Computer Networks 52 (2008) 1927–1937

Contents lists available at ScienceDirect

Computer Networks

journal homepage: www.elsevier.com/locate/comnet

A multi-layer network model based on ITU-T G.805

Freek Dijkstra [a,*], Bert Andree [a], Karst Koymans [a], Jeroen van der Ham [a,b], Paola Grosso [a], Cees de Laat [a]

[a] System and Network Engineering Group, University of Amsterdam, Kruislaan 403, 1098 SJ Amsterdam, The Netherlands
[b] TNO Defence, Security and Safety, Postbus 96864, 2509 JG Den Haag, The Netherlands

ARTICLE INFO

Article history:
Received 1 June 2007
Received in revised form 13 December 2007
Accepted 6 February 2008
Available online 13 March 2008

Keywords:
Network
Information model
Multi-layer network
Adaptation
Path finding

ABSTRACT

In recent years, dynamic multi-layer networks have emerged. Unlike regular networks these multi-layer networks allow users and other networks to interface on different technology layers. While path finding on a single layer is currently well understood, path finding on multi-layer networks is far from trivial. Even the constraints (the possible incompatibilities) are not always clear.

This paper proposes a model for multi-layer circuit-switched computer networks, based on ITU-T G.805 and GMPLS standards. Furthermore, it defines a simple algebra that can be used to verify the validity of network connections through such networks.

The most important contribution of our model and algebra is that they are technology independent: they can describe any circuit-switched network technology without modifications or tuning to the model and algebra. The model and algebra have been implemented in a syntax and network tool, which are briefly discussed.

1. Introduction

Since the introduction of computer networks *network models* have been developed to support users, administrators and others in managing their resources. Simple drawings of the network topology are often used as the first step in diagnosing problems.

In this article we study models of multi-layer networks. Multi-layer networks are computer networks where the configuration of the network can be changed dynamically at multiple layers. Examples of multi-layer networks include an optical network where both the WDM, TDM (SONET/SDH) and ethernet layers can be dynamically reconfigured and Hybrid networks [1,2]. Network models can help users and applications to understand the complexity of multi-layer networks, and can support path finding, scheduling, fault isolation, and visualisation applications.

This paper proposes a network model which is technology independent, but layer aware. This network model is based on ITU-T Recommendation G.805 [3] and the label concept in GMPLS [4]. Furthermore, we show that it is possible to use a simple algebra to verify the validity of an end-to-end network connection, traversing multiple layers.

As running example we use the optical networks in the global lambda integrated facility (GLIF) [5], as we are familiar with this community. GLIF is a collaboration of national research and educational networks across the globe. We will only look at circuit-switched connections, including ethernet VLANs and MPLS, and not at packet-switched connections that use lookup tables.

The organization of this paper is as follows. In Section 2 we show that a path finding algorithm needs to have knowledge about adaptation functions. Our model is introduced in Section 3, along with a simple algebra to verify validity of network connections through the network. Section 4 demonstrates the usability of our network model by

* Corresponding author. Tel.: +31 205257531.
E-mail address: fdijkstr@science.uva.nl (F. Dijkstra).

1389-1286/$ - see front matter
doi:10.1016/j.comnet.2008.02.013

an example network and describes our experience with an implementation of this model. Finally, we conclude the paper with related work and conclusions in Sections 5 and 6.

2. Multi-layer networks

One of the reasons to describe networks is to expose potential incompatibilities to path finding algorithms. Examples of such incompatibilities include MTU settings of two nodes leading to packet loss, a laser transmitting light at a wavelength undetectable by a receiver, or two devices supporting a different encapsulation (*adaptation*) of data of one layer in another layer. This section will give an example of incompatible adaptations.

2.1. GLIF example

Let us introduce our example network, as depicted in Fig. 1. Each circle in the picture represents an administrative domain. The domains are interconnected by links: the edges in the figure. Each domain is a National Research and Education Network (NREN), participating in the global lambda integrated facility (GLIF) [5]. Participants in the GLIF collaborate to provide researchers with circuit-switched connections across the globe, referred to as *lightpaths*. While this example is based on a real-life scenario, the mentioned incompatibility has been resolved, and we modified the topology a bit to emphasize our point.

The network in our example is not only a multi-domain network, but also a multi-layer network: the connection between the Université du Quebec and CA*net, as well as the connection between the Universiteit van Amsterdam and NetherLight is a gigabit/second ethernet (GE) connection. All other connections are OC-192 connections, based on SONET technology, and carrying 192 STS channels. Three of the domains in our example are capable of adapting gigabit ethernet in STS channels. To be exact, CA*net can embed gigabit ethernet in 24 concatenated STS channels (an STS-24c), and NetherLight can embed gigabit ethernet in 7 VC-4 containers, each in 3 concatenated STS channels: 21 STS channels in total (an STS-3c-7v). StarLight supports both methods to adapt ethernet in STS channels.

In our example, an application wants to have a gigabit/second ethernet (GE) connection between the Université du Quebec in Montreal (Canada) and the University of Amsterdam (the Netherlands). This can be achieved by creating a switched circuit through the interconnected research networks.

In this picture, the shortest path from the Université du Quebec to the University of Amsterdam would traverse CA*net, MAN LAN and NetherLight, respectively. However, in practice this would be a non-functioning network connection since the adaptation performed at CA*net, which adapts the GE in 24 STS channels, is incompatible with the adaptation of GE in 21 STS channels, as performed in NetherLight.

As a first approach, we can model the network of Fig. 1 as a two layer network as shown in Fig. 2. If we look at the SONET layer in the first subfigure and consider the adaptation capabilities, we can find the *potential* (as well as impossible) links on the ethernet layer, as shown in the second subfigure.

With that information, we can determine that a valid network connection is possible from Université du Quebec via CA* net to StarLight, where the GE is extracted from the STS-24c and re-adapted in a STS-3c-7v, and transported to NetherLight via MAN LAN.

The incompatibility in this example occurs between CA* net and NetherLight, which are not directly connected to each other. Apparently, a multi-layer path finding algorithm must not only have information of the layers and adaptations of a the direct neighbours of each domain, but also of the layers and adaptations of domains elsewhere in the network. Another way to look at this is that a path finding algorithm must not only take the *topological neighbors* into account, thus the neighbors at the physical layer, but also the *technological neighbors*: the neighbors on higher layers.

2.2. Graphs

The two layer model of Fig. 2 does not explicitly describe the adaptation functions. Instead, all possible end-to-end connections on the SONET layer occur as potential

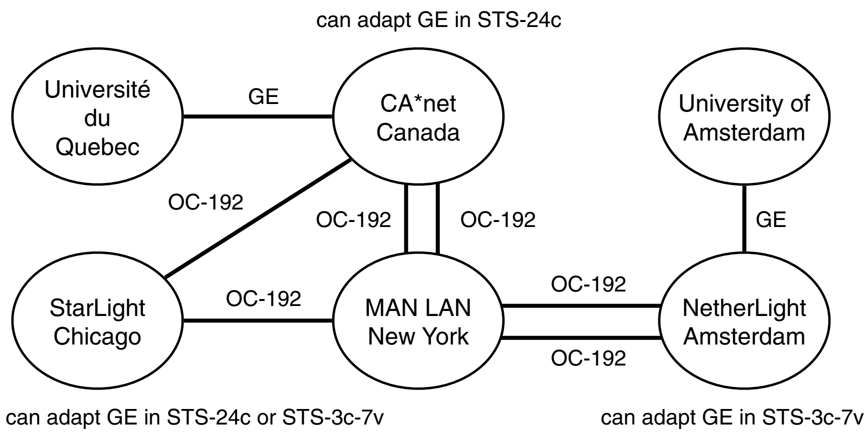


Fig. 1. Example of a multi-layer and multi-domain network.

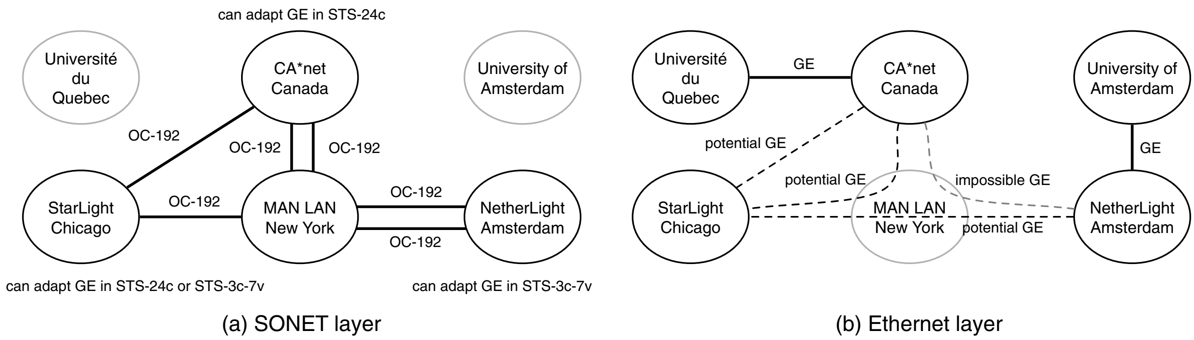


Fig. 2. Two layers (SONET and ethernet, respectively) of the multi-layer network of Fig. 1. This still does not visualise the adaptations between the two layers.

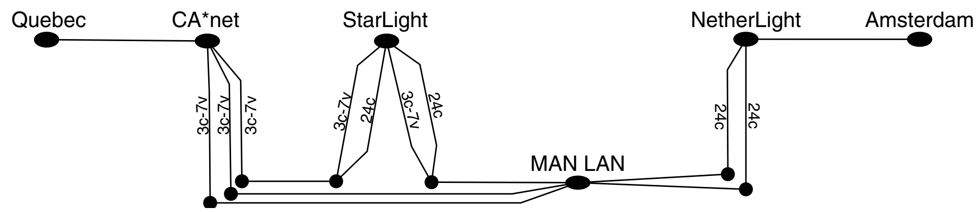


Fig. 3. The network of Fig. 1, modelled as graph with edge properties.

or impossible connections on the ethernet layer. This does not scale for more layers or larger networks.

One way to describe the constraints resulting from the use of different adaptation functions is the use of a graph with edge properties, as shown in Fig. 3. *Link-constraint* algorithms, such as variants of Dijkstra's shortest path algorithm [6] cannot handle the complexity of conditional constraints based on the chosen path (e.g., an edge with a certain de-adaptation can only be used if the corresponding adaptation occurs earlier in the path). *Path constraint* algorithms can find a valid path through this graph, but are computationally considerably harder than link-based constraints algorithms.

The fundamental limitation of graphs is that they only provide two basic building blocks, edges and vertices, while multi-layer computer networks have at least three building blocks: links, devices and adaptations, and perhaps four if you count interfaces. Consider the following choices:

- A vertex in a graph may either represent a device or an interface.
- An edge in a graph may either represent a link, a channel in a link (for instance wavelength 1310 nm in a fiber), or an adaptation function.

In Fig. 3 both links and adaptations are represented as an edge. This is not ideal, since links and adaptations have different properties.

As graphs do not provide us with the proper set of building blocks, we base our model on ITU-T G.805 functional elements.

3. Network model

The ITU-T G.805 recommendation can be used for describing connections in multi-layer networks. The model we present here is based on the ideas in G.805, and to a lesser extent, the ideas in GMPLS routing protocols.

We assume that readers are familiar with the terms connection point, termination, adaptation, link connection, tandem connection, network connection, subnetwork, subnetwork connection, client layer and server layer. Readers who are not familiar with these terms are advised to read the short *Introduction to ITU-T Recommendation G.805* [7] or turn to the recommendation itself [3].

3.1. Mapping to functional elements

Let us look at how the definitions of G.805 apply to networks. In other words, how to map real-life network elements (for instance links, and devices with interfaces) to G.805 functional elements.

Table 1 shows an overview of our mapping. We model the switching core of a network device as a subnetwork.

Table 1
Mapping of network elements to G.805 functional elements

Network element	Functional elements
Domain	Subnetwork(s)
Device	Matrix (Subnetwork)
Interface	Connection point(s) and adaptation function(s)
Link	Link connection

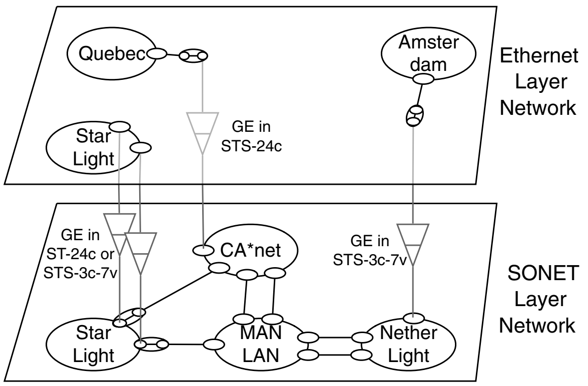


Fig. 4. The network of Fig. 1, modelled as functional elements.

A network device contains interfaces, which are modelled as multiple connection points (one or more for each layer) and optional adaptation capabilities. Finally, we map links between interfaces to link connections in G.805.

An interface is modelled as multiple connection points, one for each channel on each layer. For example, an OC-192 interface in a SONET device is modelled as 194 connection points: one connection point representing the logical fiber interface, one connection point representing the wavelength, and 192 connection points representing the 192 available STS channels.

The switching capability of a device is modelled as a switch matrix on a specific layer. For example, an SDH device which is capable of switching data with the granularity of STS channels has a switch matrix at the STS layer, while an SDH device which is capable of switching data with the granularity of virtual tributaries groups (VTG) has a switch matrix at the VTG layer.

Domains are treated as 'virtual' devices, and modelled as subnetworks, just like devices are. A difference is that physical devices in general can only switch on one granularity, represented by a subnetwork at a specific layer, while a domain may switch at different granularities, represented by multiple subnetworks.

Physical links are modelled as link connections on one of the physical layers. So a fiber is modelled as a link connection at the fiber layer and an unshielded twisted pair (UTP) cable is modelled as a link connection at the UTP layer.

An adaptation function defines the relation between the connection points that represent the different layers of an interface.

Fig. 4 shows an example network description using functional elements. The network is a slightly simplified version[1] of the network described in Fig. 1. Unlike Fig. 2, we explicitly modelled the adaptation functions. The two layers are separated vertically, while the different domains are separated horizontally. For example CA*net is represented by one subnetwork, five connection points and one adaptation function: the device is represented as a subnetwork, each SONET interface as one connection point and the ethernet interface as two connection points (one on the ethernet layer, one on the SONET layer), with an adaptation function in between.

Since StarLight can both switch at the ethernet layer as well as the SONET layer, it is represented as two subnetworks: one at the ethernet layer, one at the SONET layer. In this drawing, each interface only has one adaptation function (either STS-3c-7v or STS-24c), while in practice it may be possible to dynamically switch between these two adaptation functions at the same interface. It is possible to model this as two adaptation functions with a multipoint connection point (MPCP) to dynamically switch between them. These kind of choices needs to be made in order to turn the information model of this paper into a data model. We briefly discuss these choices in Section 4.

3.2. Notation

We define the function that combines the adaptation of data flow T from client layer to data flow U at the server layer, and the termination of the data flow U as

$$A : T^n \rightarrow U^m$$

with n and m equal to 1 for regular adaptation functions, $n > 1$ for multiplexing adaptation functions, and $m > 1$ for inverse multiplexing adaptation functions. For simplicity, we will simply write $A : T \rightarrow U$, and refer to both the data as well as the layers as T and U.

Except for Section 3.5, we will simply refer to the combined adaptation and termination function as the adaptation.

Given an adaptation function $A : T \rightarrow U$, then by definition a de-adaptation function[2] $D : U \rightarrow T$ exists such that $D \circ A = id : T \rightarrow T$.

Two adaptation functions A_1 and A_2 are considered a pair if $A_2^{-1} \circ A_1 = id$. Typically, because $A_1 = A_2$.

We will denote the adaptation performed between connection points *cp1t* at the client layer T and *cp1u* at the server layer U as $A^{cp1t}_{cp1u} : T \rightarrow U$. The corresponding de-adaptation function will be named $D^{cp1u}_{cp1t} : U \rightarrow T$, or equivalently, $(A^{cp1t}_{cp1u})^{-1} : U \rightarrow T$.

Unless noted otherwise, a function A will refer to an adaptation function, and a function D to a de-adaptation function.

Fig. 5 shows an example of a description of a network connection between two computers. As we can see, both interfaces are modelled (as connection points) on all layers they are implemented on. For instance for interface *if1*, as *cp1f* at the fiber layer, *cp1e* on the ethernet layer and *cp1i* at the IP layer.

3.3. Channel labels

In 3.1 we wrote that each channel is represented as a connection point. So an OC-192 interface has 192 STS connection points, a tagged ethernet interface has 4096 VLAN

[1] For simplicity, the ethernet-in-STS-channels adaptation is modelled as a one-to-one relation, instead of the actual one-to-many relation.

[2] In mathematical terms D is a retraction or a split epimorphism.

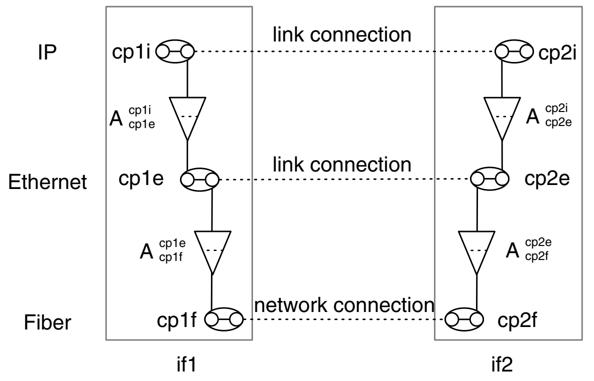


Fig. 5. Example of a multi-layer network connection. Interfaces *if1* and *if2* are modelled as connection points at all three layers. The relation between the connection points is defined by the adaptation and termination functions.

connection points and an ATM VPI can contain 65536 VCI channels.

Seemingly, this does not scale very well. However, that would be a misunderstanding, since it is often not needed to describe all individual connection points in a syntax. Only the channels that are configured or actively in use need to be described in detail. The other channels can simply be described as a set or range of available channels. This is an important distinction between the model and the syntax describing a model: a model can be verbose, while the syntax is compact.

The use of channels requires an addition to our model. Consider the adaptations pair $A^{cp1t_1;cp1t_2;cp1t_3;\ldots;cp1t_n}_{cp1u} : T^n \to U$ and $A^{cp2t_1;cp2t_2;cp2t_3;\ldots;cp2t_n}_{cp2u} : T^n \to U$ in Fig. 6. This is an example of a multiplexing adaptation function with client layer connection points $cpit_1; cpit_2; cpit_3; \ldots; cpit_n$ with associated notation.

ITU-T Recommendation G.805 defines the logic that a pair of adaptation function, connection with a network connection at the server layer, yields a link connection at the client layer.

This logic dictates that since there is a network connection on layer *U* and the two adaptations are equal, there is a link connection on layer *T*. However, it is not obvious between which pair of connection points there is a link connection. Without further specification, it could for example be between $cp1t_1$ and $cp2t_3$. As a remedy, we introduce the concept of *labels*, inspired by GMPLS [4].

Each connection point has two associated labels for each link connection connected to it: the *ingress label* and *egress label*. These labels uniquely identify the channel of an adaptation. Examples of labels are STS timeslots, IEEE 802.1Q (VLAN) tags or wavelengths.

Fig. 7 shows two connection points and a link connection. For labels, we distinguished between the two uni-directional link connections that constitute a bi-directional link connection.

For uni-directional connections a link connection from *cp1* to *cp2* can only exist if the egress label of connection point *cp1* is equal to the ingress label of connection point *cp2*. For a bi-directional link connection, we also require that the egress label of connection point *cp2* is equal to the ingress label of connection point *cp1*.

For bi-directional circuit-switched connections, the ingress and egress label are typically the same, and we simply talk about the label of a connection point, meaning both the ingress and egress label.

3.4. Validation of network connections

In this section, we introduce a mathematical concept to check the validity of a network connection. We use a recursive definition to verify that a network connection is *valid*.

Given connection points *cp1* and *cp2*, we will define the following binary relations:

- $L(cp1, cp2) \Leftrightarrow$ a bi-directional Link[3] between *cp1* and *cp2* exists.
- $\mathrm{SNC}(cp1, cp2) \Leftrightarrow$ a bi-directional subnetwork connection between *cp1* and *cp2* exists.
- $\mathrm{LC}(cp1, cp2) \Leftrightarrow$ a bi-directional link connection between *cp1* and *cp2* exists.
- $\mathrm{TC}(cp1, cp2) \Leftrightarrow$ a bi-directional Tandem connection between *cp1* and *cp2* exists.

In addition, we define the function:

- $Lb_{\mathrm{out}}(cp)$ to be the egress label of connection point *cp*.
- $Lb_{\mathrm{in}}(cp)$ to be the ingress label of connection point *cp*.

If the egress and ingress labels are equal, as for bi-directional circuit-switched network connections, we can define the equality:

$$Lb(cp) := Lb_{\mathrm{out}}(cp) = Lb_{\mathrm{in}}(cp).$$

We postulate a network *N* as a set of connection points, links, subnetworks, and adaptations, and a network configuration *C* as a set of labels, and subnetwork connections. Given these basic truths, we deduce the link connections and tandem connections: the valid connections through the network.

G.805 defines a tandem connection as a transport entity formed by a series of contiguous link connections and/or subnetwork connections. We define a tandem connection recursively to be either a link connection, a subnetwork connection or a tandem connection followed by another tandem connection.

A link connection is defined either to be a link or a combination of an adaptation source, a terminated tandem connection at the server layer, and an adaptation sink.

Mathematically the definitions of tandem connection and link connection can be written as:

$$\mathrm{TC}(cp1, cp2) = \begin{cases} \mathrm{LC}(cp1, cp2) \vee \\ \mathrm{SNC}(cp1, cp2) \vee \\ \exists cp3 : \mathrm{TC}(cp1, cp3) \wedge \mathrm{TC}(cp3, cp2) \end{cases} \quad (1)$$

[3] Actually: a *transport entity* across a link, but we will use the term *link* for simplicity.

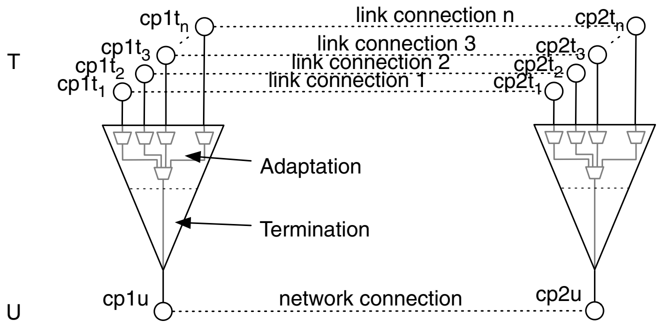


Fig. 6. Channels correspond with multiple link connections at the client layer over one link connection at the server layer.

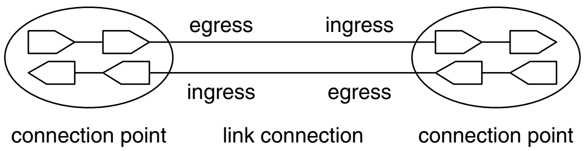


Fig. 7. The ingress and egress part of an connection point with respect to a link connection.

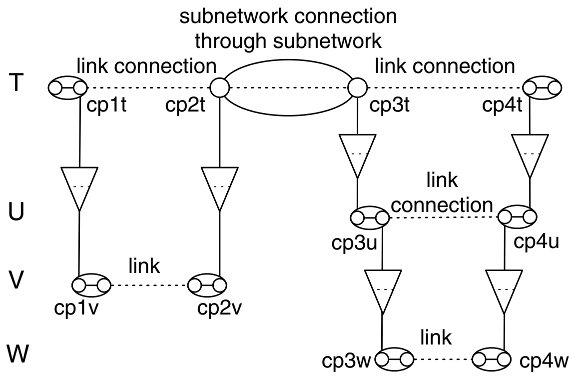


Fig. 8. Example of a valid network connection. A valid tandem connection consisting of two link connections and a matrix connection.

and

$$\mathrm{LC}(cp1, cp2) = \begin{cases} L(cp1, cp2) \vee \\ \exists cp3, cp4, T, U, A^{cp1}_{cp3}, D^{cp4}_{cp2} : \\ \mathrm{TC}(cp3, cp4) \wedge \\ A^{cp1}_{cp3} : T \to U \wedge \\ D^{cp4}_{cp2} : U \to T \wedge \\ D^{cp4}_{cp2} \circ A^{cp1}_{cp3} = Id : T \to T \wedge \\ Lb_{\mathrm{out}}(cp1) = Lb_{\mathrm{in}}(cp2) \wedge \\ Lb_{\mathrm{in}}(cp1) = Lb_{\mathrm{out}}(cp2) \end{cases} . \tag{2}$$

Furthermore, since we restrict ourself to bi-directional connections:

$$\begin{aligned} &L(cp1, cp2) \to L(cp2, cp1) \\ &\mathrm{LC}(cp1, cp2) \to \mathrm{LC}(cp2, cp1) \\ &\mathrm{TC}(cp1, cp2) \to \mathrm{TC}(cp2, cp1) \\ &\mathrm{SNC}(cp1, cp2) \to \mathrm{SNC}(cp2, cp1) \end{aligned} \tag{3}$$

and even

$$\left(A^{cp1t}_{cp1u} : T \to U\right) \to \left(D^{cp1u}_{cp1t} : U \to T\right) \tag{4}$$

with

$$D^{cp1u}_{cp1t} \circ A^{cp1t}_{cp1u} = Id : T \to T.$$

These definitions can easily be transformed to those for uni-directional connections, or explicitly allowing multiplexing and inverse multiplexing adaptation functions.

These recursive definitions, in particular the one for link connections, need a short explanation. We will refer to Fig. 8 to illustrate the concepts. This figure shows two links, five link connections, nine tandem connections and one subnetwork connection in total.

Formally, we postulate the network $N = \{cp1t, cp2t, cp3t, cp4t, cp1v, cp2v, cp3u, cp4u, cp3w, cp4w, L(cp1v, cp2v), L(cp3w, cp4wv), A^{cp1t}_{cp1v}, A^{cp2t}_{cp2v}, A^{cp3t}_{cp3u}, A^{cp4t}_{cp4u}, A^{cp3u}_{cp3w}, A^{cp4u}_{cp4w}\}$ and its configuration $C = \{\mathrm{SNC}(cp2t, cp3t)\}$. Also, $A^{cp1t}_{cp1v} = A^{cp2t}_{cp2v}$, $A^{cp3t}_{cp3u} = A^{cp4t}_{cp4u}$, and $A^{cp3u}_{cp3w} = A^{cp4u}_{cp4w}$.

The most simple link connection is simply a link. So $L(cp1v, cp2v)$ implies $\mathrm{LC}(cp1v, cp2v)$ and $L(cp3w, cp4w)$ implies $\mathrm{LC}(cp3w, cp4w)$. By definition of a tandem connection, a link connection is also a tandem connection, so $\mathrm{LC}(cp1v, cp2v)$ and $\mathrm{LC}(cp3w, cp4w)$ imply $\mathrm{TC}(cp1v, cp2v)$ and $\mathrm{TC}(cp3w, cp4w)$, respectively.

We just saw that $\mathrm{TC}(cp1v, cp2v)$ holds. Furthermore, $A^{cp1t}_{cp1v} = A^{cp2t}_{cp2v}$, thus:

$$D^{cp2v}_{cp2t} \circ A^{cp1t}_{cp1v} = Id : T \to T \tag{5}$$

with $D^{cp2v}_{cp2t} = (A^{cp2t}_{cp2v})^{-1}$. Therefore, from Eq. 2 we must conclude that $\mathrm{LC}(cp1t, cp2t)$. In G.805 terminology, the *adaptation source cp1t* and the *adaptation sink cp1v* are *paired*.

Similarly, $\mathrm{LC}(cp3u, cp4u)$, and therefor $\mathrm{TC}(cp3u, cp4u)$ hold because $\mathrm{TC}(cp3w, cp4w)$ and $D(cp4w, cp4u) \circ A(cp3u, cp3w) = Id : U \to U$, and $\mathrm{LC}(cp3t, cp4t)$ holds because $\mathrm{TC}(cp3u, cp4u)$ and $D(cp4u, cp4t) \circ A(cp3t, cp3u) = Id : T \to T$.

Furthermore, $\mathrm{LC}(cp1t, cp2t)$, $\mathrm{SNC}(cp2t, cp3t)$, and $\mathrm{LC}(cp3t, cp4t)$, respectively imply $\mathrm{TC}(cp1t, cp2t)$, $\mathrm{TC}(cp2t, cp3t)$, and $\mathrm{TC}(cp3t, cp4t)$. Two consecutive tandem connections are also a tandem connection, so from this follows that $\mathrm{TC}(cp1t, cp3t)$ and $\mathrm{TC}(cp2t, cp4t)$. Finally, $\mathrm{TC}(cp1t, cp4t)$ holds because $\mathrm{LC}(cp1t, cp2t)$ and $\mathrm{TC}(cp2t, cp4t)$.

3.5. Well typed adaptations

So far, we combined the adaptation and termination function.

We did so to make our definition of LC($cp1,\ cp2$) in Eq. 2 compatible with the definition of link connection in G.805, where a link connection *represents a pair of adaptation functions and a trail in the server layer network*. Since a *trail* is a terminated network connection in G.805, the adaptation and termination functions are always combined.

For validation, in the definition of link connections we required that the server layer network connection was terminated. In this section we will loosen this restriction. We call a link connection that is formed by a combination of an adaptation source, a server layer tandem connection, and an adaptation sink *well typed*, even if the server layer network connection is not terminated as required for *validity*.

Refer to Fig. 9 for a *well typed*, but invalid link connection between *cp1t* and *cp2t*. An example of such an invalid link connection could be if A^{cp1t}_{cp1v} adds a header to a packet, and A^{cp1v}_{cp1w} adds a tail to the result. Then, D^{cp2w}_{cp2u} first removes the header and finally D^{cp2u}_{cp2t} removes the tail. While the result is the very same packet, the intermediate result for adaptation and de-adaptation was different: a packet with header (layer V) during adaptation and a packet with tail (layer U) during de-adaptation. Since *cp1v* and *cp2u* are on different layers, no termination is possible at A^{cp1t}_{cp1v} and D^{cp2u}_{cp2t}.

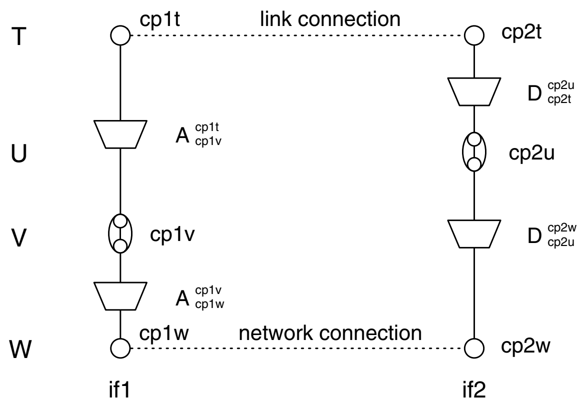


Fig. 9. Example of a well typed, but invalid connection. *U* and *V* are different layers.

Loosening the restriction that each adaptation function is followed by a termination function has consequences for a possible definition of atomic or combined adaptation functions. We will not pursue this idea further in this paper, but assume that each adaptation function is followed by a termination function.

4. Application

4.1. Example validation

In the introduction, we sketched an example network which had some restrictions in the validity of connections through the network. We will now show how the model in Section 3 can be used to make this explicit.

See Fig. 10 for a representation of the network of Fig. 1, as functional elements, using the mapping of Table 1.

This network is identified by $N = \{q1, c1, c2, c3, c4, c5, s1, s2, s3, s4, \quad m1, m2, m3, m4, m5, n1, n2, n3, n4, a1, L(q1, c2), L(c3, s3), L(c4, m2), L(c5, m3), L(s4, m1), L(m4, n2), L(m5, n3), L(n1, a1), A^{c1}_{c2}, A^{s1}_{s3}, A^{s2}_{s4}, A^{n1}_{n4}\}$ where $A^{c1}_{c2} = STS24c$ and $A^{n1}_{n4} = STS3c7v$.

The shortest path (traversing fewest link connections) between connection point $q1$ at the Université du Quebec and connection point $a1$ at the University of Amsterdam traverses through StarLight, MAN LAN and NetherLight. This would result in connection 1 in the Fig. 10. Formally, connection 1 is dataflow through the network elements $[L(q1, c1), A^{c1}_{c2}, \mathrm{SNC}(c2, c5), L(c5, m3), \quad \mathrm{SNC}(m3, m5), L(m5, n2), SNC(n3, n4), D^{n4}_{n1}, L(n1, a1)]$ and is identified by the subset $C1 = \{\mathrm{SNC}(c2, c5), \mathrm{SNC}(m3, m5), \mathrm{SNC}(n3, n4)\}$ of the network configuration.

Since $\mathrm{SNC}(c2, c5)$, $L(c5, m3)$, $\mathrm{SNC}(m3, m5)$, $L(m5, n2)$, and $\mathrm{SNC}(n3, n4)$, it follows that $\mathrm{TC}(c2, n4)$. However, from A^{c1}_{c2}, $\mathrm{TC}(c2, n4)$, D^{n4}_{n1} does *not* follow $\mathrm{LC}(c1, n1)$ since $D^{n4}_{n1} \circ A^{c1}_{c2} = \mathrm{STS3}c7v^{-1} \circ \mathrm{STS24}c \neq \mathrm{Id} : \mathrm{Ethernet} \to \mathrm{Ethernet}$.

Therefore, connection 1 does not lead to a valid tandem connection from Quebec to Amsterdam, given these links and subnetwork connections: $N, C1 \nvdash TC(q1, a1)$.

StarLight is capable of supporting either adaptation function. This is modelled in Fig. 10 using two multi-point connection points (MPCP). A^{s1}_{s3} is either equal to STS24c, or to STS3c7v.

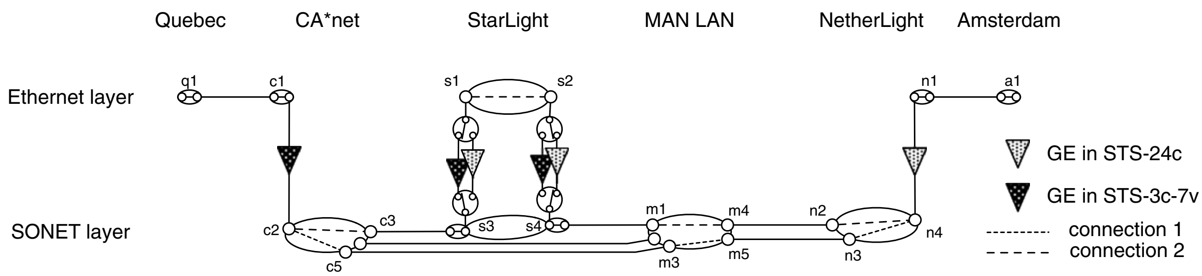


Fig. 10. A network representation of the network of Fig. 1, using functional elements. Dark-gray adaptation functions represent adaptation of gigabit/second ethernet (GE) over STS-24c, while light-gray adaptation functions represent GE over STS-3c-7v. StarLight is capable of either adaptation function.

Let's now consider connection 2, identified by the subset $C2 = \{\mathrm{SNC}(c2, c3), \mathrm{SNC}(s1, s2), \mathrm{SNC}(m1, m4), \mathrm{SNC}(n2, n4)\}$ of the network configuration, $A^{s1}_{s3} = \mathrm{STS24}c$ and $A^{s2}_{s4} = \mathrm{STS3}c7v$.

We defined in the network configuration to be equal to $\mathrm{STS24}c^{-1}$.

$A^{s1}_{s3} = \mathrm{STS24}c$, so $D^{s3}_{s1} = \mathrm{STS24}c^{-1}$, not $\mathrm{STS3}c7v^{-1}$. It now follows that $\mathrm{LC}(c1, s1)$ is true, since $A^{c1}_{c2} = \mathrm{STS24}c$, $\mathrm{TC}(c2, s3)$, $D^{s3}_{s1} = \mathrm{STS24}c^{-1}$ and $D^{s3}_{s1} \circ A^{c1}_{c2} = \mathrm{STS24}c^{-1} \circ \mathrm{STS24}c = Id : \mathrm{Ethernet} \to \mathrm{Ethernet}$.

Similarly, $\mathrm{LC}(s2, n1)$ is true because $A^{s2}_{s4} = \mathrm{STS3}c7v$, $\mathrm{TC}(s4, n4)$ (because $L(s4, m1)$, $\mathrm{SNC}(m1, m4)$, $L(m4, n3)$ and $\mathrm{SNC}(n3, n4)$), $D^{n4}_{n1} = \mathrm{STS3}c7v^{-1}$, and also $D^{n4}_{n1} \circ A^{s2}_{s4} = \mathrm{STS3}c7v^{-1} \circ \mathrm{STS3}c7v = Id : \mathrm{Ethernet} \to \mathrm{Ethernet}$.

Since we now have $L(q1, c1)$, $\mathrm{LC}(c1, s1)$, $\mathrm{SNC}(s1, s2)$, $\mathrm{LC}(s2, n1)$ and $L(n1, a1)$, thus $\mathrm{TC}(q1, a1)$ must be true. This proves that there is now a valid tandem connection from $q1$ at the Université du Quebec to $a1$ at the University of Amsterdam: $N, C2 \vdash \mathrm{TC}(q1, a1)$.

4.2. Implementation

To show that this work has practical applications, we created an implementation of our model. We did so by extending our current work on the network description language (NDL) [8], which was already able to describe single layer networks, to describe multi-layer networks [9]. NDL and our multi-layer extension are implemented as a resource description framework (RDF) schema The schemas are publicly available on our website [10].

The NDL multi-layer schema describes the basic concepts of network layers, and allows descriptions of actual technologies. We have successfully described WDM, Fiber, SONET, SDH, ATM, Ethernet and MPLS.

In addition to the schemas, we created an experimental framework in Python [11]. The framework is now in use for various tools:

- Description of the current configuration of our network, and trace network connections;
- Generation of sample networks;
- Path finding of multi-layer connections through the network;
- Fault Isolation of errors in multi-layer network connections.

The path finding tools not only require information about the current state of the network, but also about the potential state – the capabilities. This was included in the syntax by adding specific properties to the subnetwork connections, such as its ability to convert between different labels (e.g., do wavelength conversion or not).

The framework is able to distinguish between equivalent and non-equivalent wavelengths, VLANs, and other labels, as well as compatible and incompatible adaptations. For example, the path finding demonstration was not only able to find the network connection of Fig. 10, but even more advanced examples where the shortest path has a loop (it uses the link StarLight – MAN LAN twice) due to additional label constraints [12,13].

The software package is freely available for download [11].

4.3. Extensions and future work

One of our goals is to describe actual networks in a technology independent way to implement some of the extensions mentioned in this section. In that process, it is likely that some of the (mathematical) simplicity of the current model will be lost while gaining a model able to describe more technology specific parameters, without becoming very verbose.

This section highlights a few of the possible extensions to our current model.

4.3.1. Layer properties

One motivation to describe networks is to make incompatibilities between interfaces specific. We did so for incompatible adaptations (for instance ethernet over STS-24c or over STS-3c-7v described in Section 2), and for incompatible labels (for instance a wavelength with label "1310 nm" or a wavelength with label "850 nm").

This does not cover all possible incompatibilities. For example, a network connection may not be possible due to a difference in the allowed packet size (for instance ethernet packets with an MTU of 1500 bytes or 9000 bytes, or anything in between). While it is technically possible to model this as a few thousand different adaptation functions, this is not efficient. The solution in our syntax is to model it as a property of the layer itself, rather then a property of the adaptation function.

4.3.2. Inverse multiplexing

Both G.805 as well as our syntax support inverse multiplexing: the adaptation of one data stream in multiple channels. Ethernet in STS channels, as described in examples in this article, is an example of inverse multiplexing. The model as presented in this paper is limited to a single underlying network connection. For inverse multiplexing, $cp3, cp4$ in Eq. 2 needs to be changed to $cp3_1, \ldots, cp3_n$, $cp4_1, \ldots, cp4_n$, and $\mathrm{TC}(cp3, cp4)$ must be changed to $\forall i \in [1, \ldots, n] : \mathrm{TC}(cp3_i, cp4_i)$.

Furthermore, the use of inverse multiplexing can lead to a sequence of de-adaptation and adaptation at the same interface. For example, a wavelength is demultiplexed from a signal on a fiber, and ethernet packets are demultiplexed from the wavelength. This is the de-adaptation. Then, the ethernet packets are inverse multiplexed (adapted) in multiple STS channels at the same interface.

Such sequences of demultiplexing and inverse multiplexing gives two adaptation stacks at the same interface. We coined these the external and the internal adaptation stack.

4.3.3. Multi-domain

In this document we focused on multi-layer network descriptions. Beside multi-layer, it is also possible to partition a network in administrative domains. We did not cover this topic in this document, but would like to refer interested readers to our work on the network description language (NDL) [8].

4.3.4. Broadcast and multicast

ITU-T G.805 does not explicitly support broadcast and multicast. Our model can describe broadcast networks using multiple subnetwork connections. This scales with $O(n^2)$ with n the number of nodes. Since this only works fine for small broadcast networks, we added a specific description for broadcast networks to our syntax to support ethernet VLANs. For IP and MAC layers, it is probably inevitable to define a more elaborate model for switch matrices, including lookup tables, and hop-by-hop routing.

4.3.5. Bandwidth

Like G.805, our model does not (yet) have a notion of bandwidth.

4.3.6. Physical layer properties

According to G.805, a concatenation of link connections and subnetwork connections placed in series form a valid tandem connection, which is able to transport data. We followed this concept in Section 3.4.

This assumption is not generally true on the physical layer. For example, the power loss of two individual link connections may fall within acceptable limits, but the power loss of the serial-compound link may fall outside the specified range.

G.805 implicitly considers human-engineered networks only, by assuming that if all link connections, adaptations and terminations are applied correctly, indeed everything functions properly. This is generally true on higher layers (TDM and above), but not on the physical layer, where signal degradation is an important factor to take into account.

In order to apply G.805 on the physical layer, including wireless networks, layer parameters as mentioned in Section 4.3.1 must be defined for the network elements. For the lower layers, this includes power levels, signal degradation, cable length, and optical dispersion. For higher layers, parameters like delay and jitter may also be defined.

4.3.7. Uniqueness of layers

ITU-T G.805 defines a *layer* as the set X of all possible connection points of the same type. Two connection points are of the same type, if a data-transport function can be created between them.

This definition, which is taken from G.805, is ambiguous. Imagine three connection points a, b and c, where data-transport between a and b and between b and c is possible, but not between a and c. In this case, it is unclear if we are dealing with one, two or even three layers.

An example of such ambiguity is if a, b and c are ethernet interfaces with a supporting untagged ethernet, b supporting both tagged and untagged ethernet at the same time and c supporting only tagged ethernet.

Another example is if a, b, and c are all ethernet interfaces, with interface a operating at a capacity of 10 Mbit/s, c at 100 Mbit/s and b auto-sensing supporting both 10 Mbit/s and 100 Mbit/s.

Our solution to this problem is to define interfaces with potential incompatibilities as two or more different layers. In the later example, a 10 Mbit/s ethernet layer and a 100 Mbit/s ethernet layer. Interface b would then support two adaptations functions. We have in fact shown this earlier in Fig. 10, where the Interfaces at StarLight supported two adaptation functions.

4.3.8. Uniqueness of adaptations

We started our paper with a short discussion on reasons to describe networks. One of our goals is to be able to describe potential incompatibilities we like to expose to path finding algorithms. However, what is incompatible may change over time.

For example, if everyone would use 850 nm lasers, there is no need to describe the wavelength, since there are no incompatibilities. As soon as lasers with other colors are deployed, this might lead to incompatibilities, so it has to be described. However, as soon as every device is able do color conversion on the fly, the incompatibility would again disappear. The progress in technology means that potential incompatibilities come and go.

There is no unique way to defined adaptations in practice. For most purposes it is sufficient to distinguish between "WAN PHY" or "LAN PHY" for 10 Gb/s ethernet. However, new technologies may emerge that require explicit description of the XGMII, XAUI, PCS or PMD sublayers.

The great advantage of an abstract model is that a path finding algorithm, such as the one we implemented, can use a technology independent network description; it only knows about the generic concepts such as "layer", "adaptation" and "label", but not about the specific technologies. It does need to be tuned or adjusted as new network technologies come along.

5. Related work

Mathematical models exist to describe networks. Our interest lies in technology independent, multi-layer network descriptions. Table 2 shows some of the related work.

Earlier in this paper we discussed ITU-T G.805 and graph theory, which are both technology independent: they can be applied to any technology.

The few models that take multiple layers into account are often geared towards very specific cases (for instance simulation of a few specific layers, like in network simulators).

As early as 1995, Laarhuis developed a model where the network was divided in three layers [14]. The physical media layer containing all network components and fibers, the optical layer consisting of wavelength channels, and the electrical layer which uses the virtual topology of the optical layer to obtain connections. Like us, he based his work on ITU-T G.805 functional elements.

Table 2
Categorization of related work

	Technology specific	Technology independent
Single layer	Most network models	Graph theory
Multi-layer	GMPLS, CIM, network simulators	ITU-T G.805

Single layer technology specific models are not listed, since they are of no interest to us.

We describe two efforts that have generated a considerable momentum at the moment of this writing, generalized multi-protocol label switching (GMPLS) and common information model (CIM).

5.1. Generalized multi-protocol label switching

Generalized multi-protocol label switching (GMPLS) is a set of protocols for routing and signaling in circuit-switched networks [4,15,16].

Routing protocols distribute topology knowledge among the devices in a network. Open shortest path first – traffic engineering (OSPF–TE), the most commonly used routing protocol in GMPLS, can describe a link at a specific layer. OSPF–TE does so by specifying the encoding and switching capability of each interface in a link state announcement (LSA) message.

The signaling protocol of GMPLS, ReSerVation protocol – traffic engineering (RSVP–TE) can announce the available *labels*, like available wavelengths or VLAN tags to its neighbours.

GMPLS can describe the layers and switching capability of devices at a layer. However, it currently only has a limited concept of adaptation, by using a Generalized Protocol IDentifier (G-PID) to specify the payload of the channels. But this information is only used during the signalling phase, when the path is already established. In agreement with our findings, it was independently determined that *the advertisement of the internal adaptation capability of hybrid nodes* is required in the routing protocol [17]. A proposal for these routing extensions is in draft as of this writing [18].

One of the early premises of GMPLS is that incompatibilities can be solved during the signaling phase, after the path has been chosen. Thus, the available adaptation functions are not announced in the routing phase and thus are not present in the network description. Often this is a valid assumption since network engineers try to avoid possible incompatibilities when building the network. However, as we have shown in Section 2, incompatibilities can occur, even between domains that are not directly connected to each other. To work around this, a steady increase of extensions to OSPF–TE have been defined for GMPLS to still describe different possible incompatibilities.

5.2. Common information model

The common information model (CIM) [19] is a schema defined by the distributed management task force (DMTF). CIM is an object oriented schema which can describe hardware elements in high detail. It can describe networks and has a collection of schemes to describe a configuration of IP, BGP, OSPF, ethernet (including VLAN), NAT, pipes and filters.

This makes CIM useful for describing a (network) configuration of access networks, especially if the data is automatically generated using SNMP. CIM is less suitable for core networks since it cannot describe DWDM or TDM networks. CIM is a technology specific model, which makes it less suitable for our purpose.

6. Conclusion

At the beginning of this paper we have posed the hypothesis that it is necessary to describe adaptations between layers for path finding in multi-layer networks. We have shown this with an example in Section 2, and set two goals: a model for multi-layer networks and an algebra to validate potential connections through a given network.

We fulfilled the first goal with a mapping from network elements to function elements, based on previous work in the ITU-T G.805 and GMPLS standards. We satisfied the second goal with a simple algebra, without relying on complex path constraints.

To validate a network connection, we postulate a network as a set of connection points, label values, and links, and the network configuration as a set of subnetwork connections and labels. Using this information and a recursive definition for link connections and tandem connections, we can deduce information about the validity of network connections.

In Section 4.1, we have explained how our approach is successful in detecting possible and impossible network connections in case of multiple incompatible adaptation functions in the network.

We have shown that our work presents a valid solution to determine valid paths through circuit-switched layer, including ethernet VLANs and MPLS. We not only applied the algebra to support this claim, but also implemented this logic in a software framework that is able to find valid paths in multi-layer networks.

Both theoretical and practical future work is necessary. We have shown that multi-layer networks cannot be represented as simple graphs, and proposed an alternative solution, that supports a path finding function. However, it is yet uncertain if this approach is an optimal strategy for path finding in large scale multi-layer networks.

For practical usage of our model, standardization is required. There is no unique way to define technologies and layers in a network. The network layers to describe will change in the future, as new technologies emerge. Since our model is technology independent, it does not enforce a particular choice. The given validation function will continue to work, even when practical network descriptions change.

Acknowledgement

Part of this research is done under the GigaPort Next Generation project led by the Dutch National Research Network (SURFnet), and the Interactive Collaborative Information Systems (ICIS) project. Both projects are supported by the Dutch Ministry of Economic Affairs (Grant Numbers BSIK03020 and BSIK03024).

References

[1] C. de Laat, E. Radius, S. Wallace, The rationale of the current optical networking initiatives, Future Generation Computer Systems 19 (6) (2003) 999–1008. URL <http://dx.doi.org/10.1016/S0167-739X(03)00077-3>.

[2] O.H. Martin, The ongoing evolution from packet based networks to hybrid networks in research and education networks, in:

International Symposium on Nuclear Electronics and Computing (NEC 2005), Varna (Bulgaria), September 2005, pp. 198–211. URL <http://www.jinr.ru/NEC/NEC-2005/proceeding2005/Martin.doc>.
[3] ITU-T Recommendation G.805: Generic functional architecture of transport networks, International Telecommunication Union, March 2000. URL <http://www.itu.int/rec/T-REC-G.805/en>.
[4] E. Mannie, Generalized multi-protocol label switching (GMPLS) architecture, RFC 3945 (Proposed Standard), October 2004. URL <http://www.ietf.org/rfc/rfc3945.txt>.
[5] Global Lambda Integrated Facility (GLIF). An international collaboration of national research and education networks promoting the paradigm of 'lambda networking'. URL <http://www.glif.is/>.
[6] E.W. Dijkstra, A note on two problems in connexion with graphs, in: Numerische Mathematik, vol. 1, Mathematisch Centrum, Amsterdam (The Netherlands), 1959, pp. 269–271.
[7] F. Dijkstra, H.M. Andree, K. Koymans, J.J. van der Ham, Introduction to ITU-T Recommendation G.805, Technical Report UVA-SNE-2007-001, December 2007. URL <http://www.science.uva.nl/research/sne/reports/>.
[8] J.J. van der Ham, F. Dijkstra, F. Travostino, H.M. Andree, C.T.A.M. de Laat, Using RDF to describe networks, Future Generation Computer Systems 22 (8) (2006) 862–867. URL <http://dx.doi.org/10.1016/j.future.2006.03.022>.
[9] J.J. van der Ham, F. Dijkstra, P. Grosso, R. van der Pol, A. Toonk, C.T.A.M. de Laat, A distributed topology information system for optical networks based on the semantic web, J. Optical Switching and Networking (2008), URL <http://dx.doi.org/10.1016/j.osn.2008.01.006>.
[10] J.J. van der Ham, F. Dijkstra, Network description language homepage. URL <http://www.science.uva.nl/research/sne/ndl/>.
[11] F. Dijkstra, J.J. van der Ham, Python NDL toolkit homepage. URL <http://ndl.uva.netherlight.nl/trac/ndl/>.
[12] F. Dijkstra, H.M. Andree, J.J. van der Ham, K. Koymans, C.T.A.M. de Laat, Going in loops to reach your goal – multi-layer path finding with NDL, Poster presented at GLIF meeting, Prague (Czech Republic), September 2007. URL <http://staff.science.uva.nl/~fdijkstr/presentations/Going-in-Loops.pdf>.
[13] F. Kuipers, F. Dijkstra, Path selection in multi-layer networks, in press. URL <http://staff.science.uva.nl/~fdijkstr/publications/multilayer-pathselection.pdf>.
[14] J.H. Laarhuis, Multichannel interconnection in all-optical networks, Ph.D. Thesis, Centre for Telematics and Information Technology, September 1995.
[15] K. Kompella, Y. Rekhter (Eds.), Routing Extensions in Support of Generalized Multi-Protocol Label Switching (GMPLS), RFC 4202 (Proposed Standard), October 2005. URL <http://www.ietf.org/rfc/rfc4202.txt>.
[16] L. Berger, Generalized Multi-Protocol Label Switching (GMPLS) Signaling Resource ReserVation Protocol–Traffic Engineering (RSVP–TE) Extensions, RFC 3473 (Proposed Standard), January 2003. URL <http://www.ietf.org/rfc/rfc3473.txt>.
[17] J.-L. Le Roux, D. Brungard, E. Oki, D. Papadimitriou, K. Shiomoto, M. Vigoureux, Evaluation of existing GMPLS protocols against multi-layer and multi region networks (MLN/MRN), Internet-Draft (Work in Progress), November 2007. URL <http://tools.ietf.org/html/draft-ietf-ccamp-gmpls-mln-eval>.
[18] K. Shiomoto, D. Papadimitriou, J.-L. Le Roux, M. Vigoureux, D. Brungard, Requirements for GMPLS-based multi-region and multi-layer networks (MRN/MLN), Internet-Draft (Work in Progress), November 2007. URL <http://tools.ietf.org/html/draft-ietf-ccamp-gmpls-mln-reqs>.
[19] Common Information Model (CIM), CIM network, version 2.17, Distributed Management Task Force, August 2007. URL <http://www.dmtf.org/standards/cim/>.

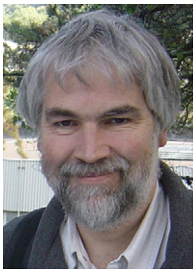
Freek Dijkstra received his M.Sc. in applied physics from the Utrecht University in 2002. He is researcher in System and Network Engineering group at the University of Amsterdam, where he pursues a Ph.D. degree. Freek's primary interest is path finding through multi-layer optical networks.

Bert Andree received his Ph.D. degree in 1995 on the topic of neural networks at Utrecht University. In 2000, he joined the System and Network Engineering group at the University of Amsterdam, where he is currently active in the European Phosphorus project.

Karst Koymans is assistant professor in the Informatics Institute of the Universiteit van Amsterdam. He likes educating students and is director of the master program in System and Network Engineering. He teaches Classical Internet Applications and InterNetworking and Routing.

Jeroen van der Ham received his M.Sc. degree in cognitive artificial intelligence from Utrecht University in 2002, and the M.Sc. degree in system and network engineering from the University of Amsterdam in 2004. In 2004, he joined the System and Network Engineering group at the University of Amsterdam, where he is currently pursuing a Ph.D. degree. His primary research interests are network topology representations, the network control plane, and its multi-domain aspects.

Paola Grosso received her Ph.D. in Physics from the University of Turin – Italy. She worked at the Stanford Linear Accelerator Center and since 2004 she is a member of the system and network engineering (SNE) group of the University of Amsterdam. She is leading the group effort in the fields of optical networking and her current research interests are lightpath provisioning and resource scheduling, and semantic network models. Dr. Grosso is chair of the OGF NML–WG, network markup language working group.



Cees de Laat is associate professor in the Informatics Institute at the University of Amsterdam. Current research includes hybrid networking, lambda switching and provisioning, policy-based networking and Authorization, Authentication and Accounting architecture. With SURFnet he implements projects in the GigaPort Research on Networks area. He serves as grid forum steering group (GFSG) infrastructure area director and IETF liaison. He is co-founder of the global lambda integrated facility (GLIF).