Available at
www.ComputerScienceWeb.com
POWERED BY SCIENCE DIRECT

Computer Communications 26 (2003) 845–860

computer communications

www.elsevier.com/locate/comcom

Quality of service provisioning through traffic engineering with applicability to IP-based production networks

Panos Trimintzios[a,*], Timothy Baugé[b], George Pavlou[a], Paris Flegkas[a], Richard Egan[b]

[a]*Centre for Communication Systems Research, University of Surrey, Guildford, Surrey, GU2 7XH, UK*
[b]*Thales Research Ltd., Worton Drive, Reading, RG2 0SB, UK*

Received 7 August 2002; accepted 7 August 2002

Abstract

Production networks require the transport of high-quality multimedia traffic between outside broadcast vans and the main studio. This is typically done through dedicated terrestrial or satellite links, with bandwidth purchased from third party network providers, which is expensive and lacks flexibility. Given the emergence of IP networks and the Internet as the multi-service network of choice, it is plausible to consider their use for transporting production network traffic with high bandwidth and low delay and packet loss requirements. Emerging technologies for quality of service such as Differentiated Services and MPLS can be used for premium quality traffic. In this paper we try to use the emerging IP technologies to support services like production network traffic. We present a Traffic Engineering and Control System that starts from agreed services with customers and provisions the network according to the expected traffic demand so as to meet the requirements of contracted services while optimising the use of network resources. We devise a non-linear programming formulation of the problem and show through extensive simulations that we can achieve the objectives and meet the requirements of demanding production network traffic. Our solution is generic enough and not only tuned to production networks, so it can be used in other contexts for supporting services with stringent quality of service requirements.

Keywords: Quality of service; Traffic engineering; Differentiated services; Production networks

1. Introduction

Differentiated Services [1] is seen as the emerging technology to support Quality of Service (QoS) in IP backbone networks in a scalable fashion. Multi-Protocol Label Switching (MPLS) [2] can be used as the underlying technology to support Traffic Engineering (TE). It is possible to use these technologies to support production network or other multimedia traffic with stringent real-time requirements. This can be done through careful traffic forecast based on contracted premium services with customers and subsequent Network Provisioning (NP) in terms of routing and resource management strategies.

Production network traffic has very stringent requirements in terms of bandwidth, delay and loss. Today, production networks use dedicated high-capacity network infrastructures. These are inflexible and very expensive. In this paper we show that it is feasible to support such demanding traffic requirements in a generic IP-based network environment. This can be achieved by taking advantage of the emerging IP technologies, Differentiated Services (DiffServ) and MPLS, and through careful provisioning and service management and control decisions. Although the key motivation behind our work is the support of production network traffic, our findings are generic enough and not only tuned to production networks, so that they can be used in other contexts for supporting services with stringent quality of service requirements.

This rest of this paper is organised as follows. Section 2 introduces production networks and their connectivity requirements and Section 3 describes the related work. In Section 4 we describe the QoS provisioning system for supporting production network traffic and describe its operational cycle. Section 5 describes the NP algorithm that is based on a non-linear programming formulation of the problem. In Section 6 we provide the experimental evaluation of our TE system through simulations. Finally, in Section 7 we present our conclusions.

* Corresponding author. Tel.: +44-1483-686005; fax: +44-1483-686001.
E-mail address: p.trimintzios@eim.surrey.ac.uk (P. Trimintzios).

0140-3664/03/$ - see front matter © 2002 Elsevier Science B.V. All rights reserved.
PII: S0140-3664(02)00218-9

2. Packet-based production networks

Two types of studios are identified in production networks: the Main Studio and Outside Broadcast (OB) Vans. The main studio is where the filming takes place, where the cameras, microphones, television monitors and speakers are found. It accommodates the producer who manages the personnel through talkback headset and controls how the programme is put together. A storage area is the part of the studio where audio, video and data need to be stored, either short term for deferred broadcast or long term for archiving. The second type of studios are the OB Vans, which are mobile studios, taken to film events outside the main studio, such as races, concerts, sport events, foreign correspondents, etc. They contain limited production facilities, but enough to produce a broadcast quality program from the range of cameras and microphones deployed at the event. If the event is to be covered live, OB vans can send the programme to the main studio in real time via dedicated networks (satellite or terrestrial networks).

The production network audio and video traffic is the industry standards. For video the dominant standard is the Serial Digital Interface (SDI). It produces 270 Mbps of video with ancillary data for audio, metadata and time-codes. The main standard for production audio (as opposed to talkback which uses low quality audio channels) is AES3 (from the Audio Engineering Society). It can be transmitted separately or embedded in SDI. AES3 contains two channels, which can accommodate a stereo pair. The transmission rate is 1536 Kbps.

2.1. Studio connectivity

The production network consists of all the video and audio devices. They are divided into three separate networks: the video network, the audio network and the talkback network for the intercommunication traffic. Additionally, control and synchronisation signals must be distributed across the site. For each network there is a central studio router to which devices connect using dedicated point-to-point links. The studio router is exclusively designed to interconnect audio or video devices using audio or video industry standards. Configuring a studio for a production session requires a significant amount of work. Audio, video, synchronisation and control connectivity must be provided via separate cabling. Many devices use proprietary protocols and cabling which increases the complexity of interconnection.

Designing an IP network to replace the existing video, audio, talkback and control networks inside a studio is not a trivial task. Switched local area network technologies such as Fast and Gigabit Ethernet can provide the required capacity. However, the performance expectations placed on the network require stringent quality of service guarantees. This paper focuses only on the use of IP between studios rather than inside them, the various approaches may be used to provide adequate quality of service within the studio, including over-provisioning and the use of the IP Integrated Services framework.

2.2. Inter-studio connectivity requirements

Inter-studio connectivity is either studio to studio or OB Van to studio. Connectivity may be through dedicated terrestrial or satellite links, depending on availability and economic factors, with bandwidth purchased from third party network providers. Given the cost of such connectivity, only one multimedia stream is sent from the van to the main studio, possibly with talkback channels and control signals if required. The latency on these links is orders of magnitude higher than for intra-studio connections, but this is acceptable as far as there are few feedback loops that use the inter-studio link. In some cases such loops are unavoidable, such as a speaker in the main studio interviewing a foreign correspondent the other side of the world. However, no control loops are required as each studio (or OB Van) is autonomous. Bandwidth requirements may also be relaxed by using compression.

The increasing availability of general purpose IP networks make them attractive candidates for interconnecting studios and OB Vans. However, the issue of quality of service over these networks must be successfully addressed before they are seriously considered for this demanding application. The current best effort paradigm is not acceptable for real time video and audio transfer requiring broadcast studio quality.

In order to provide adequate quality guarantees over an IP Wide Area Network (WAN) that serves multimedia production traffic, we propose to use the DiffServ framework together with MPLS for TE.

Fig. 1 shows a high level overview of the proposed solution. We consider a studio and an OB van requiring to exchange an SDI video flow, a number of AES3 audio streams, talkback traffic and control signals. They are connected over an IP WAN, which supports MPLS-TE and DiffServ. As the sources and sinks send packets to the WAN, routing protocols will determine paths across the networks, usually based on a shortest path algorithm. Conventional protocols do not take into account the load on different parts of the network, which may lead to certain areas of the network being congested while others under-utilised. DiffServ selectively distributes resources among the various classes of traffic, which helps maintain quality for selected traffic aggregates but does not tackle the congestion problem. In order to provide quality guarantees while optimising the use of network resources, it may be best to take a longer but less congested path through the network [3]. This requires alternative methods of routing to complement the operation of DiffServ.

This is achieved through TE, by explicitly routing traffic through the network using a set of calculated paths that provide better control of the network load. Fig. 1 shows two

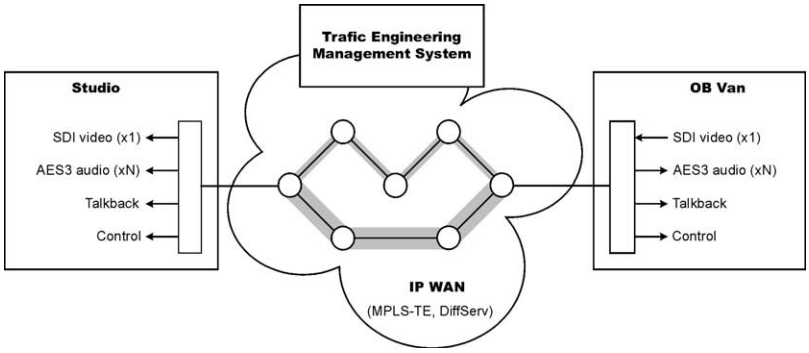

Fig. 1. Inter-studio connectivity over IP.

different paths between the OB Van and the studio, one requiring 3 hops and the other 4. In this example, the shortest path could be used to route the video stream, which requires minimal delay, whereas the less critical talkback traffic would be kept separate and made to use the longer path.

A traffic engineering management system is used to determine and enforce the traffic engineered paths. Determining these paths in order to deliver the required quality of service for the expected traffic matrix is far from trivial. The remainder of this paper focuses on the functionality, design and evaluation of the proposed TE and QoS Delivery Management System in the context of IP WANs that can serve high-volume high-quality traffic as required by IP-based inter-studio production networks.

3. Related work

The problem of TE has attracted a lot of attention in recent years. TE entails the aspect of network engineering that is concerned with the design, provisioning, and tuning of operational Internet networks. In order to deal with this important emerging area, the Internet Engineering Task Force (IETF) has chartered a Traffic Engineering Working Group (TE-WG) to define, develop, specify, and recommend principles, techniques and mechanisms for TE in the Internet. The main output of this working group until now is that it has defined the basic principles for TE [4] and the requirements to support the interoperation of MPLS and Diffserv for TE [5]. It is in the plans of this group to look into technical solutions for meeting the requirements for Diffserv-aware MPLS TE, the necessary protocol extensions, interoperability proposals and measurement requirements.

Two similar works with the work presented in this paper are Netscope [6] and RATES [7]. Both of them try to automate the configuration of the network in order to maximize network utilisation. The first one uses measurements to derive the traffic demands and then by employing the offline algorithm described in Ref. [8] it tries to offload overloaded links. The latter uses the semi-online algorithm described in Ref. [9] to find the critical links which if they are chosen for routing will cause the greatest interference (i.e. reduce the maximum flow) of the other egress–ingress pairs of the network. Both of these works do not take into account any QoS requirements and only try to minimize the maximum load of certain links.

The work described in this paper can be categorised as *time-dependent offline traffic engineering* [4]. Such problems can modelled as multi-commodity network flow optimisation problems [10]. The related works use optimisation formulations, focusing on the use of linear cost functions, usually the sum of bandwidth requirements, and in most of the cases they try to optimise a single criterion, minimize total network cost, or combine multiple criteria in a linear formula.

In Mitra et al. [11] the TE problem is seen as a multi-priority problem which is formulated as a multi-criterion optimisation problem on a predefined traffic matrix. This approach uses the notion of predefined admissible routes, which are specific for each QoS class, and each source–destination pair, where the objective is the maximization of the carried bandwidth. In Ref. [12], the authors address the resource allocation and routing problem in the design of Virtual Private networks (VPNs). The main objective is to design VPNs which will have allocated bandwidth on the links of the infrastructure network such that, when the traffic of a customer is optimally routed, a weighted aggregate measure over the service provider's infrastructure is maximized, subject to the constraint that each VPN carries a specified minimum. The weighted measure is the network revenue, which is a function of the traffic intensity. The algorithm proposed in that paper solves first the optimal routing problem for each VPN independently. Then calculates for each VPN the linear capacity costs for all the links. These quantities are used to modify appropriately the current capacity allocations so that the network revenue of the infrastructure network for the new capacities is maximized. In Ref. [12] it is shown that this is equivalent to minimizing a linear function of the capacity costs subject to constraints imposed by the link capacities.

In Ref. [13] a model is proposed for off-line centralized TE over MPLS. This uses the following objectives: resource-oriented or traffic-oriented TE [4]. The resource-oriented

problem targets load balancing and minimization of resource usage. Capacity usage is defined as the total amount of capacity used and load balancing is defined as one minus the maximal link utilization. The objective function that has to be maximized is a linear combination of capacity usage and load balancing, subject to constraints imposed by the capacity of the links. The traffic-oriented model suggests an objective function that is a linear combination of fairness and throughput, where throughput is defined as the total bandwidth guaranteed by the network and fairness as the minimum weighted capacity allocated to a traffic trunk. In Ref. [14] the authors propose an algorithm which has two phases, a pre-processing phase and an on-line one. In the pre-processing phase the algorithm uses the notion of multi-commodity flows, where commodities correspond to traffic classes. The goal is to find paths in the network to accommodate as much traffic as possible from the source to the destination node. The algorithm tries to minimize a linear cost function of the bandwidth assigned to each link for a traffic class. The second phase performs the on-line path selection for LSP requests by using the pre-computed output of the multi-commodity pre-processing phase.

Works like [8,15,16] try to achieve optimal routing behaviour by appropriately configuring the shortest path routing metrics, assuming no MPLS is supported. Wang et al. [15] proved theoretically that any routing configuration, including the optimal one, could be achieved by the appropriate setting of the shortest path routing metrics.

Finally, regarding online algorithms, they are mainly based on extensions of the QoS-routing paradigm [17]. These approaches are heuristics, recently known in the IETF as Constraint-Shortest Path First (CSPF), which utilise information kept in TE databases populated through information obtained from the routing flooding mechanisms about link capacities, unreserved capacity, colour affinities etc. Other online TE approaches [18,19] mainly focus on load balancing on multiple equal or non-equal cost paths. These works are complementary to work of this paper, since they can be used in conjunction.

4. A provisioning system for QoS delivery and traffic engineering

This section presents the functional model that will enable an Internet Service Provider (ISP) to configure and provision its autonomous system (AS) in order to satisfy the requirements of inter-studio production network traffic. We are focusing on defining a time-dependent IP Traffic Engineering and Control System that operates at medium (minutes to days) and long (weeks to months) timescales [4].

We assume that customers[1] will have bilateral Service Level Agreements (SLAs) that contain both an administrative aspect (terms, conditions, pricing information, etc.) and a technical part known as a Service Level Specification (SLS) [20]. Our system focuses to satisfy the SLSs of quality-critical customers.

In the rest of the section, we will present the overall proposed system architecture and we will elaborate on the operational timescales. Fig. 2 shows the model of the proposed system, which has evolved from our initial quality of service framework described in Ref. [21], aspects of which have also been developed further in Ref. [22].

The system in Fig. 2 is designed to provide an automated mechanism to determine, implement and monitor network configurations, which satisfy the following criteria:

1. Deliver the services as defined in SLSs (customer perspective),
2. Optimise the use of network resources (ISP perspective)

The above system is decomposed into three main sub-systems: SLS management, TE and Monitoring. While the main focus of this paper is on the TE part, we will also briefly describe the functionality of all the sub-systems for completeness.

SLS management is responsible for SLS negotiation and admission control. The negotiation and admission control process by which SLSs are accepted or refused is outside the scope of this paper. We assume that all existing SLSs, in the SLS repository, have been admitted by an external process and that no SLS will be added or deleted. Note though this is not strictly true since new SLSs are typically added and deleted in the system during its operation; these are only taken into account by the TE system in the next provisioning cycle. The basic production network SLSs we consider here are described in Section 4.3. Traffic forecast tries to predict the anticipated traffic demand based on the current subscribed SLSs, policy-based over-provisioning factors, and data both from SLS usage measurements and the history of past predictions.

We monitor network performance in order to assess our time-dependent periodic provisioning solution. Getting monitoring feedback is very important for both time and state-dependent [4] TE approaches, but only the latter uses monitoring feedback to react to network conditions.

NP performs the main task of TE by periodically (i.e. every provisioning period) running an offline algorithm which computes the network configuration in terms of explicitly routed paths and capacity requirements for the queues of each router; these are calculated according to the forecasted demands. Route and resource managers are responsible to apply the proper network configuration for every forecasting period. A route manager exists per ingress router and sends explicit MPLS Labelled Switched Path (LSP) set-up configuration requests. LSPs are set-up through a label distribution protocol, e.g. Constraint-based routing label distribution protocol (CR-LDP) or the extended

[1] In this article we assume that a customer is an owner of a production network as described in Section 2.

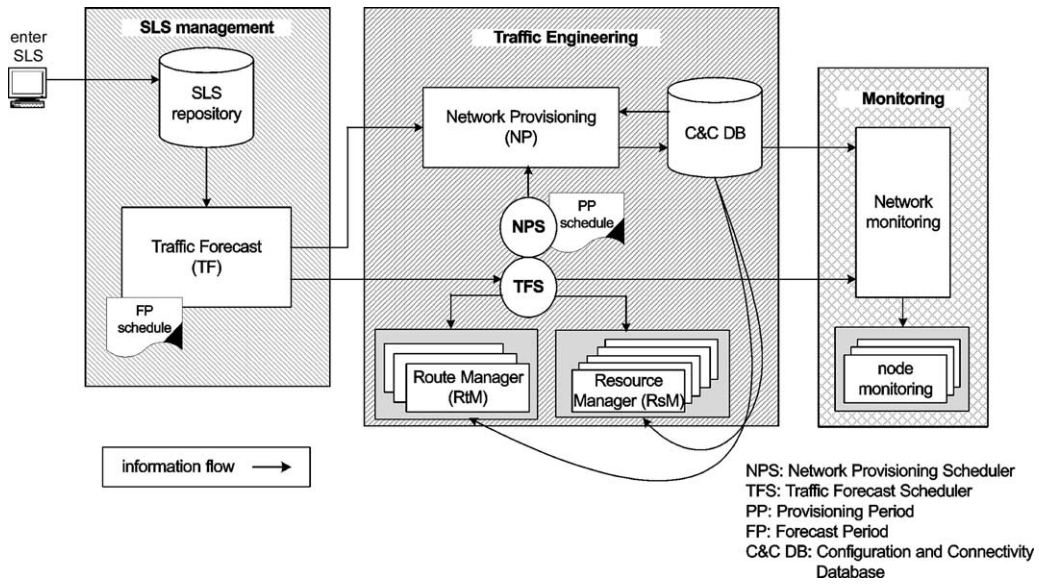

Fig. 2. Time-dependent IP TE and control system.

resource reservation protocol for TE (RSVP-TE). A resource manager exists in every router and is responsible for applying the configuration of the DiffServ Per-Hop Behaviours (PHBs) for all the router interfaces.

4.1. Provisioning and forecasting scheduling

In this section we will describe the timing and scheduling properties of our system. We assume that scheduling in the proposed model is performed at two levels:

- The NP Scheduler, which defines provisioning periods.
- The Traffic Forecast Scheduler, which defines forecasting periods.

4.1.1. Network provisioning scheduler

The IP Traffic Engineering and Control System aims to provision the network in order to provide the required QoS to contracted SLSs while at the same time optimising the usage of resources. As the traffic requirements that are derived from the SLSs change over time, the provisioning guidelines need to be updated. This is performed periodically through the NP Scheduler. The system is required to re-calculate a set of provisioning guidelines for each provisioning period. As the current period comes to an end, the Provisioning Scheduler will trigger re-provisioning of the network. Typical frequencies for the scheduler could be once a day or once a week.

The scheduler is part of the TE sub-system, and relies on the provisioning period schedule; the latter provides the frequency of the provisioning period.

4.1.2. Traffic forecast scheduler

As the provisioning periods are quite long (days, weeks), the traffic requirements may vary during a provisioning period. The granularity of the NP Scheduler is not fine enough to optimise the configuration of the network. We therefore use a second scheduler, the Traffic Forecast Scheduler, which has the following characteristics:

- It is not necessarily periodic (could schedule one 4 h period followed by one 10 h period)
- For all forecasting periods $FP(i)$ that constitute a provisioning period PP, it holds: $FP(i) \leq PP$

The principle behind the double scheduling is that the NP system is invoked once every provisioning period, and calculates multiple configurations, which will be enforced at each forecasting period. The various configurations are successively applied at the appropriate times (triggered by Traffic Forecast Scheduler) by the Resource and Route managers.

The Traffic Forecast Scheduler is part of our TE subsystem, which relies on the forecasting period schedule. The latter defines the forecasting periods.

4.1.3. Scheduling example

To clarify the roles of each scheduler, this section provides some examples.

Consider that provisioning period is a week and that Traffic Forecast Scheduler defines 4–6 h forecasting periods (forecasting period being periodic in this case). This means that once a week, four logical topologies are calculated: one for the morning, one for the afternoon, one for the evening and one for the night (6 h periods). During each day of the week, Traffic Forecast Scheduler triggers at 06:00 to enforce the morning configuration, at 12:00 to enforce the afternoon configuration, at 18:00 to enforce the evening configuration and at 24:00 to enforce the night configuration. This is shown in Fig. 3.

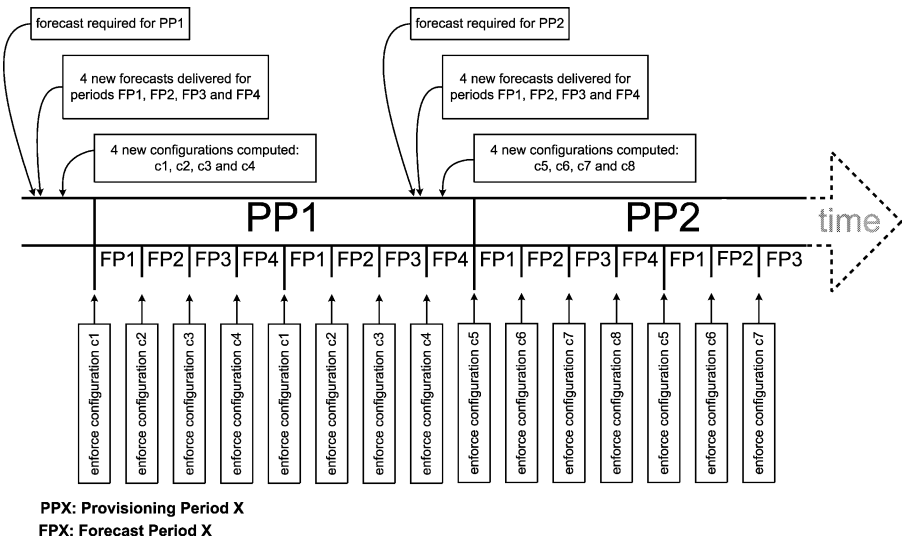

Fig. 3. Two level scheduling.

There is no requirement for Traffic Forecast Scheduler to be periodic. For example, we could have four forecasting periods per day, morning, afternoon, evening and night with different lengths as shown in Table 1.

Another more realistic and more complex example could be to have five forecasting periods: weekday morning, afternoon, evening, night, and all weekend. The interesting feature of this arrangement is that the Traffic Forecast Scheduler does not trigger the enforcement of the configurations in a simple cyclical fashion: the 4 weekday configurations must be implemented cyclically for 5 days, and the weekend configuration is enforced for the remaining two days of the week.

4.2. Operational cycle: a simple scenario

In order to illustrate the notions described above we will walk through the sequence of events, which make up a full operational cycle. Refer to Fig. 2 for the information flow described in this example. Fig. 4 shows a part of Fig. 3 in detail, covering one provisioning cycle and the first configuration enforcement.

4.2.1. Initialisation

There are two prerequisites for the operation of the model. First, a number of SLSs have been agreed between various customers (on behalf of users such as studios, OB vans etc.) and the network operator. We assume that the negotiation phase has taken place and the admitted/permitted SLSs do not exceed the overall network capacity. No SLS will be added after the initialisation phase. The SLSs are entered into the SLS repository. Second, the network topology (including the link capacities) exists in the Connectivity and Configuration (C&C) database. The topology will not be changed after the initialisation phase. The C&C database makes the topology available to Network Provisioning and Network Monitoring upon request. Then, the Traffic Forecast Scheduler requests the forecasting period schedule. It is important that the Traffic Forecast and Traffic Forecast Scheduler have identical views of the forecasting period schedule.

4.2.2. Operation

This scenario starts just before the beginning of a provisioning period, at marker 1 in Fig. 4. The NP Scheduler triggers NP to calculate a set of logical topologies for each forecasting period of the forthcoming provisioning period. In order to achieve this, NP requires the traffic forecasts for the given period, i.e. the forecasts for each forecasting period in the next provisioning period.

In order to compute these forecasts, Traffic Forecast requests from the SLS repository a list of SLSs, which are expected to be active during the next provisioning period. Based on these SLSs, forecasts are generated, one for each forecasting period, and are passed to NP (marker 2 in Fig. 4). Traffic Forecast requests the SLSs only after NP has requested the forecast.

NP calculates a logical traffic engineered topology for each received forecast (i.e. forecasting period), so as to provide the agreed QoS requirements while at the same time optimising resource utilisation. These configurations are stored in the C&C database (marker 3 in Fig. 4). NP Scheduler is inactive until the next re-provisioning is due, i.e. before the end of the next provisioning period.

The enforcement of the configurations is the responsibility of Traffic Forecast Scheduler. At the appropriate times (i.e. at each forecasting period boundary), Traffic Forecast

Table 1
Forecasting periods schedule example

Morning	08:00–12:00
Afternoon	12:00–17:00
Evening	17:00–22:00
Night	22:00–08:00

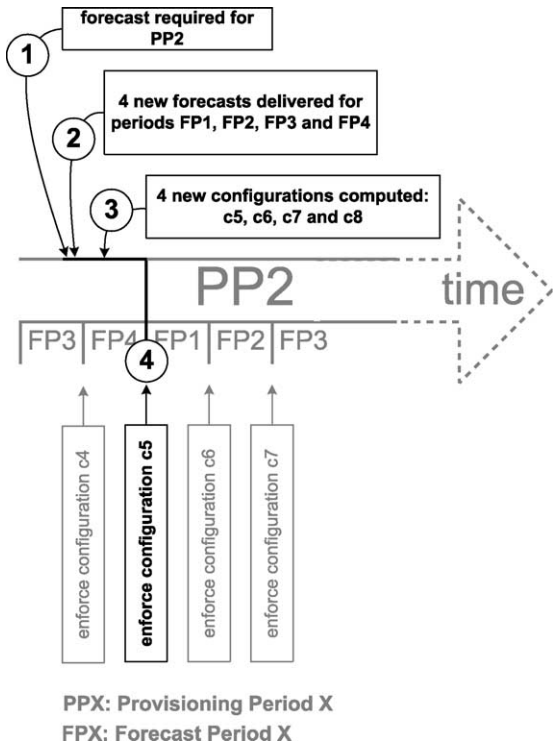

Fig. 4. Scheduling detail, re-provisioning and configuration enforcement.

Scheduler triggers the enforcement of the new configuration. Marker 4 in Fig. 4 shows the Traffic Forecast Scheduler triggering the enforcement of configuration 'c5'. Route Managers and Resource Managers enforce[2] the configuration. The Route Manager focuses on provisioning the traffic-engineered paths, while the Resource Manager configures the buffers and PHBs of the interfaces of each network node. Here we assume that the implementation of the PHBs on the various interfaces allows configuration, by defining an upper limit on the buffer, and by either setting the relative weight (Weighted Fair Queuing-WFQ) and/or the relative priority (Class-Based Queuing or Priority Queuing).

The Traffic Forecast Scheduler also causes monitoring to retrieve the new logical topology information in order to monitor the performance of the current network configuration.

4.3. From SLSs to classes of service

SLS management in the context of this paper is responsible for providing traffic forecasts to the NP. The forecasts are based on the following information:

- SLSs negotiated between the customers and network operator.
- Policy data set by the operator, including the forecasting period schedule.
- Measurements and historical data.

The SLS repository is a passive database. In this work we assume no negotiation or admission control procedure and the SLSs are entered through a user interface. The SLSs are indexed by start and end time. We are considering the following SLS types: video (SDI or compressed), audio (AES3 or compressed), which both have stringent delay and loss requirements, and production network control traffic which has less strict delay and certainly smaller throughput requirements. Note that for performance assessment purposes we also assume some background traffic SLSs, for different kinds of traffic than the studio production network types described above.

4.3.1. Traffic forecast

The role of the Traffic Forecast is to estimate the anticipated traffic for each forecasting period. The forecasting periods are determined based on the forecasting period schedule. For each forecasting period, four successive functions are performed: translation, mapping, aggregation, and forecasting.

Translation. The Customer SLSs stored in the repository use a customer-orientated terminology. These SLSs have the following semantics. They are the either bi-directional or unidirectional and follow either the pipe or the VPN model. The first function of translation is to decompose these SLSs into a number of simple unidirectional SLSs, which follow only the pipe model (one ingress to one egress). The ingress and egress are specified in geographical terms, therefore there is a need to translate those into IP source–destination addresses and from them to infer the specific ingress–egress addresses of our Autonomous System (AS). Services are given names (Olympic Gold/-Silver/Bronze Service, Virtual Leased Line). This terminology must use networking semantics (throughput, delay, loss). So, another important aspect of translation is from the customer SLS to the appropriate network parameters.

Mapping. The simple unidirectional SLSs could potentially request (or be translated to) any value of delay, loss or throughput, while the network supports only a few discrete values. The mapping function is therefore to map the SLS requirements to the services actually supported by the network, i.e. the Classes of Service (CoS). Another important function is to map the IP source–destination addresses. To illustrate the mapping function, consider the following simple example. A customer SLS may require '1 Mbps of Virtual Leased Line Premium service with 50 ms delay between London and Manchester,…'. The translation function would translate this Customer SLS into to two simple unidirectional SLSs: '1 Mbps, EF, 50 ms delay, from 122.12.2.143, egress 103.124.32.111,…' and

[2] Note that a very important operational consideration has to do with the transition from one configuration to another with the minimal disruption to the network and the current traffic. This issue, though very important, is outside the scope of this article, but a simple way out could be to apply the new configuration at the same time with the old. Then wait for all the already established flows of old configuration to drain before tearing it down, while at the same time all the new traffic demand is directed through the new configuration.

Table 2
Definitions of class of service and traffic trunk

Class of service (CoS)	PHB scheduling class (PSC)		Delay range	Loss probability range
Traffic Trunk	Ingress IP address	Egress IP address	CoS	Throughput requirement

another one with the same values but in the opposite direction. Suppose the network offers two services: a 10 and 100 ms delay premium virtual wire services, therefore we need to map the SLS to the 10 ms service. The mapping is static as far as the mapping algorithms are fixed and deterministic. The result is a list of SLSs that are active during the next provisioning period and are defined in network terms.

Aggregation. Up to this point of the forecasting procedures we had information proportional to the number of customers and SLSs. For scalability, we base our TE solution on traffic loads per ingress/egress pair and per class of service (see Section 2). We use the concept of traffic trunk [23], which is defined by the ingress/egress node pair, the class of service and the capacity requirement (Table 2). The simple unidirectional SLSs are aggregated into traffic trunks by 'adding' their throughput requirements if and only if they have the same ingress/egress pair and they require a similar treatment, i.e. they have the same CoS values. We define the distinct CoS to be uniquely identified by the required PHB Scheduling Class [24], a delay range and loss probability range (Table 2).[3] SLSs may be multiplexed. If two clients require the same service but at different times within a forecasting period, they are multiplexed in the traffic forecast. Note that multiplexing may occur within a class of service, but not between classes.

Forecasting has two main functions. On one hand, an over-subscription factor per CoS is included. This factor is defined as the ratio of the capacity reserved by all the SLSs in a given CoS to the capacity expected to be actually used. For expensive SLS types, the over-subscription factor is likely to be one. For cheaper services, the factor may be larger. On the other hand, at this stage we have all the ingress–egress demand (traffic matrix). It is possible to run an extrapolation algorithm utilising the information on the history of traffic matrices. Candidate algorithms are the spline or more generally any polynomial extrapolation method.

5. A traffic engineering solution

Network Provisioning is the main module of our TE system (Fig. 2). It provides guidelines for *provisioning* the network according to traffic estimates and resource utilisation policies (see Section 4.3.1).

[3] Normally a SLS and consequently a CoS will have a delay variation (jitter) parameter, but since our current TE solution does not take it into account (Section 5) we are omitting it from the table.

The objectives are both traffic and resource-oriented. The former relate to the obligation towards customers, through the SLSs. These obligations induce a number of restrictions about the treatment of traffic. The resource-oriented objectives are related to the network operation, more specifically they are results of the high-level business policy that dictates the network should be used in an optimal fashion. Congestion is the main cause of network performance degradation that influences both these TE objectives. Two are the main reasons behind congestion: either the demand exceeds the network capacity, or the traffic is not spread optimally, causing parts of the network to be over-utilised while others are under-utilised. The first reason can only be handled by adequate network planning, i.e. adding more physical resources. The latter can be handled by adequate bandwidth and routing management, and this is what NP is aiming for.

NP has as input the network topology (including link capacities), the estimated traffic matrix (expected traffic trunks, see Section 4.3.1), and resource-oriented policy directives[4]. It will provide as an output a list of explicitly routed paths for each source destination included in the traffic matrix, and the capacity requirements for each PHB scheduling class, and therefore physical queue [24] for every interface of all the network nodes.

5.1. Network and traffic model

The network is modelled as a directed graph $G = (V, E)$, where V is a set of nodes and E a set of links. Each link $l \in E$ is specified by the pair $l = (v_{l,i}, v_{l,e})$ where $v_{l,i}$, $v_{l,e}$ are the nodes where traffic is entering and exiting the link, respectively.

With each link $l \in E$ we associate the following parameters: the link physical capacity C_l, the link propagation delay d_l^{prop}, the set of the physical queues H, i.e. PHB Scheduling Classes (PSCs), supported by the link. For each PSC, $h \in H$ we associate a bound d_l^h (deterministic or probabilistic depending on the PSC) on the maximum delay incurred by traffic entering link l and belonging to the $h \in H$, and a loss probability p_l^h of the traffic entering link l and belonging to $h \in H$.

The basic traffic model of NP is the traffic trunk. A traffic trunk is an aggregation of a set of traffic flows characterised by similar end-to-end performance requirements [3]. Also, each traffic trunk is associated with one ingress node and one egress node, and is unidirectional. Each trunk

[4] In this work we do not consider the way policies influence our system. For a comprehensive discussion of policies in this context see Ref. [25].

corresponds to a CoS as defined in the context of this paper (see Section 4.3.1). The set of all traffic trunks is denoted by T. Each trunk $t \in T$, from ingress node v_i to egress node v_e ($v_i, v_e \in V$), is associated with a bandwidth requirement, B_t.

The following information about each traffic trunk is available because of the CoS definition (see Table 2):

- The PHB Scheduling Class (PSC) of the traffic carried on the trunk, thus inferring the PHB $h \in H$ of the trunk. For certain PSCs (e.g. AF1x) the mapping to PHB is not '1–1', providing more flexibility. This means that each traffic trunk $t \in T$ is associated (eventually) with one PHB.
- The bound D_t (deterministic or probabilistic depending on the PSC) maximum end-to-end delay expected between the ingress and egress nodes. This might be the minimum value of the delay class range of the CoS definition.
- The maximum end-to-end loss probability, P_t required between the ingress and egress nodes. Similarly to the previous case this might be the minimum value of the loss range of the CoS definition.
- The bandwidth B_t requirements.

5.2. Cost definition and optimisation objectives

We need to provide a set of routes for each traffic trunk. For each ingress–egress pair v_i, v_e these routes are implemented as LSPs at the routers. We also need to provide the amount of bandwidth each route is expected to carry, and the amount of bandwidth that is to be allocated to the PSCs at the various interfaces in the network.

The *primary objective* of such an allocation is to ensure that the requirements of each traffic trunk are met as long as the traffic carried by each trunk is at its specified minimum bandwidth. This objective ensures that our SLS requirements are met. The objective is to provide a feasible solution (i.e. routes and route bandwidths respecting the link capacities) that satisfies the SLSs delay and loss constraints.

However, with the possible exception of heavily loaded conditions, there will generally be multiple feasible solutions. Hence, the design objectives can be further refined to incorporate other requirements such as:

1. Avoid overloading parts of the network while other parts are underloaded. This way, spare bandwidth is available at various parts of the network to accommodate unpredictable traffic requests. In addition, in case of link failures, smaller amounts of traffic will be disrupted and will need to be rerouted.
2. Provide overall low network load (cost).

The last two requirements do not lead to the same optimisation objective. In any case, in order to make the last two requirements more concrete, the notion of 'load' has to be quantified. Various definitions are possible. In general, the load (or cost) on a given link is an increasing function of the amount of traffic the link carries. This function may refer to link utilization or may express an average delay, or loss probability on the link. In the context of this work, the QoS seen by the traffic using the different PHBs varies, so the link load induced by the traffic of each PHB may vary. This leads us to the following general form of the cost of link l.

Let x_l^h denote the capacity demand (flow) for PHB $h \in H$ satisfied by link l. Then the link cost induced by the load on PSC $h \in H$ is a convex function, $f_l^h(x_l^h)$, increasing in x_l^h. The total link cost per link is defined as:

$$F_l(\bar{x}_l) = \sum_{h \in H} f_l^h(x_l^h) \tag{1}$$

where $\bar{x}_I = \{x_l^h\}_{h \in H}$ is the vector of demands for all PHBs of link l. In order to take into account that link capacities may be different, the cost may be modified to reflect equivalent utilization as follows:

$$F_l(\bar{x}_l) = \frac{\sum_{h \in H} f_l^h(x_l^h)}{C_l} \tag{2}$$

Provided that appropriate buffers have been provided at each router and the scheduling policy has been defined, then $f_l^h(x_l^h)$ may specify the *equivalent capacity* needed by PHB $h \in H$ on link l in order to satisfy the loss probability associated with that PHB. Hence, the total cost per link is the total equivalent capacity allocated on link l. Note that with this approach the link costs are very naturally defined. The drawbacks are: (1) the cost definition depends on the PHB implementation at the routers, (2) the cost functions may not be known, or may be too complex. Hence approximate cost functions must be used.

In this work we are using the following cost function:

$$f_l^h(x_l^h) = \frac{x_l^h}{C_l - x_l^h} \tag{3}$$

This function combined with Eq. (1), gives as a total cost function the average number of packets in the system based on the hypothesis that each queue behaves as a M/M/1 queue [26].

Now we can express objectives (a) and (b) in mathematical terms.

Avoid overloading parts of the network:

$$\text{minimise } \max_{l \in E} F_l(\bar{x}_l) = \max_{l \in E} \left(\sum_{h \in H} f_l^h(x_l^h) \Big/ C_l \right) \tag{4}$$

Minimize overall network utilization:

$$\text{minimise } \sum_{l \in E} F_l(\bar{x}_l) = \sum_{l \in E} \left(\sum_{h \in H} f_l^h(x_l^h) \Big/ C_l \right) \tag{5}$$

It is possible to provide an objective that compromises between the previous two. More specifically,

$$\text{minimise} \sum_{l\in E} (F_l(\bar{x}_l))^n = \sum_{l\in E}\left(\sum_{h\in H} f_l^h(x_l^h)/C_l\right)^n, \quad n \geq 1 \qquad (6)$$

When $n = 1$, this objective Eq. (6) reduces to Eq. (5), while when $n \to \infty$ it can be shown that it reduces to Eq. (4). The exponent n is important for the results presented in Section 6. By using Eq. (6), even if we are using the linear cost function for $f_l^h(x_l^h)$, the problem becomes a non-linear optimisation problem.

5.3. Handling the delay and loss constraints

Each traffic trunk is associated with an end-to-end delay D_t and loss probability P_t, constraint of the traffic belonging to the trunk. Hence, the trunk routes must be designed so that these two QoS constraints are satisfied.

Given the traffic trunks' route $r = \{l_1, l_2, \ldots, l_k\}$, the end-to-end delay is bounded by the sum of the propagation delays and the per node delay over the route. The same holds for the end-to-end loss probability. Let d_l^h be the bound on the maximum delay on the link l associated with PHB h, d_l^{prop} is the propagation delay, and p_l^h is the maximum loss probability of that link.

Both the constraints above are constraints on additive path costs under specific link costs (d_l^h and p_l^h, respectively). However, the problem of finding routes satisfying these constraints is in general NP-complete [27]. Given that this is only part of the problem we need to address, the problem in its generality is rather complex. Fortunately, a reasonable simplification can be made.

Usually, loss probabilities and delay for the same PHB on different nodes (routers) are of similar order. Therefore we use the maximum delay, $d_{\max}^h = \max_{l\in E}\{d_l^h\}$ and maximum loss probability, $p_{\max}^h = \max_{l\in E}\{p_l^h\}$ of a particular PHB over the whole network, in order to translate the delay and loss constraints to an upper bound on the number of hops along a route. For each traffic trunk $t \in T$, let K_{D_t} denote the upper bound on the number of hops induced from the delay constraint, and K_{P_t} the upper bound because of the loss probability constraint. It should be noted that since loss probabilities per PHB are generally small relative to the end-to-end loss probability, the resulting bound in this case will be large and hence it will not have a big effect on the optimisation. In any case, the resulting constraint is now simpler.

5.4. Optimisation problem formulation

For each traffic trunk $t \in T$ we denote as R_t the set of (explicit) routes defined to serve this trunk. For each $r_t \in R_t$ we denote as b_{r_t} the capacity we have assigned to this explicit route. The optimisation problem can be defined now as follows:

$$\text{minimise} \sum_{l\in E}\left(\sum_{h\in H} f_l^h(x_l^h)/C_l\right)^n \qquad (7)$$

subject to:

$$\sum_{l\in r_t} 1 \leq \min\{K_{D_t}, K_{P_t}\}, \quad \forall r_t \in R_t \qquad (8)$$

$$\sum_{r_t\in R_t} b_{r_t} = B_t, \quad \forall t \in T \qquad (9)$$

$$x_l^h = \sum_{t\in T: h_t = h} \sum_{r_t: l\in r_t, r_t\in R_t} b_{r_t}, \qquad \forall l \in E, \ \forall h \in H \qquad (10)$$

$$b_{r_t} \geq 0, \quad \forall r_t \in R_t, \ \forall t \in T \qquad (11)$$

Eq. (7) provides the optimisation objective, where n is a variable that provides the relative merit of low overall cost and avoid overloading parts of the network. In Eq. (8) we add the hop-count constraint, which (see Section 5.3) is deduced from the delay and loss QoS requirements of the SLSs. Eq. (9) states that the bandwidth of the explicit routes per traffic trunk should be equal to the trunks' capacity requirements, and Eq. (11) ties the optimisation objective with the optimisation parameters.

In the formulation above (Eqs. (7)–(11)), we have not included any constraint for the physical link capacity. We handle this by defining the cost function Eq. (3) to increase very fast when the total link flow becomes greater than the physical link capacity, for example exponentially or polynomially to utilisation x_l^h/C_l, which in this case will be greater than one. This will avoid getting over the link capacity. Special care is required since the cost function and its first derivative need to be continuous at the point of change.

This is a network flow problem and considering the non-linear formulation described above, the optimal solution can based on the general gradient projection method [28]. This is an iterative method, where we start from an initial feasible solution, and at each step we find the minimum first derivative of the cost function path and we shift part of the flow from the other paths to the new path, so that we improve our objective function (Eq. (7)).

If the path flow becomes negative, the path flow simply becomes 0 (max $\{0,x\}$) in order to handle the non-negativity constraint Eq. (11). This method is based on the classic unconstraint non-linear optimisation theory, and the general point is that we try to decrease the cost function through incremental changes in the path flows.

The optimisation variables are the capacity variables b_{r_t} assigned to each route of each trunk, i.e. $\mathbf{b} = \{b_{r_t} : r_t \in R_t, t \in T\}$. In order to apply the gradient projection method we need to handle all the constraints. Substituting Eq. (10)

in the optimisation function Eq. (7), we get:

$$\bar{F}(\mathbf{b}) = \sum_{l \in E} \left(\sum_{h \in H} f_l^h \left(\sum_{t \in T: h_t = h,} \sum_{r_t: l \in r_t, r_t \in R_t} b_{r_t} \right) / C_l \right)^n \tag{12}$$

The constraint Eq. (9) states that the capacity assigned to each variable b_{r_t} should be equal to the trunks' capacity requirements, i.e. $\sum_{r_t \in R_t} b_{r_t} = B_t$, $\forall t \in T$. We can handle this by substituting at each iteration i and for each of the trunks $\forall t \in T$, one of the variables b_{r_t} $r_t \in R_t$, say $b_{\bar{r}_t}$, is substituted by

$$b_{\bar{r}_t} = B_t - \sum_{r_t \in R_t - \{\bar{r}_t\}} b_{r_t} \tag{13}$$

The hop-count constraint Eq. (8) can be handled as follows. At each step of the algorithm we are required to find a minimum weight path for each trunk $t \in T$ with the weights being the first derivative of the cost function

$$\mathrm{d}b_{r_t} = \frac{\partial \bar{F}(\mathbf{b})}{\partial b_{r_t}} = \sum_{l \in r_t} \left[n (F_l(\bar{x}_l))^{n-1} (F_l(\bar{x}_l))' \right]$$

$$= \sum_{l \in r_t} \left[\frac{n}{C_l^n} \left(\sum_{h \in H} f_l^h(x_l^h) \right)^{n-1} \frac{\mathrm{d}}{\mathrm{d}x_l^h} (f_l^h(x_l^h)) \right] \tag{14}$$

this $\mathrm{d}b_{r_t}$ is the minimum first derivative length path, which should be included in the optimum solution. In order to consider the hop-count constraint, the minimum weight path computation algorithm has to check whether the path found satisfies the hop-count constraint as well. If not, then we need to find another path (not necessarily with the minimum weight but with a total weight less than at least one of the paths in R_t), which has hop-count, at least the hop-count constraint. This procedure is done by either using a *k-shortest path* algorithm [29], or by re-trying a simple shortest path, e.g. Dijkstra's [30], on a reduced graph.

We need to have a way to control the procedure described above. We will accept a solution if the relative improvement in a step $i+1$ from step i, is less than a parameter ϵ. More specifically the iterative procedure terminates when the following equation becomes true,

$$\left| \frac{\bar{F}(\mathbf{b}^{i+1}) - \bar{F}(\mathbf{b}^i)}{\bar{F}(\mathbf{b}^{i+1})} \right| < \epsilon \tag{15}$$

6. Experimental evaluation

In this section we will investigate the performance of the system described in Section 5 in terms of long to medium term average link load and total network average delay.

The NP algorithm described in Section 5 was implemented in ANSI C and a random SLS/Forecast generator in Java (see later). For the experimental results presented in the section we used a 2×450 Mhz ULTRA SPARC II with 1024 MB memory running Solaris 7.

6.1. Topologies

When evaluating a system like the one proposed in this paper, care is required regarding the simulation conditions, including the topologies used, the traffic demand generation procedure and the intensity of the expected load.

The main topology used is shown on Fig. 5; it has 10 nodes and 34 unidirectional links. This is a rather small topology that resembles the early NSFNet backbone. More complex and bigger topologies (up to 300 nodes) where also used, according to the widely used models for random topology generation presented in Ref. [31]. We used both random and transit-stub networks, which represent the AS system hierarchical structure better than pure random. The topologies used were of 50, 100, 200 and 300 nodes with average node degree between 3 and 6 (bi-directional), which are close to the typical values for real networks. In general, for the results we opted for 90% confidence level with a confidence interval 8–10% of the corresponding values.

6.2. Initial conditions

The initial feasible solution (step 0) of the iterative procedure described in Section 5.4 is set to be the same as if the traffic trunks were to be routed with a shortest path first (SPF) algorithm. That corresponds to the case that all the traffic of a particular class from ingress to an egress will be routed through the same shortest path according to some weight (routing metric). If there exist more than one such shortest paths, the load is distributed equally among them. The metric we are using for the SPF algorithm is set to be inversely proportional to the physical link capacity. The scenario we are using for step 0 described above is the norm in today's operational networks.

The experiments shown in this section correspond to only one forecasting period, i.e. one run of the provisioning algorithm. It is straightforward to get results for the provisioning period by concatenating more than one forecasting period results. The termination parameter ϵ, see (Eq. (15)), is set equal to 0.0008.

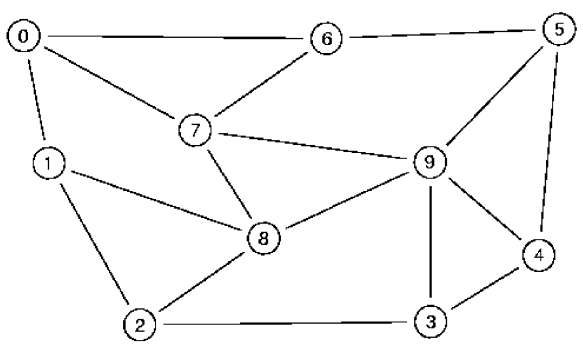

Fig. 5. Fixed topology used for experimentation.

Table 3
Loading profiles for each SLS type

	SLS type		
	Video (%)	Audio (%)	Control traffic (%)
Medium load 40%	30	9	1
High load 70%	50	18	2

6.3. Traffic load generation

In order to be able to experiment with a wide variety of traffic loads we have implemented a random SLS and traffic trunk generator in Java. This takes as input a topology file with a specified the number of edge nodes. In our experiments we selected the edge nodes to be in the range of 30–50% of the total network nodes. We have defined a number of SLS types offered by the ISP to the costumers. These are production network-oriented SLS types (see Table 3) defining video (SDI or compressed), audio, and control traffic.

We define as *total throughput of a network* the sum of the capacities of the first-hop links of all the edge nodes. This is actually an upper bound of the throughput and in reality it is a much greater than the real total throughput a network can handle. This happens because, although the sum of the first-hop link capacity imposes this limit, the rest of the backbone might not be able to handle so much traffic, which in our case is particularly true due to the random nature of the topologies used. Therefore in our experiments we used 70% load of the total throughput of the network, as the highly loaded condition, and a 40% load as a medium loaded one. The set of SLS instances that adhere to the high or medium load profile (70 or 40%) are assigned randomly at the edge node pairs and are placed into a file, which corresponds to the SLS repository of Fig. 2. Because of the random nature of the assignment to edge nodes, we can provide any number of such files for a particular load profile and a particular network topology. Each SLS file is then formatted into a traffic trunk file (corresponding to the expected traffic matrix), according to the procedure described in Section 4.3.1. SLSs are aggregated per ingress egress pair into CoS classes (see Table 2). The set of traffic trunks is generated with this procedure, resulting in one matrix for each SLS file for each loading profile.

6.4. Simulation results

Fig. 6 shows the link loads for the 10-node topology shown in Fig. 5 for the first step and after the algorithm has run. It is clear that at step 0 solution, which corresponds to the SPF with equal cost multi-path distribution enabled, parts of the network are over utilised while others have no traffic at all. The final step, which corresponds to the final output of our provisioning algorithm, balances the traffic over the whole network. Note that the largest over utilised link at step 0 is not necessarily the largest at the final solution.

Fig. 7 shows the mean and standard deviation of the link loads for the 10-node network after each iteration of the algorithm. As we can see the load becomes more balanced over the network after each iteration (standard deviation reduces). We run those experiments with the exponent $n = 2$, see Eq. (6). This value compromises between minimising the total (sum) of link costs and minimising the maximum link load. These two objectives generally lead to different solutions, with the first favouring path solutions with the least number of links while the other does not care about the number of links but only for the maximum link load (and therefore the deviation from the mean). We can observe the effect of these different objectives at the various ups and downs over the various steps of the algorithm.

The same behaviour observed with the 10-node network is also observed with larger random and transit-stub topologies. We experimented with large topologies up to 300 nodes. The maximum, mean, standard deviation and link utilisation resulted after the first step (SPF) and after algorithm finished are shown in Fig. 8.

Now we are going to look at the effect of exponent n of the cost function as we defined it in Eq. (6). This parameter compromises between the two objectives, Eqs. (4) and (5), for minimising the maximum link load and minimising the overall cost. Fig. 9 shows the results of the experiment with varying the exponent of the cost function for the 10 and 50 node networks. We can see that the maximum link load reduces as n increases, while the mean link load slightly

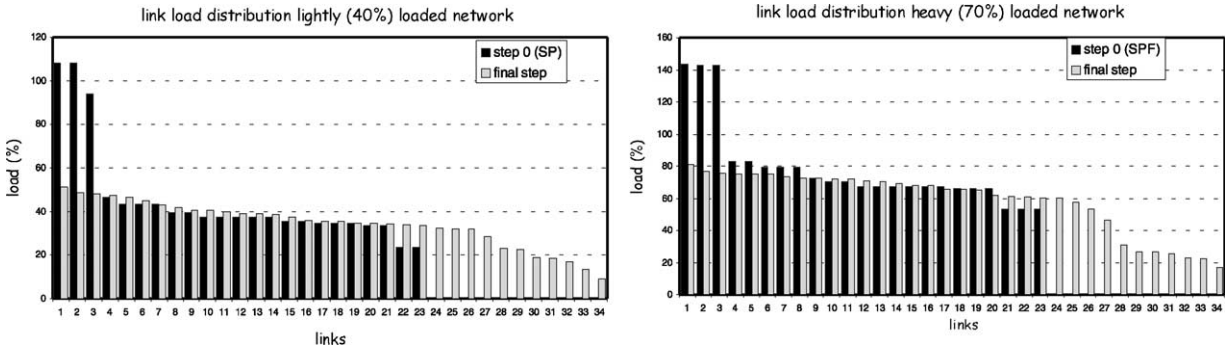

Fig. 6. Link load distribution of after the first and the last step.

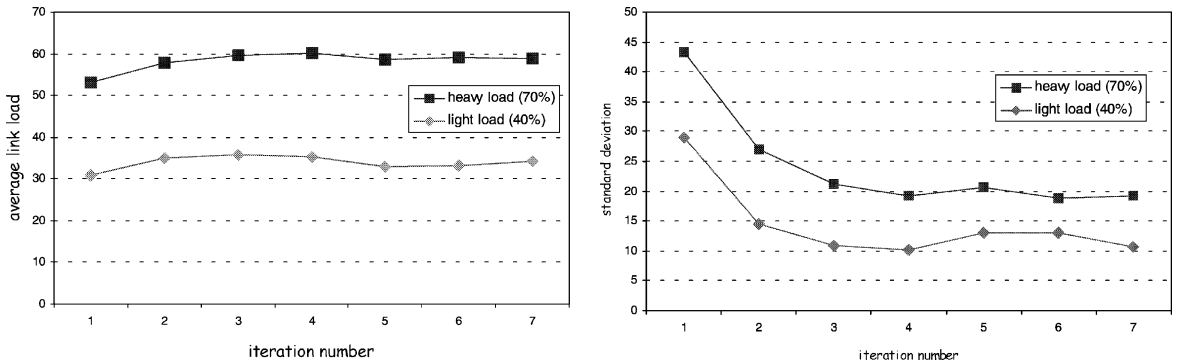

Fig. 7. Average and standard deviation of link load per iteration.

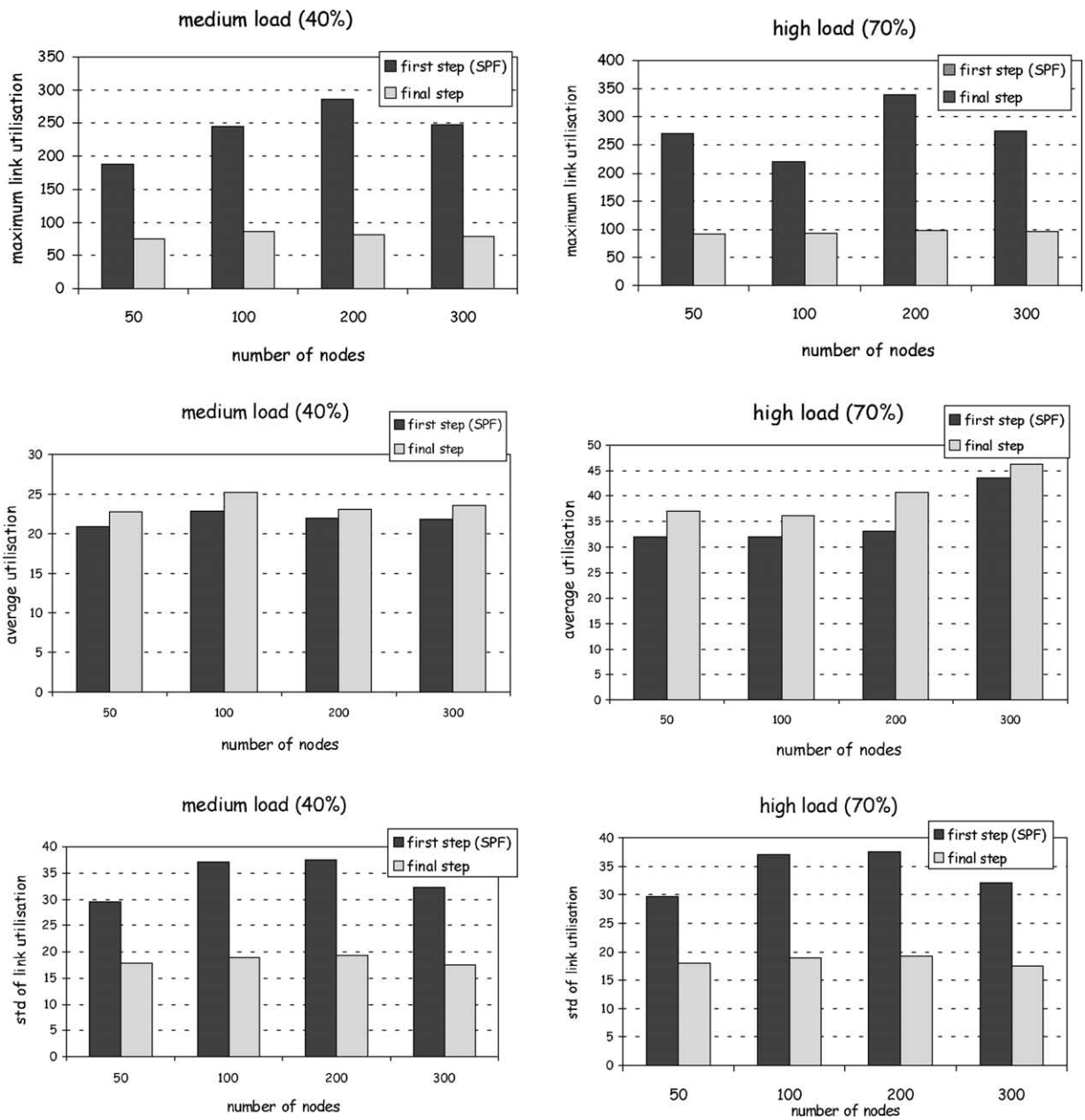

Fig. 8. Maximum, average and standard deviation of link utilisation for various network sizes.

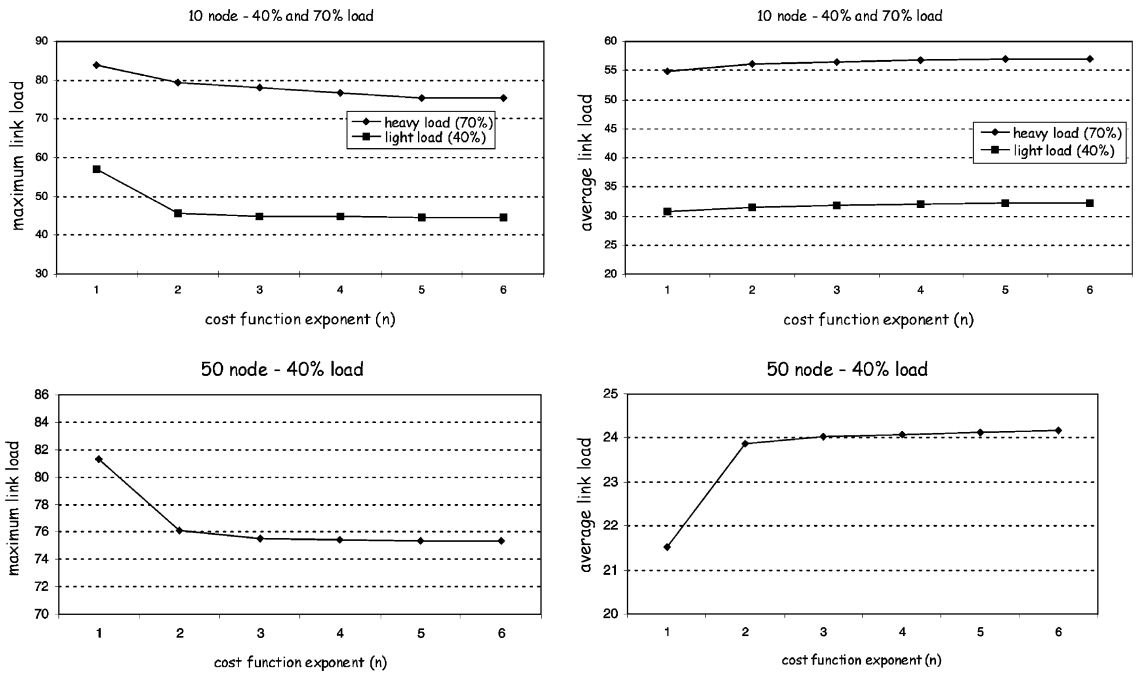

Fig. 9. The effect of the exponent for 10 and 50 node network.

increases since minimising the maximum link utilisation results in solutions with paths having more links. The solution with $n = 1$ means that we only optimise for the total cost and we do not take into account the maximum link load objective. We can observe that the reduction of the maximum link load, and the corresponding increase of the average link load, is important at the first increase of n from 1 to 2, but further increments give only a small difference. This is a consequence of the fact that we have the hop-count constraint (see Section 5.3) that limits the number of links per path and therefore very long paths, which help reducing the maximum link load, are prohibited. Finally, we can see that the same behaviour persists for the 10 and the 50 node networks, as well as all for the larger topologies we used for experimentation.

We are now going to see the average delay over the network. We calculated the average delay as a function of the link utilisation according to the following formula [26]:

$$\frac{1}{\gamma}\sum_{l\in E}\frac{x_l^h}{C_l - x_l^h}$$

where $\gamma = F_{all}/s_{avg}$ is the average of the total rate of the incoming packets in the network, F_{all} is the total in expected rate and s_{avg} is the mean packet size (set to 1 K for our experiments). The above formula is based on the assumption that each queue (PSC) at each link can be modelled as M/M/1 queue. Although, this is not always true, we can only use the qualitative nature of the result and not the exact values. Note that the only way to find the exact impact on delay and loss is by monitoring the network after applying the provisioning configuration. As part of continuing the work presented in this paper we are planning to have such results from the simulator and from the PC-based router testbeds we maintain in our laboratories.

Fig. 10 shows the average total delay in seconds of the solutions provided for the various network sizes according to the formula given above. The important observation is that the algorithm manages to keep the average total delay for all network sizes at about the same levels. This is particularly important since at the first step we have many links with utilisation more than 100%, which yields very high (almost infinite) total average delay. Another important observation is that the provisioning algorithm results in average delay for medium and high loaded networks being very close together, almost proportionally to their relative load difference. This is an indication that even for high loads

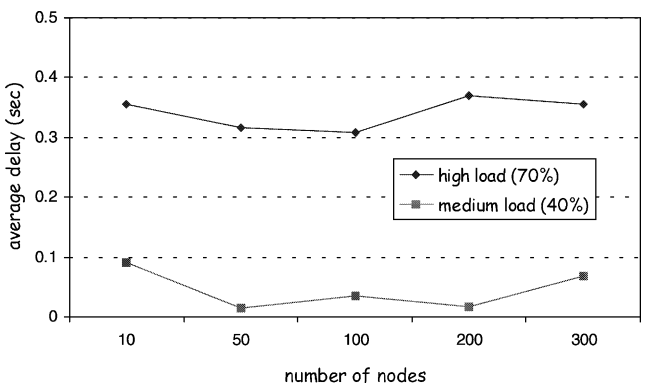

Fig. 10. Average overall network delay.

Table 4
Average running times in seconds for the various network sizes

	Medium load	High load
10	0.055	0.061
50	9.761	10.164
100	123.989	302.079
200	529.532	1002.245
300	981.175	1541.937

the provisioning algorithm manages to restrain the level of increase of average delay.

Finally, in Table 4 we provide the average running times of the various experiments conducted in this paper. We can see that even for quite large networks the running times are acceptable. For example for the 300 nodes networks, for medium load the running time is about 17 min, and for high load about 25 min. These times are perfectly acceptable taking into account the timescale of the provisioning system operation (see Section 4.1).

7. Conclusions

Supporting demanding applications such as production network traffic requires dedicated networks with high switching capacity. In this paper we investigate the possibility of using common IP-based packet network infrastructure, with Differentiated Services and MPLS as the key QoS technologies, in order to support such traffic with the appropriate Service Level Agreements and network provisioning.

We propose a management and control system for the provisioning of such networks, targeting to support demanding production network oriented Service Level Agreements while at the same time optimising the use of network resources. Our model is based on the notion of a provisioning cycle that comprises a number of forecasting periods; for each of these we need to apply a different network configuration. Traffic forecasting is based on the established SLSs, SLS usage and historical measurement data. We defined a number of CoS to which we assign the various SLSs. With this model, adequate provisioning for the CoSs ensures adequate provisioning for SLSs as well. Our objectives are then to place the CoS demands to the network in such a way as to avoid overloading parts of the network while at the same time minimising the overall network cost. We devised a non-linear programming formulation and we proved though extensive simulation that we achieve our objectives.

In conclusion, we believe it is possible to support production network traffic through IP networks with DiffServ and MPLS, as long as the appropriate SLSs are defined and agreed. They can then be used to calculate anticipated traffic while the provisioning TE algorithm takes into account the expected traffic demand and QoS constraints. Using IP networks to transport demanding traffic will result in lower cost and greater flexibility.

Acknowledgements

The work presented in this paper was partially supported by the EPSRC/LINK Project GR/M84169 ‘Production of Broadcast Content in and object-oriented IP-based Network (PRO-NET)’, and the EU IST Project IST-1999-11253 ‘Traffic Engineering for Quality of Service in the Internet at Large (TEQUILA)’. The authors would like to thank their project partners for the fruitful discussions while forming the ideas presented in this paper and the anonymous reviewers for their useful suggestions to improve the quality of presentation of this work.

References

[1] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, W. Weiss, An architecture for differentiated services, IETF Informational RFC-2475 December (1998).
[2] E. Rosen, A. Viswanathan, R. Callon, Multiprotocol label switching architecture, IETF Standards Track RFC-3031 January (2001).
[3] D. Awduche, J. Malcolm, J. Agogbua, M. O’Dell, J. McManus, Requirements for traffic engineering over MPLS, IETF Informational RFC-2702 September (1999).
[4] D. Awduche, A. Chiu, A. Elwalid, I. Widjaja, X. Xiao, Overview and principles of Internet traffic engineering, IETF Informational RFC-3272 May (2002).
[5] F. Le Faucheur, W. Lai, Requirements for support of Diff-Serv-aware MPLS traffic engineering, IETF Internet draft June (2002) ⟨ietf-tewg-diff-te-reqts-05.txt⟩, in press.
[6] A. Feldmann, J. Rexford, IP network configuration for intradomain traffic engineering, IEEE Network Magazine 15 (5) (2001) 46–57.
[7] P. Aukia, M. Kodialam, P.V. Koppol, T.V. Lakshman, nH. Sarin, B. Suter, RATES: a server for MPLS traffic engineering, IEEE Network Magazine 14 (2) (2000) 34–41.
[8] B. Fortz, M. Thorup, Internet traffic engineering by optimising {OSPF} weights, Proceedings of IEEE INFOCOM 2000,Tel Aviv, Israel March (2000) 519–528.
[9] M. Kodialam, T.V. Lakshman, Minimum interference routing with applications to traffic engineering, Proceedings of IEEE INFOCOM2000, Israel March (2000) 884–893.
[10] R.K. Ahuja, T.L. Magnanti, J.B. Orlin, Network Flows: Theory, Algorithms and Applications, Prentice Hall, Englewood Cliffs, NJ, 1993.
[11] D. Mitra, K.G. Ramakrishnan, A case study of multiservice, multi-priority traffic engineering design for data networks, Proceedings of IEEE GLOBECOM 99, Rio de Janeiro December (1999) 1087–1093.
[12] D. Mitra, J.A. Morrison, K.G. Ramakrishnan, Virtual private networks: joint resource allocation and routing design, Proceedings of IEEE INFOCOM 99, New York, USA March (1999).
[13] F. Poppe, S.V. den Bosch, P.L. Valle-Poussin, H.V. Hove, H. De Neve and G.H. Petit, Choosing the objectives for traffic engineering in IP backbone networks based on quality-of-service requirements, Proceedings of the First International workshop on Quality of Future Internet Services (QofIS’00), Berlin, Germany September (2000) 129–140.
[14] S. Suri, M. Waldvogel, P.R. Warkhede, Profile-based routing: a new framework for MPLS traffic engineering, Proceedings of the Second

International Workshop on Quality of future Internet Services (QofIS'01), Portugal September (2001) 138–157.
[15] Z. Wang, Y. Wang, L. Zhang, Internet traffic engineering without full mesh overlaying, Proceedings of IEEE INFOCOM, Alaska April (2001).
[16] Y. Breitbart, M. Garofalakis, A. Kumar, R. Rastogi, Optimal configuration of OSPF aggregates, Proceedings of IEEE INFOCOM, New York, USA June (2002).
[17] S. Chen, K. Nahrstedt, An overview of quality-of-service routing for the next generation high-speed networks: problems and solutions, IEEE Network Magazine, Special Issue on Transmission and Distribution of Digital Video 12 (6) (1998) 64–79.
[18] A. Elwalid, C. Jin, S.H. Low, I. Widjaja, MATE: MPLS adaptive traffic engineering, Proceedings of IEEE INFOCOM2001, Alaska, USA April (2001) 1300–1309.
[19] Z. Cao, Z. Wang, E. Zegura, Performance of hashing-based schemes for Internet load balancing, Proceedings of IEEE INFOCOM 2000, Tel Aviv, Israel March (2000) 332–341.
[20] D. Grossman, New terminology and clarifications for Diffserv, IETF Informational RFC-3260 April (2002).
[21] P. Trimintzios, G. Pavlou, I. Andrikopoulos, Providing traffic engineering capabilities in IP networks using logical paths, Eighth IFIP Workshop on Performance Modelling and Evaluation of ATM and IP Networks (IFPM ATM and IP 2000), Ilkey, UK July (2000).
[22] P. Trimintzios, I. Andrikopoulos, G. Pavlou, P. Flegkas, D. Griffin, P. Georgatsos, D. Goderis, Y. T'Joens, L. Georgiadis, C. Jacquenet, R. Egan, A management and control architecture for providing IP differentiated services in MPLS-based networks, IEEE Communications Magazine 39 (5) (2001) 80–88.
[23] T. Li, Y. Rekhter, Provider architecture for differentiated services and traffic engineering (PASTE), IETF Informational RFC-2430 October (1998).
[24] F. Le Faucheur, L. Wu, B. Davie, S. Davari, P. Vaananen, R. Krishnan, P. Cheval, J. Heinanen, Multi-protocol label switching (MPLS) support for differentiated services, IETF Standards Track RFC-3270 May (2002).
[25] P. Flegkas, P. Trimintzios, G. Pavlou, A policy-based management system for differentiated services networks, IEEE Network Magazine, Special Issue on Policy-Based Networking 28 (2) (2002) 50–56. In press.
[26] D. Bertsekas, R. Gallager, Data Networks, Prentice Hall, Englewood Cliffs, NJ, 1992.
[27] Z. Wang, J. Crowcroft, Quality of service routing for supporting multimedia applications, IEEE Journal of Selected Areas in Communications (JSAC) 14 (7) (1996) 1228–1234.
[28] D. Bertsekas, Nonlinear Programming, second ed., Athena Scientific, 1999.
[29] D. Eppstein, Finding k-shortest paths, SIAM Journal on Computing 28 (2) (1998) 652–673.
[30] E.W. Dijkstra, A note on two problems in connection with graphs, Numerische Mathematik 1 (1959) 269–271.
[31] E.W. Zegura, K.L. Calvert, S. Bhattacharjee, How to model an Internetwork, Proceedings of IEEE INFOCOM 96, San Francisco 2 (1996) 594–602.