University of Würzburg
Institute of Computer Science
Research Report Series

A Graph Theoretical Concept for LSP Hierarchies

Michael Menth and Norbert Hauck

Report No. 287 November 2001

Institute of Computer Science, University of Würzburg
Am Hubland, D-97074 Würzburg, Germany
E-Mail:{menth|hauck}@informatik.uni-wuerzburg.de

A Graph Theoretical Concept for LSP Hierarchies

Michael Menth and Norbert Hauck
Institute of Computer Science, University of Würzburg
Am Hubland, D-97074 Würzburg, Germany
E-Mail:{menth|hauck}@informatik.uni-wuerzburg.de

Abstract

In this paper we propose the use of Multiprotocol Label Switching (MPLS) to achieve scalability of resource management that is needed to provide quality of service guarantees in IP networks. We suggest a hierarchical structure of Label Switched Paths (LSPs) to reduce the stored reservation information to a minimum. We lay a graph theoretic foundation for LSP hierarchies and define cost functions that help to find a least cost overlay network in terms of state maintenance. The simulation results demonstrate the efficiency of the concept. The overlay network can be used for configuration of MPLS networks.

Keywords: QoS, resource reservation, scalability, MPLS

1 Introduction

The challenge of future IP networks is the provisioning of toll quality data transport for real-time applications, i.e. quality of service (QoS) for the traffic transport in terms of loss and delay bounds must be met. For this purpose, the Internet Engineering Task Force (IETF) proposed the Integrated Services (IntServ) approach [1, 2] which is tightly coupled with the Resource Reservation Protocol (RSVP) [3]. For every flow, the routers along the path from source to destination keep an account of booked resources. We call the stored information the *state* of a connection. The complexity of classification and scheduling increases with the number of tracked flows because these actions require lookups for each forwarded IP packet. In a network like the Internet the number of flows increases drastically towards the core. Hence, IntServ works well for small intranets but it fails on a large scale basis like backbone networks since the routers would be mostly busy with bookkeeping. In other words, IntServ does not scale because the amount of information in the routers increases with the number of flows in the network (cf. Figure 1).

The Differentiated Services (DiffServ) approach [4, 5] was a reaction to that problem. Traffic is aggregated into only a few service classes and the routers treat the marked IP packets with different priority. This method scales well but without an additional admission control (AC), the required QoS can not be guaranteed. Therefore, various concepts for AC have been discussed. Measurement based admission control (MBAC) [6] can provide good results for a reasonable QoS but it can not guarantee fixed delay bounds. Bandwidth brokers are discussed very often to track the traffic in a single entity and to make AC decisions based on agents [7].

Apparently, to provide QoS guarantees, it is also necessary for DiffServ to signal the required minimum capacity at least for a premium service. This can be achieved in an

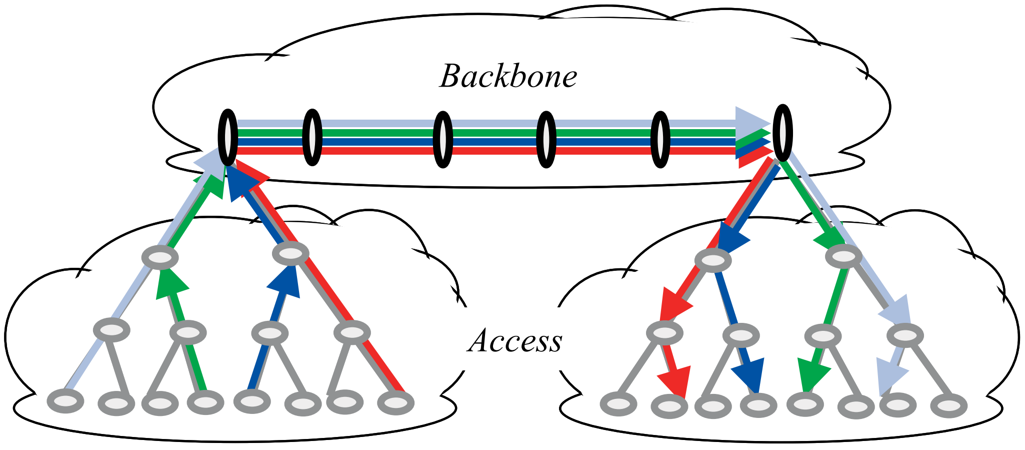

Figure 1: The number of reservation states in the routers increases towards the backbone.

aggregated manner like in [8] or in [9]. Both studies use a sink tree for traffic aggregation purposes. As opposed to that, [10] uses a kind of tunnel. Multiprotocol Label Switching (MPLS) [11, 12] helps to aggregate traffic in Label Switched Paths (LSPs) which can be used as tunnels with QoS attributes.

In this work, we explain how scalable resource management can be achieved by using tunnels to realize aggregation of individual reservations. We suggest to use the tunneling method recursively and to obtain an arbitrary degree of scalability. The result is a hierarchy of LSPs. We formulate the LSP hierarchy in graph theoretical notation which facilitates the process of optimization. Constraints from the network topology and from aggregation are derived and translated into conditions in the graph. Furthermore, we define cost functions that count the number of states in an LSP hierarchy. This concept may be used to find optimized LSP hierarchies for a given networking scenario and helps to obtain a scalable overlay topology in order to support QoS in IP networks. The resulting LSP hierarchy can be used to configure an MPLS network.

The paper is structured as follows. In Section 2, we describe the basic mechanisms of IntServ and DiffServ. We introduce MPLS and explain how it can contribute to reservation aggregation. Section 3 covers the graph theoretical description of communication concepts, the constraints from MPLS technology, and the cost functions. Section 4 provides numerical results from simulations. They underline the ability of the concept to enforce scalability in IP networks that need to support QoS. Finally, we give an outlook for further research.

2 Protocols for QoS Support

The IETF has suggested two main alternatives to enhance IP networks with real-time capabilities. These are the IntServ and the DiffServ approach. In addition, MPLS has been defined to facilitate the process of traffic engineering. Traffic aggregation can be done organized in tunnels and in funnels. The rest of this section proposes the use of hierarchical traffic aggregation using LSP tunnels to achieve scalable QoS support in IP networks.

2.1 Integrated Services

IntServ is characterized by the separate handling of each individual end-to-end (e2e) micro flow. The Resource Reservation Protocol (RSVP) [3] is used to establish a state with knowledge about every e2e flow in all routers along the path from source to destination. These traffic and reservation descriptors (T_{spec},R_{spec}) are used to manage the capacity on every outgoing interface and to enforce policing on a per flow basis. In particular, the AC uses these data to decide whether an additional flow can be admitted. A separate queue and a scheduling state are maintained for each flow to meet the booked QoS objectives. This, however, is clearly a too difficult task for routers as soon as the number of flows is in the order of a few ten thousands which can be easily reached in backbone networks.

2.2 Differentiated Services

The DiffServ approach uses the Differentiated Services Code Points (DSCPs) in the IP header to define only a few different Per Hop Behaviors (PHBs). The PHB tells the router to treat the corresponding IP packet with low or high priority in the forwarding process. No per flow information is stored and, as a consequence, this architecture scales well for large networks because the forwarding process operates on aggregated traffic and not on single micro flows. Policers and shapers try to control the traffic volume entering the network. However, without the knowledge about where flows go within the network the load can become very high on some links which leads to service degradation even for high priority traffic. In addition, without AC the policers and shapers at the network edges also impair the QoS of all flows. All flows with the same DSCPs degrade in the same way and this approach does not support high QoS for some flows at the expense of the rejection of others.

To overcome that, an entity must perform AC on per flow basis at least at the edges and be aware of the load on the links inside the network. This may be done by a bandwidth broker or in a distributed manner in the routers. The required information about the network state needs to be distributed and stored in a scalable way, e.g. by the use of aggregate reservations.

2.3 Multiprotocol Label Switching

MPLS is a mechanism to allow packet switching instead of routing over any network layer protocol [13]. A connection in MPLS is called an LSP. The first label switching router (LSR) equips the IP packet with a 4 bytes label and sends it to the next LSR. The LSRs classify a packet according to its incoming interface and label. Based on this information, label swapping is performed and the packet is forwarded to the particular outgoing interfaces. The last LSR only removes the label from the IP packet header. The label swapping process requires also entries for every LSP in the management information base (MIB) of the LSRs, so there is again a state per session like in IntServ. There are two major protocol alternatives for establishing an LSP. A modification to RSVP [14] is able to distribute the labels and the Constraint-Based Label Distribution Protocol (CR-LDP), [15] has been designed particularly for that goal.

MPLS is often viewed as modified version of the Asynchronous Transfer Mode (ATM) with variable cell size. But there is a profound difference: ATM enables with its virtual connection and virtual path concept a two-fold aggregation while MPLS allows for many-fold aggregation using multiple label stacking, i.e. an LSP may be transported over other LSPs.

2.4 Aggregation Alternatives

There are basically two ways how flows can be aggregated. We explain these fundamental concepts using MPLS terminology. Aggregation using MPLS means that two different flows are equipped with the same label and are forwarded in the same manner. In principle, there are two alternatives to accomplish this, funnels and tunnels.

2.4.1 Funnels

We say that LSP flows are merged into a new aggregate if the uppermost labels in their packets are substituted by a new common label. Figure 2 visualizes the resulting sink tree towards a common destination and motivates the name funnel for this kind of aggregation. Reservations for aggregates on different links can be (naively spoken) summed up in case of merging and they can be propagated along the paths from the sources to the destinations. This is described in [9]. Thus, funnels reduce the amount of state information in the LSRs towards the destination router. At the end of the funnel, only one reservation exists and the information about the demands of the individual original flows is lost. Hence, flow related information can be aggregated but it can not be deaggregated.

To achieve full connectivity in a network with N nodes, every node needs to hold $N-1$ LSPs since every router can be reached by equipping the packets with the corresponding label of the destination machine. This means that the number of paths scales linearly with the network size.

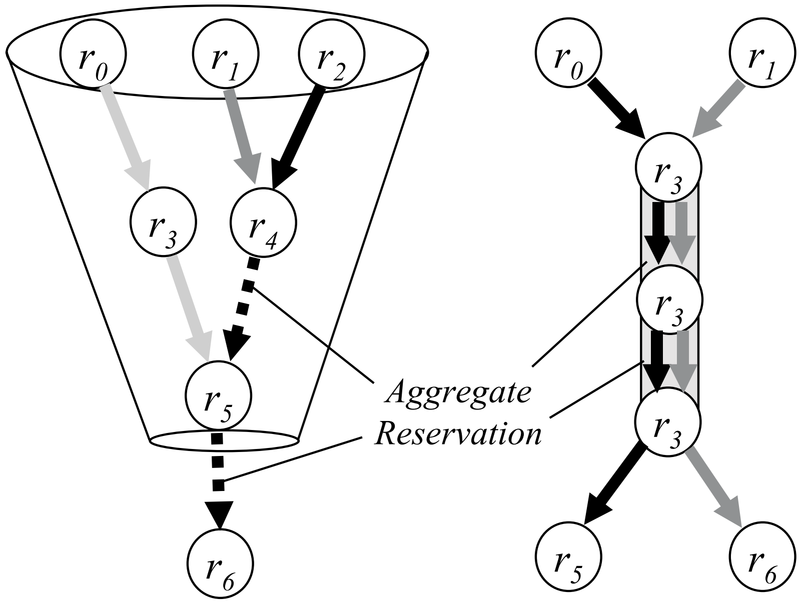

Figure 2: Funnel aggregation in contrast to tunnel aggregation.

2.4.2 Tunnels

We talk about tunneling flows if the uppermost labels in their packets remain in place and a new common label is put onto the label stack. The reservation size for the aggregate tunnel can be computed like above as the sum of the reservations of the contained flows. The flows are transported over the LSP and when the exit router of that LSP removes the uppermost label, the original flows can be restored. The LSP acts as a logical link and the intermediate LSRs do not see the individual reservations because the RSVP control messages are bypassed as MPLS packets at the LSRs. Hence, tunnels reduce the state information in the intermediate routers, they allow for reservation aggregation and deaggregation.

However, the concept scales poorly in some network topologies. When we want to realize full connectivity in a star network, the center node has to handle all possible tunnels which amounts to exactly $N \cdot (N - 1)$ LSPs. Thus, the number of aggregates scales quadraticly with the network size.

2.5 Scalable QoS Support Using an LSP Hierarchy

We imagine n virtual private networks (VPNs). They all consist of two sites with m clients each and these sites are linked by a hierarchically structured transit network. Single E2E connections are aggregated from the source client to the destination client using LSPs.

The funnel approach requires $m \cdot n$ different aggregates in every transit router. The tunnel approach can take advantage of hierarchical traffic aggregation (Figure 3) due to its deaggregation capability. These LSPs can again be aggregated into other LSPs according to geographical regions and again, e.g. for the transit over a transatlantic carrier. Instead of $m \cdot n$ destinations, the backbone carrier knows only a single tunnel T_{ABCD}. With hierarchical tunneling, the number of required LSPs in the core of a transit network can be reduced to a small number that depends on the network structure but it is definitely independent of m.

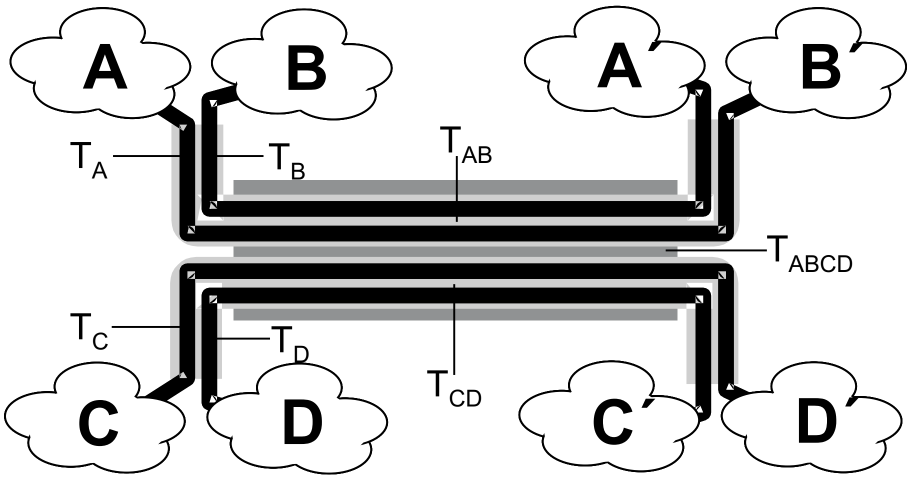

Figure 3: Hierarchical traffic aggregation by tunneling reduces the number of flows in the core.

We see that hierarchical tunneling of flows is a powerful means to reduce the number of flows in a network in order to achieve scalability concerning the reservation states in the router MIBs. Only the tunneling approach is able to support it since with funnels, deaggregation is not possible.

We have the idea to introduce a hierarchical tunnel and funnel structure as an overlay topology to reduce the number of reservations in IP networks. This should be done not only in networks with an obvious hierarchy but also for arbitrarily structured networks. In this case, the overlay structure is not fixed, on the contrary, an overlay topology that effectively decreases the reservation states is even hard to find.

3 Graph Theoretical Notations for an LSP Hierarchy

In this section, we present a graph theoretical notation for the reservation aggregation concept. We first explain the basic terms for network topology, flows, and paths, and develop a concept to write down the LSP hierarchy which translates directly into a network configuration. If an LSP hierarchy is constructed for a given networking scenario, some constraints, that arise from flow tunneling and merging, must be respected. We formulate these constraints as rules within the hierarchy graph. A simple cost function evaluates the corresponding amount of RSVP and MPLS states in the router MIBs. This yields eventually an optimization problem that helps to find the best overlay model.

3.1 Network Topology, Paths, and Flows

A network topology consists of a set of routers $\mathcal{R}$ which are connected by a set of network segments $\mathcal{S}$. A network segment is a directed edge $s_0 : r_0 \rightarrow r_1$, $(r_0, r_1 \in \mathcal{R})$.This means that the network segment s_0 is identified in router r_0 by an outgoing interface and in router r_1 by an incoming interface. Links (forwarding adjacencies) may either be network segments or LSPs. An LSP $a = [g_i]_{0 \leq i < n}$ can be built by concatenating several links such that $g_i : r_i \rightarrow r_{i+1}$ is true for $0 \leq i < n$. This results in a new forwarding adjacency $g_n : r_0 \rightarrow r_n$ in router r_0.

A route is a concatenation $[g_i]_{0 \leq i < n}$ of links $g_i : r_i \rightarrow r_{i+1}$ such that for $i \neq j$ holds $r_i \neq r_j$, i.e. a route must not contain any loops. The path is the sequence of network segments that contribute to a route. Also the path of a route must not have any loops. We define an order among network segments or links. We have $g_0 < g_1$ if $g_0 : r_0 \rightarrow r_1$ and $g_1 : r_1 \rightarrow r_2$ hold. This order is not absolute. However, an LSP is an ordered list of links and a path is an ordered list of network segments without loops. Therefore, the order among the links or network segments is absolute within an LSP or a route. A path p contains another path q if all edges of q are also in the same order in p ($p \supseteq q$). The intersection ($p \cap q$) of two paths p and q is the set of maximum paths that are contained both in p and in q.

A flow is a subset of packets that are transported on the same path and it can be further specified by some attributes. The path of a flow f is returned by the operator $path(f)$. For example, the packets of a point-to-point (p2p) e2e flow are described by a common source and destination IP address and UDP port number. We denote the set of e2e flows by $\mathcal{E}$. An LSP a aggregates packets by using the same MPLS label ($label(a)$) on the outgoing interfaces of the concerned LSRs, i.e. the classification of multiple flows is unified by the label in an LSP. An LSP flow a is a subset of packets that share link-wise their labels along a common route. We refer to the set of LSP aggregated flows by $\mathcal{A}$. An IP packet can be part of an e2e flow but, simultaneously, it can be part of several LSPs that are specified in the label stack of the header.

For funnels, we have a merging condition: LSP tunnels can be merged into a funnel LSP along a contiguous subset of their path. In particular, their destination router must be contained in that subset. Essentially, it is the union of several tunnels (Figure 4). In particular, the LSP funnel does not create other forwarding adjacencies compared to the merged LSP tunnels. But the funnel stores the information for label swapping and reservations in a more compact way.

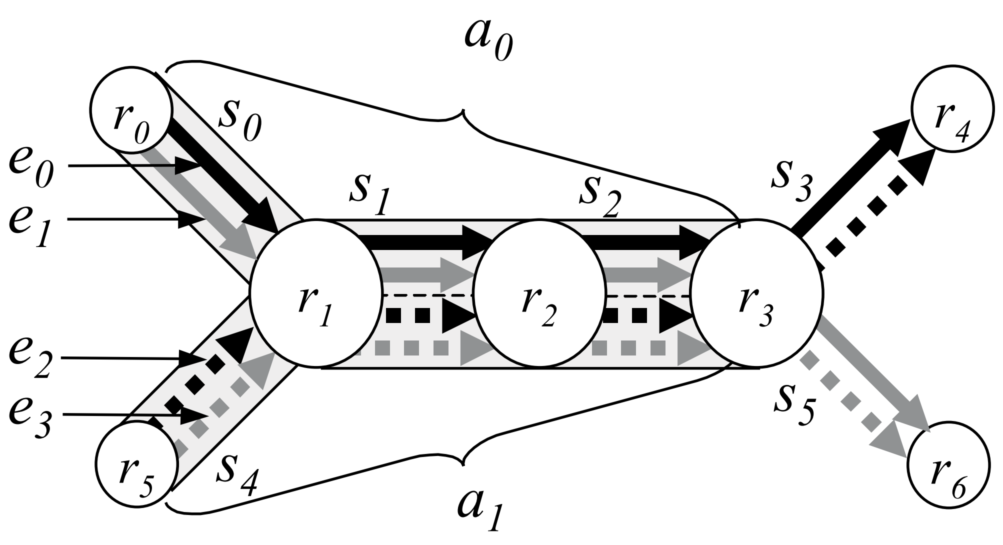

Figure 4: An LSP funnel is the union of several LSP tunnels.

For tunneling and merging, the computation of the aggregated reservation is necessary, which is performed at the tunneling or merging router, respectively. To that aim, the reservation information of the aggregated flows must be accessible and can not be tunneled within another LSP at that router. This prohibits that a funnel is tunneled by another funnel. However, the p2p sections of a funnel may be tunneled. Hence, the original LSP tunnels are cut into pieces for further tunneling and these pieces lose their general tunneling ability since they may only be used for future merging and not for tunneling. We represent a funnel m (multipoint-to-point) by the set of its p2p sections. The set $\mathcal{M}$ contains the union of all these sets.

To reduce state information, flows to a common destination can first be tunneled by LSPs. Then, these LSP tunnels may be merged into a common LSP funnel. After popping the label, the original flows are restored.

3.2 LSP Hierarchy

As outlined previously, recursive aggregation as well as merging of reservations may contribute to a scalable information exchange about resource demands between routers. This can be achieved by the use of hierarchical aggregation of p2p LSPs and multipoint-to-point LSPs.

We specify the LSP hierarchy generally by a graph $\mathcal{H} = (\mathcal{F} = \mathcal{S} \cup \mathcal{A} \cup \{g|g \in m, m \in \mathcal{M}\} \cup \mathcal{E}, \mathcal{T})$ and $\mathcal{M}$. The set of segments $\mathcal{S}$ corresponds to the physical links in the network. The LSP tunnels $\mathcal{A}$ are flows that act as virtual links. They support other flows but have to be carried themselves over other links. The set $\mathcal{M}$ contains the set of funnels and $\{g|g \in m, m \in \mathcal{M}\}$ is the set of their p2p sections. The e2e flows $\mathcal{E}$ are the actual flows that have to be transported. The set of flows and links $\mathcal{F}$ corresponds to the nodes in the graph and the set of directed edges $\mathcal{T}$ represents the transport relationship among the links and the flows. $\mathcal{M}$ is additionally required to identify the funnels in the graph.

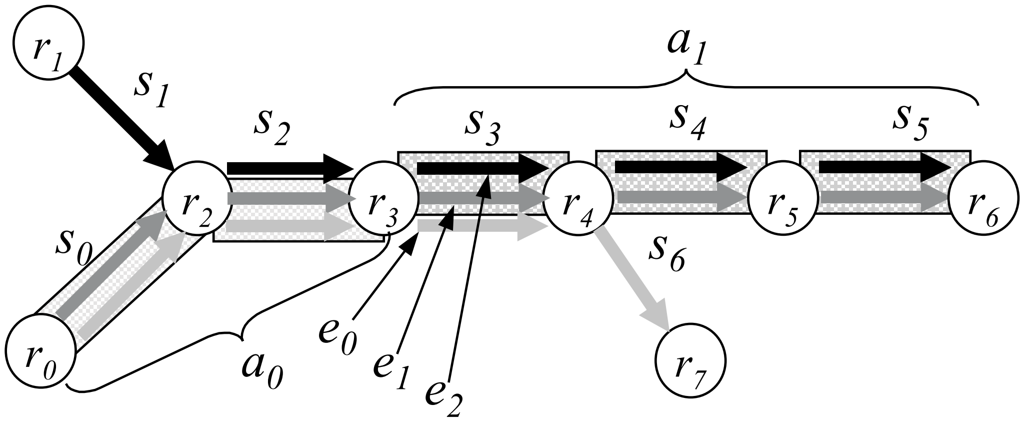

Figure 5: An aggregation scenario: network segments, LSPs, and e2e flows.

If flow b is transported over the link a, we call a a parent of flow b along $path(a)$ and b a child of a. If b is an e2e flow, then a is either a physical link or $label(a)$ is the uppermost label in the stack. If b is an LSP tunnel, $label(b)$ is the lowest label in the stack in case that a is a physical link, otherwise, $label(a)$ sits directly on top of $label(b)$. The parent-child relationship is marked in the graph by a directed edge. Figure 6 shows an LSP hierarchy that corresponds to the network structure depicted in Figure 5.

The operator $children(a)$ returns the set of flows that are children of a and, analogously, $parents(b)$ returns all parents of b. It is also possible to apply the $parents$ and the $children$ operator on a set of flows which yields the union of their parents or children, respectively. Given this, we can define the offspring of a flow by *offspring(f)*$= \cup_{1 \leq i < \infty} children^i(f)$ and the ancestors of a flow $ancestors(f) = \cup_{1 \leq i < \infty} parents^i(f)$ by applying the $children$ ($parents$) operator several times.

For an easier identification of the p2p sections that belong to a single funnel, we mark them as a set in the graph and connect them together in downstream direction. Figure 7 shows the corresponding LSP hierarchy graph to the networking scenario in Figure 4. Flows in these sets distinguish from normal LSP tunnels because they can only be used for the funnel and not for other tunnels. But they can be further aggregated as well.

This graph theoretical notation can easily be applied to larger structures and processed by algorithms. In principle, there are many choices for aggregation, therefore, there are

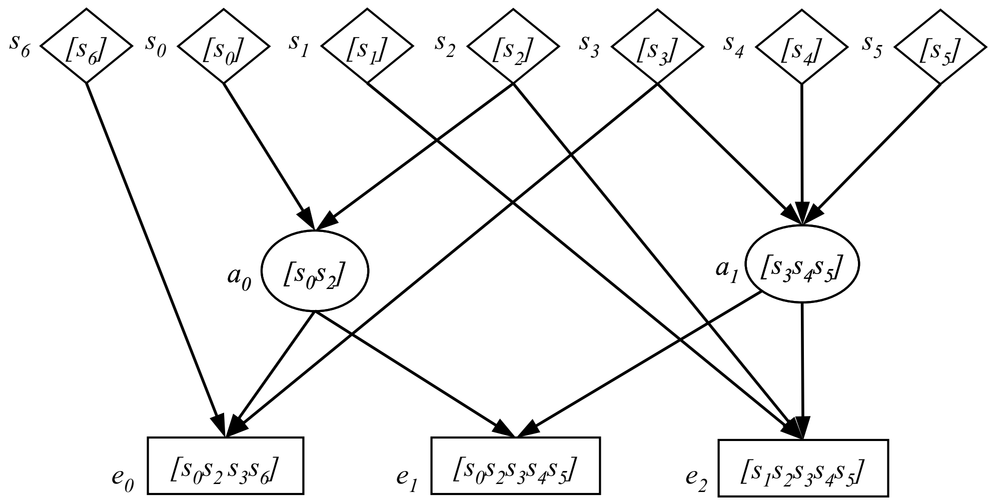

Figure 6: An LSP hierarchy consists of network segments (rhombuses), aggregates (ovals) and e2e flows (rectangles).

many possible LSP hierarchies for a networking scenario. Figure 8 offers an alternative structure to the one given in Figure 6. Our objective is to find an overlay topology that is best suited for the support of real-time services.

3.3 Properties of a Valid LSP Hierarchy

In this section we reflect properties of a valid LSP hierarchy. These are constraints that arise from tunnel and funnel aggregation, properties that we can conclude from others, and properties that we postulate in order to build an efficient hierarchy. We formulate them as graph theoretical rules so that they can be easily verified.

3.3.1 Constraints Due to Aggregation in General

There are several constraints that are due to flow aggregation by tunnels and funnels.

- The network segments $\mathcal{S}$ are physical links. Links can carry traffic, so they can support other flows, i.e. they can be parents. But they are not flows, so they can not be children. Hence, network segments constitute the roots in the LSP hierarchy.
- LSPs in $\mathcal{A}$ and p2p sections of funnels flow over other physical or virtual links. They have always parents and can not be roots. Since they are virtual links, they can also be parents.
- E2E flows $\mathcal{E}$ are carried over physical or virtual links. But they can not serve as links for other flows, so they can not be parents. As a consequence, they are the leaf nodes in the LSP hierarchy.
- A circle in $\mathcal{H}$ would denote that a flow is an ancestor of itself. This is not possible because a flow can not be transported over itself. Hence, the LSP hierarchy does not contain any circles.

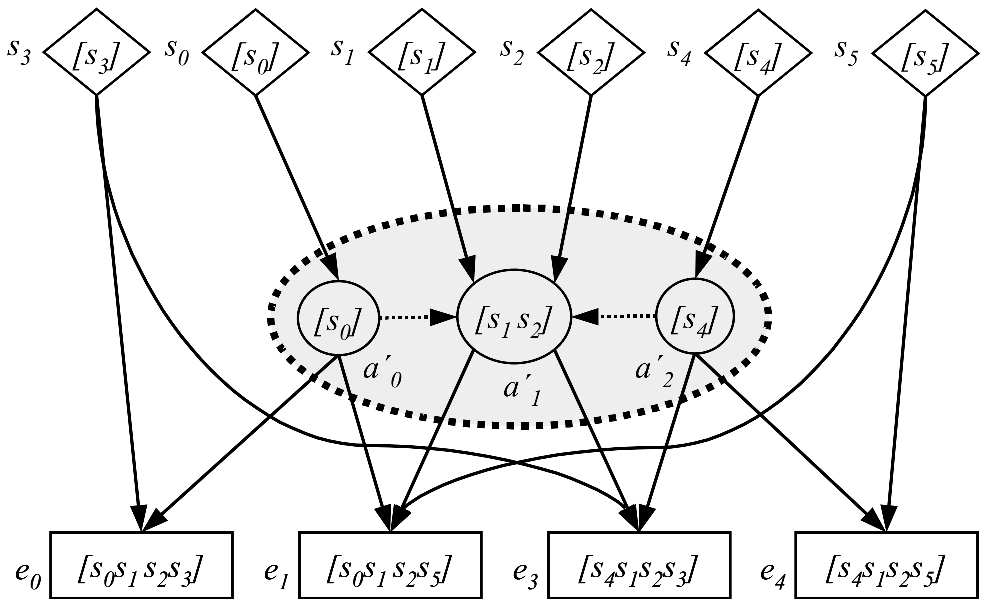

Figure 7: The set of links for a funnel must meet some merging constraints.

- A flow can have several parents if it is transported over several links. However, the paths of the parents can not have any network segments in common. This denotes that the intersection by pairs of paths of the parents is empty:

$$a, c \in parents(b) \Rightarrow path(a) \cap path(c) = \emptyset.$$

- If a is parent of b, then the path of a is a subpath of b ($path(a) \subseteq path(b)$) because $label(a)$ sits along the path of a on top of the one of b in the label stack or the IP header, respectively. Since "$\subseteq$" is transitive, $path(a)$ is still a subpath of $path(b)$ even if b is not a direct child of a but if it is in the offspring of a.

$$a \in ancestors(b) \Rightarrow path(a) \subseteq path(b)$$

- Note that the inversion of the last sentence is not true. Figure 9 shows that $path(e_0)$ contains $path(a_1)$ but a_1 does not aggregate e_0 since a_1 would conflict with a_0 as a parent of e_0.

- Finally, we need to specify the merging condition. If a set of network segments contains $s_0 : r_0 \rightarrow r_1$ such that there is no other edge $s_1 : r_1 \rightarrow r_2$, then r_1 is called a *destination* or a *sink*. A funnel m is a set of links such that

$$\bigcup_{a \in m} path(a) \quad = \bigcup_{a \in \{children(b) | b \in m\}} path(a)$$

holds and there is only one destination router in $\bigcup_{a \in m} path(a)$. The links in m are furthermore grouped into p2p sections such that we have for $f, g \in \{children(b) | b \in m\}$:

$$path(f) \cap path(g) = \bigcup_{a \in m, a \subseteq f, a \subseteq g} path(a)$$

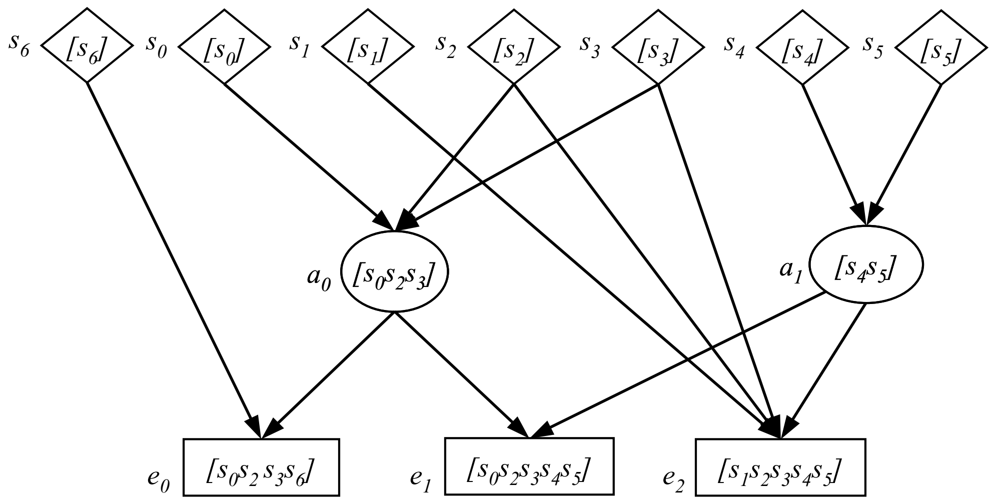

Figure 8: An alternative LSP hierarchy.

and $a \in m$ must be maximal with this property.

These are the technical constraints from aggregation that hold in every LSP hierarchy.

3.3.2 Flow Properties in the LSP Hierarchy

We can furthermore derive some properties for flows within the LSP hierarchy.

- We have argued that $\mathcal{H}$ does not contain any circles. Therefore, we can define the depth of a flow f within $\mathcal{H}$ ($depth(f)$). If f is a leaf node, i.e. an e2e flow, its depth is $depth(f) = 0$. If f is an aggregate, its depth is given by $depth(f) = max_{g \in children(f)}\big(depth(g)\big) + 1$.

- The path of a link (network segment, LSP tunnel) is an ordered set of edges. Circles are prohibited in a route, therefore, the order "$<$" is absolute and transitive among the network segments in a route. The paths of $parents(f)$ have an empty intersection even by pairs, i.e. they do not overlap but they cover the whole path of f. So, the order "$<$"is even meaningful for these subpaths within the path of f. The relation "$<$" can be further extended to the parents of a flow. Hence, we have an absolute order among the parents of a flow.

Advantage can be taken from these properties if LSP hierarchies are constructed automatically for a given networking scenario.

3.3.3 Postulates for an Effective LSP Hierarchy

There are further conditions on the LSP hierarchy that should hold so that the LSP hierarchy reduces the number of states in the routers as efficiently as possible.

- An LSP is only useful if it aggregates at least n flows. This translates into the condition that every aggregate flow in the graph must have at least n children.

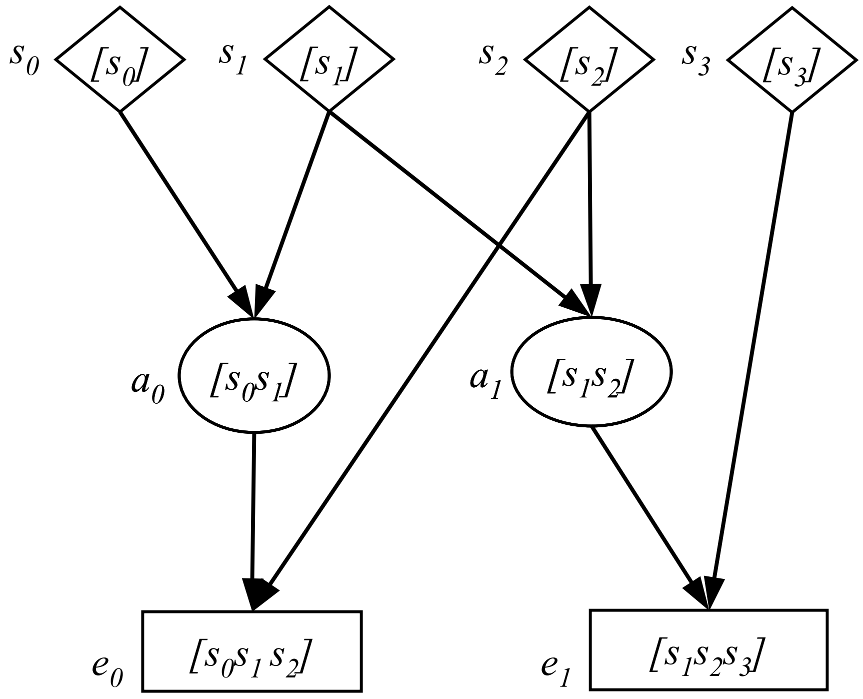

Figure 9: $path(a_1) \subseteq path(e_0) \nRightarrow a_1 \in ancestors(e_0)$

- An LSP is only useful if it is longer than l links. This length l should be larger than 1.

3.4 Cost Function

As elaborated earlier, the flow parameters are tracked for RSVP connection or LSP establishment in the MIBs of the participating routers. Therefore, a vast amount of such states is prohibitive. Our goal is to take advantage of an LSP hierarchy for QoS support in IP networks to reduce the number of states in the involved forwarding machines. Therefore, we suggest a cost function that counts the RSVP states for e2e connections, the states for LSP tunnels, and their reduction by merged reservations in the hierarchy graph.

3.4.1 Counting RSVP States for E2E Connections

Routers need the traffic descriptors and QoS requirements for e2e connections whenever these flows are sent over a link. Therefore, we assign n cost points to a router that carries n flows over a link. In the context of this work, the link-flow relation is represented by the parent-child relationship in the graph. Therefore, the first router of each concerned parent that is a LSP tunnel or a network segment, respectively, is charged with n cost points. If the parent is a p2p section that belongs to a funnel, its first router receives the cost point only if it is the ingress router for the e2e flow into the funnel.

3.4.2 Counting LSP Tunnel States

With LSP tunnels, this is slightly different. Like above, the ingress routers of the used links keep track of the reservation states. But in addition, every LSR in the LSP holds an entry for label pushing, swapping, or removing. Therefore, we decide that every LSP tunnel imposes one cost point on the first router in each its parents (like above) plus one cost point on the destination router that removes the label. For a parent that belongs to a funnel, the same procedure as above applies. Note that this cost function could also be differently designed, e.g. by differentiating between reservation states and label states.

3.4.3 Counting State Reduction by Merged LSPs

An LSP funnel keeps track of labels like above in every participating router because of label pushing, swapping and removing. According to [9], reservation states are necessary for the reservations themselves at every outgoing interface and at every incoming interface for the reconciliation with the next outgoing interface (except for the destination router). So we decide to assign one cost point on an LSR for each outgoing or incoming interface of its links, i.e. the number of cost points assigned to a router equals the number of links of the LSP it is attached to. Again, this metric is motivated but could also be defined slightly different.

3.4.4 Evaluation of the Complete LSP Hierarchy

The overall number of states in a router is the sum of states induced by e2e RSVP connections, LSP tunnels and LSP funnels. In a real world scenario, we prefer a network with equally loaded machines (in terms of reservation states) to a network with many little loaded machines and a few heavily loaded machines. Therefore, a meaningful measure for scalability of the entire hierarchy is the maximum of the number of states in each router of the network.

3.5 Application of the Concept

Assume that we have a network topology and a flow matrix. Then, we need algorithms that construct LSP hierarchies that lead to a maximum state reduction in the routers. The outcome is an LSP hierarchy graph that can be easily translated into a network configuration. The graph does not need to be computed in real-time and it suffices to recompute it if the flow matrix has changed significantly. This configuration can be useful in MPLS supported DiffServ networks to track the amount of admitted premium traffic in a scalable way.

4 Simulation Results

In Section 2, we have motivated that the use of a hierarchical LSP structure enforces the reduction of the number of states in the traversed network nodes. In Section 3, we introduced some abstract notation to facilitate the construction of a valid LSP hierarchy. We also proposed a cost function that estimates the number of states in the routers. In this section, we want to demonstrate that an LSP hierarchy can in fact reduce the number of states in the routers.

The aggregation of n flows generates a single new flow, which denotes that only one reservation must be maintained in transit as opposed to n reservations. This is a trivial result that does not need to be confirmed by numerical results and shows that aggregation makes sense. The benefit of layered LSPs in hierarchical network topologies is also easy to see as we discussed in Section 2 (Figure 1).

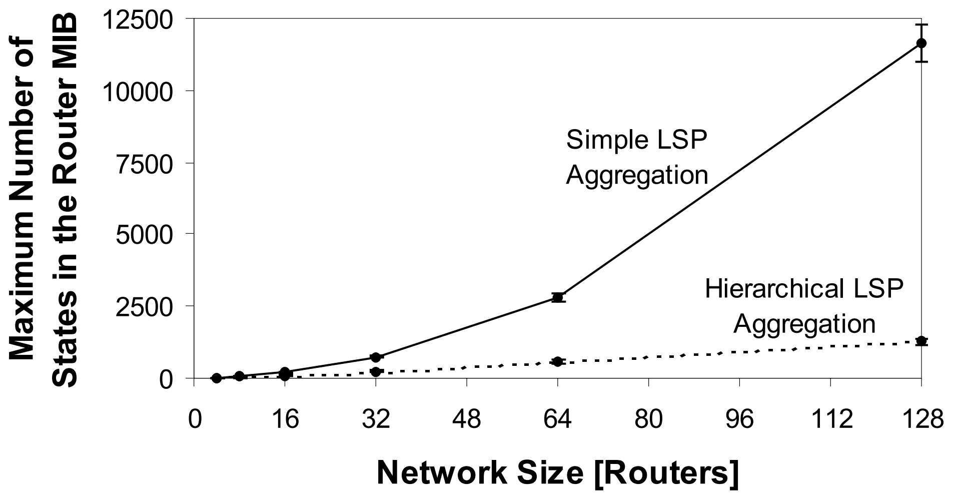

Figure 10: State reduction in the router MIBs applying an LSP hierarchy.

Regularly structured topologies, where only the leaf nodes are able to be source and destination routers for e2e connections, are best for the effectiveness of hierarchical traffic aggregation. But now, we would like to evaluate our approach in arbitrary IP networks. We found algorithms based on the presented graph theoretical concept which will be published soon. Their outputs are valid LSP hierarchies. We used hierarchies without funneled reservations since there is no standardized BGRP-like reservation protocol, yet. The input networking scenarios are randomly constructed spanning trees where, in contrast to Figure 1, every router is an access router. We count the number of MPLS states and Figure 10 shows the maximum number of states in the routers averaged over multiple simulation runs. On the one side we aggregated the traffic in an e2e LSP to eliminate the RSVP connections in the network. On the other side, we continued to aggregated even the resulting LSPs. The error bars are computed for a reliability of 95%. Without further LSP aggregation, the growth of the number of states for the administration of the transit LSP is quadratic. As mentioned before, the worst case scenario for tunneling with respect to scalability is the pure star topology which can not be optimized by a hierarchical LSP structure. However, Figure 10 shows that the LSP hierarchy is able to reduce the growth of that number to be linear for the average of randomly constructed networks. For 128 routers, the maximum number of MPLS states is reduced by 90%.

These results show that LSP hierarchies are able to provide full e2e connectivity where number of reservation states in the router MIBs scales on average only linearly with the network size. This effect can be enforced by suitable network topologies where the transit routers can not serve as source and destination for e2e connections. Flow merging will further reduce the number of reservation states. Hence, we have shown that scalable resource management can be achieved by hierarchically structured overlay networks.

5 Conclusion and Outlook

We gave a short introduction to IntServ and DiffServ and pointed out their shortcomings. We explained the scalability problem due to per flow signaling. We suggested to solve that problem by reservation aggregation. In contrast to many other works, we presented a hierarchical approach that can be realized in MPLS technology using LSPs.

The main contribution of this paper is the graph theoretical formulation of the LSP hierarchy and the translation of technical constraints into that language. A cost function allows for evaluating the LSP hierarchy. That leads to an optimization problem that yields a scalable LSP structure to support e2e reservations. The simulation results showed that due to the hierarchically structured overlay network, the maximum number of states in a router scales rather linearly than quadraticly with the network size in a fully connected average spanning tree network. Almost 90% of the reservation states can be saved in a network with 128 routers. Smart topologies and merging of flows (reservations) can further reduce the number of states that the routers need to track. The resulting LSP hierarchy can be immediately used for the configuration of an MPLS network.

Currently, we are working on algorithms that optimize the LSP hierarchy. Furthermore, we explore network structures that support the scalability of reservation signaling [16]. Finally, enhancements of existing protocol suites are required to set up LSP hierarchies and traffic engineering methods need to be integrated to operate the networks efficiently.

References

[1] B. Braden, D. Clark, and S. Shenker, “RFC1633: Integrated Services in the Internet Architecture: an Overview.” http://www.ietf.org/rfc/rfc1633.txt, June 1994.

[2] J. Wroclawski, “RFC2210: The Use of RSVP with IETF Integrated Services.” ftp://ftp.isi.edu/in-notes/rfc2210.txt, Sep. 1997.

[3] B. Braden, L. Zhang, S. Berson, S. Herzog, and S. Jamin, “RFC2205: Resource ReSerVation Protocol (RSVP) - Version 1 Functional Specification.” ftp://ftp.isi.edu/in-notes/rfc2205.txt, Sep. 1997.

[4] S. Blake, D. L. Black, M. A. Carlson, E. Davies, Z. Wang, and W. Weiss, “RFC2475: An Architecture for Differentiated Services.” ftp://ftp.isi.edu/in-notes/rfc2475.txt, Dec. 1998.

[5] K. Nichols, V. Jacobson, and L. Zhang, “RFC2638: A Two-bit Differentiated Services Architecture for the Internet.” ftp://ftp.isi.edu/in-notes/rfc2638.txt, July 1999.

[6] A. Terzis, L. Wang, J. Ogawa, and L. Zhang, “A Two-Tier Resource Management Model for the Internet,” in *Global Internet Symposium 1999*, Dec. 1999.

[7] M. Günther and T. Braun, “Evaluation of Bandwidth Broker Signaling,” in *International Conference on Network Protocols ICNP’99*, pp. 145–152, Nov. 1999.

[8] O. Schelén and S. Pink, “Aggregating Resource Reservations over Multiple Routing Domains,” in *IFIP Sixth International Workshop on Quality of Service (IWQoS’98)*, may 1998.

[9] P. Pan and H. Schulzrinne, “BGRP: A Tree-Based Aggregation Protocol for Interdomain Reservations,” *Journal of Communications and Networks*, vol. 2, pp. 157–167, June 2000.

[10] F. Baker, C. Iturralde, F. Le Faucheur, and B. Davie, “Aggregation of RSVP for IPv4 and IPv6 Reservations.” http://www.ietf.org/internet-drafts/draft-ietf-issll-rsvp-aggr-03.txt, February 2001.

[11] T. Li and Y. Rekhter, “RFC2430: A Provider Architecture for Differentiated Services and Traffic Engineering (PASTE).” ftp://ftp.isi.edu/in-notes/rfc2430.txt, Oct. 1998.

[12] K. Kompella and Y. Rekhter, “LSP Hierarchy with MPLS TE.” http://www.ietf.org/internet-drafts/draft-ietf-mpls-lsp-hierarchy-02.txt, Feb. 2001.

[13] E. C. Rosen, A. Viswanathan, and R. Callon, “Multiprotocol label switching architecture.” http://www.ietf.org/rfc/rfc3031.txt, Jan. 2001.

[14] D. O. Awduche, L. Berger, D.-H. Gan, T. Li, V. Srinivasan, and G. Swallow, “RSVP-TE: Extension to RSVP for LSP Tunnels.” http://www.ietf.org/internet-drafts/draft-ietf-mpls-rsvp-lsp-tunnel-08.txt, Feb. 2001.

[15] B. Jamoussi, O. S. Aboul-Magd, P. Ashwook-Smith, R. Dantu, N. Feldman, E. Gray, J. Heinanen, A. G. Malis, K. Sundell, and T. Worster, “Constraint-Based LSP Setup Using LDP.” http://www.ietf.org/internet-drafts/draft-ietf-mpls-cr-ldp-04.txt, July 2000.

[16] M. Menth, “A Scalable Protocol Architecture for End-to-End Signaling and Resource Reservation in IP Networks,” in *Proc. ITC-17*, (Salvador de Bahia, Brazil), 2001.