Framework for path finding in multi-layer transport networks

Dijkstra, F.

Chapter 4

Multi-Layer Network Model

This chapter is based on *A Multi-Layer Network Model Based on ITU-T G.805* by F. Dijkstra, H.M. Andree, K. Koymans, J.J. van der Ham, P. Grosso, and C.T.A.M. de Laat [a10].

4.1 Introduction

In this chapter we examine abstract representations of multi-layer networks. Network models can help users and applications to understand the complexity of multi-layer networks. In particular models can support path finding, scheduling, fault isolation, and visualisation applications.

Path finding in multi-layer networks requires knowledge about the relation between different network layers (the adaptation).

The most common representation of a network is a graph. As we saw in the previous chapter, a simple graph with devices represented as nodes and links as edges is not able to describe multi-layer networks, so we turn to more elaborate models.

For our network model, we set two goals: first, such a model should be able to explicitly describe multi-layer networks, including the relation between the layers, and second, it should be able to find valid potential connections through a given network. For example, it should be possible to distinguish between the invalid paths and the valid paths in the example networks presented in the previous chapter.

Since we did not find suitable models to describe multi-layer computer networks, we developed a new model, which is technology independent, but layer aware. This network model is based on functional elements as defined in ITU-T Recommendation G.805 [s42], combined with the label concept in GMPLS [s20] and the addition of capability information. 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.

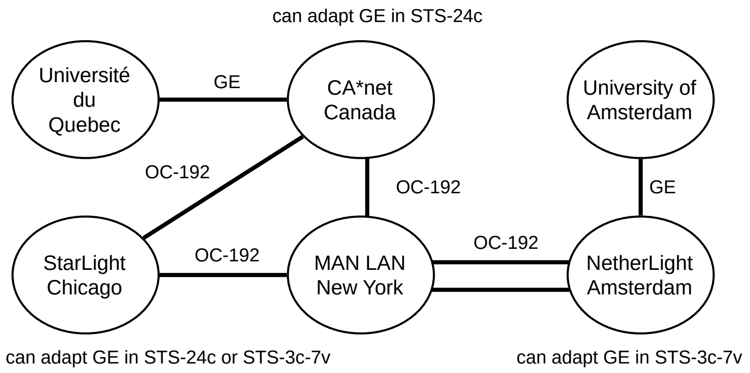


Figure 4.1: *Example of a multi-layer and multi-domain network.*

We will explain the model using the example network in figure 4.1. This is the same example as we have seen in the previous chapter, but we will ignore the availability constraints for now.

Each circle in the picture represents an operational domain. The domains are interconnected by links: the edges in the figure.

The organisation of this chapter is as follows. Section 4.2 starts with the related work. In section 4.3 we will introduce ITU-T Recommendation G.805. The network model is introduced in section 4.4, along with a simple algebra to verify validity of network connections through the network. Section 4.5 demonstrates the usability of the network model by an example network and section 4.6 describes a few possible extensions of this model. Finally, we present related work and conclusions in section 4.7.

4.2 Related work

The great advantage of technology-independent network description is that a path finding algorithm only needs to know about the generic concepts such as ‘layer’ and ‘adaptation’, but not about the specific technologies. It does not need to be tuned or adjusted as new network technologies come along.

Since new technologies and thus incompatibilities will come along as time progresses (see our claim in section 2.3.1), our interest lays in technology independent, multi-layer network description. Table 4.1 shows some of the related work.

	Technology Specific	**Technology Independent**
Single Layer	*most network models*	Graphs
Multi Layer	GMPLS model, CIM, network simulators	ITU-T G.805, G.800

Table 4.1: *Categorization of related work. Single layer technology specific models are not listed, since they are of no interest to us.*

ITU-T Recommendations G.805 and G.800 and graph theory are all technology independent: they can be applied to any technology.

G.805 is the first standard to clearly define the terms *adaptation* and *termination* to describe the relation between different layers. G.805 is based on transport networks and can only be applied to circuit-switched data networks. ITU-T G.809 [s43] extends these definitions to include packet switched networks.

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 [p21]. The physical media layer containing all network components and fibres, 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.

G.800 [s41] is a recent extension of ITU-T G.805 that adds the concepts of domains, symbols and labels. Both G.805 and G.800 provide functional elements to describe the state of a network. Neither provides a description how the state can be changed, and by whom. For path-finding, the information how to change a network is just as crucial as knowing the current state. The capability of a network describes how the state can be changes, while the policy defines who may change the state.

Whereas Graphs, and the functional models in G.805 and G.800 are technology independent, there are many more models that are specific for a limited set of technologies. We describe two of these models that have generated a considerable momentum at the moment of this writing, the model in Generalized Multi-Protocol Label Switching (GMPLS) and the Common Information Model (CIM).

4.2.1 Generalized Multi-Protocol Label Switching

Generalized Multi-Protocol Label Switching (GMPLS) is a set of protocols for routing and signalling in circuit switched networks [s20, s22, s18]. Its path finding properties were already discussed in section 3.2.4, and this section concentrates on its network model only.

GMPLS does not specify a formal network model, but merely specifies parameters for different technologies, as required by a network control plane.

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 [s32]. A proposal for these routing extensions is in draft as of this writing [s31, s33].

4.2.2 Common Information Model

The Common Information Model (CIM) [s2] 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 a very useful tool for describing a (network) configuration in detail. In particular, it makes a great database for access networks, especially if tools like SNMP can automatically generate the data.

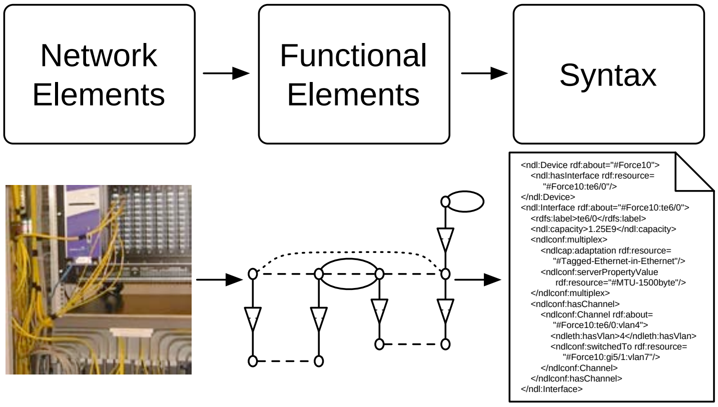
CIM is less suitable for core networks since it can not describe DWDM or TDM networks. CIM is a technology specific model, which makes it less suitable for our purpose: a technology independent model.

4.3 ITU-T G.805 Concepts

This section serves as a short explanation to ITU-T G.805 terminology [s42]. A few concepts are simplified for readability. For example, we do not distinguish between a link, and the transport entity across a link. The section on functional elements and connection partitioning are our own additions.

4.3.1 Functional Elements

The process of creating an abstract description of a network involves two steps, as shown in figure 4.2.



```
<ndl:Device rdf:about="#Force10">
   <ndl:hasInterface rdf:resource=
      "#Force10:te6/0"/>
</ndl:Device>
<ndl:Interface rdf:about="#Force10:te6/0">
   <rdfs:label>te6/0</rdfs:label>
   <ndl:capacity>1.25E9</ndl:capacity>
   <ndlconf:multiplex>
      <ndlcap:adaptation rdf:resource=
         "#Tagged-Ethernet-in-Ethernet"/>
      <ndlconf:serverPropertyValue
          rdf:resource="#MTU-1500byte"/>
   </ndlconf:multiplex>
   <ndlconf:hasChannel>
      <ndlconf:Channel rdf:about=
         "#Force10:te6/0:vlan4">
         <ndleth:hasVlan>4</ndleth:hasVlan>
         <ndlconf:switchedTo rdf:resource=
            "#Force10:gi5/1:vlan7"/>
      </ndlconf:Channel>
   </ndlconf:hasChannel>
</ndl:Interface>
```

Figure 4.2: *The two steps required to create an abstract description of a network.*

The first step consists of the creation of an abstract representation of the physical network elements. Individual components in the abstract representation are so-called functional elements, like *device* and *interface*. The mapping of network elements to functional elements is called information modelling [s17]. A second step is the mapping of the functional elements to a certain syntax. In this chapter, we only examine the first step, the modelling.

The distinction between model and syntax is important for interoperability. If two control planes use a different syntax, but the same model it is straightforward to translate between the to syntaxes. It is hard, and sometimes impossible without making assumptions, to translate between two different models.

ITU-T G.805 provides a set of generic functional elements, without actually specifying this mapping between network elements and functional elements. Neither does it provide a syntax to describe the functional elements.

The functional elements defined by ITU-T G.805 allow one to describe a circuit switched network connection through multiple layers, or a network at a single layer. In addition, ITU-T G.809 [s43] allows the same for packet switched network connections. However, in our view, G.809 tries to map the terminology of circuit switched networks to packet switched networks, ignoring important characteristics of packet switched networks like packet sizes or buffer sizes. We mostly ignore packet switched networks in this chapter.

4.3.2 Connection Point and Layer

ITU-T G.805 defines a *connection point* as a source and sink for data transport. A good way to think about it is as a hop or (virtual) interface on a network connection. One physical interface can consist of multiple logical interfaces. For instance one for each distinguishable data flow.

A *layer* is defined as the set 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. So each connection point resides at one specific layer.

4.3.3 Connections

Informally, a *connection point* can be thought of as a vertex in a graph, and a *link connection* as an edge in the graph. A *tandem connection* is a series of contiguous link connections (a *path* in graph theory), and a *network connection* is a tandem connection between two connection points where the connection is terminated for that layer: an end-to-end connection on a certain layer.

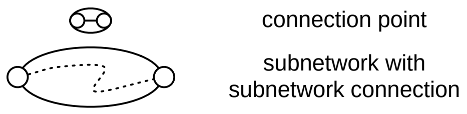
More formally G.805 defines a link connection as "a *transport entity* that transfers information between parts across a link". A network connection is defined as "a series of contiguous link connections and/or subnetwork connections between *termination connection points*".

The termination connection point in the previous definition means that the network connection is an end-to-end connection on that layer.

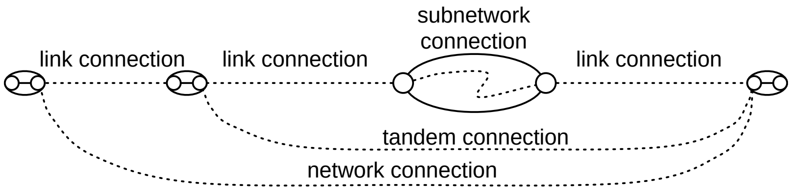
Subnetworks can represent parts of the network *at a single layer*. In general, subnetworks may be partitioned into smaller subnetworks interconnected by link connections. The minimal subnetwork in G.805 is called a *matrix*. A connection through a subnetwork is called a *subnetwork connection*. An indivisible subnetwork connection is called a *matrix connection*. As subnetworks and matrices are defined at a single layer, subnetwork connections and matrix connections can only be described at a single layer as well.

We will later use subnetworks to represent network devices.

The diagrammatic conventions are depicted in figures 4.3a and 4.3b.



(a) *connection point and subnetwork connection*



(b) *link connection, tandem connection and network connection*

Figure 4.3: *Graphical representations of functional elements.*

4.3.4 Adaptation and Termination

If we want to send data belonging to layer X over a different layer Y, the data needs to be transformed. This transformation is defined by an *adaptation function.*

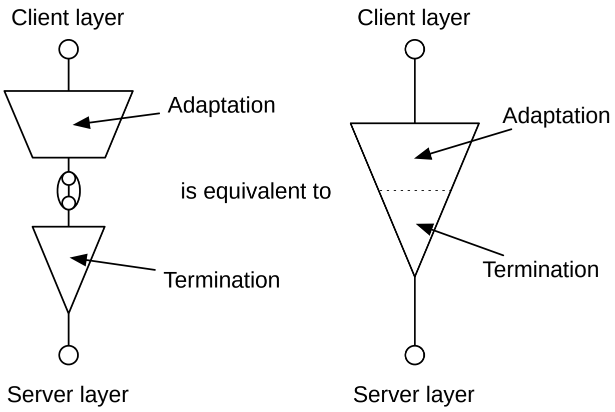


Figure 4.4: *Graphical representation of the adaptation and termination functions. In this chapter we will use the simplified representation shown at the right.*

ITU-T G.805 defines the *adaptation function* and the *termination function.*

The adaptation function defines how data belonging to a client layer (the 'higher' layer) network is embedded into data of a server layer (the 'lower' layer) network. The termination function adds monitoring information to the server layer network connection, taking care of a reliable data transmission.

The trail termination function is defined by G.805 as *a transport processing function that consists of the trail termination source where monitoring information is added and the trail termination sink which removes the monitoring information*. So in short, a termination function just adds *monitoring information*. For example, a termination function adds a checksum field to each data packet.

A graphical representation of these functional elements is depicted in figure 4.4. The adaptation function is visualised by the upper part of the triangle and the termination function is visualised by the lower part of the triangle.

Adaptation and termination have analogies in the real world. For example, we want to send five pairs of wooden shoes from Amsterdam to Quebec. Rather than sending them as-is, we wrap them in a box for shipping. This is the adaptation. However, a box can only contain 3 pairs of shoes, so we use two boxes and mark them as 'box 1/2' and 'box 2/2'. This allows the recipient to verify that the shipment arrived complete and unmodified. This is the termination.

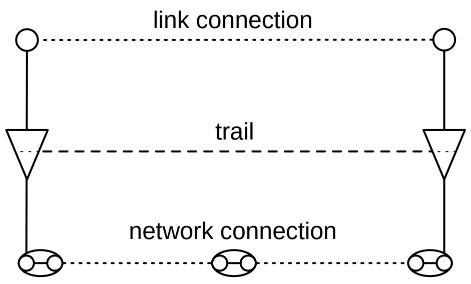


Figure 4.5: *A network connections on the server layer with two adaptations functions yields a link connection on the client layer.*

Figure 4.5 shows how adaptations can be used to build network connections on a higher layer, the client layer, using network connections on an underlying layer, the server layer. According to the G.805 definition, a link connection "represents a pair of adaptation functions and a trail in the server layer network", where the trail is a termination network connection. So if there is a network connection on a (lower) server layer network, and both ends have the same termination and adaptation functions, then there is a link connection at the client layer above.

The embedding of data of one layer into another is a recursive process. For example, the OSI model [s53] defines 7 network layers where the data of each layer is embedded in the layer directly underneath. We call such a sequence of adaptation in adaptation an *adaptation stack*.

4.3.5 Multiplexing

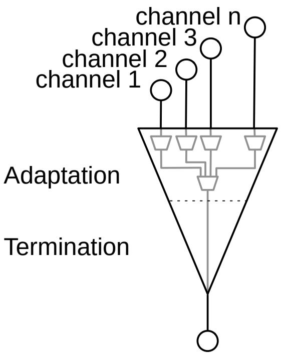
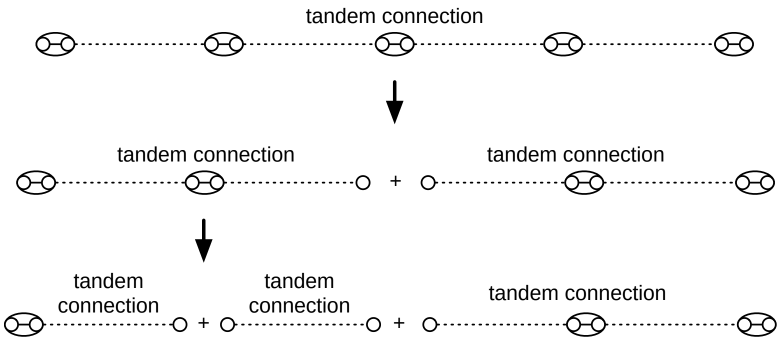


Figure 4.6: *Implementing multiple connections over a network link is equivalent with multiplexing at the adaptation function.*

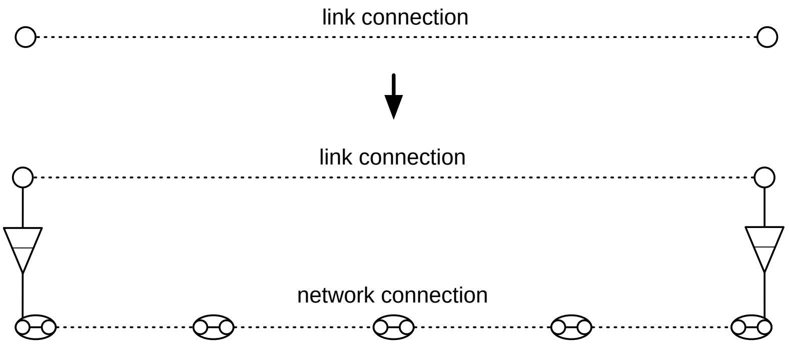
Channeling, implementing multiple connections over a network link, is equivalent to multiplexing at the adaptation function in G.805. As figure 4.6 shows, the adaptation function may consist of specific processes for each channel at the client layer and one common process that converts these adapted client layer channels to the server layer. Each logical channel interface is represented as a connection point on the client layer, while there is only one (termination) connection point at the server layer.

4.3.6 Connection Partitioning

Figure 4.7 repeats the two different ways how a connection can be partitioned. A tandem connection can be split in multiple tandem connections, up to the smallest unit, a link connection. This is a partitioning on a single layer, and we refer to it as horizontal partitioning. Horizontal partitioning is ambiguous. For example, the connection $A-B-C-D$ can either be split in $A-B-C$



(a) *Horizontal partitioning of a tandem connection*



(b) *Vertical partitioning of a link connection*

Figure 4.7: *Two partitionings of a connection in smaller parts.*

and $C - D$ or in $A - B$ and $B - C - D$. The chosen partitioning depends on organisational reasons like the boundary of the operational domains.

The other partitioning is of a link connection on a client layer into a network connection on another layer. This is partitioning between different layers, and we refer to it as vertical partitioning. It is determined by the actual technology and therefore unambiguous, not driven by organisational decisions.

4.4 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 the label concept in GMPLS. In addition, we model the capability, thus how the state of a network can be changed.

4.4.1 Mapping to Functional Elements

Table 4.2 shows an overview of our mapping from real-life network elements (for instance links, and devices with interfaces) to G.805 functional elements. We model the switching core of a network device as a subnetwork. 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.

Network Element	**Functional Elements**
Domain	Subnetwork(s)
Device	Matrix (Subnetwork)
Interface	Connection point(s) and adaptation function(s)
Link	Link connection

Table 4.2: *Mapping of network elements to G.805 functional elements*

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 interface at the fibre layer, 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 fibre is modelled as a link connection at the fibre 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.

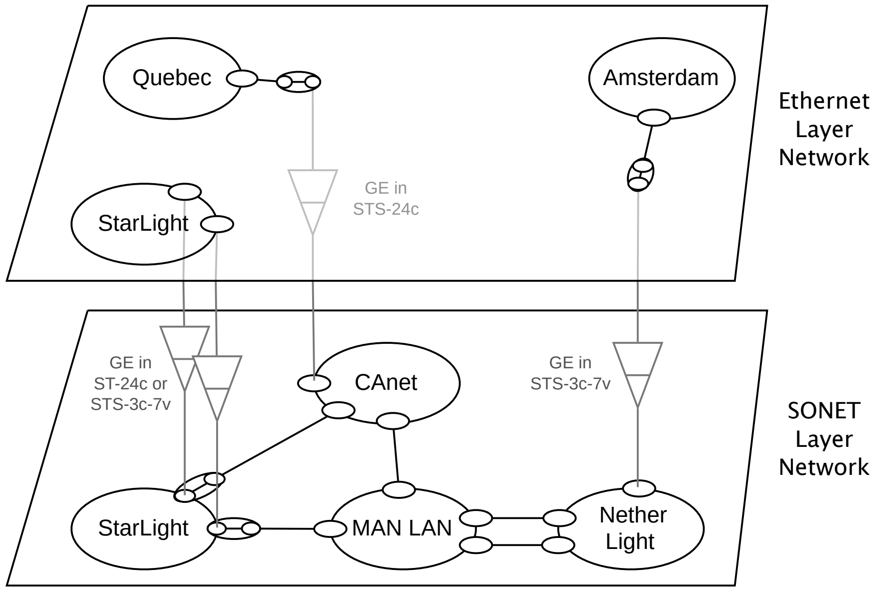


Figure 4.8: *The network of figure 4.1, modelled as functional elements.*

Figure 4.8 shows an example network description using functional elements. The network is a slightly simplified version[1] of the network described in figure 4.1. Unlike figure 3.7 in the previous chapter, we explicitly modelled the adaptation functions. The two layers are separated vertically, while the different domains are separated horizontally. For example CAnet 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

[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.

to dynamically switch between these two adaptation functions at the same interface. It is possible to model this as two adaptation functions with a multi-point 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 chapter into a data model. We will come back to these decisions in the next chapter.

4.4.2 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

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

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 $A : T \to U$, and refer to both the data as well as the layers as T and U.

Except for section 4.4.6, we will simply refer to the combined adaptation and termination function as the adaptation. This implies that A is noncommutative.

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

$$\forall (A : T \to U)\ \exists (D : U \to T) : D \circ A = id : T \to T \tag{4.1}$$

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 $A_{cp1u}^{cp1t} : T \to U$. The corresponding de-adaptation function will be named $D_{cp1t}^{cp1u} : U \to T$, or equivalently, $(A_{cp1u}^{cp1t})^{-1} : U \to T$.

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

Figure 4.9 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 applicable layers. For instance for interface $if1$, as *cp1f* at the fibre layer, *cp1e* on the Ethernet layer and *cp1i* at the IP layer.

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

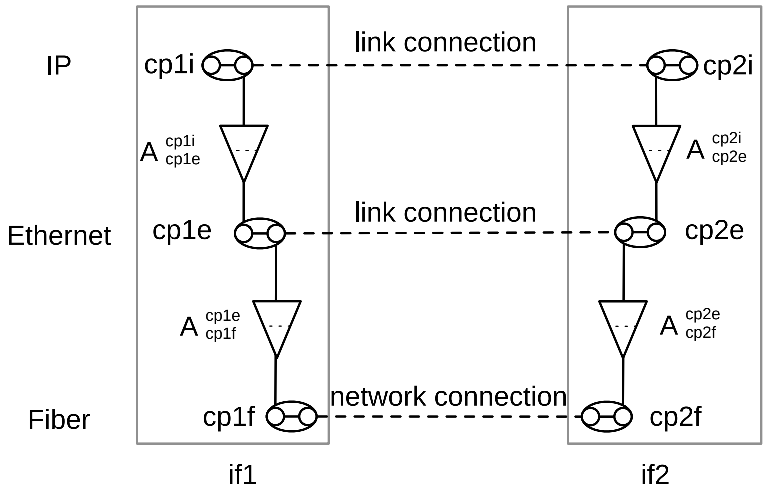


Figure 4.9: *Example of a multi-layer network connection. Interfaces $if1$ and $if2$ are modelled as connection points at all three layers. The relation between the connection points is defined by the adaptation and termination functions.*

4.4.3 Channel Labels

In 4.4.1 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 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 $A_{cp1u}^{cp1t_1;cp1t_2;cp1t_3;\ldots;cp1t_n} : T^n \to U$ and $A_{cp2u}^{cp2t_1;cp2t_2;cp2t_3;\ldots;cp2t_n} : T^n \to U$ in figure 4.10. 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.

From the logic of section 4.3.4 it follows that since there is a network con-

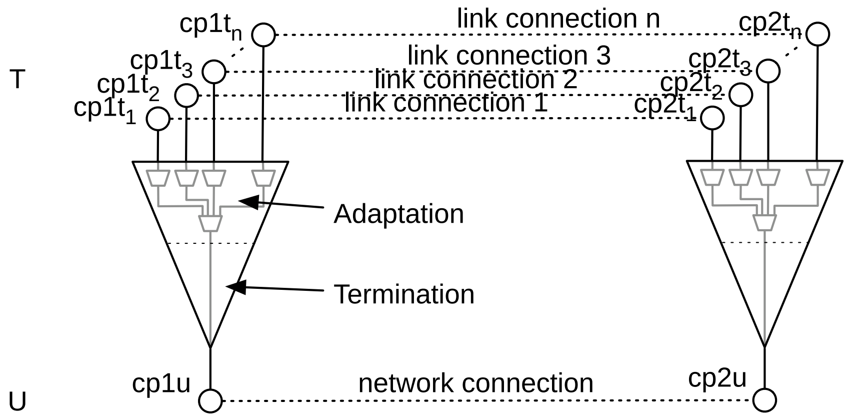


Figure 4.10: *Channels correspond with multiple link connections at the client layer over one link connection at the server layer.*

nection 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 [s20].

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 would be STS timeslots, IEEE 802.1Q (VLAN) tags or wavelengths.

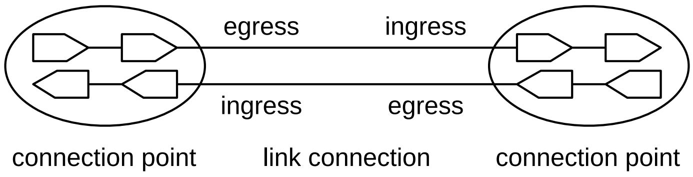


Figure 4.11: *The ingress and egress part of an connection point with respect to a link connection.*

Figure 4.11 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.

We define the functions:

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

$$Lb(cp) := Lb_{out}(cp) = Lb_{in}(cp) \tag{4.2}$$

If an interface does not have a particular label, we consider it to have an "empty" label, ϵ. For all other purposes, we consider ϵ just to be a regular label.

4.4.4 Capability Model

The functional elements of G.805 and G.800 only allow a description of the state of a network at a certain point in time. The capability describes how the state of a network can be changed.

In our model, there are only two parameters that can change:

- the labels of a connection point and
- the subnetwork connections within a subnetwork.

A label can only be changed to a predefined value. Each connection point has an egress and an ingress labelset, $Ls_{out}(cp)$ and $Ls_{in}(cp)$ respectively. The label of a connection point *cp* can be changed to any value $Lb(cp) \in Ls(cp)$.

A subnetwork is a set of connection points. A subnetwork connection exists between two connection points in this set. Rather than allowing subnetwork connections between all connection points in a subnetwork, we place a restriction on it. Subnetworks can have either or both of the *switching* and *swapping* capability. The switching capability refers to a switch matrix that can forward data as long as the label is the same. For example a WDM device without wavelength conversion, or an Ethernet switch that can not convert between VLAN tags. The swapping capability refers to a switch matrix that can forward data while changing the label. For example a WDM device with wavelength conversion capabilities or an SDH device with virtual concatenation (VCAT) [s47] and Link Capacity Adjustment Schema (LCAS) [s46] support that can forward data from one timeslot to another timeslot without constraints.

4.4.5 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:

- $L(\mathit{cp1}, \mathit{cp2}) \iff$ a directional Link from *cp1* to *cp2* exists;
- $\mathit{SNC}(\mathit{cp1}, \mathit{cp2}) \iff$ a directional Subnetwork Connection from *cp1* to *cp2* exists;
- $\mathit{LC}(\mathit{cp1}, \mathit{cp2}) \iff$ a directional Link Connection from *cp1* to *cp2* exists;
- $\mathit{TC}(\mathit{cp1}, \mathit{cp2}) \iff$ a directional Tandem Connection from *cp1* to *cp2* exists.

We postulate a network $N = (\mathit{CP}, L, \mathit{SN}, A)$ as a set of connection points CP, physical links L, subnetworks SN, and adaptations A. The network configuration $C = (\mathit{LB}, \mathit{SC})$ is a set of labels LB, and subnetwork connections SC. Given these basic premises N and C, 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:

$$TC(cp1, cp2) = \begin{cases} LC(cp1, cp2) \vee & (4.3a) \\ SNC(cp1, cp2) \vee & (4.3b) \\ \exists cp3 : TC(cp1, cp3) \wedge TC(cp3, cp2) & (4.3c) \end{cases}$$

and

$$LC(cp1, cp2) = \begin{cases} L(cp1, cp2) \vee & (4.4a) \\ \exists cp3, cp4, T, U, A^{cp1}_{cp3}, D^{cp4}_{cp2} : & \\ \quad TC(cp3, cp4) \wedge & \\ \quad A^{cp1}_{cp3} : T \to U \wedge & \\ \quad D^{cp4}_{cp2} : U \to T \wedge & (4.4b) \\ \quad D^{cp4}_{cp2} \circ A^{cp1}_{cp3} = Id : T \to T \wedge & \\ \quad Lb_{out}(cp1) = Lb_{in}(cp2) \wedge & \\ \quad Lb_{in}(cp1) = Lb_{out}(cp2) & \end{cases}$$

For simplicity, we will now restrict ourselves to bidirectional connections. Thus:

$$L(cp1, cp2) \to L(cp2, cp1) \qquad (4.5a)$$
$$LC(cp1, cp2) \to LC(cp2, cp1) \qquad (4.5b)$$
$$TC(cp1, cp2) \to TC(cp2, cp1) \qquad (4.5c)$$
$$SNC(cp1, cp2) \to SNC(cp2, cp1) \qquad (4.5d)$$

and even:

$$(A^{cp1t}_{cp1u} : T \to U) \to (D^{cp1u}_{cp1t} : U \to T) \qquad (4.6)$$

with

$$D^{cp1u}_{cp1t} \circ A^{cp1t}_{cp1u} = Id : T \to T \qquad (4.7)$$

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 figure 4.12 to illustrate the concepts. This figure shows a network N_{ex} with links, five link connections, nine tandem connections and one subnetwork connection in total.

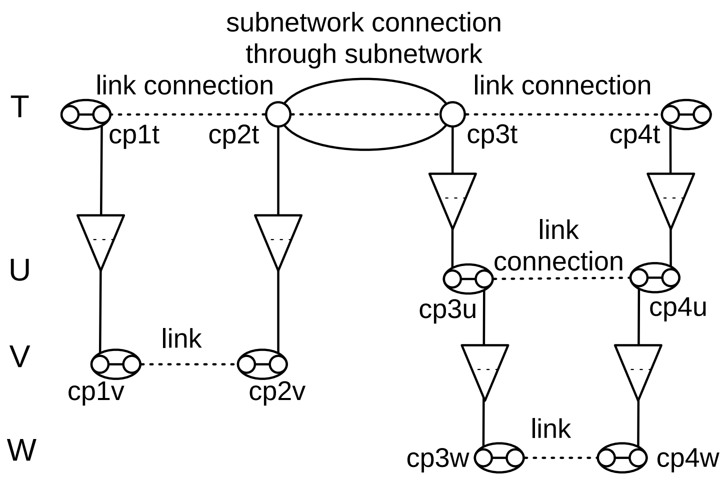


Figure 4.12: *example of a valid network connection. A valid tandem connection consisting of two link connections and a matrix connection.*

Formally, we postulate the network $N_{ex} = (CP_{ex}, L_{ex}, SN_{ex}, A_{ex})$ as the sets:
$CP_{ex} = \{cp1t, cp2t, cp3t, cp4t, cp1v, cp2v, cp3u, cp4u, cp3w, cp4w\}$,
$L_{ex} = \{L(cp1v, cp2v), L(cp3w, cp4w)\}$,
$SN_{ex} = \{\{cp2t, cp3t\}\}$, and
$A_{ex} = \{A^{cp1t}_{cp1v}, A^{cp2t}_{cp2v}, A^{cp3t}_{cp3u}, A^{cp4t}_{cp4u}, A^{cp3u}_{cp3w}, A^{cp4u}_{cp4w}\}$.

In this network, $A^{cp1t}_{cp1v} = A^{cp2t}_{cp2v}$, $A^{cp3t}_{cp3u} = A^{cp4t}_{cp4u}$, and $A^{cp3u}_{cp3w} = A^{cp4u}_{cp4w}$.

The configuration $C_{ex} = (LB_{ex}, SC_{ex})$ of network N_{ex} are the sets of labels and subnetwork connections: $LB_{ex} = \{\forall_{cp \in CP_{ex}} : Lb(cp)$ with $Lb(cp) = \epsilon$ (no explicit labels defined)$\}$, and
$SC_{ex} = \{SNC(cp2t, cp3t)\}$.

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

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

$$D^{cp2v}_{cp2t} \circ A^{cp1t}_{cp1v} = Id : T \to T \tag{4.8}$$

with $D^{cp2v}_{cp2t} = (A^{cp2t}_{cp2v})^{-1}$. Therefore, from equation 4.4 we must conclude that

$LC(cp1t, cp2t)$. In G.805 terminology, the *adaptation source* $cp1t$ and the *adaptation sink* $cp1v$ are *paired*.

Similarly, $LC(cp3u, cp4u)$, and therefore $TC(cp3u, cp4u)$ hold because $TC(cp3w, cp4w)$ and $D(cp4w, cp4u) \circ A(cp3u, cp3w) = Id : U \rightarrow U$, and $LC(cp3t, cp4t)$ holds because $TC(cp3u, cp4u)$ and $D(cp4u, cp4t) \circ A(cp3t, cp3u) = Id : T \rightarrow T$.

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

4.4.6 Well Typed Adaptations

So far, we combined the adaptation and termination function.

We did so to make our definition of $LC(cp1, cp2)$ in equation 4.4 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 figure 4.13 for a *well-typed*, but invalid link connection between $cp1t$ and $cp2t$. An example of such an invalid link connection could be if A_{cp1v}^{cp1t} adds a header to a packet, and A_{cp1w}^{cp1v} adds a tail to the result. Then, D_{cp2u}^{cp2w} first removes the header and finally D_{cp2t}^{cp2u} 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_{cp1v}^{cp1t} and D_{cp2t}^{cp2u}.

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, but assume that each adaptation function is followed by a termination function.

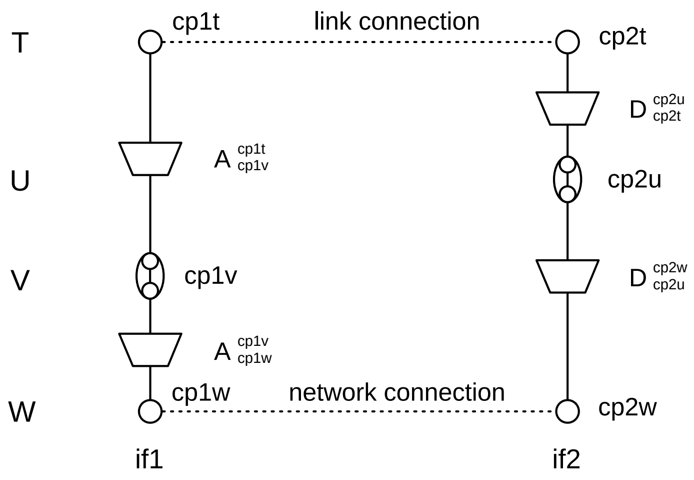


Figure 4.13: *Example of a well typed, but invalid connection. U and V are different layers.*

4.5 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 4.4 can be used to make this explicit.

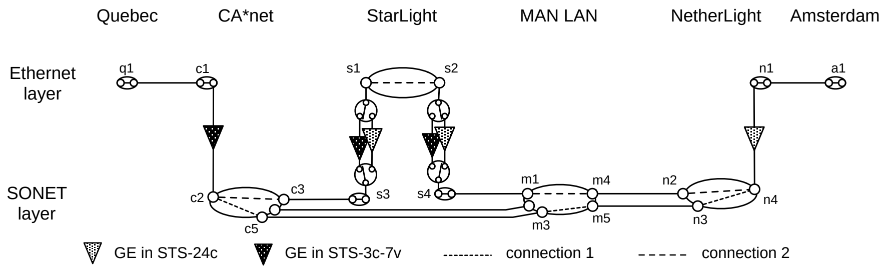


Figure 4.14: *A network representation of the network of figure 4.1, using functional elements. Dark-gray adaptation functions represent adaptation of Gigabit/second Ethernet (GE) over STS-24c, while light-gray adaptation functions represent GE over STS-3c-7v. StarLight is capable of either adaptation function.*

Figure 4.14 gives a representation the network N_g of figure 4.1 as functional elements, using the mapping of table 4.2.

This network is identified by $N_g = N_g(CP_g, E_g, SN_g, A_g))$, with:
$CP_g = \{q1, c1, c2, c3, c4, c5, s1, s2, s3, s4, m1, m2, m3, m4, m5, n1, n2, n3, n4, a1\}$,
$E_g = \{L(q1, c1), L(c3, s3), L(c4, m2), L(c5, m3), L(s4, m1), L(m4, n2), L(m5, n3), L(n1, a1)\}$,
$SN_g = \{\{c2, c3, c5\}.\{s1, s2\}, \{m1, m2, m3, m4, m5\}, \{n2, n3, n4\}\}$, and
$A_g = A^{c1}_{c2}, A^{s1}_{s3}, A^{s2}_{s4}, A^{n1}_{n4}\}$ 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 StarLight, MAN LAN and NetherLight. This would result in connection 1 in the figure 4.14. Formally, connection 1 is dataflow through the network elements $[L(q1, c1), A^{c1}_{c2}, SNC(c2, c5), L(c5, m3), SNC(m3, m5), L(m5, n3), SNC(n3, n4), D^{n4}_{n1}, L(n1, a1)]$ and is identified by the subset $C1 = \{SNC(c2, c5), SNC(m3, m5), SNC(n3, n4)\}$ of the network configuration.

Given the above sets, we can use the definitions of section 4.4.5 to derive the valid link connections and tandem connections in this network.

$$SNC(c2, c5) \xrightarrow{equation\ 4.3b} TC(c2, c5) \tag{4.9}$$

$$L(c5, m3) \xrightarrow{equation\ 4.4a} LC(c5, m3) \xrightarrow{equation\ 4.3a} TC(c5, m3) \tag{4.10}$$

$$SNC(m3, m5) \xrightarrow{equation\ 4.3b} TC(m3, m5) \tag{4.11}$$

$$L(m5, n3) \xrightarrow{equation\ 4.4a} LC(m5, n3) \xrightarrow{equation\ 4.3a} TC(m5, n3) \tag{4.12}$$

$$SNC(n3, n4) \xrightarrow{equation\ 4.3b} TC(n3, n4) \tag{4.13}$$

$$TC(c2, c5) \wedge TC(c5, m3) \wedge TC(m3, m5) \wedge TC(m5, n3) \wedge TC(n3, n4) \xrightarrow{equation\ 4.3c\ recursive} TC(c2, n4) \tag{4.14}$$

However, from A^{c1}_{c2}, $TC(c2, n4)$, D^{n4}_{n1} does *not* follow $LC(c1, n1)$ since $D^{n4}_{n1} \circ A^{c1}_{c2} = STS3c7v^{-1} \circ STS24c \neq Id : Ethernet \rightarrow Ethernet$. Therefore, connection 1 does not lead to a valid tandem connection from Quebec to Amsterdam, given these links and subnetwork connections:

$$N, C1 \nvdash TC(q1, a1) \tag{4.15}$$

StarLight is capable of supporting either adaptation function. This is modelled in figure 4.14 using two multi-point connection points (MPCP). A_{s3}^{s1} is either equal to $STS24c$, or to $STS3c7v$.

Let's now consider connection 2, identified by the subset $C2 = \{SNC(c2, c3)$, $SNC(s1, s2)$, $SNC(m1, m4)$, $SNC(n2, n4)\}$ of the network configuration, $A_{s3}^{s1} = STS24c$ and $A_{s4}^{s2} = STS3c7v$

$$SNC(c2, c3) \xrightarrow{equation\ 4.3b} TC(c2, c3) \tag{4.16}$$

$$L(c3, s3) \xrightarrow{equation\ 4.4a} LC(c3, s3) \xrightarrow{equation\ 4.3a} TC(c3, s3) \tag{4.17}$$

$$TC(c2, c3) \wedge TC(c3, s3) \xrightarrow{equation\ 4.3c} TC(c2, s3) \tag{4.18}$$

$$A_{s3}^{s1} = STS24c \xrightarrow{equation\ 4.6} D_{s1}^{s3} = STS24c^{-1} \tag{4.19}$$

$$TC(c2, s3) \wedge \tag{4.20}$$

$$D_{s1}^{s3} \circ A_{c2}^{c1} = STS24c^{-1} \circ STS24c = Id : \text{Ethernet} \to \text{Ethernet} \wedge \tag{4.21}$$

$$Lb_{out}(c1) = \epsilon = Lb_{in}(s3) \wedge \tag{4.22}$$

$$Lb_{in}(c1) = \epsilon = Lb_{out}(s3) \tag{4.23}$$

$$\xrightarrow{equation\ 4.4b} TC(c1, s1) \tag{4.24}$$

Similarly for the connection between $s2$ and $n1$:

$$L(s4, m1) \xrightarrow{equation\ 4.4a} LC(s4, m1) \xrightarrow{equation\ 4.3a} TC(s4, m1) \tag{4.25}$$

$$SNC(m1, m4) \xrightarrow{equation\ 4.3b} TC(m1, m4) \tag{4.26}$$

$$L(m4, n2) \xrightarrow{equation\ 4.4a} LC(m4, n2) \xrightarrow{equation\ 4.3a} TC(m4, n2) \tag{4.27}$$

$$SNC(n2, n4) \xrightarrow{equation\ 4.3b} TC(n2, n4) \tag{4.28}$$

$$TC(s4, m1) \wedge TC(m1, m4) \wedge TC(m4, n2) \wedge TC(n2, n4) \xrightarrow{equation\ 4.3c} TC(s4, n4) \tag{4.29}$$

$$A_{n4}^{n1} = STS3c7v \xrightarrow{equation\ 4.6} D_{n1}^{n4} = STS3c7v^{-1} \tag{4.30}$$

$$TC(s4, n4) \wedge \tag{4.31}$$
$$D_{n1}^{n4} \circ A_{s4}^{s2} = STS3c7v^{-1} \circ STS3c7v = Id : \text{Ethernet} \rightarrow \text{Ethernet} \wedge \tag{4.32}$$
$$Lb_{out}(s2) = \epsilon = Lb_{in}(n1) \wedge \tag{4.33}$$
$$Lb_{in}(s2) = \epsilon = Lb_{out}(n1) \tag{4.34}$$
$$\xrightarrow{equation\ 4.4b} LC(s2, n1) \tag{4.35}$$

We can now combine the derived truth statements $LC(c1, s1)$ and $LC(s2, n1)$ with other statements in the network description N_g and it's configuration $C2$:

$$L(q1, c1) \xrightarrow{equation\ 4.4a} LC(q1, c1) \xrightarrow{equation\ 4.3a} TC(q1, c1) \tag{4.36}$$
$$LC(c1, s1) \xrightarrow{equation\ 4.3a} TC(c1, s1) \tag{4.37}$$
$$SNC(s1, s2) \xrightarrow{equation\ 4.3b} TC(s1, s2) \tag{4.38}$$
$$LC(s2, n1) \xrightarrow{equation\ 4.3a} TC(s2, n1) \tag{4.39}$$
$$L(n1, a1) \xrightarrow{equation\ 4.4a} LC(n1, a1) \xrightarrow{equation\ 4.3a} TC(n1, a1) \tag{4.40}$$
$$\tag{4.41}$$

$$TC(q1, c1) \wedge TC(c1, s1) \wedge TC(s1, s2) \wedge TC(s2, n1) \wedge TC(n1, a1)$$
$$\xrightarrow{equation\ 4.3c} TC(q1, a1) \tag{4.42}$$

Thus $TC(q1, a1)$ is 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:

$$N, C2 \vdash TC(q1, a1) \tag{4.43}$$

4.6 Extensions of the Model

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

One of our goals is to describe actual networks in a technology-independent way. In order to implement some of the extensions mentioned in this section, it is likely that some of the (mathematical) simplicity of the current model will be lost while gaining a model able to describe additional technologies, or additional logic used in some technologies only. Care should be taken to retain the basic logic.

4.6.1 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), 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).

It is technically possible to model this as a few thousand different adaptation functions, but this is not efficient. An alternative is to define the layer properties, and extend the model by defining when two values cause an incompatibility. This later approach is used by the stitching framework in GÉANT2 [t9]. This stitching framework defines a "method of logic" for each property (same, different, comparable, overlapping, open, different, min, or sum).

4.6.2 Inverse Multiplexing

Both G.805 as well as our model support inverse multiplexing: the adaptation of one data stream in multiple channels. Ethernet in STS channels, as described in examples in this chapter, is an instance of inverse multiplexing. The model as presented in this chapter is limited to a single underlying network connection. For inverse multiplexing, $cp3$, $cp4$ in equation 4.4 needs to be changed to $cp3_1, \ldots, cp3_n, cp4_1, \ldots, cp4_n$, and $TC(cp3, cp4)$ must be changed to $\forall i \in [1, \ldots, n] : TC(cp3_i, cp4_i)$.

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 fibre, 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.

It is simple to prove that there are at most two (de-)adaptation stacks for valid descriptions of an interface: one de-adaptation stack and one adaptation stack. Two adaptation stacks can be collapsed to a single adaptation stack, and two de-adaptation stacks can be combined into a single de-adaptation stack. Furthermore, a single adaptations followed by the inverse de-adaptation function cancel each other out, and for a (de-)adaptation stack this process can be repeated. Thus an adaptation stack followed by a de-adaptations stack can be collapsed till one of the stacks is fully cancelled out, and only a single adaptation or a single de-adaptation stack remains. Unless of course, an adaptation was followed by a different de-adaptation, which would leave us with an invalid (non-working) interface description.

4.6.3 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.

4.6.4 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 4.4.5.

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 subsection 4.6.1 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.

4.6.5 Uniqueness of Layers

ITU-T G.805 defines a *layer* in section 4.3.2 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 figure 4.14, where the interfaces at StarLight supported two adaptation functions.

4.6.6 Tunnels

Because there is no ordering in the layers in the G.805 model, it is entirely possible to model layer A over layer B over layer A, effectively describing network tunnels.

4.6.7 Uniqueness of Adaptations

One of our goals is to be able to describe potential incompatibilities we like to expose to path finding algorithms. We already mentioned in chapters 2 and 3 that technologies and incompatibilities change over time.

The progress in technology makes that potential incompatibilities come and go. 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 colours are deployed, this might lead to incompatibilities, so it has to be described. However, as soon as every device is able do colour conversion on the fly, the incompatibility would again disappear.

At first Ethernet over an optical fibre may be described as an adaptation of Ethernet over fibre. However, later on, the same adaptation may be described as Ethernet over a wavelength over a fibre. A mechanism is needed to described that these two representations are in fact the same.

4.7 Conclusion

At the beginning of this chapter we 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. We satisfied the second goal with a simple algebra, without relying on complex path constraints.

Our mapping from network elements to functional elements is based on previous work in the ITU-T G.805, G.800 and GMPLS standards. Our contribution is the use of the subnetwork matrix to represent the switching capability of a device, the integration of the label concept, the distinction between the switching and swapping capability. Furthermore, we supplemented the model with an algebra, and confirmed the mapping with two example networks.

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 4.5, 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.

A technology-independent network model, as we defined in this chapter, means that a path finding algorithms only needs to know about the generic concepts such as ‘layer’, ‘label’ and ‘adaptation’, but not about the specific technologies. The advantage is that path finding algorithms does need to be tuned or adjusted as new network technologies come along.