EuroFGI Workshop on IP QoS and Traffic Control
P. Pereira (Ed.)
Lisbon, Portugal, December 6-7, 2007

IMPROVING LOAD BALANCE AND MINIMIZING SERVICE DISRUPTION ON ETHERNET NETWORKS WITH IEEE 802.1S MSTP

Amaro F. de Sousa*, Gil A. S. Soares*

* Instituto de Telecomunicações, Universidade de Aveiro
3810-193 Aveiro, Portugal
e-mails: asou@ua.pt, gsoares@av.it.pt

Keywords: Multiple Spanning Tree Protocol, Ethernet, Traffic Engineering.

Abstract. *In this paper, we address the combined resource allocation and protocol parameter assignment problem of Ethernet networks using IEEE 802.1S Multiple Spanning Tree Protocol (MSTP). MSTP enables (i) the definition of multiple network regions, where a Common Spanning Tree is set up between all switches, (ii) the set up of additional Spanning Trees inside each region and (iii) the assignment of each traffic VLAN to a different Spanning Tree. We propose an algorithm to compute the appropriate MSTP parameters aiming to optimize the network load balance and minimize service disruption, and show its efficiency through computational results. We study the relationship between load balance and service disruption under single link failures when either a single region or different regions are adopted and for different number of additional Spanning Trees. We show that the single region approach is always better when MSTP is optimally configured and that a very small number of Spanning Trees can obtain near optimal solutions, thus, not penalizing the switching processing overhead.*

1 INTRODUCTION

Current Ethernet switches are compliant with IEEE 802.1D and IEEE 802.1Q protocols. The 802.1D Spanning Tree Protocol (STP) sets the routing paths between any pair of switches based on a logical set of links spanning all switches without cycles, i.e., a Spanning Tree (ST). It includes detection of network topology changes and recalculation of the ST to reacquire global connectivity. In a network with n switches interconnected by l point-to-point links ($l > n - 1$), there are n–1 active links and the remaining ones act as backup links. On the other end, 802.1Q VLAN Protocol enables the definition of different VLANs, therefore creating different broadcast domains within the same switching nodes. A VLAN is a set of ports, belonging to the same or to different switches, with full connectivity between each other. Traffic demands of different clients are supported on different VLANs to prevent packets from one client to reach ports of others, thus optimizing network resources and preventing Layer 2 security attacks.

STP was proposed years ago when recovering connectivity after a network failure within a minute was considered adequate performance. Recently, two new protocols were proposed by IEEE to enhance the survivability and the traffic engineering capabilities of switching networks. One of these protocols is the IEEE 802.1W Rapid Spanning Tree Protocol (RSTP), an evolution of STP where the port states and roles are redefined and a negotiation mechanism is used in order to accelerate the convergence of the ST whenever the network topology changes. The other protocol is the IEEE 802.1S Multiple Spanning Tree Protocol (MSTP) [1] which makes use of both RSTP and VLAN protocols. With MSTP, a network operator can define different network regions where a Common Spanning Tree (CST) connecting all switches of all regions is set-up in such a way that a topology change inside a region does not change the CST outside the region. MSTP also enables additional STs to be configured inside each region.

In this paper, we address the following problem. Given (a) a network composed by a set of switches connected through point-to-point links, (b) a set of regions defined on the network and (c) a set of

VLANs, each one defined by a set of traffic flows that are to be supported by the network, we determine (i) the appropriate MSTP parameters implementing the CST and the required additional STs inside each region and (ii) the assignment of each traffic VLAN to one of the STs. The aim is to optimize the network load balance and to minimize service disruption. As will be shown later, for any desired set of STs, it is always possible to determine a set of MSTP parameters that makes active its links. This fact enables the separation of the problem in two distinct stages. In a first stage, the set of desired STs is determined applying a GRASP (Greedy Randomized Adaptive Search Procedure) based optimization algorithm. In a second stage, a MSTP parameter assignment algorithm is proposed that computes the appropriate protocol parameters minimizing the service disruption for single link failures.

MSTP does not define how networks should be divided in regions, which Spanning Trees should be created on each region and how VLANs should be assigned to each Spanning Tree. One possibility is to consider the whole network as a single region. The motivation for the multiple regions is twofold: (i) the inter-working between MSTP enabled networks and non MSTP enabled networks is possible since switches outside regions do not need to support MSTP and, (ii) the STs reconfiguration due to topology changes inside a region is administratively confined to that region (limiting network operational instability). This second reason might let us conclude that multiple regions provide lower service disruption than a single region. However, since inter-region VLANs can be supported by a single ST between regions, it limits the amount of load balancing that can be obtained with multiple regions. Moreover, the use of multiple STs in the single region case also improves service disruption because the STs not using a failed link are not reconfigured and, therefore, their supported traffic does not suffer service disruption. We will show through the computational results that the single region approach is always better not only in network load balance (the obvious result) but also in service disruption (the non-obvious results) provided that MSTP is optimally configured. We will also show that in both cases (single region and multiple regions) a very small number of STs can obtain near optimal solutions, thus, not penalizing significantly the switching processing overhead.

Recently, we have addressed the load balance optimization problem in [2]. In that work, thought, the proposed optimization algorithm is less efficient than the one presented here and the impact of link failures on service disruption was not addressed. The use of MSTP as a means to enhance traffic engineering capabilities of Ethernet network has been addressed by other authors [3-5] in recent years. Nevertheless, these works address other issues and traffic engineering objectives. In [6], the authors address the problem of how to divide the network into regions. Other works propose MSTP as a means to improve the support to other important aspects like mobility [7] and QoS [8].

This paper is organized as follows. Section 2 reviews the MSTP protocol and, in particular, how the different ST instances are configured. In Section 3, we propose the MSTP parameter assignment algorithm that computes the appropriate protocol parameters implementing any desired STs minimizing at the same time the service disruption for single link failure. Section 4 presents a GRASP based optimization algorithm for the determination of the set of STs that optimize network load balance. In Section 5, a selected set of computational results is presented and discussed.

2 MSTP PROTOCOL

In this section, we review MSTP and, in particular, how the different ST instances are configured. Let us focus first on the configuration of the additional STs inside each region. MSTP configures each ST (i.e., each set of active links) in the same way as STP, which is based on two types of positive integer parameters: a BridgeID assigned to each switch and a PortCost assigned to each port of each switch. These parameters can be administratively configured and, for each ST, different sets of BridgeID values and PortCost values are to be configured.

Consider the example (Figure 1) of a single region for one additional ST (the thick lines represent the active links forming the ST). First, the switch with the lowest BridgeID becomes the Root Bridge (the BridgeID includes the switch MAC address which guarantees that all BridgeID values are different), which is switch D. Then, the active links are the ones that belong to the minimum cost paths from each switch to the Root Bridge. The cost of a path is the sum of the PortCost values of all forwarding ports towards the Root Bridge (designated as Root Ports). In the example, the minimum cost path from switch A to the Root Bridge is A-B-D with a path cost of 20 and, therefore, links {A,B} and {B,D} become active links (the Root Port of switch A is the port that connect it to switch B). In the example, there are two minimum cost paths from switch E to the Root Bridge (E-D and E-B-D) both with a path cost of 20. When a switch has multiple minimum cost paths to the Root Bridge, the switch chooses the one whose next switch has the lowest BridgeID. Therefore, in the case of switch E, the first path is chosen. Note that the set of active links defines a unique routing path between any pair of switches and the traffic flows of all VLANs assigned to this ST are routed through these unique paths.

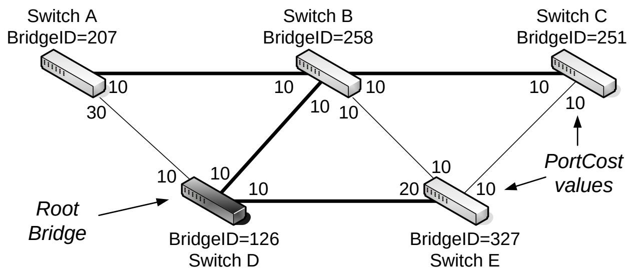


Figure 1: Spanning Tree resulting from the given sets of BridgeID and PortCost values.

The non-active links are used as backup links, i.e., links that can become active when the current active links fail. Figure 2 shows the new ST that is configured if link {B,D} fails. Once again, the new ST is the set of links in the minimum cost paths from each switch to the Root Bridge.

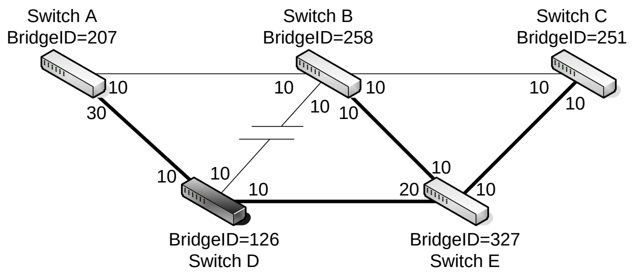


Figure 2: Spanning Tree after the failure of link {B,D}.

The comparison between Figures 1 and 2 shows that, in the case of link {B,D} failure, there is only one routing path that does not change (the one between switches D and E). All traffic demands whose routing paths change suffer temporary service disruption. In this case, most of the routing paths have changed and, therefore, most of the traffic suffers temporary service disruption. Service disruption can be minimized if whenever there is a failure, all active links prior to the failure remain active after the failure. Figure 3 shows the same example where we have changed two Port Cost values (with a circle around) and the resulting ST after the failure of link {B,D}. In this case, it is possible to verify that the failure has activated link {B,E} and all other previous active links remain active after the failure (it is straightforward to verify that the service disruption is minimum for all other single link failure cases).

Let us now focus on the configuration of the CST. The administrator defines a set of network regions. The switch with the lowest BridgeID becomes the CST Root Bridge. The part of CST outside the regions is configured in a similar way as described before considering each region as a single switch: the active links outside regions are the ones that belong to the minimum cost paths from each switch to the CST

Root Bridge considering that all Port Cost values of links internal to regions are NULL. The switch inside each region whose Root Port belongs to a link outside the region becomes the CST Regional Root Bridge (if the CST Root Bridge belongs to a region, it is its CST Regional Root Bridge). Inside each region, the active links are the ones that belong to the minimum cost paths from each region switch to the CST Regional Root Bridge.

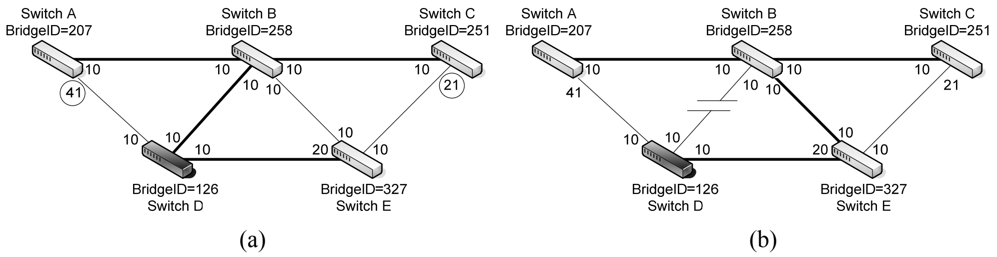


Figure 3: Spanning Tree with two updated PortCost values.

Figure 4 illustrates this architecture with 3 regions: the CST Root Bridge belongs to region 2 and the CST is given by the thick lines (in the general case, the CST Root Bridge can also be a switch outside any region). At regions 1 and 3, the CST Regional Root Bridges are the ones that are closer in the routing paths towards the CST Root Bridge. Note that by the method that MSTP uses to determine the CST, its set of links has the in-region spanning tree property, i.e., it forms also a spanning tree inside each region.

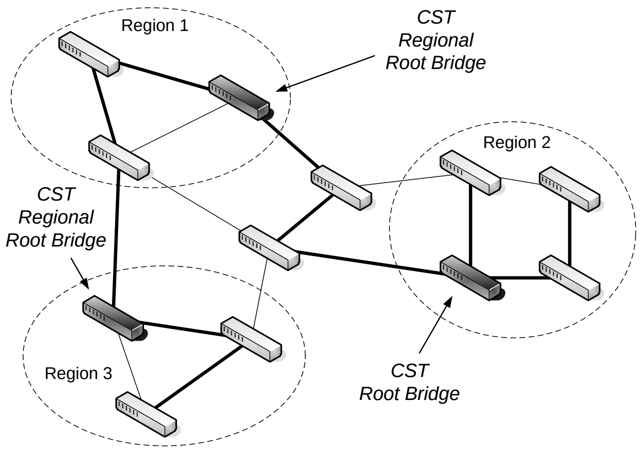


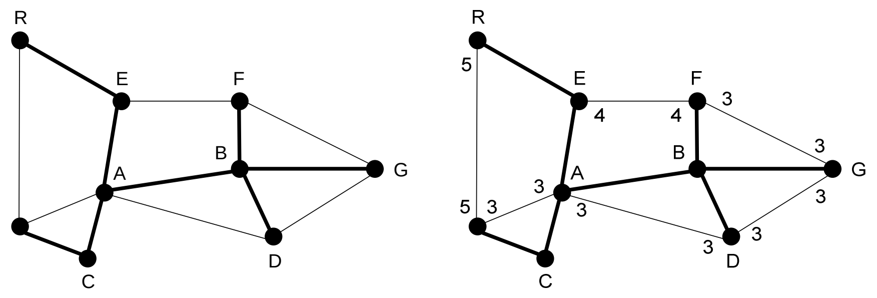
Figure 4: Common Spanning Tree (CST), Root Bridge and Regional Root Bridges.

The reconfiguration of the CST depends on where the link failures occur. Failures inside a region will not affect the CST outside the region and will not change its CST Regional Root Bridge. Failures outside regions change the CST between regions and can also change the CST inside each region because the CST Regional Root Bridges might change. Nevertheless, the minimum service disruption property can be observed if a set of PortCost values can guarantee that two conditions. In a failure inside region *r*: (i) a new link of region *r* is activated and (ii) all links of region *r* that were active before the failure remain active after the failure. In a failure outside regions: (i) a new link outside regions is activated, (ii) all links outside regions that were active before the failure remain active after the failure and (iii) the active links inside all regions do not change even if the CST Regional Root Bridge changes.

3 PortCost VALUES ASSIGNMENT

In previous section, we have reviewed how STs are configured based on the sets of BridgeID and PortCost values. In this section, we deal with the opposite operation, *i.e.*, how to assign the MST parameters in order to obtain a given set of desired STs. Since we can use a different set of parameters to define each ST, the parameter assignment operation can be applied to each ST individually. Assume that the Ethernet switches have some current BridgeID and PortCost values. The assignment algorithms proposed here update the current PortCost values without any change in the current BridgeID values.

Consider the example of Figure 5(a) where a desired ST is defined (with thick lines) over a network of Ethernet switches. Assume that switch R is the Root Bridge (the one that has the lowest BridgeID value) and that all current PortCost values are 1.



(a) (b)

Figure 5: (a) A ST defined on a Ethernet switched network. (b) PortCost values assigned at first stage.

In the first step, we start by fixing the current PortCost values of the links belonging to the desired ST. Then, we calculate the cost c_i of the paths from every switch i to the Root Bridge in the desired ST and the cost c_{ij} of the routing path from every switch i to every switch j in the desired ST. In the example of Figure 6, c_D is 4, c_C is 3 and c_{DC} is 3. Let $\{i,j\}$ represent the link between switches i and j and p_{ij} represent the PortCost value of the port of switch i used on link $\{i,j\}$. In order to guarantee the desired ST, the minimum PortCost values that must be assigned to each link $\{i,j\}$ not belonging to the desired ST are $p_{ij} = c_i - c_j + 1$ and $p_{ji} = c_j - c_i + 1$. However, any other higher value can be assigned and, in general, these minimum values do not guarantee the minimum service disruption property.

A failure on an active link divides the current ST in two STs. The ST that contains the Root Bridge, which we designate by ST_R, does not change because the minimum cost paths remain the same. The aim is that the active links of the other ST, which we designate by ST_B, remain active and that one of the backup links connecting the two STs becomes active. For a failure of a particular active link $\{a,b\}$, let us partition the backup links in two sets: set $B_{\{ab\}}$ with the backup links connecting a switch of ST_R to a switch of ST_B and set $\underline{B}_{\{ab\}}$ with the remaining backup links. In Figure 6, $B_{\{ab\}} = \{\{A,D\},\{E,F\}\}$.

In order to guarantee that the backup links $\{i,j\} \in \underline{B}_{\{ab\}}$ do not become active, the minimum PortCost values of their ports must be $p_{ij} = c_{ij} + 1$ and $p_{ji} = c_{ji} + 1$. With these minimum values we guarantee that the routing path on the desired ST between switches i and j has always a lower cost than the path provided by link $\{i,j\} \in \underline{B}_{\{ab\}}$. In Figure 5(a), c_{GD} and c_{DG} are both equal to 2 and , if $p_{GD} \geq 3$ and $p_{DG} \geq 3$, we guarantee that the backup link {D,G} will never become active provided that there is no failure in {B,D} or {B,G}. In a first stage, all non-active links whose current PortCost values are lower than $c_{ij} + 1$ are assigned with this value. In the following stages, these values can only be increased, which guarantee that the backup links $\{i,j\} \in \underline{B}_{\{ab\}}$ do not become active, whatever the active link $\{a,b\}$ fails.

Figure 5(b) shows the assigned PortCost values for all non-active links at this first stage. It is easy to see that this PortCost assignment does not yet guarantee the minimum service disruption property. If link {A,B} fails, the minimum cost path from switch F to the Root Bridge is F-E-R with cost 5 and the minimum cost path from switch D to the Root Bridge is D-A-E-R with cost 5. Therefore, both links {E,F} and {A,D} become active and some of the active links {B,F}, {B,D} or {B,G} become backup.

To guarantee the minimum service disruption property for a failure of an active link $\{a,b\}$, we have to guarantee that the minimum cost paths of all switches belonging to $\mathrm{ST_B}$ will be through one single backup link $\{i,j\} \in \mathrm{B}_{\{ab\}}$. First, we select one of these backup links. Then, (i) we fix the current PortCost values of the selected backup link, (ii) we determine for each of the other backup links $\{i,j\} \in \mathrm{B}_{\{ab\}}$ the costs δ_i and δ_j of the routing path from switches i and j to the Root Bridge in the desired backup ST and (iii) we increase the PortCost values of each of the other backup links $\{i,j\} \in \mathrm{B}_{\{ab\}}$ to the values $p_{ij} = \delta_i - \delta_j + 1$ and $p_{ji} = \delta_j - \delta_i + 1$ if their current values are lower. These are the minimum values that guarantee that the path of each switch of $\mathrm{ST_B}$ to the Root Bridge has always a lower cost through the selected backup link than to any other possible backup link. In Figure 5(b), if we select link {A,D} to become active when link {A,B} fails, the other backup link is {E,F}. In the backup ST (composed by all active links with {A,D} and without {A,B}), δ_E is equal to 1 (the cost of the path E-R) and δ_F is equal to 7 (the cost of the path F-B-D-A-E-R). Since the current PortCost values of link {E,F} are 4, the PortCost value p_{EF} remains unchanged (4>1–7+1) and the PortCost value p_{FE} is increased to 7–1+1 = 7 to ensure the minimum service disruption property for the failure of link {A,B}.

Finally, we must ensure the minimum service disruption property for the single link failure of all active links. Note that for a particular failure of an active link $\{a,b\}$, we select one backup link $\{i,j\} \in \mathrm{B}_{\{ab\}}$ and start by fixing its two PortCost values. Therefore, when we proceed to the failures of other active links, we must ensure that the PortCost values of the previously selected backup links are not required to be increased. This is guaranteed if, whenever we select a backup link for a particular failure, we select it also for all other active links that can be recovered by it. In the example of Figure 5(b), if link {A,D} is selected to recover from failure of link {A,B}, it is also selected to recover from failure of link {B,D}, since {A,D} also belongs to $\mathrm{B}_{\{BD\}}$. The algorithm for assigning the PortCost values that guarantees the minimum service disruption property for each additional ST on each region is now straightforward (consider ρ the Root Bridge, p_{ij} the current PortCost value of the port of switch i used on link $\{i,j\}$, and V, L and X three sets of links):

```
1. Keep PortCost values $p_{ij}$ of all desired active links unchanged
2. For each backup link $\{i,j\}$:
   2.1. Determine the costs $c_{ij}$ and $c_{ij}$ of the routing paths on the desired active links
   2.2. Do $p_{ij} = \max[p_{ij}, c_{ij}+1]$ and $p_{ji} = \max[p_{ji}, c_{ji}+1]$
3. Make $V = \{\}$ and $L = \{\}$
4. While ($V$ does not contain all desired active links) do:
   4.1. Select one backup link $\{y,z\} \notin L$
   4.2. Add link $\{y,z\}$ to set $L$
   4.2. Compute $X = \{$all desired active links $\{a,b\} \notin V$, such that link $\{y,z\} \in \mathrm{B}_{\{ab\}}\}$
   4.2. For each link $\{a,b\} \in X$ do:
      4.2.1. Add link $\{a,b\}$ to set $V$
      4.2.2. For all backup links $\{i,j\} \in \mathrm{B}_{\{ab\}}$ except $\{y,z\}$ do:
         4.2.2.1. Determine the costs $\delta_i$ and $\delta_j$ of the routing path from switches $i$ and $j$ to $\rho$ in the
                  desired backup ST
         4.2.2.2. Do $p_{ij} = \max[p_{ij}, \delta_i - \delta_j + 1]$ and $p_{ji} = \max[p_{ji}, \delta_j - \delta_i + 1]$
```

For the CST, we use the previous algorithm as a building block to define an appropriate algorithm. We assign the PortCost values to the part of CST between regions in the following way: first, we reduce each region in the complete network topology to a representative single switch (assigned with the lowest BridgeID value among all switches of the region); then, we assign ρ to the switch with the lowest BridgeID value and run the previous algorithm on the reduced topology. We assign the PortCost values to the part of CST inside each region in the following way: first, we assign ρ to the CST Regional Root Bridge (if the region includes the CST Root Bridge, it is by definition its CST Regional Root Bridge); then, we run the previous algorithm on the region network topology. Note that the previous algorithm still guarantees the desired ST inside each region even if ρ is different than the considered switch. Therefore, if a failure occurs in a link between regions, the solution given by the algorithm for each region will ensure that its part of the CST will remain the same, even if its CST Regional Root Bridge changes.

4 OPTIMIZATION ALGORITM

Consider an Ethernet network represented by a directed graph $G = (N,A)$ where N is the set of switches and A is the set of directions (i,j) of all links (E is the set of links $\{i,j\}$). The capacity of link $\{i,j\}$ is $b\{i,j\}$. Consider R regions defined on the network, each one identified with a positive number r between 1 and R. Each region is defined by the sub-graph $G_r = (N_r,A_r)$ where $N_r \subset N$ is the set of switches and $A_r \subset$ A the set of direction of links belonging to region r. The network supports a set of VLANs represented by V. Each VLAN $v \in V$ is characterized by a set of traffic flows $T(v)$ and each traffic flow $t \in T(v)$ is characterized by its origin switch $o(t)$, destination switch $d(t)$ and demand $b(t)$.

For a particular set of STs and a particular mapping of VLANs to STs, each traffic flow is routed through the routing path defined in the STs that its VLAN was assigned to. Assume that $a(v)$ indicates the ST assigned to VLAN v. Assume a binary parameter $s[(i,j), t]$ that is one if arc (i,j) is in the path of traffic flow $t \in T(v)$ defined by the ST $a(v)$. The load on arc (i,j) is the sum of the demands of all traffic flows that use it. Consider $l(i,j)$ the resulting load (in percentage):

$$l(i,j) = \frac{\sum_{v \in V} \sum_{t \in T(v)} \big(b(t)s[(i,j),t]\big)}{b\{i,j\}} \times 100\% \qquad (1)$$

The aim is the lexicographical minimization of the link loads. We define the load array L of a particular solution, the array which is formed by all $l(i,j)$ values given by (1) and sorted in a non increasing order. Given two solutions 1 and 2 whose load arrays are L_1 e L_2, solution 1 is better than solution 2 if L_1 has a smaller value than L_1 in the first array position whose values are different. The aim is to select the solution whose worst load is better; if the worst load is equal, the one whose second worst load is better; and so on… Maximizing the unused resources of the worst load links maximizes the network robustness to unpredicted demand growth and minimizes the amount of traffic that suffers service disruption when a single link fails.

The proposed algorithm is based on a GRASP (Greedy Randomized Adaptive Search Procedure) with a local search technique at each GRASP iteration. In the following description, Ω is a set of STs (one CST and n additional STs at each region) and Φ is an assignment array defined by a set in the form $a(1…|V|)$, where the element $a(v)$ indicates the ST assigned to VLAN v. A solution S is defined by a set of STs and an assignment array: $S = \{\Omega,\Phi\}$. The set *Neighbors*(S) is the set of all neighbor solutions of S defined as follows: $S' = \{\Omega', \Phi'\}$ is a neighbor of S if either it has the same set of STs (Ω' equal to Ω) and its assignment array Φ' differs from Φ in a single value or it has the same assignment array (Φ' equal to Φ) and its set of STs Ω' differ from Ω on a single link of a single ST. Considering S_{best} as the best solution found so far and L_{best} its load array, the proposed algorithm is then:

```
1:   Set all values of $L_{best}$ to $+\infty$
2:   **while** [Computing time is lower than *MaxTime*] **do:**
3:           Determine $S$ by generating randomly a set of STs Ω and an assignment array Φ
4:           Compute the load array $L$ of solution $S$
```

```
5:      repeat
6:          $S' \leftarrow S$
7:          for [$S'' \in Neighbors(S')$] do:
8:              Compute the load array $L''$ of solution $S''$
9:              if [$L''$ is better than $L$] then: $S \leftarrow S''$ , $L \leftarrow L''$
10:     until [$S'$ equal to $S$]
11:     if [$L$ is better than $L_{best}$] then: $L_{best} \leftarrow L$ , $S_{best} \leftarrow S$
```

In step 1, we start by setting all elements of L_{best} to $+\infty$ since there is no solution found yet. The **while** cycle (steps 2 to 11) is the GRASP main cycle and it runs until a maximum computing time (given by *MaxTime*) is reached. Inside the **while** cycle, first, a solution S is randomly generated (step 3) and its load array L is computed (step 4). Then, the **repeat** cycle (steps 5 to 10) implements the local search procedure: all neighbors of the present solution are computed (**for** cycle in steps 7 to 9) and, whenever a neighbor solution is better than the present one (step 9), it is stored as the present solution; the local search stops if there is no better neighbor solution (step 10). Finally, the present solution is stored as the best solution if it is better than the previously found best solution (step 11).

In step 3, a set of STs Ω is randomly generated (remember that this set is the CST and n additional STs for each region). We generate each additional ST on each region r as follows. First, we assign all nodes of N_r with a different label. Then, we repeat $| N_r | - 1$ times the following operations: (i) select randomly one link of A_r among all links whose end-nodes have different labels and (ii) choose the label of one end-node (of the selected link) and assign it to all nodes of N_r with the label of the other end-node. At the end, all nodes have a single assigned label and the selected links form a ST over all nodes of region r. Concerning the random generation of the CST, in order to agree with MSTP, its set of links must be an in-region Spanning Tree (as explained in Section 2), *i.e.*, it should form a ST on each sub-graph G_r. In order to guarantee this property, we generate a random CST using the same algorithm (as described before) applied to all nodes of N with the following adaptation: the algorithm starts by selecting links among the ones that belong to regions and, when all these links have the same label on their end-nodes, the algorithm proceeds selecting links not belonging to any region. In this way, we guarantee that all nodes inside each region are fully connected before the links outside regions start to be selected and, therefore, we guarantee the in-region property.

5 COMPUTATIONAL RESULTS

We have solved different random generated problem instances with the proposed algorithms. The case studies presented here consider the network shown in Figure 6, an Ethernet network with 23 nodes, 42 links (all links with a capacity of 1 Gbps) and 3 defined regions. Switches A to D, H to K and O to R are access switches where customer equipment is connected to and switches V and W are gateway switches that connect the network to other core networks. Concerning traffic demands, we have considered 51 VLANs (each VLAN with 2 traffic flows) randomly selected between all access switch pairs and all access-gateway pairs. Demand values were randomly selected among 4 different values (10, 20 50 and 100 Mbps) considering three different case studies: the percentage of total demand internal to regions is 20% for case study A, 50% for case study B and 80% for case study C.

All case studies were solved by the optimization algorithm proposed in Section 4 and, then, the obtained ST solutions were used to assign the appropriate MSTP parameters using the PortCost assignment algorithm presented in Section 3. The proposed algorithms were implemented in C programming language and were run in a PC with a 3.0 GHz Pentium 4 processor and 512 MB of RAM memory. The executable code runs on Windows OS and is available at [9]. In the following sub-sections, we make a separate analysis of the computational results in three different aspects.

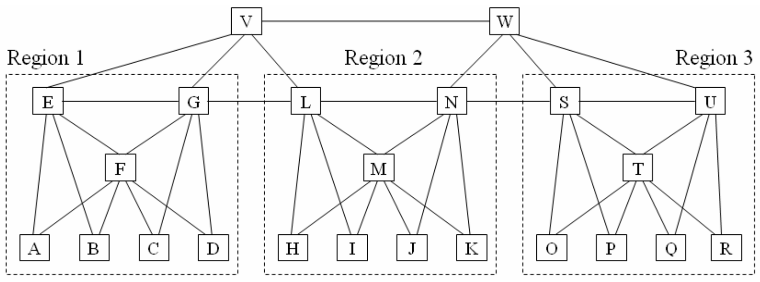


Figure 6: Topology of the Ethernet network used on the case studies.

5.1 Network Load Balance Analysis

We have solved all three case studies assuming a maximum number *n* of additional STs inside each region equal to 0, 1 and 2. Table 1 shows the 8 worst link load values of each obtained solution. These results show that one additional ST obtains: (i) a significant load balance improvement for case study C where traffic is mainly internal to regions (worst link loads decreased from 41% to 29%), (ii) a small load balance improvement for case study B (worst link loads decreased from 55% to 51%) and (iii) no significant improvement for case study A. These results show that adopting multiple regions limits the amount of load balance that can be obtained with additional STs. Significant load balance gains are obtained only in network scenarios with highly clustered traffic matrices provided that the adopted regions reflect the traffic clustering.

Table 1: Results of the case studies based on 3 regions.

		Index of first 8 positions of Load Array (%)							
Case Study	*n*	1	2	3	4	5	6	7	8
A	0	76.0	76.0	70.0	70.0	68.0	68.0	58.0	58.0
	1	76.0	76.0	68.0	68.0	59.0	59.0	58.0	58.0
	2	76.0	76.0	68.0	68.0	59.0	59.0	58.0	58.0
B	0	55.0	55.0	51.0	51.0	49.0	49.0	46.0	46.0
	1	51.0	51.0	49.0	49.0	43.0	43.0	36.0	36.0
	2	51.0	51.0	49.0	49.0	43.0	43.0	36.0	36.0
C	0	41.0	41.0	39.0	39.0	34.0	34.0	32.0	32.0
	1	29.0	29.0	22.0	22.0	21.0	21.0	21.0	21.0
	2	29.0	29.0	21.0	21.0	21.0	21.0	20.0	20.0

Table 2: Results of the case studies based on 1 region.

		Index of first 8 positions of Load Array (%)							
Case Study	*n*	1	2	3	4	5	6	7	8
A	0	70.0	70.0	68.0	68.0	59.0	59.0	58.0	58.0
	1	31.0	31.0	31.0	31.0	30.0	30.0	26.0	26.0
	2	31.0	31.0	31.0	31.0	30.0	30.0	26.0	26.0
B	0	55.0	55.0	51.0	51.0	46.0	46.0	43.0	43.0
	1	23.0	23.0	23.0	23.0	21.0	21.0	20.0	20.0
	2	23.0	23.0	23.0	23.0	20.0	20.0	20.0	20.0
C	0	41.0	41.0	39.0	39.0	34.0	34.0	32.0	32.0
	1	21.0	21.0	20.0	20.0	20.0	20.0	19.0	19.0
	2	21.0	21.0	20.0	20.0	20.0	20.0	19.0	19.0

We have also considered the same three case studies but assuming the whole network as a single region. The 8 worst link load values of each obtained solution are presented in Table 2. These results show that one additional ST obtains a significant load balance improvement for all case studies (worst link loads decreased from 70% to 31% in case study A, from 50% to 23% in case study B and from 41% to 21% in case study C). Comparing both tables, we observe than in all cases the single region case is better. This difference is very large in case studies A and B and it is only small in case study C since it is the case where traffic is mainly internal to regions and, as observed before, the additional ST was efficient in improving the load balance. Moreover, two additional STs do not provide significant better performance than one additional ST, which means that near optimal load balance is obtained with one single additional ST, thus, not penalizing significantly the switching processing overhead.

5.2 Service Disruption Analysis

We have performed a service disruption analysis of all best solutions with the MSTP parameters assigned by the algorithm of Section 3. For each solution and for each link that is used by at least one ST, we have computed (i) the new STs when this link fails, (ii) the traffic flows whose routing paths are changed in the new STs, (iii) the sum of the demands of such flows and (iv) the percentage of the total traffic that suffer service disruption. Then, we have determined (a) the average of all traffic percentages for all links that are used by at least one ST and (b) the percentage of all traffic demand that suffers service disruption in the worst case link failure (Table 3).

Table 3: Service disruption results.

		Average service disruption		Worst case service disruption	
Case Study	*n*	3 regions	1 region	3 regions	1 region
A	0	16.64	16.69	37.81	34.83
	1	9.59	9.29	37.81	15.42
	2	9.59	8.34	37.81	15.42
B	0	16.19	16.51	27.36	27.36
	1	8.90	7.72	25.37	11.44
	2	8.42	7.52	25,4	11.44
C	0	13.25	13.25	20.40	20.40
	1	7.89	6.72	14.43	10.45
	2	7.60	6.81	14.43	10.45

In both average and worst case analysis, the results show one important observation. The single regions approach is always better when additional STs are considered (even in case study C where traffic is mainly internal to regions). This might seem unexpected since the multiple region scenarios guarantee that link failures inside a region do not change the STs outside that region. Nevertheless, the better network load balance and the proposed MSTP parameter assignment algorithm, that minimizes the number of links changing their state when a failure occurs, results in better service disruption by the single region approach. Moreover, two additional STs do not provide significant better service disruption than one additional ST, which means that, like in the previous analysis, near optimal solutions are obtained with one single additional ST, thus, not penalizing significantly the switching processing overhead.

5.3 Algorithms Performance Analysis

The MSTP parameter assignment algorithm proposed in Section 3 has a polynomial complexity relative to both the number of switches and number of links and, therefore, it runs with very low computational times (below a tenth of a second). Concerning the efficiency of the optimization algorithm proposed in Section 4, we have run the algorithm in all cases with *MaxTime* equal to 30 minutes. In practice, a solution that minimizes a small number of the worst load values (lets say the 6 worst load values) is enough since the remaining values become small. With this criterion, a lot less computing time is required. The proposed algorithm is a stochastic process, which means that different runs might give different results. We have run 5 times all cases presented here and in all runs, a solution equal to the best solution in the 6 worst load values was always found below 2 minutes of computing time for the cases with 1 and 2 additional STs (the cases without additional STs are much easier to solve). These times make us confident that the obtained results are indeed optimal for the 6 worst case load values (further work is required to obtain theoretical bounds that can be used to prove near optimality of the solutions).

6 CONCLUSIONS

In this paper, we have addressed the problem of how to use the IEEE 802.1S MSTP to improve load balance and service disruption in Ethernet networks. We have proposed an MSTP parameter assignment algorithm that implements any desired set of STs minimizing the number of links changing their state between active and backup when a single link failure occurs (the so-called minimum service disruption property). We have also proposed an optimization algorithm to determine the set of STs that optimize network load balance and service disruption. The computational results show that the combination of the two algorithms is very efficient and able to obtain good solutions within very short computing times. The MSTP parameter assignment algorithm (presented in Section 3) is an important result by itself since it enables the determination of the appropriate MSTP parameters for other sets of STs that might be computed by any other methodology, possibly addressing other optimization objectives. We have studied the relationship between load balance and service disruption under single link failures when either a single region or multiple regions scenarios are adopted. We showed that the single region approach is always better when MSTP is optimally configured and that a very few number of additional Spanning Trees can obtain near optimal solutions. Note that due to length limit, we have presented only part of a much larger set of case studies. Nevertheless, the conclusions drawn by the presented results also stand on all other computational results.

REFERENCES

[1] IEEE Standard 802.1S, "Virtual Bridged Local Area Networks – Amendment 3: Multiple Spanning Trees, 2002

[2] A. de Sousa, G. Soares, "Improving Load Balance of Ethernet Carrier Networks using IEEE 802.1S MSTP with Multiple Regions", 5th Int. IFIP-TC6 Networking Conference, LNCS, Vol. 3976, Springer, 1250-1260, 2006

[3] M. Ali, G. Chiruvolu, A. Ge, "Traffic Engineering in Metro Ethernet", IEEE Network, Vol. 19, No. 2, 10-17, 2005

[4] A. Kolarov, B. Sengupta, A. Iwata, "Design of Multiple Reverse Spanning Trees in Next Generation of Ethernet-VPNs", IEEE GLOBECOM'04, Vol. 3, 1390-1395, 2004

[5] S. Sharma, K. Gopalan, S. Nanda, T. Chiueh, "Viking: A Multi-Spanning-Tree Ethernet Architecture for Metropolitan Area and Cluster Networks", IEEE INFOCOM'04, Vol. 4, 2283-2294, 2004

[6] M. Padmaraj, S. Nair, M. Marchetti, G. Chiruvolu, M. Ali, "Traffic Engineering in Enterprise Ethernet with Multiple Spanning Tree Regions", Proc. of System Communications (ICW'05), Montreal, Canada, 261-266, 2005

[7] K. Ishizu, M. Kuroda, K. Kamura, "SSTP: an 802.1s Extention to Support Scalable Spanning Tree for Mobile Metropolitan Area Network", IEEE GLOBECOM'04, Vol. 3, 1500-1504, 2004

[8] Y. Lim, H. Yu, S. Das, S.-S. Lee, M. Gerla, "QoS-aware multiple spanning tree mechanism over a bridged LAN environment", IEEE GLOBECOM'03, Vol. 6, 3068-3072, 2003

[9] http://www.av.it.pt/asou/mstp.html