Guaranteeing Service Continuity in Traffic Engineered Networks

Sven Van den Bosch, Gert Van Hoey, Natalie Degrande, Paloma de La Vallée-Poussin, Jan Janssen and Hans De Neve, Alcatel Bell - Network Strategy Group, Francis Wellesplein 1, B-2018 Antwerp, Belgium

Abstract

The interest in Differentiated Services and Quality of Service in networks has raised considerable interest in traffic engineering. Traffic engineering allows the operator to optimise the routing in his network. Now, as a next step, network engineering offers the capability to dynamically configure the capacity of links over a so-called Automatically Switched Transport Network. A combined optimisation of routing and topology becomes a challenging task. However, when this problem has been solved, a related problem occurs when the operator wants to convert the existing configuration into the one that was optimised by the algorithm. In general, the transition between both configurations in a make-before-break fashion cannot be guaranteed. This paper outlines an approach for optimising the transition between different LSP and optical path configurations.

1 Introduction

RFC 2702 [1] defines Internet traffic engineering as that aspect of Internet network engineering that deals with the issue of performance evaluation and performance optimisation of operational IP networks. This definition, when applied at different time scales, is broad enough to cover both routing optimisation and planning. For optimisation algorithms, however, both aspects are typically considered separately. Planning decisions such as the addition of routers, links or link capacity, are typically taken at large time scales. They intend to match the configuration of the network to the long-term traffic trends or forecasts. A more restricted, but widely used, definition of traffic engineering considers routing optimisation only. Routing optimisation operates at much smaller time scales and may involve tailoring the routing configuration in the network to periodical and/or trend behaviour. ISPs that are finding that their network traffic is both asymmetrical and changes direction with time may apply routing optimisation in order to increase the utilisation of their network. This optimisation may ultimately postpone network investments. At this time scale, however, topology and (physical) capacity of the network are typically fixed. Depending on the required responsiveness to traffic changes an on-line or off-line traffic engineering algorithm can be used.

In many cases, the ISP does not own the entire network it operates. Instead, (transport) capacity can be leased from a transport provider. This effectively offers the ISP the possibility to dynamically reconfigure its topology as required. The resulting network is known as an Automatically Switched Transport Network (ASTN) [2]. This kind of network is based on dynamic circuit provisioning, i.e. reconfiguring the connectivity at the optical layer or SDH/SONET layer in response of the dynamic behaviour of IP traffic. Although changes in connectivity pattern can be in the order of minutes/seconds, this type of networks is referred to as "routed" because it makes use of IP/MPLS based methods, protocols and algorithms. It is also referred to as a G-MPLS network [3], because use is made of a generalised MPLS control plane.

A traffic engineered solution for an ASTN such as the one described in [4] can specify both the optical paths (ΛSP) to be configured over the ASTN and the explicit path over this logical topology for each LSP. An operator may wish to carry out such optimisation at regular time intervals. In this case, he may wish to keep his existing configuration and add new ΛSPs s or LSPs as needed. It is, however, likely that better optimisation can be obtained when all ΛSPs and LSPs are left to be optimised. At this point, subsequent solutions may require some ΛSPs to be torn down and/or some LSPs to change paths. However, although both the current situation and the desired configuration must clearly be feasible, it cannot be guaranteed that enough resources are available to implement all new ΛSP and LSP before deleting the old ones. Hence, service continuity (make-before-break [5] transition) cannot be guaranteed. The feasibility of the make-before-break transition between both solutions typically depends on the order in which the modifications are carried out. The aim of this contribution is to outline a transition algorithm that optimises the order in which to set up and tear down the LSPs and ΛSPs in the network. The main objective of such an algorithm obviously is to find a make-before-break transition if it exists.

The remainder of the text is organised as follows. Section 2 presents the algorithm for LSP and ΛSP transition. Section 3 provides an example of the LSP transition optimisation in a fixed topology. Finally, the conclusion summarises the most important contributions and results.

2 Transition optimisation

This section describes the transition algorithm used for optimising the order in which to set up and/or tear down ΛSP and LSPs over an ASTN. At this point we assume that we have two LSP and ΛSP configurations one of which is currently implemented in the network (the old configuration *O*) and one of which is the desired configuration (the new configuration *N*). Possible transition steps include adding or deleting an ΛSP, adding or deleting an LSP and changing the path of an existing LSP.
Section 2.1 gives a functional overview of the features that are required to support such a transition algorithm. Section 2.2 provides a mathematical (ILP) formulation of the algorithm used in our current implementation.

2.1 Functional overview

The aim of the transition algorithm is to find the optimal order in which to set up and tear down the LSPs and ΛSPs in the network. LSPs that are only present in the old configuration are deleted prior to all other transition actions. The associated ΛSP capacity is maintained during the transition (as long as possible). LSPs that only occur in the new configuration are not set up until all other LSPs have been dealt with, thus effectively contributing to the capacity needed during the transition. Some LSPs will be present both in the new and the old configuration and may need to change their explicit path. The transition algorithm outlined in the remainder of this document will be run on these LSPs only.
The main objective of the algorithm obviously is to find a make-before-break transition if it exists. It will try to do so by defining a sequence of network states that allow for a gradual transition from the original state to the desired state. The feasibility of each intermediate state is determined by two constraints. First, the LSP load on each link may not exceed the link capacity. Second, the number of links emanating from each router may not exceed the number of ports available on that router. The transition algorithm is based on the observation that in order to guarantee make-before-break transition of LSPs and ΛSPs, capacity and ports need to be available, not only for the current state, but *for the next state as well*.
If a complete make-before-break transition is not possible, then the number of broken LSPs should be minimised. Note that breaking an ΛSP breaks all LSPs running over it. We capture this by minimising the number of broken ΛSP prior to minimising the number of broken LSPs.
The transition should allow for complete rollback if possible. Complete rollback refers to the situation in which the set-up of ΛSPs and LSPs can be reversed in order to return to a previously existing configuration. Complete rollback can only be guaranteed for a single transition step because in a multi-user environment, once an ΛSP is released, it cannot be guaranteed that it will not have been taken by another user at one time in the future.
All other things being equal, the number of transition steps should be minimised in order to increase the number of simultaneous LSP set-ups.
A transition step is can now be defined more formally as the following sequence of events.

- Set up any number of new ΛSPs.
- Re-route any number of existing LSPs from their old explicit path to the new desired one. This includes setting up new paths for all involved LSPs before tearing down the old paths.
- Tear down any number of old ΛSPs.

For computational efficiency, we also request that LSPs that are transitioned cannot return to their old path, that capacity on links with increasing capacity should be monotonically increasing and that capacity on links with decreasing capacity should remain constant until the last transition (if complete rollback is required) and should be monotonically decreasing if piecewise rollback is sufficient.

2.2 Mathematical description of the transition algorithm

This section describes a mixed-inter linear program (MILP) capturing the functional description of the previous section. This MILP will be used to optimise the transition sequence in the network.
We assume that K is the set of LSPs that needs to be changed in the transition. Er is the set of links that are in both the old and the new topology. Ed is the set of links that will be deleted and E a is the set of links that will be added. Table 1 lists the variables and constants used in the MILP formulation.

p_k^s	Equals 1 if the k-th LSP uses the old path in state s and 0 otherwise
π_k^s	Equals 1 if the k-th LSP is make-before-break transitioned between state s and s+1 and 0 otherwise
l_e^s	Equals 1 if the e-th ΛSP is available in state s and 0 otherwise
ε_e^s	Equals 0 if the e-th ΛSP is make-before-break transitioned between state s and s+1 and 1 otherwise
c_e	Capacity of the e-th ΛSP
$\omega_{ke}^o \left(\omega_{ke}^n\right)$	Equals 1 if the old (new) path of the k-th LSP uses the e-th ΛSP and 0 otherwise
δ_{ep}	Equals 1 if the e-th ΛSP uses the p-th port and 0 otherwise

Table 1: Variables and constants used in the MILP formulation of the transition algorithm

With the above definitions, the functional model can be implemented as follows. Equation (1) with $l_e^0 = 1$ and $e \in E_d, s \in S \setminus s_{max}$ ensures that deleted ΛSPs cannot be reused in subsequent states. Equation (2) with $e \in E_d, s \in S \setminus s_{max}$ expresses that break-before-make can only occur when a ΛSP is deleted prior to offloading all LSPs from it. Equation (3) $l_e^0 = 0$ $e \in E_a, s \in S \setminus s_{max}$ ensures that capacity is monotonically increasing on ΛSPs that are added. With $l_e^s = 1$ $e \in E_r, s \in S$, it expresses that ΛSPs that are in both the old and the new topology are available at all times. Equation (4) $p \in P, s \in S$ expresses that if port constraints are violated, all LSPs on a ΛSP that is deleted are broken (we call this a broken ΛSP).

$$l_e^{s+1} \le l_e^s \tag{1}$$

$$\varepsilon_e^s \le l_e^s - l_e^{s+1} \tag{2}$$

$$l_e^s \le l_e^{s+1} \tag{3}$$

$$\sum_{a \in E_a} l_a^{s+1} \cdot \delta_{ap} + \sum_{d \in E_d} \left(l_d^{s-1} - \varepsilon_d^s \right) \cdot \delta_{dp} \le 1 \tag{4}$$

Similar equations govern the transition behaviour of the LSPs. Equation (5) with $k \in K, s \in S\_$ ensures that LSPs cannot return to their old path once they are transferred to the new path. Equation (6) with $k \in K, s \in S\_$ expresses that break-before-make of LSPs can only occur when they are transferred from their old to their new path. Equation (7) with $e \in E_r \cup E_d \cup E_a, s \in S$ ensures that enough capacity is available for all LSPs on the old or new path and for all LSPs that are transitioned in a make-before-break way during the next transition.

$$p_k^{s+1} \le p_k^s \tag{5}$$

$$\pi_k^s \le p_k^s - p_k^{s+1} \tag{6}$$

$$\sum_{k \in K} \left(\omega_e^o \cdot d_k^o \cdot p_k^s \right) + \sum_{k \in K} \left(\omega_e^n \cdot d_k^n \cdot \left(1 - p_k^s \right) \right) + \sum_{k \in K} \left(\left(\omega_e^n \cdot d_k^n - \omega_e^o \cdot d_k^o \right) \pi_k^s \right) \le l_e^s \cdot c_e \tag{7}$$

Three performance objective are derived from these variables. B_{link} equals the number of broken ΛSPs (8). B_{LSP} equals the number of broken LSPs (9). D equals the delay with which the transition takes place (10).

$$B_{link} = \sum_{e \in E_d} \sum_{s \in S \setminus s_{max}} \varepsilon_e^s \tag{8}$$

$$B_{LSP} = \sum_{k \in K} \sum_{s \in S \setminus s_{max}} \left(1 - \pi_k^s \right) \tag{9}$$

$$D = \sum_{k \in K} \sum_{s \in S \setminus s_{max}} p_k^s \tag{10}$$

The objectives are combined into an objective function which is to be minimised (11).

$$M^2 \cdot B_{link} + M \cdot B_{LSP} + D \tag{11}$$

It is instructive to calculate the number of variables in this mixed-integer linear program. The number of 0/1 variables equals the product of the number of LSPs and the number of transitions plus the product of the number of ΛSPs and the number of transitions. In a realistic example, the number of LSPs may be a few thousands and the number of transitions may be limited to about 100. The number of ΛSPs could be around 200. In this case, the number of 0/1 variables is in the order of magnitude of 100000. Despite the considerable number of variables in the problem, the algorithm proves to solve fairly quickly for practical examples.

3 Example

An example is included in order to demonstrate the capabilities of the algorithm. This example is deliberately kept simple for presentation purposes. We verify the transition algorithm described in section 2 for the make-before-break transition of LSPs in a constant ΛSP configuration.

The transition problem is solved for the four node network shown on **Fig. 1**. The links are assumed to be OC-3 (155 Mb/s).

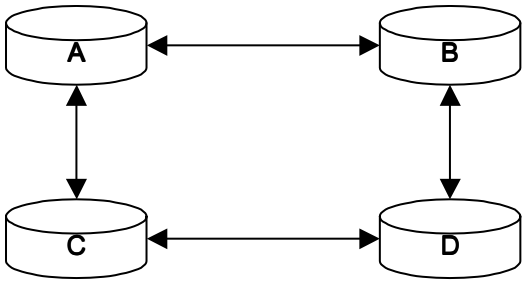

Fig. 1: Make-before-break LSP transition: problem description

The problem encompasses the optimisation of the make-before-break transition of eight LSPs from their old path, shown in Table 2, to their new path, shown in Table 3. The reserved capacity for the LSPs is either 45 or 60 Mb/s.

When the LSPs are treated sequentially (or in random order), the transition between both LSP configurations is usually not feasible in a make-before-break fashion. However, with the transition optimisation, the optimal order in which to set up and tear down LSPs is found. The optimal transition for our problem is shown in Table 4.

The interested reader can easily check that in each intermediate state capacity is available both on the old and the new path for the LSPs that are about to change their path. Also, in all intermediate states, the capacity available on each of the links is sufficient for the LSP configuration in that state.
In our example, no LSPs were broken and the delay was 0.306.

LSP	Path	Size [Mb/s]
0	C-A-B	45
1	A-B-D	60
2	B-D-C	60
3	D-C-A	60
4	B-A-C	45
5	A-C-D	45
6	C-D-B	60
7	D-B-A	45

Table 2: Old LSP configuration

LSP	Path	Size [Mb/s]
0	C-D-B	45
1	A-C-D	60
2	B-A-C	60
3	D-B-A	60
4	B-D-C	45
5	A-B-D	45
6	C-A-B	60
7	D-C-A	45

Table 3: New LSP configuration

Step 1
add LSP2 on path B-A-C
delete LSP2 from path B-D-C
Step 2
add LSP4 on path B-D-C
add LSP5 on path A-B-D
add LSP7 on path D-C-A
delete LSP4 from path B-A-C
delete LSP5 from path A-C-D
delete LSP7 from path D-B-A
Step 3
add LSP1 on path A-C-D
add LSP3 on path D-B-A
delete LSP1 from path A-B-D
delete LSP3 from path D-C-A 2
Step 4
add LSP6 on path C-A-B
delete LSP6 from path C-D-B
Step 5
add LSP0 on path C-D-B
delete LSP0 from path C-A-B

Table 4: Required set up order for make-before-break transition

4 Conclusion

In this paper, we presented a transition approach between subsequently optimised solutions. This transition approach was verified on a small LSP transition example on a fixed ΛSP topology.

5 References

[1] D. Awduche, J. Malcolm, J. Agogbua, M. O'Dell and J. McManus, “Requirements for Traffic Engineering Over MPLS,” RFC 2702, informational, September 1999.

[2] “Architecture of Optical Transport Networks,” ITU-T Recommendation G.872, Februari 1999.

[3] Banerjee, “Generalised Multiprotocol Label Switching: An Overview of Routing and Management Enhancements,” IEEE Communications Magazine, Jan. 2001.

[4] G. Van Hoey, S. Van den Bosch, P. de La Vallée-Poussin, N. Degrande and H. De Neve, “MPLS Traffic Engineering over Automatically Switched Transport Networks,” submitted to European Transaction on Telecommunications – special issue on Internet Traffic Engineering

[5] D. O. Awduche, L. Berger, D.-H. Gan, T. Li, V. Srinivasan and G. Swallow, “RSVP-TE: Extensions to RSVP for LSP Tunnels,” draft-ietf-mpls-rsvp-lsp-tunnel-08.txt, work in progress