Hashing based traffic partitioning in a multicast-multipath MPLS network model

Xavier Hesselbach
Technical University of Catalonia
Jordi Girona,1-Mod.C3-Campus Nord
Barcelona 08034 (Spain)
+ 34 93 401 59 87
xavierh@entel.upc.edu

Ramón Fabregat
IIiA University of Girona
EPS – P4 - Campus Montilivi
Girona 17071 (Spain)
+ 34 972 41 84 84
ramon@eia.udg.es

Benjamín Baran
Centro Nacional de Computación
National University of Asuncion
San Lorenzo (Paraguay)
+ 595 21 585550
bbaran@cnc.una.py

Yezid Donoso
Universidad del Norte
Km.5 Vía Puerto Colombia
Barranquilla (Colombia)
+ 57 5 3509509
ydonoso@uninorte.edu.co

Fernando Solano
IIiA University of Girona
EPS – P4 - Campus Montilivi
Girona 17071 (Spain)
+ 34 972 41 84 75
fernando@eia.udg.es

Mónica Huerta
Technical University of Catalonia
Jordi Girona,1-Mod.C3-Campus Nord
Barcelona 08034 (Spain)
+ 34 93 401 59 94
mhuerta@entel.upc.edu

ABSTRACT

Load Balancing is a key mechanism in traffic engineering. One interesting strategy for load balancing enhancement is the multipath approach, in which data is transmitted through different paths. The use of effective hashing functions for load balancing optimizes the network utilization and reduces packet disordering and imbalance. This paper address the problem of packet ordering in multipath – multicast MPLS networks, studies the impact of the hashing function to effectively partition the traffic to implement the flow splitting values issued from an optimized model and analyzes the traffic allocation to the LSPs of the network and the mis-ordering problem at the egress node using buffer schemes. The buffer allocation levels are calculated according to end-to-end delays. Finally, the paper presents some experimental results from an optimized network.

Categories and Subject Descriptors

C.2.1 [**Network Architecture and Design**]: Network communications. C.2.2 [**Network Protocols**]: Routing protocols. C.2.3 [**Network Operations]**: Network management.

General Terms

Management, Design.

Keywords

Multicast traffic, traffic engineering, load balancing, multiobjective, splitting, hashing function, state dependent.

1. INTRODUCTION

The emergence of Multiprotocol Label Switching (MPLS) technology provides mechanisms in IP backbones for explicit routing using virtual circuits called Label Switched Paths (LSP), encapsulating the IP frame in an MPLS packet, where a label defines the end-to-end virtual circuit in the MPLS network. Additionally, it facilitates traffic engineering (TE). Traffic engineering is concerned with improving the performance of operational networks, usually taking into account QoS (Quality of Service) requirements.

The main objectives are to reduce congestion hot spots, improve resource utilization and provide adequate QoS for final users. These aims can be achieved by setting up explicit routes through the physical network in such a way that the traffic distribution is balanced across several traffic trunks, giving the best possible service, i.e., minimum delay, packet losses, jitter, etc. Therefore, traffic may be controlled to flow optimally through certain routes.

Load Balancing (also known as Load Sharing or Traffic Splitting) is a fundamental mechanism to implement traffic engineering. For a load balancing model to be general, unicast considerations are not enough and multicast should also be considered. One interesting option for load balancing is the multipath approach, in which data is transmitted through different paths.

In a previous work [13], a multiobjective traffic engineering scheme (GMM-model) using different distribution trees to multicast several flows has been proposed. Solving the GMM-model allows to compute the fraction of flow demanded from the ingress node to the set of egress nodes assigned to each link. In this paper we propose effective hashing strategies to handle traffic partitioning obtained by GMM-model.

In Section 2, an overview of traffic engineering and load balancing is presented. Section 3 overviews the GMM-model. According to optimum flow values obtained by calculated from the GMM-model. Section 4 analyses partition of traffic strategies. The use of the hashing function is studied in section 5. In Section 6, is evaluated the dimensioning of buffers at the egress nodes for packets re-ordering. Section 7 presents some results considering a numerical example. Final conclusions and future work are left for section 8.

2. TRAFFIC ENGINEERING

In conventional IP forwarding, a particular router will typically consider two packets to be in the same Forwarding Equivalence Classes (FECs) if there is some address prefix X in that router's routing tables such that X is the "longest match" for each packet's destination address. As the packet traverses the network, each hop in turn reexamines the packet and assigns it to a FEC.

In Multiprotocol Label Switching (MPLS), the assignment of a particular packet to a particular FEC is done just once, as the packet enters the network. The FEC to which the packet is assigned is encoded as a short fixed length value known as a "label". When a packet is forwarded to its next hop, the label is sent along with it; that is, the packets are "labeled" before they are forwarded. At subsequent hops, there is no further analysis of the packet's network layer header. Rather, the label is used as an index into a table which specifies the next hop, and a new (or the same) label. The old label is replaced with the new label, and the packet is forwarded to its next hop.

Basis for MPLS load balancing is the fact that a backbone technology connecting several Internet Service Providers (ISPs) provides multiple paths in order to guarantee an appropriate redundancy level and capacity enough. Therefore, these different paths can be used to provide a balanced traffic splitting over the links.

Several advantages of using multipath routing are discussed in [1] and [2]. In short, links do not get overused and therefore do not get congested, and so they have the potential to aggregate bandwidth, allowing a network to support a higher data transfer than with any single path. Furthermore, some authors have expanded this idea by proposing to split each flow into multiple subflows in order to achieve better load balancing [3], [4], [5] and [6].

Additionally, multiple paths can be used as backup paths for one primary path when paths are configured maximally disjoint. Each working path is protected by an alternative disjoint path. If the primary path goes down, alternative paths can quickly be deployed. Therefore, load balancing emerges as a fast response path protection mechanism and the flow splitting approach can be used for protecting the path. Splitting the working path has the advantage of reducing the amount of bandwidth to be protected [2].

One of the most difficult issues in load balancing is how to guarantee the end to end delays among several Label-Switched Path (LSPs) while maintaining the packet sequence. This requirement is especially critical for the network throughput for TCP protocol, because packet disordering can produce false congestion detections.

The load balancing techniques are usually classified in two groups [7]: *Based in connection*, which are represented by some parameters and routing decisions that affect the whole flow; and *based in packet*, where decisions depend only on every packet and is therefore simpler. This is the technique considered in this paper.

When an IP packet ingresses into the MPLS domain, the ingress Label Edge Router (LER) analyzes the header. Depending on the information of destiny, type of traffic, etc. an MPLS label value is assigned. It consists on an identifier of fixed length, associated to the path that the packet will have to take into account in order to reach the exit node [8], [9] and [10].

According to the scale where applied, the TE mechanisms are classified in two basic types, [11]:

- In the Time Dependent case, the traffic control algorithms optimize the network resources in response to traffic variations in long time range.
- In the State Dependent case, the traffic control algorithms responds immediately to state variations in the network. In other words, it adapts changes in a short time range.

According to load balancing goals, the State Dependent is the type used in this work.

A load balancing system is formed by a Traffic Splitter and several Outgoing Links [12] (figure 1).

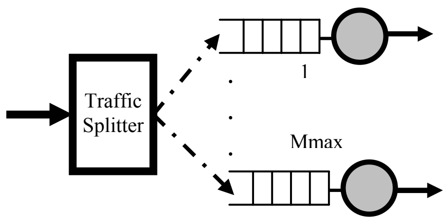

Figure 1. Reference model for Load Balancing.

Conceptually, the input traffic is partitioned according to certain criteria (that will be discussed later) into Mmax bins at the MPLS ingress node.

The value for Mmax will be considered in the section 5.1 and is fundamental for the hash function. The Mmax bins are mapped onto the pre-established LSPs. The fraction assigned to each LSP could be calculated using a mathematical optimization model.

Specifically, a load balancing mechanism is functionally constituted by a Splitting Function and an Allocation Function (figure 2). The Splitting Function splits the incoming traffic to the outgoing links, guaranteeing the order of packets. The Allocation Function determines the LSP where every packet have to be forwarded to the egress router and the moment when it must be delivered.

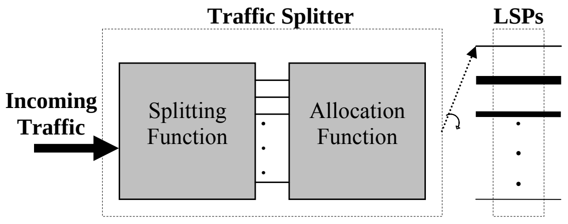

Figure.2. Functional model for Load Balancing.

3. GMM-MODEL

Inspired by the TE load balancing taxonomy, [13] proposes a Generalized Multiobjective Multitree model (GMM-model) in a pure multiobjective context that, for the first time, considers simultaneously multicast flow, multitree, and splitting.

When load balancing techniques are translated into a mathematical formulation different conflicting objectives are found. The GMM-model follows the general mathematical framework of any Multi-Objective Problem (MOP). This model considers 11 objective functions: maximal link utilization, total hop count, hop count average, maximal hop count, maximal hop count variation for a flow, total delay, average delay, maximal delay), maximal delay variation for a flow, total bandwidth consumption, and number of subflows; and several constraints: flow conservation constraints, a subflow uniformity constraint, to ensure that a subflow f_k always transports the same information, link capacity constraint and constraint on the maximum number of subflows. Clearly, it is not difficult to increase the number of objectives or constraints of the proposed model if new ones appear in the literature or they are useful for a given situation.

The GMM-model considers a network represented as a graph G(N,E), with N denoting the set of nodes and E the set of links. The cardinality of a set is denoted as | |. The set of flows is denoted as F. Each flow $f \in F$ can be split into K_f subflows that after normalization can be denoted as f_k, k = 1, … K_f, which indicates the fraction of f transported.

Among the nodes in the network, T is the set of destination nodes (egress nodes). Let $t \in T$ be any egress node. In a multicast transmission, $f_k t$ denotes de multicast subflow f_k to the destination egress node t.

The solution of the GMM-model, $X_{ij}^{f_k t}$, is the fraction assigned to each Point-to-Multipoint (P2MP) LSP by load balancing system. In particular, $X_{ij}^{f_k t}$ denote the fraction of subflow f_k to egress node t assigned to link (i,j) $\in$ E, i.e. $0 \le X_{ij}^{f_k t} \le 1$. Note that $X_{ij}^{f_k t}$ uses five indexes: i, j, f, k and t for the first time, unlike previous publications that only used a smaller subset of indexes because they did not deal with the same general problem [4] and [16]. The novel introduction of a subflow-index k gives an easy way to identify subflows and define LSPs in a MPLS implementation. $X_{ij}^{f_k t}$ values are the fractions assigned to each LSP by load balancing system.

It is important to point out that the mathematical solution of the GMM-model is a set of Pareto optimal solutions [13]. In this set no one solution can be considered better than the others if all the objectives are taken into consideration at the same time. They derives from the fact that there may exist - and in general there exist - a conflict between the different objectives that make up the problem. For this reason, when dealing with MOPs a new concept of “optimal” is necessary [14].

In [19], the GMM-model is solved with a multi-objective evolutionary algorithm (MOEA) inspired by the Strength Pareto Evolutionary Algorithm (SPEA) [15].

4. PARTITION OF TRAFFIC

If the topology is such that equal cost paths exist, then at least 2 methods can be used for partitioning traffic among paths [17]:

- Simple methods to partition the input traffic using a per-packet basis are round robin, weighted round robin and weighted fair queuing, for example. These methods do not take into account specific parameters for load balancing, and usually suffer in practice from the possibility of excessive packet reordering. Also, they are only applicable if the delays on the paths are almost equal.
- The second method consists on filtering the ingress traffic using a hash function on a subset of the IP packet header fields [18] (commonly, at least, the source and destination addresses, but more frequently considering the 5-tuple formed by source and destination address, source and destination socket, and protocol identification) and taking decisions based on the network state. In this method, the hash space is split among the available paths by setting thresholds or performing a modulo operation, which is more frequently used. Taking into account that the incoming flow is in general an aggregated traffic formed by several end-to-end communications, the traffic can be split considering those individual connections (i.e.: TCP connections). Therefore, in this scenario the individual connections keep the packet sequencing and no packet reordering is required for each individual flow $f_k t$. However, a general reordering method is required for rebuild the original flow f.

Hence, traffic between any source and destination nodes is transported over the same path. This alternative is detailed in the next section.

5. HASHING FUNCTIONS

Hashing based traffic partitioning algorithms are simple to compute and independent of the state of the network.

A good hash function satisfies the assumption of simple uniform hashing, that is, each key is equally likely to hash to any of the L outgoing links, independently of where any other key has hashed to. In practice, heuristic techniques are frequently used to define a hash function that works well.

Hashing schemes for load balancing can be classified in Direct Hashing (the splitter considers simply a hash function using a set of fields in the data packet in order to split the traffic) and Table-Based Hashing (that first splits the traffic in M bins and then maps them to L outgoing links based on an allocation table, as represented in Figure 2) [12]. The last one is of course less simple than Direct Hashing.

Direct Hashing can be divided in the following methods:

- The division method: h(w) = w mod M. This method is quite fast. It is well known that it is convenient not to use a power of 2 for M, when low order bits ok w are not fair enough, since if M=2p, then h(w) is the p-lowest-order bits of w. In this situation, a prime not too close to an exact power of 2 is often a good choice for M.
- The multiplication method: $h(w) = \lfloor M\ (wA \bmod 1)\rfloor$, where (wA mod 1) means the fractional part of wA. (A is a constant, 0<A<1). In this method, the value of M is not critical, so in this sense, is better than the division method. However, the most suitable value for A depends on the characteristics of the data to be hashed.
- Universal hashing: This method aims to avoid vulnerability in the sense that several keys hash to the same link. The only effective way is to choose the hash function randomly in a way that is independent of the keys that are actually going to be stored.

However, these methods do not take into account the partition of traffic according to unequal weights.

Studies on performance of hashing based schemes for load balancing can be found. Since [12], hashing with only the destination IP address causes significant imbalance across two links. Order in packet delivering is preserved if hash function considers the 5-tuple of the packet headers (at network and transport layer): source and destination address, source and destination socket, and protocol identification. This is really effective in a network trunk scenario, where many connections share the same link. Three alternatives offer good performance:

- Using the Internet checksum or the exclusive OR of both the source IP address and destination IP address improves the performance significantly, though moderate imbalance persists.
- Also, the 16-bit CRC using the 5-tuple gives excellent load balancing performance, and keeps the load and queue lengths very similar on two links. Unfortunately, is computationally complex. In this scheme, the traffic splitter computes CRC16 for the 5-tuple and takes the modulo M in calculate the outgoing link. The hash function is H(.)°= CRC16(5-tuple)°mod M. This is the most frequently used expression for hashing. In case of use it, it is recommended not to select a power of 2 (as seen before) for M. When possible, choose for M a prime not to close to a power of 2. For instance, for a desired M = 16, is better to use 13 or 11. In this case, expression is finally H(.) = CRC16(5-tuple) mod 13.

Hence, from this discussion, a simple way to implement this is to associate an outgoing link with an amount of M bins.

When a packet arrives, the hash value is computed (using the effective 16-bit CRC using the 5-tuple, H(.) = CRC16(5-tuple) mod M, for instance). M defines the granularity of the adjustment.

Then using the hash value, the splitted traffic is sent to a cluster of M buffers according to the hash value, referenced by H_i, $i \in [1..M]$.

By changing the allocation of the bins to the outgoing links (changing the Hi values), traffic can be distributed in a pre-defined ratio. Figure 3 shows the detailed table for the model of figure 2.

Finally, the traffic must be allocated in the LSPs of the network. Therefore, the traffic coming from the set of M queues will be distributed according to the traffic partition indexes $X_{ij}^{f_k t}$ among the N network links.

At each buffer in the allocation function box, the packets are ordered according to the splitting function. These sets of buffers are FIFO queues. We consider in this paper for simplicity transmission according to a WFQ (weighted fair queuing) model at every outgoing LSP.

The packets from every outgoing LSP are selected taking into account the available bandwidth $X_{ij}^{f_k t}$ at each outgoing link.

The selection of packets coming from the M queues is made using the criteria of the Dynamic Load Balancing algorithm [20].

Therefore, some degree of disordering is produced, and reordering buffers are required at the destination nodes.

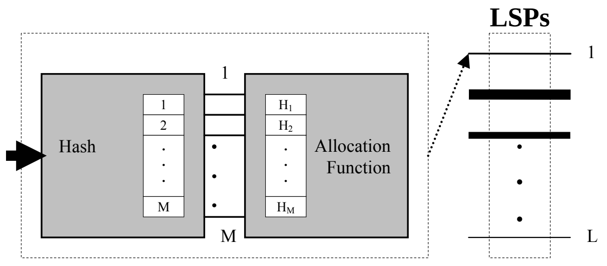

Figure. 3. Functional table-based hashing.

5.1 The M and L values

The L value is the number of different paths that can be established from the source ingress router to a certain egress router. Therefore, L depends on the topology of the network.

At the other hand, the value of M can be tuned according to the allocation function and the Load Balancing Mechanisms.

In general, it is recommended that M should be greater or equal than L ($M \geq L$). This criterion permits mapping between M and L, easily.

Let's consider M separated flows coming from the hashing function, ready to be allocated to one or more of the L disjoint paths in the network. Therefore, there is some granularity in the allocation that come decrease the overall throughput of the network. According to this, in general, M should be several times greater than L in order to minimize the remainder unused bandwidth.

In this paper, we consider that perfect allocation can be done, and practical allocation methods (let's say PGPS for packet-by-packet Generalized Processor Sharing [21] - WFQ, Weighted Fair Queuing -, or other ones) are left for a future work.

6. BUFFER ALLOCATION

The scheme presented may require packets reordering at the egress node to avoid packet disordering. A buffer is therefore effective.

Taking into account that faster paths require buffer allocation in order to synchronize packets received from slower paths, we need to calculate the buffer size required. We include the following notation:

Consider a single flow f. For a f_k subflow from this flow f to an egress node t, the end to end delay is:

$d^{f_k t} = \sum_{(i,j)\in E} d_{ij} \cdot Y_{ij}^{f_k t}$, where according to our model, d_{ij} is the delay (in ms.) of each link (i,j).

The delay for slowest f_k belonging to the flow f to egress node t is:

$$d_{slowest}^{f_k t} = \max_{\forall k \in K_f}(\{d^{f_k t}\}),$$

and let $f_k t_{slowest}$ the f_k with a $d_{slowest}^{f_k t}$ delay.

Then buffer size $B^{f_k t}$ required for each f_k flow is:

$$B^{f_k t} = (d_{slowest}^{f_k t} - d^{f_k t}) \cdot (b_f \cdot X_{closest\_link\_to\_node\_t}^{f_k t})$$

$b_f \cdot X_{closest\_link\_to\_node\_t}^{f_k t}$ is the bit rate arriving to node t from flow f_k. Note that a buffer for the slowest path is not required.

And the total buffer size in an egress node t for a single flow f is:

$$B^{ft} = \sum_{k=1}^{|K_f|} B^{f_k t}$$

7. NUMERICAL EXAMPLE

This section presents a simple problem, the corresponding results using the mathematical model and the table based hashing function considering $X_{ij}^{f_k t}$ coefficients. The chosen topology is the well-known 14-node NSF (National Science Foundation) network shown in figure 4 (|N|=14). The costs on the links represent the delays (d_{ij}) and all links are assumed to have 1.5 Mbps of bandwidth capacity (c_{ij} = 1.5 Mbps (i,j)∈E).

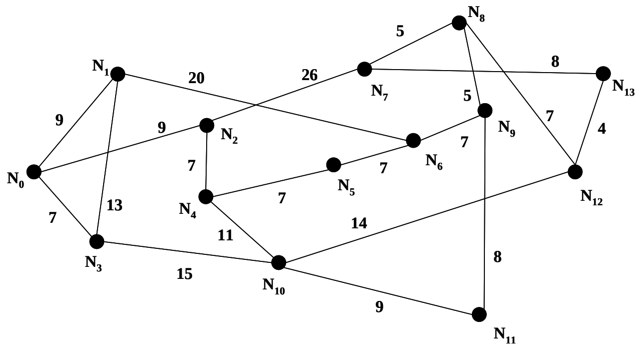

Figure 4. NSF network

Two flows with the same source, s_f=N_0, are considered. The egress subsets are T_1={N_5, N_9} and T_2={N_4, N_9, N_{12}}. The transmission rates are b_1=256 Kbps for the first flow and b_2=512 Kbps for the second flow. In this case, flow 1 is transmitted without splitting (fraction 1_1 = 1.0) while flow 2 is split in two subflows transmitting 256 Kbps per subflow (fraction 2_1 = fraction 2_2 = 0.5) (see figure 5).

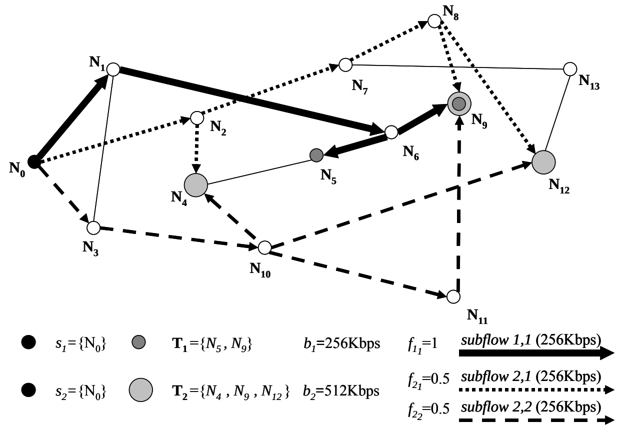

Figure 5. Subflows fraction representation

According to these values, we can calculate buffer allocation required in each egress node (Table 1).

For first flow, there is no multipath transmission. The egress nodes N_5 and N_9 receive only one flow contribution and therefore, no buffer is required for packets re-ordering.

For the second flow, 3 egress nodes and two subflows are generated for each egress node. In this case, $d^{f_1 4} = \sum_{(i,j)\in E} d_{ij} \cdot Y_{ij}^{f_1 4}$ = $d_{0,2}$ + $d_{2,4}$ = 9 + 7 = 16.

In the same way for the rest, we obtain the values of table 1.

Table 1. Buffer allocation required in each egress node.

	d_kt *(ms)*	max	Partial Buffer (bits)	Total buffer (Σ) for egress node t
$f_1$4	16		(33-16)·256 = 4352	4352 bits
$f_1$4	33	33	0	
$f_1$9	45	45	0	3584 bits
$f_2$9	31		(45-31)·256 = 3584	
$f_1$12	47	47	0	2816 bits
$f_2$12	36		(47-36)·256 = 2816	

In order to restore the packet sequencing, the packet number can be considered. This is a light processing load for the egress node.

8. CONCLUSIONS

In this paper, we have considered a novel Generalized Multi-objective Multitree model to examine the hashing function in the case of multicast and multipath MPLS network. We considered classical hashing methods and, according to the results of the GMM model, applied the load balancing coefficients to hash appropriately the incoming traffic.

We have found that the table based hashing should be employed, better than direct hashing, due to load distribution according to unequal weights. Hashing values are tuned according to the $X_{ij}^{f_kt}$ parameters calculated from the network optimization. The buffer size for the egress node has been examined, and some experimental results from the GMM model have been presented.

Next works will evaluate the overall network throughput using the hashing based traffic partitioning in a multicast-multipath MPLS network model, compared with traditional not-multipath routing.

The M parameter should be tuned. It will depend on the mapping and the allocation functions used.

Additionally, buffer management at the splitter should be analyzed in a future work in order to minimize the buffer size at the egress routers.

9. ACKNOWLEDGMENTS

Our thanks to the Spanish national projects MCyT-TIC2003-08129-C02-02 and MCyT-TIC2003-05567 for their support in this work.

10. REFERENCES

[1] J.C. Chen, and S.H. Chan. "Multipath Routing for Video Unicast over Bandwidth-Limited Networks". GLOBECOM 2001.

[2] R. Izmailov and D. Niculescu. “Flow splitting approach for path provisioning and path protection problems”. HPSR 2002.

[3] Y. Donoso, R. Fabregat, J. Marzo. “Multi-Objective Optimization Algorithm for Multicast Routing with Traffic Engineering”. IEEE ICN 2004.

[4] Y. Donoso, R. Fabregat, L. Fàbrega. “Multi-Objective Scheme over Multi-Tree Routing in Multicast MPLS Networks”. ACM/IFIP LANC 2003.

[5] J. Kim, C. Kim, S. Seok, C. Kang. “Traffic Engineering using Adaptive Multipath-Forwarding Against Dynamic Traffic in MPLS Networks”. IEEE ICN 2004.

[6] C. Kim; Y. Choi; Y. Seok. Y. Lee. “A Constrained Multipath Traffic Engineering Scheme for MPLS Networks”. ICC 2002.

[7] Casellas R., Rougier Louis J., and Kofman D. “Packet Based Load Sharing Schemes in MPLS Networks”. 2nd. European Conference on Universal Multiservice Networks. Colmar, France. April 2002.

[8] Black, Uyless. “MPLS and Label Switching Networks” Second Edition. Prentice Hall, 2002.

[9] Rosen E. et al, “Multiprotocol Label Switching Architecture”, RFC 3031, Jan. 2001.

[10] Awduche D. O. and Jabbari B.. “Internet traffic engineering using multi-protocol label switching (MPLS)”. Computer Networks, Vol. 40, Issue 1, september 2002.

[11] Awduche, D. et al., “Overview and Principles of Internet Traffic Engineering”, RFC 3272, May 2002.

[12] Cao Z., Wang Z. and Zegura E., “Performance of Hashing-Based Schemes for Internet Load Balancing”, INFOCOM 2000.

[13] B. Barán, R. Fabregat, Y. Donoso, F. Solano, J.L. Marzo. “Generalized Multiobjective Multitree model”. IIiA Research Report, ref. IIiA 04-08-RR, Institut d'Informàtica i Aplicacions, University of Girona.

[14] D. A. van Veldhuizen. “Multiobjective Evolutionary Algorithms: Classifications, Analyses and New Innovations”. Ph.D. thesis. Air Force Institute of Technology, 1999.

[15] E. Zitzler and L. Thiele. “Multiobjective Evolutionary Algorithm: A comparative case study and the Strength Pareto Approach”. IEEE Trans. Evolutionary Computation. Vol. 3, No. 4, 1999.

[16] Y. Seok, Y. Lee, Y. Choi, C. Kim. “Explicit Multicast Routing Algorithms for Constrained Traffic Engineering”. IEEE ISCC 2002.

[17] Deyun Gao; Yantai Shu; Shuo Liu; Yang, O.W.W.; “Delay-based adaptive load balancing in MPLS networks”, Communications, 2002. ICC 2002. IEEE International Conference on , Volume: 2 , 28 April-2 May 2002.

[18] Villamizar C., “OSPF optimized multipath (OSPF-OMP)” Internet draft <draft-ietf-ospf-omp-02.txt>, feb. 1999.

[19] B. Barán, R. Fabregat, Y. Donoso, F. Solano, J.L. Marzo. "Generalized Multiobjective Multitree model solution using MOEA". Accepted at 6th WSEAS Int.Conf. on EVOLUTIONARY COMPUTING (EC '05). 16 -18. June 2005

[20] Long K., Zhang Z. and Cheng S., “Load Balancing in MPLS Traffic Engineering”, IEEE Workshop on High Performance Switching and Routing, Dallas May 2001.

[21] Parekh, A.K.; Gallager, R.G., “A generalized processor sharing approach to flow control in integrated services networks: the single-node case”. Networking, IEEE/ACM Transactions on. Volume 1, Issue 3, June 1993 Page(s):344 – 357.