Virtual-Trunk-Based Inter-Domain QoS Routing

Rui Prior[1] and Susana Sargento[2]
[1]Instituto de Telecomunicações – Universidade do Porto, R. Campo Alegre 1021/1055, 4169-007 Porto, Portugal
Tel.: +351-220402933 Fax: +351-220402950 E-Mail: rprior@dcc.fc.up.pt
[2]Instituto de Telecomunicações – Universidade de Aveiro, Campus Universitário de Santiago, 3810-193 Aveiro, Portugal
Tel.: +351-234377900 Fax: +351-234377901 E-Mail: ssargento@det.ua.pt

Abstract—**This paper addresses the problem of inter-domain QoS routing with Service Level Agreements (SLA) for data transport between peering domains, using virtual-trunk type aggregates. The problem is formally stated and formulated in Integer Linear Programming. We consider the comparison of this optimal approach with a proposed solution for inter-domain QoS to be part of the framework for end-to-end QoS control. This solution, QoS_INFO, defines an extension to the Border Gateway Protocol (BGP) to transport three different QoS metrics (light load delay, assigned bandwidth and a congestion alarm), and a path selection algorithm using a combination of these metrics. We present simulation results, obtained in ns-2, of standard BGP, BGP with the QoS_NLRI extension, and our own QoS_INFO proposal, and compare them with those of the optimized route set provided by our formulation. The results show that there is no congestion or packet losses using the optimized routes, and that our QoS_INFO proposal yields better QoS than standard BGP or BGP with the QoS_NLRI extension, since it is able to efficiently avoid congested paths. They also show that the impact of QoS_INFO in route stability is relatively low.**

I. Introduction

The provision of multimedia services with real-time requirements through the Internet is conditioned by its ability to ensure that certain Quality of Service (QoS) requirements are met. The introduction of QoS routing mechanisms able to select paths with the required characteristics, namely avoiding congestion, is of major importance towards this goal.

Though much attention has been paid to QoS in IP networks, most of the effort has been centered on intra-domain; much less has been done in the scope of inter-domain, since it is a much more complex problem. In [1] we proposed the QoS_INFO extension to the Border Gateway Protocol (BGP) for inter-domain QoS routing based on both static and coarse-grained dynamic metrics: it uses the light load delay and assigned bandwidth (both static) in order to improve the packet QoS and make better use of network resources, and a coarse-grained dynamic metric for path congestion to avoid overloaded paths. This extension is used as part of the inter-domain QoS solution of the IST Daidalos project [2], where Service Level Agreements (SLA) for data transport between peering operators are represented by virtual trunks.

In this paper we formally state the problem of inter-domain QoS routing with virtual trunks and formulate it as an Integer Linear Programming (ILP) optimization problem. Using this formulation in a Mixed Integer Programming (MIP) code (Xpress-MP from Dash Optimization [3]), we obtain the optimal solution to the inter-domain QoS routing with virtual trunks problem in a given topology and traffic demand matrix. This optimal solution is then used as a baseline for comparison with the solutions provided by different practical protocols with the same topology and traffic matrix, obtained by simulation in ns-2 [4]. These protocols are the standard BGP, BGP with the QoS_NLRI extension [5] conveying static one-way delay information (expected route delay in light load conditions) and our QoS_INFO proposal [1] (based on both static and coarse-grained dynamic metrics). Results show that the QoS parameters of the route set obtained with QoS_INFO are the closest to those of the optimal route set. Specifically, we show that congestion, packet losses and delay are much lower with QoS_INFO than with standard BGP or with QoS_NLRI.

The rest of the paper is organized as follows. Section II contains the formal description of the problem, and section III describes the ILP formulation of the problem. In section IV we compare the optimal results with simulation results from BGP, QoS_NLRI and QoS_INFO. Finally, section V draws our conclusions.

II. Formal Description

In this section we formally describe the problem of inter-domain routing with virtual trunks.

A. Virtual trunk model of the Autonomous Systems

Though the use of some inner information of the ASs is important for inter-domain QoS routing, the exact topology and configuration of the ASs should not be used for two reasons: (1) the level of detail would be excessive, complicating the route computation task and, most important, (2) network operators usually want to disclose the minimum possible amount of internal information about their networks. In this work, we use a "black box" model where only externally observable AS information is disclosed. The intra-domain connections between edge routers are replaced by virtual trunks with specific characteristics interconnecting the peering ASs. Each virtual trunk has a specific amount of reserved bandwidth and an expected delay. These values depend on the internal topology of the AS, on the intra-domain routing and on resource management performed by the operators, and usually reflect SLAs established between the operator of the AS and the operators of the peering ASs.

The virtual trunk model of ASs is illustrated in figure 1. Each virtual trunk corresponds to a particular (*ingress point, egress point*) pair. Figure 1 illustrates the concept: a Service

This work is based on results of the IST FP6 Integrated Project DAIDALOS, funded by the European Community's Sixth Framework Programme. It reflects the authors' views, and the EC is not liable for any use that may be made of this information.

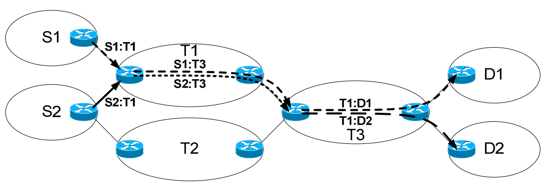


Figure 1: Virtual trunk type SLSs

Level Specification (SLS) between domain S1 and domain T1 states that X traffic may flow between S1 and domain T3; an SLS between domain T1 and domain T3 states that Y traffic may flow between T1 and domain D1. Aggregates are managed internally within each (transit) domain, ensuring that enough resources are assigned.

The configuration of the virtual trunks must be consistent with the inter-domain links. In particular, the summed bandwidth of all virtual trunks traversing AS *j* and going to AS *k* must be less than the bandwidth of the inter-domain link connecting ASs *j* and *k*; similarly, the summed bandwidth of all virtual trunks coming from AS *i* and traversing AS *j* must be less than the bandwidth of the inter-domain link connecting ASs *i* and *j*.

B. Problem statement

Let $G=(V,E)$ be an undirected graph with edge capacities $c_{i,j}$ and edge delays $w_{i,j}$. Each node represents an AS, and the edges correspond to the inter-domain links. Additionally, we define a set F of aggregate flows between pairs of nodes and a corresponding matrix of traffic demands $a_{s,d}$ for all $(s,d)\in V^2$, where s and d denote the source and destination nodes, respectively.

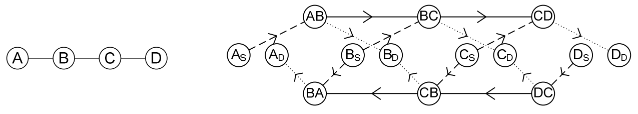
Given any three nodes i, j and k, where i is directly connected to j and j is directly connected to k, there may be a traffic contract (SLS) stating that j provides a virtual trunk between i and k with reserved capacity $r_{i,j,k}$. The amount of data transported from i to k via j is, therefore, bounded by $r_{i,j,k}$. If no such contract exists, we say that $r_{i,j,k}=0$. Since each virtual trunk is mapped to an actual path inside the AS, it has an associated delay $y_{i,j,k}$, corresponding to the delay of that path. We denote L the set of all virtual trunks (i,j,k).

The virtual trunks must satisfy the conditions $o_{j,k} + \sum_i r_{i,j,k} \le c_{j,k}$, where $o_{j,k}$ is the minimum capacity for traffic originated at node j and destined to or traversing node k, and $t_{i,j} + \sum_k r_{i,j,k} \le c_{i,j}$, where $t_{i,j}$ is the minimum capacity for traffic destined to node j and originated at or traversing node i.

The expected total delay suffered by packets of a given flow is the sum of the $w_{i,j}$ and $y_{i,j,k}$ parameters along the path followed by the flow. Our goal is to find the set of hop-by-hop routes that minimize the delay while guaranteeing that inter-domain link and virtual trunk capacities are not exceeded.

C. Problem statement transform

In order to formulate the stated problem as an ILP problem, we first transform the original graph into a transformed graph



a) Original graph b) Transform graph

Figure 2: Simple network with 4 nodes

where the virtual trunks are explicitly accounted for.

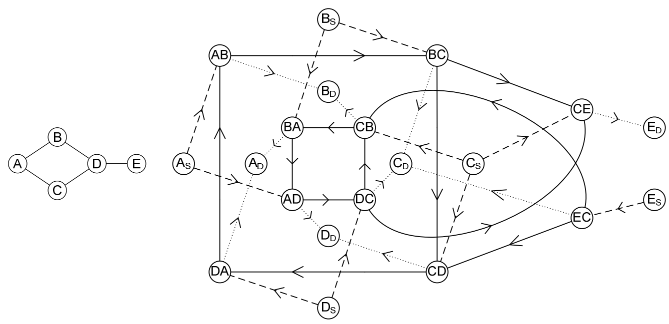
1) Transform graph

It is possible to transform the graph into a directed multigraph where each edge corresponds to a virtual trunk; however, it is difficult to account for the delays of all links in the original graph (inter-domain links) without counting some of them twice. Therefore, we add virtual nodes to the directed multigraph in order to obtain a resulting directed graph.

Virtual trunks are established between an entry link and an exit link. Therefore, we add two virtual vertices per link of the original graph, one for each direction, and virtual trunks are represented by edges connecting these virtual nodes. Moreover, in order to forbid a node of the original graph from being traversed directly (instead of via a virtual trunk), we split each original node into two: one source virtual node with outgoing edges only, and one destination virtual node with incoming edges only. Flows on the transform graph exist between source and destination virtual nodes.

A very simple example of an original graph and its transform with all possible virtual trunks is shown in figure 2. Link (A,B) on the original graph is represented by virtual nodes AB and BA; virtual trunk (A,B,C) is represented by an edge connecting AB to BC; node A is represented by the virtual source and destination nodes A_S and A_D; and flow (A,D) is represented by flow (A_S,D_D), for example.

The solid edge connecting the virtual nodes *ij* and *jk* correspond to the virtual trunk for sending traffic from node i to node k via node j, and has capacity $r_{i,j,k}$ (that of the virtual trunk), and delay $y_{i,j,k} + w_{j,k}$, where $y_{i,j,k}$ is the internal delay of the virtual trunk and $w_{j,k}$ the delay of the inter-domain exit link. Each dashed edge (j_S, jk) corresponds to the inter-domain exit link from node j to node k, and has delay $w_{j,k}$ and infinite capacity. Each dotted edge (ij, j_D) corresponds to the inter-domain entry link in node j from node i, and has zero delay and infinite capacity.



a) Original graph b) Transform graph

Figure 3: Cyclic network with 5 nodes

In the example there is only one possible path from A to C, the virtual trunk through node B. Traffic sent from A to C is subject to a delay equal to the sum of $w_{a,b}$ (from the dashed edge $A_S \rightarrow AB$) and $y_{a,b,c} + w_{b,c}$ (from the solid edge $AB \rightarrow BC$); the dotted edge $BC \rightarrow C_D$ has zero delay. Regarding bandwidth, it is constrained by $r_{a,b,c}$, shared with other traffic.

Figure 3 provides an example with a cyclic graph and its transform containing all possible virtual trunks. Though the transform graph looks overly complex when compared to the original one, the number of variables and constraints in the ILP formulation is not increased, since a formulation based on the original graph would require variable unfolding in order to be linear. Also keep in mind that an undirected graph has half the number of edges of the equivalent directed graph.

2) Generation of the transform graph

In this section we present an algorithm for the generation of the transform graph $G'=(V',E')$ from the original graph G and the set of virtual trunks, informally described above. The algorithm is as follows:

1. For each node $i \in V$
   1.1. Add node i_S to the set S of sources and to the set V' of nodes; add node i_D to the set D of destinations and to V'
2. For each (undirected) edge $\{i,j\} \in E$
   2.1. Add node ij to V'; add node ji to V'
   2.2. Add edge (ij, j_D) to the set E' of edges, with capacity $c'_{ij,j_D} = \infty$ and delay $w'_{ij,j_D} = 0$; add edge (ji,i_D) to E', with capacity $c'_{ji,i_D} = \infty$ and delay $w'_{ji,i_D} = 0$
   2.3. Add edge (i_S,ij) to E', with capacity $c'_{i_S,ij} = \infty$ and delay $w'_{i_S,ij} = w_{i,j}$; add edge (j_S, ji) to E', with capacity $c'_{j_S,ji} = \infty$ and delay $w'_{j_S,ji} = w_{j,i}$
3. For each (directed) virtual trunk $(i,j,k) \in L$
   3.1. Add edge (ij, jk) to E' and to the set L' of virtual trunk edges, with capacity $c'_{ij,jk} = r_{i,j,k}$ and delay $w'_{ij,jk} = y_{i,j,k} + w_{j,k}$
4. For each flow $(i,j) \in F$
   4.1. Add flow (i_S, j_D) to the set F' of flows; set traffic demand $a'_{i_S,j_D} = a_{i,j}$

When the algorithm finishes, we have the transform graph $G'=(V',E')$, the associated edge capacity and edge delay matrices C' and W', a set $L' \subset E'$ of virtual trunk edges, a set $S \subset V'$ of source nodes, a set $D \subset V'$ of destination nodes, a set F' of flows, and the respective traffic demand matrix A'.

3) Complexity of the transform graph

The number of nodes and edges of the transform graph G' is related to the original (undirected) graph G and the set of virtual trunks in the following way. The number of nodes is two per node of the original graph (one source and one destination, e.g., A_S and A_D) plus two per edge of the original graph (one for each direction, e.g., AB and BA). The number of edges is four per edge of the original graph (combinations of source/destination and transmission/reception, e.g., (A_S,AB), (AB, B_D), (B_S,BA) and (BA,A_D)) plus one per virtual trunk (e.g., (AB,BC)). In the example of figure 3, the original graph has 5 nodes, 5 edges and 12 possible virtual trunks. The transform, therefore, has 20 nodes (2*5+2*5) and 32 edges (4*5+12).

4) Back conversion of the routes

A route p' on the transform graph may be converted back to a route p on the original graph by analyzing the traversed edges. Each traversed edge on the transform graph corresponds to a traversed node on the original graph according to the following: (i_S, ij) corresponds to i, (ij, jk) to j, and (jk, k_D) to k. For example, the route (B_S,BC,CE,E_D) on the transform graph of figure 3.b) corresponds to the route (B,C,E) on the original graph.

III. Formulation as ILP Problem

We now formulate our bandwidth-constrained global route and delay optimization hop-by-hop routing problem as an ILP problem with boolean variables using the transform graph. In section IV this formulation will be used in a Mixed Integer Programming (MIP) solver to obtain optimal route sets against which we compare the results of our proposal.

In the transform graph, formulation is somewhat simpler than in a generic graph, since some restrictions are already enforced by the topology: since there are no incoming edges in source nodes, it is not necessary to use a restriction disallowing incoming traffic for flows originated at those nodes (similarly for destination nodes).

Our objective is to minimize the global delay while respecting the bandwidth limits, assuming that the network has enough capacity to satisfy all demands. In addition to the transform data obtained by the above described algorithm, let us define a set of positive flow weights $b_{s,d}$ for all $(s,d) \in F'$. Two different optimizations may be obtained by using different weight values. The first alternative uses $b_{s,d}=1, \forall (s,d) \in F'$, stating that all flows have equal importance - optimization is performed on a per-route basis. The second alternative uses $b_{s,d} \propto a'_{s,d}, \forall (s,d) \in F'$, stating that a flow's importance is proportional to its traffic demand - optimization is performed on a traffic volume basis. We define the boolean decision variables x_{ij}^{sd} which take the value 1 if the flow $(s,d) \in F'$ is routed through the edge $(i,j) \in E'$ and 0 otherwise.

The problem can, thus, be formulated as follows:

$$\text{Minimize} \sum_{(i,j)\in E'} \sum_{(s,d)\in F'} b_{s,d} w'_{i,j} x_{i,j}^{s,d} \text{ subject to}$$

$$x_{i,j}^{s,d} \in \{0,1\}, \forall (s,d) \in F', (i,j) \in E' \quad (1)$$

$$\sum_{(s,d)\in F'} a'_{s,d} x_{i,j}^{s,d} \le c'_{i,j}, \forall (i,j) \in L' \quad (2)$$

$$\sum_{i\in V':(i,j)\in E'} \sum_{(s,d)\in F'} a'_{s,d} x_{i,j}^{s,d} \le c_i, \forall i \in (V'-S-D) \quad (3)$$

$$\sum_{(j,k)\in E'} x_{j,k}^{s,d} - \sum_{(i,j)\in E'} x_{i,j}^{s,d} = 0, \forall (s,d) \in F', j \in (V'-\{s,d\}) \quad (4)$$

$$\sum_{(s,j)\in E'} x_{s,j}^{s,d} = 1, \forall (s,d) \in F' \quad (5)$$

$$\sum_{(i,d)\in E'} x_{i,d}^{s,d} = 1, \forall (s,d) \in F' \quad (6)$$

$$\sum_{j\in V':(i,j)\in E'} \sum_{\substack{t\in S\\ t\ne d}} x_{i,j}^{t,d} \le |S| \cdot x_{s,i}^{s,d}, \quad \forall (s,d) \in F', i \in S : (s,i) \in E' \quad (7)$$

$$\sum_{(i,j)\in E':(j,d)\in E'} \sum_{\substack{s\in S \\ s\neq d}} x_{i,j}^{s,d} = \sum_{j\in V':(j,d)\in E'} \sum_{\substack{s\in S \\ s\neq d}} x_{j,d}^{s,d}, \forall d \in D \tag{8}$$

Constraint set (1) imposes boolean decision variables, meaning that flows cannot be split over multiple paths; (2) states that the sum of all flows traversing a virtual trunk edge will not exceed its capacity; (3) states that the sum of all flows traversing (leaving) a virtual node i corresponding to an inter-domain link in the original graph must be less than c_i, the capacity of the inter-domain link.

Constraint sets (4), (5) and (6) are the mass balance equations: (4) means that each flow entering a node that is neither source nor destination for that flow must leave it and vice-versa; (5) means that each flow leaves the source node once and, conversely, (6) means that each flow enters the destination node once.

Constraint set (7) imposes hop-by-hop routing on the original graph: it means that, if a flow from a given node to a certain destination leaves that node by a given link, no flow to the same destination traversing that node may leave that node by a different link. On the transform graph, it means that, if a flow from a source to a destination traverses a given virtual node directly connected to that source, no other flows to the same destination may traverse a different virtual node connected to the same source. Finally, (8) prevents routing loops at the destination nodes of flows in the original graph, by forcing flows arriving at a node directly connected to their destination virtual node to use that direct path. Failing this, a flow would be counted twice (or more) on the left hand side and only once on the right hand side, invalidating the equality.

Routes obtained through this optimization can be proven to contain no cycles [6].

IV. Optimal and Practical Results

In this section we compare the characteristics of the optimal route set obtained using the ILP formulation of section III with simulation results obtained in ns-2 [4] using our QoS_INFO proposal for inter-domain QoS routing, the standard BGP, and BGP with the QoS_NLRI extension conveying static one-way delay information (the expected delay of the route in light load conditions – scenario illustrated in [5]). We consider the network topology depicted in Figure 4. The presented results concern congestion, packet losses and delay.

In the simulations we used three different packet sizes: 50% of packets with 40 bytes (representing 4% of the traffic volume), simulating SYN, ACK, FIN and RST TCP segments; 20% of packets with 80 bytes, simulating packetized voice (3% of traffic volume); and 30% of packets with 1500 bytes, simulating full size TCP segments (93% of traffic volume). These packet sizes reflect the bimodality currently observed in internet traffic [7], complemented with voice packets, whose frequency tends to increase.

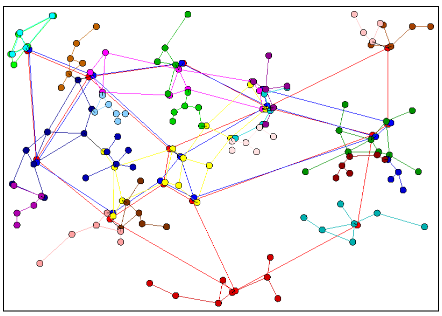


Figure 4**:** Simulated topology

We ran simulations for 8200 simulated seconds, discarding data for the first 1000 seconds, to filter out transient effects.

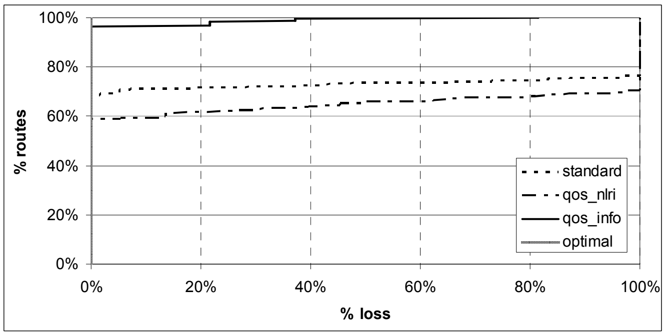


Figure 5: Percentage of routes with loss probability ≤ X

Figure 5 shows the packet loss probability cumulative distribution function (CDF) for the routes at the end of the simulation[1] in the different scenarios. The QoS_INFO approach yields better results, with 96.5% of the routes having a negligible packet loss probability, contrasting to only 58.8% in QoS_NLRI and 68.0% in the standard BGP. As expected, in the route set obtained from optimization, 100% of the routes had no packet losses (vertical line on the y axis).

The fact that congestion (and, consequently, packet loss) is worse in QoS_NLRI than in standard BGP is probably related to the fact that by minimizing the number of AS hops, standard BGP tends to exploit the hierarchical character of the network by preferring a more logical path comprising a small number of transport operators with broad geographical coverage[2] to a path consisting on a large number of operators with small coverage that may, nevertheless, have a lower light load delay value.

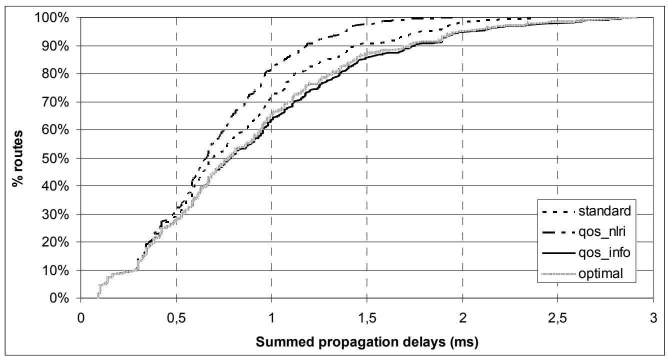


Figure 6: Percentage of routes with light load delay ≤ X

[1] Since routing with QoS_INFO is based on dynamic information, routes do change in the course of the simulations; in standard BGP and BGP with QoS_NLRI all routes are stable during the useful simulation period.

[2] In non-hierarchical topologies standard BGP performed worse than QoS_NLRI with respect to congestion.

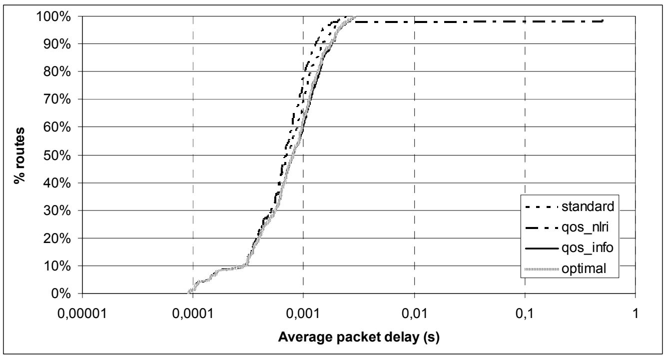


Figure 7: Percentage of routes with average packet delay ≤ X

Figure 6 shows CDFs of the summed propagation delays for the routes in the three scenarios (in the cases of QoS_NLRI and QoS_INFO, they correspond to the announced delay values). As expected, QoS_NLRI performs better in this respect, even better than the optimizations, since the routes with the lower delay metric are always chosen, ignoring virtual trunk capacity and congestion. Interestingly, standard BGP also does better than the optimal solutions in this respect, since it also ignores capacities and congestion. The QoS_INFO curve follows the optimal curves very closely. It is worth noting that the light load expected delay holds little significance if routes are congested (heavily loaded); therefore, a much more meaningful parameter is the expected packet delay for the routes (sum of propagation and transmission delays with the expected queuing delays along the path). The packet delay results in Figure 7 show that there is no link congestion in most of the cases, therefore the route delays are kept low. Nevertheless, 0.5% of the routes in QoS NLRI traversed a congested link and suffered large delays. Except for these routes, the delays are close in all cases, with the QoS_INFO curve practically overlapping those of the optimizations.

Table 1 shows a summary of the results. Since the traffic in the virtual trunks is policed and the virtual trunks are consistent with the capacity of the inter-domain links, there are practically no overloaded links, as link overload can only stem from excess traffic generated at the transmitting AS. Even so, there is one overloaded link (1.4%) in QoS_NLRI. Regarding virtual trunks, 7.6% were congested with standard BGP and 8.5% with QoS_NLRI, whilst with QoS_INFO it was only 1.4%. Even more significant is the fact that, both in standard BGP and QoS_NLRI, there was traffic sent on virtual trunks that were not established (that is, on triplets (a,b,c) for which there was no SLS), corresponding to 15.6% and 15.2%, respectively, of the established virtual trunks. This problem did not occur with QoS_INFO.

Overloaded links and virtual trunks mean that routes traversing them will suffer losses. The percentage of routes with losses was 32.0% with standard BGP, 41.2% with QoS_NLRI, and only 3.5% with our QoS_INFO proposal. In terms of overall packet losses, the loss figures were 17.1% of the offered traffic with standard BGP, 28.2% with QoS_NLRI and a much lower 0.4% with QoS_INFO. Obviously, the optimization results suffered none of the aforementioned problems.

Table 1: Summary of results

	Overloaded		AS triplets with traffic and no SLS (% of v. trunks)	Packet losses	Routes with losses
	links	v. trunks			
Standard	0.0%	7.6%	15.6%	17.1%	32.0%
QoS_NLRI	1.4%	8.5%	15.2%	28.2%	41.2%
QoS_INFO	0.0%	1.4%	0.0%	0.4%	3.5%
Optimal	0.0%	0.0%	0.0%	0.0%	0.0%

The drawback of using dynamic metrics is route stability. Even so, the nature of the congestion alarm greatly alleviates this problem in QoS_INFO: 95% of the routes were completely stable (did not change) during the useful simulation period.

V. Conclusions

This paper addressed the problem of inter-domain QoS routing using virtual trunk type aggregates for the indirect transport of traffic between two different administrative domains across a third one. These virtual trunks are usually (though not necessarily) defined by means of agreements between the operators of peering domains, and each inter-domain path consists on a concatenation of virtual trunks. We formally stated the problem of SLA-aware inter-domain QoS routing and formulated it as an Integer Linear Programming optimization problem. Simulations were performed to evaluate our QoS_INFO proposal and to compare it to standard, QoS-unaware BGP and to the QoS_NLRI extension, and also with the results obtained through the optimization procedure. The results show that although the use of the QoS_NLRI extension represents an improvement over standard BGP, routing using only static QoS parameters is equally unable to avoid path congestion. With our QoS_INFO extension, congested paths and their consequences on QoS are almost entirely avoided, and the results approach the optimal ones. Although there is a penalty in overhead and route stability in QoS_INFO, most of the routes are stable.

References

[1] R. Prior, S. Sargento, “Towards Inter-Domain QoS Control.” In Proc. ISCC-2006, Pula-Cagliari, Sardinia, Italy, June 2006.

[2] DAIDALOS project (IST-2002-506997), <http://www.ist-daidalos.org/>

[3] Dash Optimization, <http://www.dashoptimization.com/>

[4] The Network Simulator – ns-2, <http://www.isi.edu/nsnam/ns/>

[5] G. Cristallo and C. Jacquenet, “The BGP QOS_NLRI Attribute.” IETF Internet Draft, February 2004.

[6] R. Prior, S. Sargento, ”Inter-domain QoS Routing: Optimal and Practical Study”. To appear in IEICE Transactions on Communications.

[7] R. Sinha, C. Papadopoulos and J. Heidemann, “Internet Packet Size Distributions: Some Observations.” October 2005. <http://netweb.usc.edu/~rsinha/pkt-sizes/>