Resource Allocation in Multiservice MPLS*

Magda Chatzaki, Stelios Sartzetakis, Nikos Papadakis+, Costas Courcoubetis+

Telecommunications and Networks Division
Institute of Computer Science - Foundation of Research and Technology Hellas
{magda, stelios, npapadak, courcou}@ics.forth.gr

ABSTRACT

This work deals with the problem of routing flows with QoS requirements across MPLS capable networks. We identify the functional capabilities that should be supported in order to optimize network resource utilization, satisfy business or service or network level QoS requirements. We focus on the design of an appropriate architecture to combine MPLS with QoS-Policy based routing capable components. Existing label switching technologies assume that routing is accomplished by just using traditional network layer routing protocols. Consequently network administrators have no control on the way traffic navigates the network. This can easily lead to unpleasant situations where some parts of the network are congested and some others are underutilized. Future Internet imposes the demand for advanced network control and management mechanisms. Combining label switching technologies with routing mechanisms that take into account network performance metrics and at the same time give network administrators the ability to influence routing decisions according to their preferences appears to be a very clear solution to cost-effectively and efficiently engineer future Internet. In order to accomplish that, we endorse some of the already proposed extensions to MPLS, and introduce a routing selection mechanism based on implied costs. Such implied costs reflect the network congestion as well as the interdependencies among flows in the network, and can provide the management signals for expanding or reducing the amount of resources allocated to different service classes and routes as well as performing the route selection.

1. Introduction

The traditional Internet that supports only best effort service transforms very quickly into a commercial environment demanding support for quality of service. Furthermore new compound services that start to appear, such as, multiparty collaborative working sessions, videoconferencing, and especially Internet telephony, have to be provided with the required total QoS in attractive prices, in order to create demand and subsequently increase optimized network usage. ISPs need mechanisms to differentiate between services and guarantee appropriate QoS. On the other hand, new telecom operators aim for new IP-optimized networks where they own the transport in order to: control the pace of deployment, control the costs and support PSTN quality voice and fax, in addition to the traditional Internet services.

In order for a network to guarantee customers' QoS requirements, it should be able to reserve resources and exercise network control functionality at different time scales. In connection oriented networks, the transfer of information between two end-users is accomplished by network functions that select and allocate network resources along an acceptable path. In connectionless networks this is an emerging need. The goal of QoS provisioning is to provide better and more predictable network service by providing dedicated bandwidth, controlled jitter and latency, and improved loss characteristics. These goals can be achieved by developing tools for managing network congestion, for shaping network traffic, for using the expensive wide area links more efficiently, and for setting traffic policies across the network. Furthermore a QoS provisioning framework should provide differentiation between the services (as far as transfer characteristics are concerned) and above all differentiation among the customers in terms of services they may request.

IETF has already proposed several frameworks and mechanisms for QoS support i.e. Integrated Services and RSVP [18] framework, Differentiated Services framework [17], MPLS (Multi-Protocol Label Switching) [10] and QoS based Routing frameworks [14][15].

Integrated Services framework is closely related with the definition of new classes of service and with resource allocation. When a real-time application requests a connection it should reserve resources on existing routes.

* This work was supported in part by the European Commission under ACTS Project IthACI

+ Also with the Dept. of Computer Science, University of Crete.

This is accomplished by using RSVP as the signaling protocol.

Differentiated Services framework supports a discrete model for service differentiation. Packets are classified and marked at the ingress edge routers of the network. This way, packets are assigned to different service classes that receive different quality of service.

MPLS is a packet label-based switching technique. Packets are assigned a label as they enter a MPLS capable network. Subsequent packet treatment is based on the labels only.

QoS-Policy based Routing is a routing process that is able to find paths that are subject to multiple quantitative (i.e. bandwidth, delay requirements) as well as qualitative (i.e. according to a number of high level admission control policies) constraints.

Integrated Services and Differentiated Services provide a framework for service classification, differentiation and prioritization, with the objective to guarantee network performance, avoid network congestion and satisfy customer requirements. MPLS and QoS-Policy based routing provide the mechanisms to control and influence the way traffic is forwarded across the network with the same objectives as the former two.

The focus of this paper is on the enhancements to the MPLS with resource allocation among several service classes and virtual links in order to control the way traffic flows in the network. Existing label switching technologies assume that routing is accomplished using traditional network layer routing protocols. Consequently network administrators have no control on the way traffic navigates the network. This can easily lead to unpleasant situations where some parts of the network are congested and some others are underutilized. We propose to combine label switching technologies with routing mechanisms that take into account on-line network performance metrics. In order to avoid the overhead of continuous monitoring of the link traffic we calculate on the fly the link usage based on the effective bandwidth of the already routed calls. One can use the recently appeared QoS policy editing tools to assign flows to classes according to administrative policies.

We propose to apply a route selection mechanism based on ***implied costs*** defined for circuit switched networks [20] (recently applied to ATM [19] as well). It has been demonstrated that implied costs reflect the network congestion as well as the interdependencies among flows in the network. Implied costs are independent of the way routes are computed but influence the way routes are selected. Whenever there is an incoming request a route should be selected for servicing this request. The selection is made among an initial set of routes that have been computed in advance. For each such route we compute the net benefit of the network provider if it routes the request through the specific route. Implied costs are used for the net benefit computation as we explain in section 2.2. They have been used successfully in the past in circuit switched networks. One key issue is whether the semantics of circuit switched networks where implied costs are defined, carry naturally for the case of networks using MPLS. But this is largely the case due to the connection oriented nature of the majority of the network flows that will use the MPLS network. Another issue concerns scalability. When the number of nodes becomes large, then the calculation of implied costs becomes prohibitive. There are recent results that use aggregation and extend the work in [20] for networks using hierarchical routing, see [19].

In the following section we review the concept of MPLS and stress the added value if combined with QoS-Policy based routing mechanisms. We also review the background theory in implied costs and propose it's use in MPLS. In section 3 we identify the functional and architectural enhancements in order to support this combination. For every component of the architecture we describe its functionality and we discuss implementation alternatives.

2. Motivation

2.1 Label Switching, Routing and Traffic Engineering in MPLS

In connectionless network routing protocols packet forwarding decision is taken independently at each hop as the packet is sent from one hop to the next. In the traditional IP forwarding paradigm each router forwards the packet by using the IP destination address field in the packet header. Every router has a routing table that contains tuples of the form, *<destination address, output interface>*. The router reads the destination address from the header of an incoming packet and uses the routing table to forward it on the appropriate output interface.

MPLS [10] is an advanced framework for fast label switching. In MPLS, a short fixed length value called label is assigned to the packet, as the packet arrives at an entry point of the network. The packet is forwarded to its next hop together with this label. At subsequent hops, there is no further analysis of the packet's network layer header. The label is used as an index into the forwarding table, which contains tuples of the form *<next hop, new*

label>. The old label is replaced with the new label and the packet is forwarded to its next hop. In traditional IP some routers analyze the network layer packet header not only for choosing packet's next hop but also for determining the packet's class of service. MPLS allows this information to be inferred from the label so that no further header analysis is needed.

The fact that a packet is assigned a label as it enters the network allows the use of advanced forwarding techniques. A packet entering the network at a particular router can be labeled differently than the same packet entering the network at a different router. As a result some kind of policy routing can be easily made. Since MPLS decouples forwarding from routing, it is able to support a large variety of routing policies that are difficult or impossible to support with just conventional network layer forwarding.

MPLS uses the Label Distribution Protocol (LDP) [11] to exchange the label/traffic trunk binding between Label Switching Routers (LSRs). A traffic trunk in MPLS terminology is an aggregation of traffic flows of the same service class, which are placed inside the same Label Switched Path (LSP). A LSP is defined to be a sequence of labels from ingress LSR to an egress LSR. LSPs are very similar to unidirectional ATM VCs. The route taken by a LSP between two LSRs can be the same as the conventional network layer route, or the sender LSR can specify an explicit route for this LSP. An explicit route is specified as a sequence of hops rather than being determined by conventional layer 3 routing algorithms on a hop by hop basis. Thus, MPLS apart from conventional IP routing facilitates also the use of the routing technique called explicit routing [7],[12]. Explicit routes can be used to support policy routing and traffic engineering. An explicit route needs to be specified at the time that labels are assigned and does not have to be specified with each IP packet. The ability to setup explicit routes is one of the most useful features of MPLS because it allows network administrators to control how traffic flows through their network. In MPLS an explicitly routed LSP is considered as a tunnel. When a packet enters the network its path, QoS, and forwarding is already fully determined. Explicit routes can be selected either by configuration (manual setup by the network administrator) or dynamically.

Multipath routing can also be supported by MPLS. Specifically label switched routers may support multiple routes for a particular traffic trunk. MPLS assigns multiple labels to the traffic trunk, one for each route. There are a number of obvious reasons why it may be desirable to use explicit and multipath routing instead of conventional IP routing. For example, this allows selection of routes based on administrative policies and careful route design to allow traffic engineering. As we show here, multipath and explicit routing features can also be used to facilitate QoS support for multiple class of services in MPLS capable networks. The end result will be a load balanced network optimized to maximize operators' revenue and customer satisfaction.

MPLS architecture [10] assumes that all nodes in the network use traditional IP routing protocols in order to determine a routed path. The concept of explicit routing is introduced as an add-on feature that coexists with traditional IP routing. MPLS does not support multiple class of services and multi-path routing yet. Since LDP has been designed to interoperate with the traditional IP routing, explicit routing set up is currently accomplished by using RSVP [7],[10],[12] with some extensions for label allocation. The resulting label switched paths may have associated resource reservations or may be used simply as best effort paths. As mentioned earlier explicit routes are determined either manually or dynamically. MPLS implicitly suggests mechanisms like QOSPF [14]for the dynamic determination of explicit routes but currently there is not any implementation available. Current implementations of Tag Switching support only the manual option and implement static, pre-specified explicit routes by configuring TSP (Tag Switched Path) tunnels. Furthermore for route selection and traffic engineering even with the promising proposal [13], there are no satisfactory implementations yet. Specifically in tag switching it is implemented with a filtering mechanism based on destination address. This provides basic traffic engineering functionality.

A drawback of the QOSPF and of other similar approaches [1],[4],[5] is that they compute feasible paths instead of optimal paths. Although this is by no means trivial and requires relatively low overhead, it does not lead to any globally optimal situation. In other words the computed paths are such that satisfy the QoS requirements of the requests but do not optimize overall network resource utilization. Viewed as a game, the OSPF decisions made by individual source nodes could lead to unstable situations, or to inefficient equilibria. We see the following evolutionary steps to the current state of the art:

- Develop routing protocols with the objective to determine and then dynamically select optimal routes that on one hand accommodate predicted traffic requirements of various service classes defined in the network and on the other increase overall network resource utilization [2],[3]. They should take into

account the QoS requirements of the service, the network resource availability, the traffic paterns, and any other policy imposed for administrative purposes [7],[16]. As any routing protocol they should provide route computation and route selection mechanisms. They should be able to discover multiple paths for each class of service and optimize the route choice in terms of efficient resource utilization, load balancing and other policies imposed by the network administrator.

- Extend LDP to support multiple class of services and advanced resource reservation mechanisms. Add the capability of setting up the paths determined by the advanced routing protocols mentioned above. A recent draft proposal [21] specifies mechanisms to support constraint-based LSP setup that uses [7]. CRLSP service classes (best effort, throughput sensitive, and delay sensitive) defined in [21] seem to satisfy our requirements.
- On top of that, use a mechanism based on ***implied costs*** for the link state advertisement and routing selection. This is a very advanced and optimized approach since implied costs reflect the network congestion as well as the interdependencies among flows in the network. A key idea is that such costs can signal the need for expanding the resources allocated to various routes, optimizing thus the allocation of network resources among competing routes. The theory is introduced in the subsection below, and the mechanisms for its application are in the next section (sec.3).

2.2 Background Theory And Proposed Use

We briefly introduce Kelly's work on routing and capacity allocation in circuit-switched networks [20], and propose its use in policy based resource allocation in multiservice MPLS networks. There are finitely many links, labeled $k = 1,2,3.....K$, and link *k* comprises C_k circuits. A subset $r \subset \{1,2,3,.....k\}$ identifies a route. Calls requesting route *r* arrive as a Poisson process of rate v_r, and as *r* varies it indexes independent Poisson streams. A call requesting route *r* is blocked and lost if on any link $k \in r$ there are no free circuits. Write *R* for the set of possible routes. Let B_j be the blocking probability at link *j*. For each $k = 1,2,3.....K$ we have $B_k = E(\sum_{r:k\in r} v_r \prod_{j\in r-\{k\}}(1-B_j), C_k)$ (1) where for scalar *v* and *C*, $E(v,C)$ is Erlang's formula for the proportion of calls lost on a single link of capacity *C* circuits offered Poisson traffic at rate *v*. Note that the rate offered by route *r* at a link $k \in r$ is not v_r but is thinned by factor $\prod_{j\in r-\{k\}}(1-B_j)$. This means that the stream v_r is thinned by a factor $(1-B_j)$ at each link $j \in r-\{k\}$ before being offered to link k. The factor $(1-B_j)$ indicates the probability to not block route *r* at link *j*. An approximation for the proportion of calls requesting route *r* that are lost is

$B_r = 1 - \prod_{k\in r}(1-B_k)$. Let $B = (B_1,.....B_k)$, $C = (C_1,.....C_k)$, $v = (v_r, r \in R)$ and $B = B(v;C)$ be the solution to Equation (1). Assume a call accepted on route r that generates expected revenue w_r. Then the total revenue of the network is $W(V;C) = \sum w_r \lambda_r$, where $\lambda_r = v_r \prod_{k\in r}(1-B_k)$. Let c_k be the implied cost of link *k* and $c = (c_1,.....c_k)$. Then

$$c_k = \eta_\kappa (1-B_\kappa)^{-1} \sum_{r:k\in r} \lambda_r (w_r - \sum_{j\in r-\{k\}} c_j) \quad (2)$$

where

$$\eta_\kappa = E(\rho_\kappa, C_k - 1) - E(\rho_\kappa, C_k) \quad (3)$$

and

$$\rho_\kappa = \sum_{r:k\in r} v_r \prod_{j\in r-\{k\}}(1-B_k)$$

The ρ_κ is simply the traffic offered to link *k* and η_κ is the increase of blocking probability at link *k* if we remove one curcuit from the capacity. Note that the implied cost of link *k* indicates how much the network revenue will decrease if we accept one call at the link *k*. This is the decrease implied by the decrease of the blocking probability at link *k*, because we reserve one circuit at link *k* in order to serve the call, thus decreasing the available capacity. Also note that one change on a link *k* affects all routes that use the link *k* and thus it affects all other links (because these routes use some other links too). For example the increase of the blocking probability on a link *k* may decrease the acceptance rate of route *r* which uses the link *k* and so for each other link $j \in r$ we can increase the acceptance rate of other routes on these links. This increases the blocking probability at some other links and so on. The equation (2) captures all these interactions. The implied costs *c* measure the

knock-on effects of accepting a call upon the other routes in the network. Kelly [20] proved that

$$\frac{d}{dv_r}W(v;C) = (1-B_r)(w_r - \sum_{k\in r} c_k) \quad (4)$$

The equation (4) says that an additional call offered at route *r* would be accepted with probability $(1-B_r)$; if accepted it will earn w_r directly, but at a cost c_k for each link $k \in r$. He also proved that

$$\frac{d}{dC_k}W(v;C) = c_k \quad (5)$$

Equation (5) shows how much the revenue changes if we change the capacity of link *k.* Specifically the implied cost c_k measures the sensitivity of the rate of return to the capacity C_k of the link *k*. Let $s_r = w_r - \sum_{j\in r} c_j$ be the surplus revenue of route *r.* We can rewrite the equation (2) in the form $c_k = \eta_\kappa (1-\mathrm{B}_\kappa)^{-1} \sum_{r:k\in r} \lambda_r (s_r + c_k)$ (6).

Under the approximation $\rho_r = \lambda_r (1-\mathrm{B}_\kappa)^{-1}$, where ρ_r is the traffic offered at link *k* by route, we can rewrite (6)

$$c_k = \sum_{r:k\in r} \eta_\kappa \rho_r (s_r + c_k) \quad (7).$$

We compute on the fly $\rho_k = \sum_{k\in r} \rho_r$ and $\rho_r = \sum_{f\in r} \alpha_f$ and use them to calculate the implied cost (knowing C_k to calculate η_k using (3)). Where *f* are the flows routed through *r* and α the effective bandwidth of the flow *f.*

The above equations can be modified in order to include the case that some routes need more than one circuit. Specifically if a route *q* needs A_{kq} circuits from link *k,* then the implied cost for the acceptance of one call on route *q* is

$$c_{kq} = \sum_{r:k\in r} \eta_{krq} \rho_{kr} (w_r - \sum_{j\in r-\{k\}} c_{jr}), \quad (8)$$

where

$$\eta_{krq} = B_{kr}(\overrightarrow{\rho_k}, \overrightarrow{\mathrm{A}_k}, C_k - A_{kq}) - B_{kr}(\overrightarrow{\rho_k}, \overrightarrow{\mathrm{A}_k}, C_k), \quad (9)$$

$$\rho_{kr} = \lambda_r (1-\mathrm{B}_{kr})^{-1}, \qquad (10)$$

$$B_{kr}(\overrightarrow{\rho_k}, \overrightarrow{\mathrm{A}_k}, C_k - A_{kq}) = E(\sum_r A_{kr} v_r \prod_{j\neq k} (1-B_{jr})^{A_{jr}}, C_k).$$

The η_{krq} is the expected increase in blocking probability at link *k* for route r given that A_{kq} circuits are removed from link *k* (the blocking probability if the capacity of link *k* after we reserve A_{kq} circuits on link *k* -in order to service the call- minus the blocking probability if we reject the call -thus changing the available capacity-).

Proposed Use of implied costs in MPLS

Now, assume an MPLS capable IP network of considerable size that supports at least three different service classes (for example: best effort, throughput sensitive, and delay sensitive as in [21]). We initially allocate bandwidth to the different classes according to a common practice policy (operators' or ISPs' strategy). For each service class we define virtual links that constitute the virtual network for the specific service class (so there are three virtual networks in this case).

There are several ways that one could map the circuit switched model (CSM) into the MPLS framework. The key observation is that a call in CSM can correspond to a flow in MPLS. When a new flow is to be routed through the network, the router determines the virtual network it will be routed through (in terms of the QoS requirements of the flow), the amount of resources A_{kq} (by assigning an effective bandwidth to the flow), and the reward W obtained by the flow. Moreover for simplicity, we consider the case of all rewards being equal to one, which corresponds to minimizing the overall number of flows being blocked (although we do not require a call blocking mechanism we operate as if there is one). An obvious extension could be the assignment of real prices to the different service classes thus giving a way to maximise the real network revenue.

Once the c_k s are known, then they can be used either for expanding the resources of a logical link at the expense of another (sharing the same physical link) or for doing route selection when more than one routes are available.

As we propose in the next section, the dynamic redistribution of bandwidth among the supported classes will be done by a higher level decision process (a policy enforcer we named "**Policer**") based on link state alarms depending on optimization criteria. For example, blocking probability threshold exceed alarm $(B_j \geq \varepsilon)$ for a given service class invokes recalculation using (8).

3. Mechanism for QoS Provisioning for Multiple Class of Service Supporting MPLS.

In this section we propose a mechanism to support multiservice, multipath routing in MPLS capable networks. We provide for initial allocation and dynamic reallocation of network resources whenever necessary for multiple Service Classes. We consider a network that supports a number of Services Classes (SCs) with different QoS guarantees like the one given in a recent Internet draft [21]. In our approach, network resources are initially allocated to the different Service Classes according to the administrative policies of the network operator. This means that we create one virtual network for every Service Class. These virtual networks share actual network resources as described above. Then a set of routes is computed for every source/destination pair on each virtual network. When the blocking on a virtual link increases significantly in any of the virtual networks, network resource reallocation is enforced among the virtual networks. To achieve this we adopt the implied cost calculation theory introduced above.

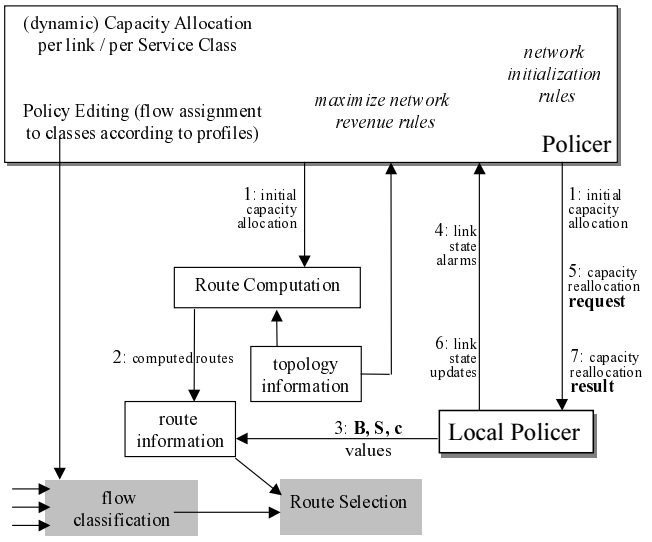

Figure 1. Policy based resource allocation and Routing

For this purpose our architecture supports a high level "policy enforcement" process named **Policer** that is responsible of initial allocation and dynamic reallocation of network resources among the different Service Classes, and as a consequence among the different virtual networks [see Fig.1]. Furthermore every network node supports a **Local Policer** process that is able to calculate the implied cost and blocking probability per outgoing link as well as the surplus revenue of every route to which this node is the source node. Such computations are made independently for every virtual network. The **Policer** and the underlying **Local Policer** processes closely interact in order to coordinate the network resource allocation among the virtual networks.

In addition every network node is able to advertise the values calculated by the **Local Policer**. This is the purpose of the **Link State Advertisement** process [see Fig.2]. Finally, upon a flow request on an ingress network node **Route Selection** selects the route that gives the largest surplus revenue. A process that maps a flow request to a specific Service Class, the **Flow Classification** process is available at every ingress node of the network.

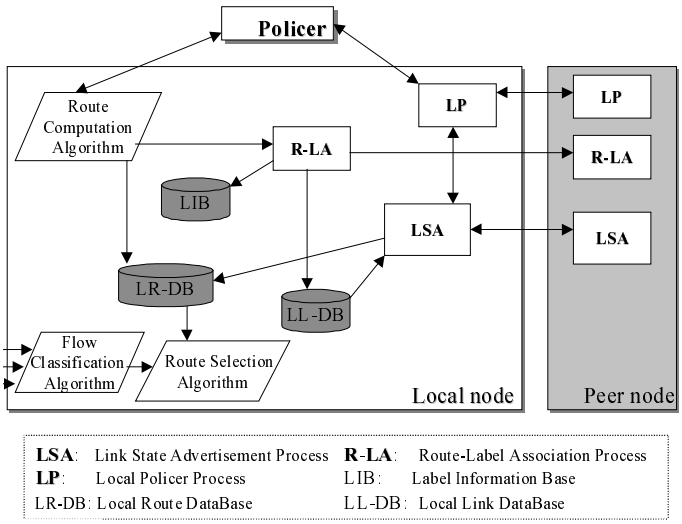

Figure 2. Network Node Functional Architecture

In the following subsections we discuss design and implementation issues and the MPLS extensions needed to realize this architecture. We foresee distributed implementations for the **Route Computation** and **Route Selection** and **Local Policer** (LP) as well as for **Link State Advertisement** (LSA). Figure 2 presents the functional architecture of a network node. The edge nodes support all the functional components shown in Figure 2 while core nodes of the network should support only the LP Process, LSA Process, the R-LA Process and the LL-DB and of course the LIB. The Policer is implemented with a centralized logic.

3.1 Policer

The main responsibility of the Policer is to distribute the resources of the network between the Service Classes (SC). Knowing the network resources available initially, and a number of administrative rules, it distributes them among the different Service Classes. This distribution is

made per link per Service Class. Consequently, physical links are split to many virtual links. A virtual link services a single Service Class. Virtual links of the same Service Class construct a virtual network servicing the particular Service Class. In other words, the Policer creates one virtual network for each Service Class. The Policer downloads the virtual network configurations to the network nodes (this corresponds to action 1 in Fig. 1).

The Policer receives link state alarms from the Local Policers running on the network nodes. Link state alarms are produced whenever a virtual link is congested i.e. the blocking probability of a link exceeds a threshold (action 4 in Figure 1). The Policer is responsible to resolve the congestion on the specific virtual link. To achieve this, it applies a decision procedure to decide whether it should increase/decrease the capacity allocated to the virtual links that correspond to the same physical link as the congested one. The outcome of this decision procedure is a set of possible deltas to the current capacity allocation for every virtual link. This set of deltas is then sent to the Local Policer who produced the link state alarm (action 5 in Figure 1). The Local Policer estimates all possible implied costs, with the goal to satisfy the different deltas on the relevant virtual links. It then sends the result back to the Policer (action 6 in figure 1). As we have already mentioned in section 2.2: $\frac{d}{dC_j}W = c_j$, for every link j. W is the revenue of the network, C_j is the capacity of link *j*, and c_j is the implied cost of link *j.* The Policer estimates the sum: $\sum_{s \in SC}(c_{j_{old}} - c_{j_{new}})$. The delta that gives the higher value to the above sum gives the new split of network resources among the different virtual networks. The Policer updates the Local Policer with the new resource allocation state (action 7 in Figure 1). If the above quantity is negative there's going to be no change since any change leads the network to a worse state.

3.2 Route Computation

As mentioned before, Route Computation computes a set of routes for every <*source, destination, SC*>. It is not the intention of this paper to propose a specific algorithm. We investigate several alternatives based on results in [3],[6],[14]. We should note that whichever the Route Computation algorithm would be, it should support multiple Service Classes and compute multiple routes per Service Class (action 2 in Figure 1). The result of route computation is a table containing the computed routes. The entries of the table are of the form:

$[< source, output_link >, < node_i, output_link >, < node_j, output_link >, ... < destinatio\ n, . >, SC, B, S]$

We call this table Local Route DataBase (LR-DB). Specifically, each entry contains a list of pairs $< node_i, output_link >$ that are the nodes and links, which are contained in a route. This list is called *Transit List*, where SC indicates the service class for which this route has been created, B indicates the blocking probability of route *r*. The blocking probability B of route $r = (i, j, ..., m, n)$ is equal to $[1 - ((1\text{-}B(i,j))*....*(1\text{-}B(m,n)))]$ where $i, j, ..., m, n$ are the nodes visited by route *r*. S indicates the surplus benefit. The surplus revenue S of route $r = \{i, j, ..., m, n\}$ is equal to $[w - (c(i,j) + \ldots c(m,n))]$, where w is the total benefit and $c(i,j)$ is the implied cost of the link between node i and node j. Note that the metrics B and S change dynamically as the network load changes (we'll discuss how these metrics change later on).

3.2.1 Label Information Base construction

Each source node creates the LR-DB that contains the routes from it to all possible destinations. It is clear that every source node must advertise the computed routes to the transit nodes, the nodes a route traverses. To accomplish this, for each such route the node binds a label and sends a message to the next node on the route that contains the *Transit List* of the route together with this label. The Route-Label Association Process (R-LA) is responsible for this operation.

When a node receives this message it binds a new label and adds a new entry <*input label, input interface, output label, output interface*> in its Label Information Base (LIB). The *input label* is the label which is contained in the message, the *output label* is the label that has just been allocated by this node, the *input interface* is the link from which the message has been received and the *output interface* is the next link on the route. Then it is its turn to forward a similar message that contains the same *Transit List* and the new label - the output label - to the next hop on the route.

The label, Service Class and Transit List information is transferred in a LDP message. The label is encapsulated in the label TLV, and the Service Class and the Transit List in the CR-TLV of the Label Request message according to [11], [21]. The LDP messages exchanged

for the Label Information Base construction, are the Label Request Message and the Label Mapping Message.

3.3 Local Policer

Every network node supports a Local Policer process. The Local Policer is responsible for three operations. **First** to estimate the implied cost and blocking probability on every outgoing virtual link and the surplus revenue of routes of which this node is a source node [using (8) from section 2.2]. **Second** to advertise these values to other Local Policers. This advertisement is necessary for the calculation of the implied costs as well as for updating the LR-DBs on the ingress network nodes (action 3 in Figure 1) and is detailed below. And **third** to communicate with the Policer in case network resource reallocation is needed, as described above [section 3.1].

3.3.1 Link State Advertisement

The nodes need topology and link state information to initially allocate (and whenever necessary reallocate) capacity to links and as a consequence to routes. The policy adopted for Link State Advertisement determines when a network node should trigger Link State updates as well as specify the contents of such updates. Since link state advertisement creates traffic on the network we want the link state advertisement policy to trigger updates only when the current value of a link metric *"differs significantly"* from the previously advertised value. We are interested to advertise the values of two link parameters, specifically implied cost and the blocking probability.

3.3.2 Link State Advertisement Messages

Our goal is to minimize the link state advertisement messages in the network. That is why we propose to send link state updates only to nodes that are interested in the specific update. For example a node n sends a Link State update message, for the outgoing link *j* only to the nodes that are source nodes of routes that use this link *j*. In order to accomplish this, a node must maintain a Local Link DataBase (LL-DB) the entries of which are of the form $< j, list_of_nodes, list_of_routes >$. There is one such entry in the LL-DB for every output link of every node *n*. *list_of_nodes* contains all nodes that are source nodes of routes that use j. *list_of_routes* contains all routes that use j. Every element of *list_of_routes* has the form $< label, S_r, B_{jr} >$, where *label* is the label corresponding to route *r*, S_r is the surplus revenue of route *r* and B_{jr} is the blocking probability of link *j* for route *r*. The R-LA process is responsible for this operation. Note that LL-DB is created at the same time as LIB is created.

When a node decides to make a link state advertisement it sends a message to all nodes in *list_of_nodes* of this specific link. The message contains the previous and current implied cost and the previous and current blocking probability of the link. When a node receives this message it checks in its LR-DB to find the route using this link – let us name this link *j*. For every such route it updates its values of B (blocking probability) and S (surplus revenue). Specifically:

$$B_{r_{new}} = \frac{B_{r_{old}} * (1 - B_{jr_{new}})}{(1 - B_{jr_{old}})},$$

$$S_{r_{new}} = S_{r_{old}} - c_{jq_{old}} + c_{jq_{new}},$$

where the $B_{r_{old}}$ and $B_{r_{new}}$ are the previous and the current blocking probabilities of link j. The $c_{jq_{old}}$ and $c_{jq_{new}}$ are the previous and current implied cost of link j for the Service Class q. Note that route *r* belongs to the Service Class q. As we have already mentioned every node *n* must keep the S_r of each route r such that outgoing link $j \in r$. The source node of route *r* should advertise the S_r to the rest of the nodes of route *r*. The advertisement of the implied cost, blocking probability and surplus revenue values is accomplished by using LDP messages. An option is to use the KeepAlive message [11].

3.4 Route Selection

Route Selection selects a route *r* among the computed ones using B_r and S_r of route *r*. Specifically the Route Selection algorithm tries to select a route that is capable to satisfy the requirements of a flow, while at the same time tries to minimize the amount of network resources that need to be allocated and to do load balancing.

The Route Selection algorithm selects a path among the computed ones that are stored in the LR-DB. When a flow request arrives at an ingress node, the routing selection estimates the product $(1 - B_r) * S_r$ for every route *r* that corresponds to the Service Class of the flow request and orders the routes in descending order. The Route Selection algorithm chooses the first route from the list (for negative $(1 - B_r) * S_r$ it selects the route with the smallest $|(1 - B_r) * S_r|$).

4. Evaluation results

Our evaluation is based on simulation results. We used the C language to implement our algorithms. We used the network topology shown in Fig. 3 that represents a typical large ISP network. It is the same network topology as in [8].

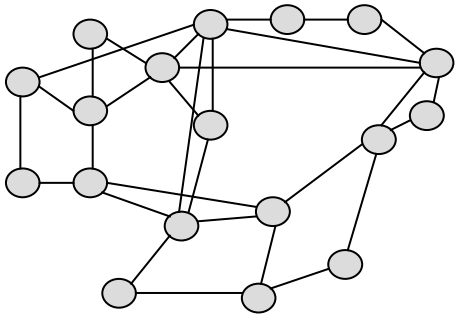

Figure 3. Network topology used in the experiments

We assumed links of 155 Mbits/sec capacity and we divided the link capacity to 6,200 circuits of 25kbits/sec each.

4.1 Traffic Load

We consider a network that supports three Service Classes (SCs), namely SC0, SC1, SC2, with different QoS guarantees each. Specifically SC0 imposes hard QoS guarantees for both bandwidth and delay requirements (delay sensitive service class), SC1 imposes hard QoS guarantees for throughput (throughput sensitive service class) and SC2 is best effort with a guaranteed minimum throughput. Our target is to succeed very low blocking probability (e.g. lower than 1%) for SC0 sessions and e.g. lower than 20% for SC1 sessions. For SC2 sessions we don't assume any constraint.

Our traffic load consists of two types of traffic for SC0 and SC1, namely MPEG video and voice. For SC0 and SC1, sessions arrive according to a Poisson distribution. The mean rate of arrivals for SC0 is smaller than the mean rate for SC1 (approximately 1:5). Furthermore for both SC0 and SC1 the ratio of arrivals of MPEG video and voice sessions is 1/10. For SC2, we also assume that sessions arrive according to a Poisson distribution with mean arrival rate larger than that of SC0 and SC1 (approximately 1:5:15). Furthermore sessions of SC2 do not express any QoS requirements in terms of delay or throughput. The holding time for SC0 sessions follows a Poisson distribution with mean value 3min, for SC1 with mean value 5min and for SC2 with mean value 10min.

4.2 Performance evaluation – Discussion of the experimental results

For the performance evaluation of our algorithms we measured three parameters namely the mean value of the run time of the algorithms, the mean value of the time between two link state advertisements, between any two nodes of the network and the blocking probability per Service Class. We found out that for the scenario described above, the mean value of the run time of the algorithms is 0.012sec. and the mean value of the time between two links state advertisement between any two nodes is 12.2sec. As it is shown in Fig.4 the time between link state advertisements converges very quickly to this maximum value. Concerning the blocking probability per Service Class, for SC0 it is 0.05% for SC1 is 19,7% and for SC2 is 58,2%.

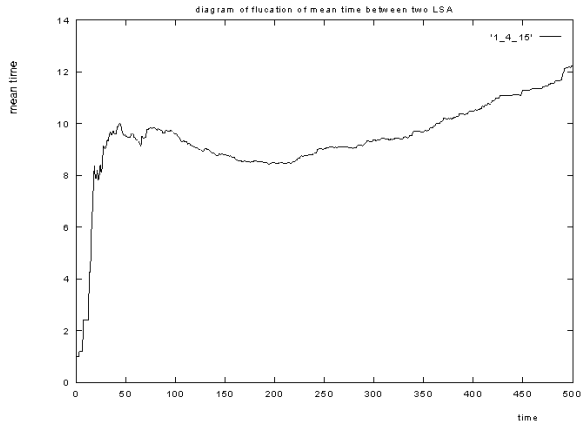

Figure 4. Mean time between two link state advertisements

We observe that the time between two link state advertisements is quite large and significantly larger than the run time of the algorithms. Based on this result we can conclude that our algorithms work with accurate network state information. Furthermore the complexity of our algorithms is very low since the time to run on the network topology (18 nodes, 30 bi-directional links) of Figure 3 is approximately 0.012sec.

In order to evaluate the usefulness of our approach we compared the blocking probability per Service Class with the blocking probability per Service Class in the case that our route selection algorithm is replaced with a simple shortest path algorithm. The blocking probability in the case of the simple shortest path algorithm, for SC0 is 9,23%, for SC1 is 41,2% and for SC2 is 61%. It is evident from the results above that with our approach applied all the three Service Classes get significant lower blocking probability. Specifically for SC0 and SC1 we manage to guarantee their blocking probability constraints, while this is not the case with the shortest path algorithm, without making SC2 any harm. In the

contrary we see an improvement in the performance of the SC2 class.

5. Conclusions

We introduced an architecture for resource allocation among several service classes with varying QoS requirements across MPLS networks. We focused on design and performance of resource management. Our on demand approach selects the best of the pre-computed routes that is capable to satisfy users' requirements. In order to accomplish that, we adopted MPLS extensions and proposed a routing selection mechanism based on implied costs. Implied costs reflect the network congestion as well as the interdependencies among flows in the network. Work is ongoing implementing policing schemes following this architecture using cisco switch-routers with tag swiching. We also examine extensions to hierarchical routing and applicability to IP over ATM or DWDM layered networks.

6. References

[1]. W.C. Lee, M.G. Hluchyj, P.A. Humblet. "Routing Subject to Quality of Service Constraints in Integrated Communication Networks", IEEE Network, July/August 1995.

[2]. P. Georgatsos, D. Griffin, "A Management System for Load Balancing through Adaptive Routing in Multi-Service ATM Networks", INFOCOM 1996.

[3]. P. Georgatsos, D. Griffin, "A General Framework for Routing Management in Multi-Service ATM Networks", 5th IFIP/IEEE International Symposium on Integrated Network Management, pp.343-354, 1997.

[4]. O.Crochat, J. Le Boudec, T. Przygienda. "A Path Selection Method in ATM using Pre-Computation", Tecnical Report, Laboratoire de Reseaux de Communication, http://lrcwww.epfl.ch.

[5]. J. Le Boudec, T. Przygienda. "A Route Pre-Computation Algorithm for Integrated Services Networks", Tecnical Report No 95/113, Laboratoire de Reseaux de Communication, http://lrcwww.epfl.ch.

[6]. L. Georgiadis, G. Gikas, M. Chatzaki, S. Sartzetakis, "Cost-effective VPC network design", DRCN'98, Brudge, Belgium, May 1998.

[7]. M. Chatzaki, and S. Sartzetakis. "QoS-Policy based Routing in Public Heterogeneous Broadband Networks". In Proceedings of Interworking'98 Conference, Ottawa, Canada, 6-10 July 1998.

[8]. Q. Ma and P. Steenkiste. "On Path Selection for Traffic with Bandwidth Guarantees". In Proceedings of IEEE International Conference on Network Protocols, October 1997.

[9]. R. Guerin et. al., "Setting up Reservations on Explicit Paths using RSVP", draft-guerin-expl-path-rsvp-01.txt.

[10]. E. Rosen et. al., "Multiprotocol Label Switching Architecture", draft-ietf-mpls-arch-02.txt.

[11]. L.Andersson et. al, "LDP Specification", draft-ietf-mpls-ldp-02.txt., Nov.98

[12]. B. Davie et. al., "Use of Label Switching With RSVP", draft-ietf-mpls-rsvp-00.txt. ***AND*** "Explicit Route Support in MPLS", draft-davie-mpls-explicit-routes-00.txt.

[13]. D. Awduche et. al. "Requirements for Traffic Engineering over MPLS", draft-ietf-mpls-traffic-Engineering-00.txt., Nov.98

[14]. R. Guerin et. al., "QoS Routing Mechanisms and OSPF Extensions", draft-guerin-qos-routing-ospf-03.txt.

[15]. E. Crawl et.al., "A Framework for QoS-based Routing in the Internet". RFC 2386.

[16]. R. Yavatkar et. al., "A Framework for Policy-based Admission Control", draft-ietf-rap-framework-01.txt, Nov. 1998.

[17]. S. Blake et. al., "An Architecture for Differentiated Services", draft-ietf-diffserv-arch-02.txt.

[18]. R. Braden et. al., "Integrated Services in the Internet Architecture: an Overview", RFC 1633.

[19]. M. Montgomery & G. De Veciana. "Hierachical Source Routing Through Clouds", INFOCOM 98.

[20]. F. P. Kelly. "Routing in Circuit-Switched Networks: Optimization, shadow prices and decentralization", Adv. Appl. Prob. 20 112-144, 1988.

[21]. B. Jamoussi et.al, "Constrained-Based LSP Setup using LDP", draft-jamoussi-mpls-cr-ldp-00.txt, Nov. 98.