 deg 6	128 deg 8	256 deg 4	256 deg 6	256 deg 8
U/D max	0,20	0,23	0,25	0,17	0,20	0,22	0,14	0,18	0,19	0,12	0,16	0,16
HURP max	0,17	0,20	0,22	0,15	0,18	0,20	0,13	0,17	0,18	0,12	0,15	0,16
U/D average	0,13	0,17	0,19	0,10	0,15	0,16	0,09	0,12	0,15	0,08	0,11	0,13
HURP average	0,10	0,13	0,16	0,08	0,13	0,14	0,08	0,11	0,14	0,08	0,10	0,12
U/D min	0,09	0,12	0,16	0,07	0,12	0,13	0,07	0,10	0,12	0,06	0,09	0,11
HURP min	0,06	0,09	0,13	0,05	0,10	0,11	0,05	0,08	0,11	0,05	0,08	0,10

Fig. 10. Averages, maximum and minimum fractions of prohibited turns Up/Down vs. HURP for Barabasi–Albert topologies of varying sizes (32, 64, 128 and 256 nodes) and average node degrees of 4, 6, 8.

Table 1
Comparative results with fixed node degree topologies (120 Node). Averaged values.

d	U/D	HURP	TBTP	TP	Spanning tree
4	0.25	0.23	0.23	0.21	0.73
6	0.29	0.27	0.27	0.23	0.86
8	0.30	0.28	0.28	0.25	0.91
10	0.31	0.30	0.29	0.26	0.93

Table 2
Fraction of prohibited turns for fixed node degree topologies. All nodes have degree four.

Nodes	U/D	HURP	TP	Spanning tree
16	0.27	0.20	0.23	0.72
32	0.26	0.21	0.23	0.73
64	0.25	0.22	0.22	0.73
128	0.25	0.23	0.21	0.73

above, fixed node degree topologies perform worse with any turn prohibition algorithm than any other topologies evaluated.

Similarly, Table 2 shows the comparison with varying network sizes for fixed node degree topologies with degree $d = 4$ including TP results from [2]. TBTP results are not available for $d = 4$, only for d=8 [10]. HURP performs slightly better than Turn Prohibition in smaller topologies and slightly worse in bigger ones, but stays well below Up/Down.

The conclusions regarding fraction of prohibited turns are the following:

– HURP performs better than Up/Down for average node degrees 4 to 8, and slowly tends asymptotically to Up/Down with increasing average node degree and with increasing network size.
– The topology type influences heavily the results for all protocols without changing its relative position: Scale free topologies perform best and random fixed node degree perform worst, for all protocols.
– The dependency of Up/Down and HURP with the bridge elected as root bridge is low and bounded in all simulated networks. It seems that this dependency appears just in very specific topologies in which only one or a few nodes have high degree and are positioned in the lowest levels of the spanning tree would exhibit this dependency.

6.3. Throughput

In this section we compare the maximum throughput obtained with the algorithms studied and the same topologies described in the evaluation of the number of turns prohibited and path lengths. To evaluate the dependency of compared algorithms with the root bridge elected, the calculation is iterated N times per topology, with a different bridge root bridge at each iteration.

Throughput is obtained considering a flow model. We assume that each client is connected at every node and establishes a session (flow) with $N - 1$ different clients, each one located at every other node. Then, routes are computed according to each algorithm, (shortest path is obtained applying Dijkstra algorithm) and we obtain the number of flows traversing each link. With this information, the *bottleneck link*, i.e. the link shared by the highest number of sessions, can be determined. The *relative throughput* is represented as the quotient of the number of sessions at bottleneck link with the compared protocol, divided by the number of sessions at bottleneck link with shortest path. Note that shortest path may be not optimum in terms of throughput because equal-cost routes tie-breaking mechanisms may result in the accumulation of flows in a given link, and therefore may increase the number of flows in the bottleneck link.

6.3.1. 120 node Waxman topologies

Table 3 shows the results for the random topologies with Waxman model with varying degree topologies. Results are referred as the percentage relative to throughput obtained with Shortest Path routing (no turn prohibitions at all). HURP results are clearly superior to Up/Down. HURP performance approaches to Shortest Path as the network connectivity degree increases. The reason is that with high node degrees, there are many shortest paths to choose from for a route, so restricting some of them has overall little effect.

Table 3
Relative throughput for 120 node Waxman Topologies as a function of average node degree.

Nbr of Nodes	Degree	Shortest paths	HURP	Up/Down	Spanning tree
120	4	100	81	57	12.9
120	6	100	86	65	7.4
120	8	100	88	69	5.6
120	10	100	96	73	4.9

Table 4
Relative throughput of fixed node degree ($D = 4$) topologies.

Nbr of nodes	Shortest paths	HURP	Up/Down	Spanning tree
16	100	88	66	42
32	100	79	55	36
64	100	65	41	26
128	100	47	25	15

6.3.2. Fixed node degree topologies

Table 4 shows the results for the random topologies with fixed node degree ($d = 4$). As for turn prohibition, these topologies show lower performance than other topologies for all the algorithms compared.

Up/Down relative performance in random fixed node degree topologies is low, likely due to the relatively higher number of nodes with high degree (more possible turns) located in a low tree position.

6.3.3. Barabasi varying sizes topologies

Table 5 shows the throughput for the random topologies with Barabasi–Albert degree distribution. HURP performs close and even exceeds Shortest Path throughput, Exceeding Shortest Path is a matter of path diversity and has no special meaning in this context. Due to the high connectivity of BA topologies, path lengths (see Section 7) are practically identical for HURP and Shortest Path. The small relative differences in throughput obtained for HURP and Shortest Path are more related with statistical variations for each topology derived from the addressing system dependent of root bridge for HURP and path selection and tie cost resolution mechanisms in shortest path routing.

6.4. Path length

In this section we compare the results for path lengths of the Spanning Tree, Shortest Path, Up/Down and HURP protocols in the above mentioned topologies. HURP consistently improves path length compared to Up/Down.

6.4.1. 120-node Waxman topologies

Table 6 shows the average path lengths for the 120 node Waxman topologies with random node degree distribution as a function of average node degree of topology. Path lengths for HURP are very close (less than 3%) to Shortest Path and always shorter than Up/Down.

Table 5
Throughput of Barabasi (Power law) topologies relative to shortest paths.

Nbr of nodes	Degree	Shortest paths	HURP	Up/Down	Spanning tree
32	3.81	100	97	89	29
32	5.63	100	95	91	17
32	7.38	100	94	91	12
64	4.00	100	99	88	26
64	5.81	100	97	86	14
64	7.69	100	101	90	10
128	3.95	100	102	81	21
128	5.90	100	101	83	11
128	7.84	100	109	93	9
256	3.98	100	103	80	19
256	5.95	100	104	82	10
256	7.92	100	97	81	6

Table 6
Path length for 120 node Waxman topologies (average node degree).

Nbr of nodes	Degree	Shortest paths	HURP	Up/Down	Spanning tree
120	4	3.47	3.56	3.76	5.65
120	6	2.84	2.87	2.98	4.84
120	8	2.53	2.54	2.62	4.43
120	10	2.34	2.35	2.41	4.19

Table 7
Average path length of fixed node degree topologies.

Nbr of nodes	Shortest paths	HURP	Up/Down	Spanning tree
16	2.16	2.19	2.41	3.43
32	3.2	3.34	3.82	5.15
64	4.30	4.71	5.52	7.21
128	5.38	6.2	7.35	9.38

6.4.2. *Fixed node degree topologies*

Table 7 shows the average path lengths for fixed node degree topologies. Note that HURP results are much closer to Shortest Path than to Up/Down.

6.4.3. *Barabasi varying sizes topologies*

Table 8 shows the path length results for the random topologies with Barabasi–Albert topologies, scale free degree distribution. HURP path lengths are very close or coincide with shortest paths. It is worth noting that path length is nearly protocol independent, i.e., the resulting values are very close for Up/Down, HURP and Shortest Path. This path

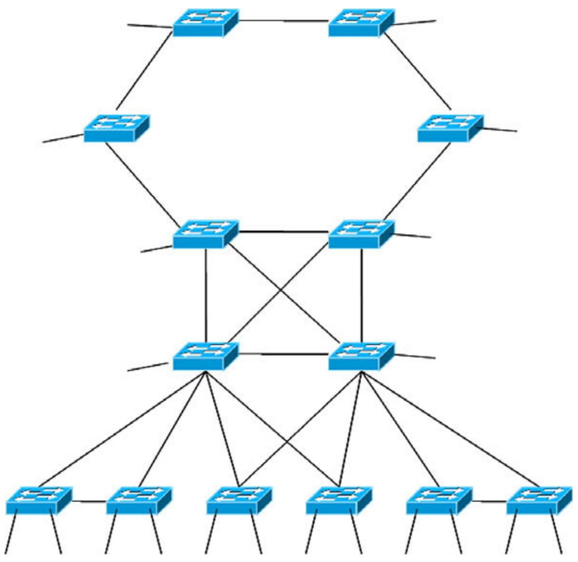


Fig. 11. Metro topology.

Table 8
Average path length of Barabasi–Albert topologies with varying average node degrees.

Nbr of nodes	Degree	Shortest paths	HURP	Up/Down	Spanning tree
32	3.81	2.43	2.44	2.48	3.51
32	5.63	2.07	2.07	2.09	3.20
32	7.38	1.88	1.88	1.90	3.05
64	3.91	2.78	2.79	2.85	3.93
64	5.81	2.40	2.40	2.44	3.78
64	7.69	2.17	2.17	2.19	3.50
128	3.95	3.16	3.18	3.27	4.63
128	5.91	2.70	2.71	2.76	4.30
128	7.84	2.43	2.43	2.45	3.94

Table 9
Performance comparison of metro topology with protocols: fraction of prohibited turns, througput and average path lengths.

	Shortest paths	HURP	Up/Down	Spanning tree
Prohibited Turns	0	0.09	0.15	0.64
Throughput	100	93	83	54
Path length	2.29	2.29	2.37	2.92

shortening is probably the result of the "hub effect" of nodes with high connectivity.

We can see that the worst topology for path length of all the ones considered in this paper is fixed node degree topologies. HURP path lengths are always shorter than Up/Down and closer to the optimum of Shortest Paths than to Up/Down.

6.4.4. Metro topology

Besides the above described topologies, a typical Metro topology (Fig. 11) was also evaluated for throughput, path length and fraction of prohibited turns. Traffic distribution is uniform, as in previous topologies, assuming one flow between each client connected to a node every other node. Table 9 shows the results.

7. Related work

Up/Down and the Turn based routing derivatives are the closest proposals to the HURP architecture described in this paper. Up/Down is used by Autonet [2]. *Turn Prohibition* [2] is a centralized algorithm based on selecting iteratively the node with lower connectivity degree and prohibiting turns around it, taking care that the network is not partitioned. In this way it may guarantee bounded performance on the fraction of total turns prohibited. Turn Prohibition can provide a loop-free topology by eliminating less than 1/3 of the turns. The experimental results presented at [2] show an average improvement of performance around 10–20% compared to its predecessor Autonet Up/Down routing paradigm on random topologies with fixed node degree Turn Prohibition has a high computational complexity $O(N^2d)$ that limits its scalability.

Tree Based Turn Prohibition (TBTP) [10], is an evolution of Turn Prohibition. TBTP relies partially on spanning tree information to prohibit less than half of the turns of any graph. The improvement in the fraction of prohibited turns increases with the node degree. TBTP requires a global knowledge of the network topology to determine the turns to prohibit, and the complexity of the algorithm is $O(N^2 * d^2)$ where N is the number of nodes and d the degree of node with max degree in the graph. This complexity leads to a limited scalability when network size increases. TBTP includes a version that is backward compatible with 802.1D. A distributed version of TBTP (dTBTP) is proposed at [10] to improve scalability and compatibility with standard bridges, although performance results are slightly inferior to that of TBTP and complexity remains $O(N^2 * d^2)$.

Segment-Based Routing (SBR) [19], another variation on turn prohibition, divides the network in subnets and subnets into disjoint segments, enforcing one turn prohibition per segment. Segmentation is performed by the successive visitation of nodes from a visited node to a not visited node using a not visited link. The segments found are labeled and classified in three basic types: *initial*, *regular* and *unitary*. Each segment is a set of links that completes a potential cycle in the network Cycles are prevented by selecting at every segment a restriction (bidirectional local turn prohibition) that depends on the type of segment. Performance of SBR is better than Up/Down and TBTP. However, it is not compatible with standard Ethernet switches, since all bridges must execute SBR. SBR can operate in a zero-configuration mode, although an alternative is the use of centralized computation of tables and static configuration of switches with SNMP. Fully distributed dynamic reconfiguration is thus not possible.

RBridges [9] use a link state protocol derived from IS–IS to acquire the bridge topology. The Rbridges broadcast their local connectivity to other RBridges to allow the computation of a global view of the topology. Rbridges also learn which end nodes are located on its link by observing the source address of packets that have originated on that link and interchange the list of addresses of end nodes connected to them. This enables all Rbridges to know which Rbridge is the appropriate destination for each end node. The egress Rbridge from a link encapsulates the packet with an additional header that contains a hop count (to discard looped frames), and a destination Rbridge identifier. Rbridges still require a spanning tree for delivering layer 2 multicast packets and packets to unknown destinations. The RBridges proposal is being standardized at the IETF TRILL [8] working group. It is worth to note that RBridges will likely require complex configuration.

LSOM (Link State over MAC) [11] relies on a link state protocol to calculate shortest paths. It is intended for being used at backbone switches, i.e. interconnecting a limited number of bridges, so scalability is not considered to be a concern. Only the backbone ports (the ports that connect to other backbone bridges) implement LSOM. Link state frames, sent to all the bridges in the MAN, are composed of its identifier, the list of neighbors, the cost to reach them, and a list of the MAC addresses on its boundary ports. Each bridge is able to compute a complete map for the topology of the backbone network from the link states it has received from the rest of the bridges, and from this map it can decide the best route to each destination.

STAR (*Spanning Tree Alternate Routing Protocol*) [6] bridges attempt to forward frames over alternate paths that are shorter than the corresponding paths on the standard spanning tree, provided that such alternate paths can be identified without excessive protocol complexity. Otherwise, STAR uses the standard spanning tree for default forwarding. The metric considered for STAR operation may be delay, cost, or in general, any additive metric. STAR bridges are backward compatible with the current IEEE 802.ID standard and can be mixed with them without restriction. The improvements in performance are modest due to the restrictions for cross links eligibility. In case of topology changes, full reinitialization of STAR tables seems necessary after the spanning tree has converged again. Convergence of STAR is constrained by the slow convergence of the spanning tree protocol STP, but migration to RSTP protocol seems feasible.

Viking [23] is a Multiple Spanning Tree VLAN-based architecture oriented to Storage Area Networks. It aims to optimize overall throughput performance by using shortest paths, and to provide protection in case of link failure by the definition of multiple and redundant links. To do so, it uses per-VLAN spanning tree instances that are calculated in a centralized device called *Viking Manager*. The Viking Manager computes optimum routes between hosts, and alternative routes to be used in case of failure, maps them to VLANs, and uses SNMP to configure each bridge. An important drawback lies in the fact that the hosts must run additional software to select the assigned VLAN.

IEEE 802.1aq Shortest Path Bridging (SPB) [24] aims to improve network infrastructure utilization and reduce path length by replacing the MSTP [5] control plane with shortest path trees. SPB operates in a Shortest Path Tree (SPT) Bridging Region in which it defines multiple tree instances rooted respectively at the edge bridges to obtain shortest paths among SPB bridges. Accordingly, a Shortest Path Trees Bridging Region corresponds to a Multiple Spanning Tree region of the 802.1Q standard. As in MSTP, multiple regions may exist and are differentiated by a per-region identifier. Each tree is associated to a specific VLAN. SPB uses Shared VLAN Learning (SVL) of MAC addresses among VLANs for frames allocated in different spanning tree instances. Different choices are available for the computation of the set of symmetric shortest path trees between each of the bridges of an SPB region: A derivation of MSTP protocol (distance vector based) with the addition of cut-bit vectors to ensure symmetry of tree instances; the use of an extension of the IS–IS protocol with additional information and procedures; and finally, the use of a new Link State Tree Protocol (LSTP). The proposal is compatible with RSTP, and with carefully configured MSTP (802.1Q).

The ABridges architecture [21,22] is a two-tiered hierarchy in which network islands running independent rapid spanning tree protocols communicate through a core formed by island root bridges (ABridges). ABridges use AMSTP, a simplified and self configuring version of the MSTP protocol, to establish shortest paths in the core using multiple spanning tree instances, one instance rooted at each core edge ABridge. ABridges encapsulate the frame with a header containing the Ingress and Egress ABridge addresses and the AMSTP Ethertype. The architecture is efficient in terms of network usage and path length due to the ability of AMSTP to provide optimum paths in the core mesh, while RSTP is used to aggregate efficiently the traffic at islands composed of legacy bridges, where sparsely connected, tree-like topologies are frequent and recommended. Convergence speed is as fast as existing Rapid Spanning Tree and Multiple Spanning Tree Protocols. Scalability is limited by the maximum number of spanning tree instances constructed with same BPDU, limited by the BPDU length (around 64). Variants of AMSTP with independent spanning trees may overcome this limitation, but at the cost of multiplying the number of BPDUs interchanged between nodes by the number of bridges in the network.

Seize [17,25] is a scalable and efficient zero-configuration enterprise networking architecture. It uses link state shortest path routing and hash-based location-resolution to avoid broadcasts. The distributed hash tables reside at switches. Switches perform location resolution on demand and cache the results to optimize routing paths and to reduce the number of location-resolution requests. Compatibility with non-Seize bridges is achieved through encapsulation. Scalability is limited by the link state routing protocol between switches that requires full topology knowledge.

8. Conclusion

We have presented HURBA, a novel layer-two architecture that combines distance vector routing and Up/Down, enhanced with the use of hierarchical bridge identifiers. HURBA is designed as a zero-configuration architecture that extends and uses RSTP as the underlying protocol for core tree building, bridge identifier assignment and core reconfiguration. The low computational complexity and the preservation of the Ethernet frame format in the HURBA core allow upgrading legacy bridges to HURBA operation with just software updates, HURBA is consistent with current 802.1D architecture and self configures building a core mesh to which legacy bridges attach building peripheral standard spanning trees.

The performance of HURP is similar or superior to other Turn Prohibition algorithms like TP and TBTP, but with lower computation complexity (similar to the one of Up/Down). HURP performs consistently better than Up/Down using the topology information carried on the hierarchical MAC addresses. Performance in all aspects is much closer to shortest path routing than to Up/Down. Performance does not vary significantly with root bridge election. The impact of the number of prohibited turns in system performance decreases with the node degree, because the number of possible turns per node grows with the square of node degree (possible turns per node is equal to $d^{*}(d-1)/2$). Therefore, as node degree grows, a large number of alternative shortest paths are created.

Acknowledgements

This work was supported in part by grants from Spanish Ministry of Education through Project CONPARTE (P3) (TEC2007-67966-C03-03/TCM) Comunidad de Madrid through Projects TOLIPAN (CCG07-UC3M/TIC-3356) and E-MAGERIT (S-0505/TIC/000251) and from Comunidad de Castilla la Mancha through Project EMARECE (PII1I09-0204-4319).

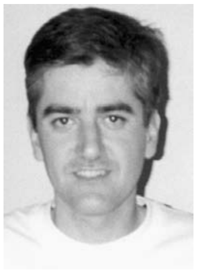
References

[1] M. Shoreder et al., Autonet: a high-speed, self-configuring local area network using point-to-point links, IEEE Journal on Selected Areas in Communications 9 (8) (1991) 1318–1335.
[2] D. Starobinski, G. Karpovsky, F. Zakrevsky, Applications of network calculus to general topologies, IEEE/ACM Transactions on Networking 11 (3) (2003) 411–422.
[3] G. Ibáñez, A. Azcorra, Application of rapid spanning tree protocol for automatic hierarchical address assignment to bridges, in: 11th International Telecommunication Networks Strategy and Planning

Symposium. Networks 2004, Wien, June 2004. <www.ieee.org/ieee.explore>.
[4] IEEE 802.1D-2004 IEEE standard for local and metropolitan area networks-Media access control (MAC) Bridges. <http://standards.ieee.org/getieee802/802.1.html>.
[5] IEEE 802.1Q-2003 IEEE standard for Local and Metropolitan Area Networks-Virtual Bridged Local Area Networks.
[6] K.-S. Lui, W.C. Lee, K. Nahrstedt, STAR: a transparent spanning tree bridge protocol with alternate routing, SIGCOMM Computing Communication Review 32 (3) (2002) 33–46.
[7] R. Perlman, Rbridges: transparent routing, in: Proceedings of IEEE Infocom 2004, March 2004.
[8] Transparent interconnection of lots of links (TRILL) WG. <http://www.ietf.org/html.charters/trill-charter.html>.
[9] The Rbridge archives. <http://www.postel.org/pipermail/rbridge/>.
[10] Pellegrini et al., Scalable, distributed cycle-breaking algorithms for gigabit Ethernet backbones, Journal of Optical Networking 5 (2) (2006). Feb.
[11] R. García, J. Duato, F. Silla. LSOM: A Link State Protocol Over MAC addresses for metropolitan backbones using optical ethernet switches, in: Proceedings c. Second IEEE NCA, 2003, p. 315.
[12] Boston University Representative Topology Generator-BRITE. <http://www.cs.bu.edu/brite/>.
[13] E. Sfeir et al., Performance evaluation of Ethernet resilience mechanisms, in: Workshop on High Performance Switching and Routing, 2005, HPSR, 2005 Workshop on May 2005, pp. 356–360.
[14] K. Elmeleegy et al., On count-to-infinity induced forwarding loops in Ethernet networks, in: Proceedings of the INFOCOM 2006, Barcelona, Spain, April 2006.
[15] L.S. Carmichael, Ghani et al., Characterization and comparison of modern layer-2 Ethernet survivability protocols, in: Proceedings of the Thirty-seventh Southeastern Symposium on System, March 2005, pp. 124–129.
[16] S. Owicki, A. Karlin, Factors in the performance of the AN1 computer network, in: Proceedings of the 1992 ACM SIGMETRICS and PERFORMANCE '92 Int'l. Conf. on Measurement and Modeling of Computer Systems June 15, 1992.
[17] K. Changhoon, J. Rexford, Revisiting Ethernet: plug-and-play made scalable and efficient, in: 15th IEEE Workshop on Local and Metropolitan Area Networks, 2007. LANMAN 2007, June 2007, pp. 163–169.
[18] G. Ibáñez, et al., Hierarchical up/down routing architecture for Ethernet backbones and campus networks, in: HSN 2008. INFOCOM April 2008.
[19] A. Mejia, J. Flich, J. Duato, S.A. Reinemo, T. Skeie, Segment-based routing: an efficient fault-tolerant routing algorithm for meshes and tori, in: Proceedings of the 20th IEEE International Parallel and Distributed Processing Symposium (IPDPS 2006), IEEE (2006).
[20] IEEE 802.1 Working group. <http://www.ieee802.org/1>.
[21] G. Ibáñez et al., ABridges: scalable self-configuring Ethernet campus networks, Computer Networks 1 (3) (2008) 630–649.
[22] G. Ibáñez, A. García, A. Azcorra, Alternative multiple spanning tree protocol (AMSTP) for optical Ethernet backbones, in: IEEE Conference on Local Computer Networks, Tampa, November 2004.
[23] S. Sharma, K. Gopalan, S. Nanda, T.C. Chiueh, Viking: a multi-spanning-tree Ethernet architecture for metropolitan area and cluster networks, in: INFOCOM 2004, Twenty-third Annual Joint Conference of the IEEE Computer and Communications Society (IEEE, 2004), pp. 2283–2294.
[24] S.P.B.M. Seaman, Shortest path bridging. <http://www.ieee802.org/1/files/public/docs2005/new-seaman-shortestpath-par-0405-02.htm>.
[25] C. Kim, J. Rexfor, Revisiting Ethernet: plug-and-play made scalable and efficient, in: Local and Metropolitan Area Networks, 2007, LANMAN 2007. <http://ieeexplore.ieee.org>.

Guillermo Ibáñez received his Telecommunication Engineering degree from Universidad Politécnica de Madrid in 1975 and the Ph.D. degree in Communication Technologies from Universidad Carlos III de Madrid in 2005. He worked at ITT R&D Laboratories in Madrid till 1984. He has long R&D experience in the telecommunication industry at Alcatel Spain, where he has held several technical and technical leading positions in the international development of Alcatel System 12 Switching System and Litespan 1540 Multiservice Access Nodes till 2002. He is an Associate Professor in the Telematics Engineering Area of the Universidad de Alcala' in Madrid. His current research interests are high performance and scalable Ethernet networks, wired and wireless. He is author of several publications on these subjects.

Alberto García-Martínez received his Ph.D. in Telematics Engineering from Universidad Politécnica de Madrid (UPM) in 1999. He is an Associate Professor in the Telematics Engineering Dept. of the Universidad Carlos III de Madrid, Spain. He has participated in several national and international research projects on IPv6 and QoS. His recent research interests are related with routing in layers two and three, and IPv6.



Juan Antonio Carral received his Telecommunication Engineering degree from Technical University of Madrid in 1993. He was a lecturer and research assistant at Technical University of Madrid from 1994 to 1998. He joined the University of Alcala in 1999 as an assistant professor, he promoted to Associate professor in 2000. His research activities cover the fields of layer-two routing, performance simulation and MPLS. He has been involved in international and national research projects related with these topics, including the EU ACTS INTERACT and ITTI projects. He has published several papers and technical reports.

Pedro A. González received his Telecommunication Engineering degree in 2007 from Universidad de Alcalám, where he collaborated on layer two routing protocols and simulations. He has worked at Ericsson Spain. He has worked until Nov. 2008 at Ericsson Span. He currently works as network engineer at Huawei Spain.

Arturo Azcorra received a M.Sc. degree in Telecommunications from Technical University of Madrid (Spain), in 1986, and Ph.D. degree from the same university in 1989. In 1993 he obtained an MBA from Instituto de Empresa, Madrid. He was a lecturer at Technical University of Madrid from 1987 to 1990 and promoted to associate professor. In 1998 he joined U. Carlos III of Madrid (Spain), where he is now a full professor. On 2006 he was appointed the Director of the International Research Institute IMDEA Networks. Professor Azcorra has participated and directed over 20 European research and technological development projects from ESPRIT, RACE, ACTS and IST programs. He has also performed direct consulting and engineering work for institutions such as European Space Agency, MFS Worldcom, Madrid Regional Government, RENFE, REPSOL, and the Spanish Ministry of Science and Technology. He has been program committee member of international conferences in several editions of IEEE-PROMS, IDMS, QofIS, CONEXT and IEEE-INFOCOM. He is general co-chair for CONEXT '05, and TPC co-chair for INFOCOM '06. His list of papers published in national and international magazines, books and conferences is over 100 titles. Current research projects include broadband networks, multicast teleservices, active networks, and advanced IP networks.

José Manuel Arco received his Telecommunication Engineering degree in 1996 and his Ph.D. in 2000 from Universidad de Alcalá. He is an Associate Professor in the Telematics Engineering Area of the Universidad de Alcalá. His research activities cover His research activities cover broadband networks, MPLS and VPNs. He is involved in research projects related with these topics. He is author of several publication on these subject.