NCP+: an integrated network and IT control plane for cloud computing

Jens Buysse[a], Marc De Leenheer[c], Luis Miguel Contreras[d], José Ignacio Aznar[d], Juan Rodriguez Martinez[d], Giada Landi[e], Chris Develder[b]

[a] *University College Ghent, Faculty of Business and Information Management, Valentin Vaerwyckweg 1, 9000 Gent, Belgium*
[b] *Ghent University, Gaston Crommenlaan 8 bus 201, 9050 Gent, Belgium*
[c] *ON.LAB, 1000 El Camino Real 100, Menlo Park, CA 94025, USA*
[d] *Telefonica, Telefonica I+D, Don Ramón de la Cruz 82,84. 28006, Madrid, Spain.*
[e] *Nextworks, Via Livornese, 1027, 56122 Pisa, Italy*

Abstract

Cloud computing, building on the idea of "computation as a public utility" is made possible by the increased network capabilities in terms of bandwidth and reduced latency. Today, the cloud paradigm today sees adoption in many businesses, given its advantages not only from a customer point of view (e.g., universal access to the same applications across all company branches), but also from the application provider and network operator perspective (e.g., software updates no longer need to be distributed). To be able to offer cloud computing services efficiently, service providers need not just an infrastructure comprising both network and IT resources, but especially a control system that is able to orchestrate such integrated network and IT services. This paper offers a new proposal for such a system: an enhanced network control plane, the NCP+, which is based on a GMPLS control plane with a Hierarchical Path Computation Element (PCE) architecture able to jointly make network routing and IT server provisioning decisions. Indeed, in the assumed cloud paradigm, a user generally does not care what exact server the offered service is using, as long as its service requirements are met: the anycast principle applies. The paper discusses (i) the architecture of the NCP+, (ii) two IT-aware aggregation mechanisms to be used in the hierarchical PCE approach, and (iii) routing and scheduling algorithms for those aggregation mechanisms. We conclude this work with a thorough simulation analysis of the aggregation and routing/scheduling policies showing that *Full Mesh* aggregation where the domain topology is represented by a complete graph, although less scalable in terms of computation time, is able to provision efficiently using the proposed load balancing routing and scheduling policy. However, for a scenario with stringent IT requirements, *Star* could be used in parallel for scenarios where end-to-end setup times are important.

Keywords: WDM Networks, PCE, GMPLS, Cloud Computing

1. Introduction

The adoption of cloud computing, as a manifestation of the "utility computing" idea suggested back in 1961 by J. McCarthy, can be seen as a next step in an evolution to gradually push functionality further into the network, enabled by the evolution of, e.g., optical networking (which meets high bandwidth and low latency requirements of applications varying from consumer-oriented, over more stringent business-driven to scientific cases) [1]. This shift to network-based solutions not only has benefits for users (no local configurations, automatic backups, etc.) but is also interesting for the network and service provider: updates and improvements are simpler because instead of pushing software updates to the users, the service provider only needs to update the software copy in the data center. Moreover, these services can run at a low cost as IT resources can be shared among many users.

With cloud computing, the network interactions evolved from point-to-point to interworking of many distributed components. Moreover, some services may involve multiple geographically dispersed data centers (e.g., replicas to improve throughput and reduce latency). This all implies the need for the service provider to offer joint provisioning of IT resources at multiple data center sites as well as their interconnection. Isolated and statically interconnected data centers are evolving towards warehouse scale computing data centers [2] where network connectivity cannot rely on traditional transport technologies [3]. Optical networks with Wavelength Division Multiplexing (WDM) are the ideal candidate for the required low-latency and high-bandwidth network connectivity.

Traditionally, a network provider aiming to provide cloud services is required to make substantial investments in the integration of the variety of platforms operating over the heterogeneous resources within his management system. Thus, there is a need for an automated and combined control mechanism for IT and network resources to ensure service continuity, efficient use of resources, service performance guarantees, scalability and manageability. These goals can be achieved either by reusing and combining existing separate IT and network management systems, or by developing new joint platforms. However, the former solution still implies substantial human intervention, and the efficiency of the whole system is bounded by the limits of the separate components [4]. Instead, deploying an integrated control plane would enable both scalability and efficient operation over the network and IT infrastructure, which is what this paper presents.

Note that in a cloud context, it is common to have multiple data center sites offering the same functionality (cf. aforementioned replication), which implies the flexibility to choose the most suitable resource(s). This model amounts to anycast routing, solving the problem of selecting a route to one target destination chosen from a set of nodes, as opposed to unicast where the source-destination pair is known in advance. Anycast has been shown to be beneficial for the overall network performance, either in terms of network survivability [5, 6], impairment avoidance [7], energy minimization [8, 9] or blocking probability reduction [10]. Consequently, the control mechanism of an integrated

network and IT infrastructure to select both the data center (i.e., the IT end point, without initially specifying its location) and the network resources (the path to the chosen IT end point) becomes critical for guaranteeing an efficient operation of the entire infrastructure. In this paper we refer to such a service as a Network and IT Provisioning Service (NIPS), where IT capacity is dynamically requested in combination with the network services among the selected sites, with a network capacity tailored to the real-time application requirements.

This paper introduces a set of extensions to a Generalized Multi-Protocol Label Switching (GMPLS)[11] and Hierarchical Path Computation Element (H-PCE)-based [12] network control plane, referred to as NCP+ , to enable anycast path computation for NIPS requests in a multi-domain optical scenario, comprising also the IT resources (i.e., servers). The contribution of this paper is threefold: (i) we propose the enhancement of GMPLS/H-PCE modules to disseminate and process IT resource information in the NCP+, both in terms of routing and signalling functionalities, (ii) discuss the main extensions to the existing Path Computation Element Protocol (PCEP) to disseminate the IT resource information and (iii) propose joint network and IT path computation and topology representation algorithms used by the PCEs and evaluate them in simulation case studies.

The remainder of this paper is structured as follows. In Section 2 we discuss related work, while in Section 3 we introduce the NCP+ architecture, propose new modules and PCEP protocol extensions in order to disseminate IT resource information. In Section 4 we lay out the options for topology representation, routing and resource allocation algorithms. In Section 5 we present our simulation analysis demonstrating the effectiveness of the proposed NCP+ algorithms in terms of service blocking. We summarize our final conclusions in Section 6.

2. Related work

2.1. Converged network and IT control architectures

Prior attempts to an architecture managing and controlling network and IT resources simultaneously, mainly stem from the grid computing world. In grid computing users create applications (jobs) which are scheduled to some server. Many of these jobs also require network bandwidth (large transfers of data) and consequently a network path needs to be reserved between several source-destination pairs. Cloud computing builds on this concept (similar coordination of resources is required), but manifests itself in more commercially oriented scenarios [1]. A key characteristic of cloud computing is its scalability: cloud providers virtualize their resources. This enables them to operate the infrastructure cost-effectively (avoid overprovisioning), to migrate virtual machines to other servers (making relocation possible [5]) and to share resources in a safe way. Working from the bottom up, there are three models for cloud computing: (1) Infrastructure-as-a-Service (IaaS), (2) Platform-as-a-Service (PaaS) and (3) Software-as-a-Service (SaaS)

With IaaS, companies rent the network and IT resources, with pre-loaded operating systems and barely anything else (these resources are sometimes provided as virtualized resources). IaaS users then load their own applications and platforms. In SaaS on the other hand, Cloud providers really offer a complete application that is directly usable by the consumer In between we find PaaS, where consumers rent infrastructure with a development platform which enables them to create SaaS services. Our integrated network control plane has been developped for the IaaS paradigm: once the resources have been reserved, how can we efficiently control and provision services on them?

A control plane able to control an integrated network and IT infrastructure, requires three components: (1) a signalling component to set up the service, (2) a provisioning strategy in order to choose IT end points and routes to these resources and (3) the control/management plane to glue evertyhing together and provide the service. In what follows we will investigate prior attemps for such control systems and compare them to our NCP+ proposal, using the requirements above.

One attempt for an integrated control plane spanning both the network and the IT resources in the context of grid computing has been created by the Phosphorus project [13] [14]. Phosphorus created an enhanced version of the ASON/GMPLS control plane to both monitor and co-allocate network and grid resources (denoted as Grid GMPLS or G^2MPLS). Basically, it represents the grid resources as network nodes with special capabilities and distributes this extra grid information the same way as the network resource information using OSPF-TE. G^2MPLS adopts a fully distributed path computation architecture: computations are performed by the ingress GMPLS controller (where the request originates), which means that all GMPLS controllers need to have the full view of the infrastructure, implying the need to disseminate OSPF-TE messages across all controllers. This contrasts with the hierarchical PCE architecture we adopt for the advertisement of the IT resources: since the path computation is centralized on dedicated PCE servers, resource updates are limited to these entities and not flooded among the GMPLS controllers (see Section 3). Moreover, our NCP+ provides a cloud-oriented web service interface, with a centralized access point for all the NCP+ provisioning services (service request, service monitoring and IT advertisement). This interface follows the paradigms of the REpresentational State Transfer (REST) model, commonly adopted in the current cloud environments and, in contrast with the Phosphorus approach, hides all the complexity of the internal network protocols, like OSPF-TE. This aspect is fundamental to allow an easy integration of the NCP+ into existing management systems for cloud resources and infrastructures. Finally, the semantics used to represent the IT resources within the NCP+ derives from the more recent Open Cloud Computing Interface (OCCI) [15] standards as opposed to the grid-oriented Grid Laboratory Uniform Environment (GLUE) standards [16] used in Phosphorus.

The EnLIGHTened (ENL) Computing Project had the same objective as the Phosphorus project: create an environment able to dynamically request any kind of resource (e.g., computers, storage, instruments and high-bandwidth net-

work paths). It is based on the Highly-Available Resource Co-allocator (HARC) [17], which is an open-source system that allows clients to reserve multiple distributed resources in a single step as if they were one resource (known as atomicity). The general architecture consists of *Clients* which generate resource co-allocation requests, *Acceptors* which make the reservations and lastly *Resource managers* which talk to local schedulers for each resource. There are two important Resource Managers: (i) Compute Resource Manager which communicates with some batch scheduler (e.g, Moab, Torque or Maui) and (ii) Network Resource Manager [18] which sends commands (with the fully specified route in an Explicit Route Object) to the GMPLS controller at the path's ingress point.

The G-Lambda project provides a standard web service interface between applications and the grid resource managers and network managers provided by existing network operators. In essence, the proposed architecture works with a central Grid Resource Scheduler (GRS) which accepts requests specifying the required number of CPUs and bandwidth. The GRS then sends its commands to two entities: (i) Computing Resource Managers which reserves the computing resources and (ii) Network Resource Management System which provisions the network paths using GMPLS.

For both the G-Lambda and ENL project, multi-domain connections are computed as follows: a local network resource broker communicates with the other resource brokers of the other domains to ask for network path quotations [19]. There are no state updates between the resource brokers (only for the inter-domain links) and consequently, the inter-domain path and end-points at the domain boundaries must be known beforehand. Hence, network optimization is limited to intra-domain path computations, as the inter-domain paths are fixed. Our NCP+ uses a hierarchical PCE architecture, where an abstracted view of the infrastructure is known by a central entity. Based on this abstracted information, both the network end-points and the paths towards them are computed, optimizing the complete infrastructure.

2.2. Path computation methods

Path computation in the NCP+ is performed by dedicated PCEs [12]. A PCE holds topology information and can be queried by a Path Computation Client (PCC) to determine end-to-end paths. The PCE typically stores its information in a Traffic Engineering Database (TED) and uses this information to perform constrained path computation. PCE-based path computation has been extensively studied [20], especially to facilitate inter-domain service provisioning. The PCE proposals for inter-domain path computation for unicast requests (e.g., where both source and destination are explicitly and univocally specified) can be classified into three categories, referred to as Per-Domain PCE (PD-PCE), peer-to-peer PCE (P2P-PCE) and hierarchical PCE (H-PCE). In what follows we will explain them and explain which options best fits the anycast paradigm.

2.2.1. Per-Domain PCE path computation

In a Per-Domain approach [21], the route to a destination in another domain is fixed (pre-computed or based on operator policies). For inter-domain path computations, each path segment within a domain is computed during the signaling process by each entry node of the domain up to the next-hop exit node of that same domain. When an entry border node fails to find a route, boundary re-routing crankback signalling [22] can be used: a cranckback message is send to the entry border node of the domain and a new exit border node is chosen. This mechanism has several flaws: (i) the PD solution starts from an already known domain sequence which does not allow to optimize the complete infrastructure, (ii) this method does not guarantee an optimal constrained path and (iii) the method may require several crankback signaling messages, thus increasing signaling traffic and delaying the LSP setup. Consequently, PD path computation is not a suitable method for anycast path computation in a multi-domain, optical network scenario.

2.2.2. Backward Recursive Path Computation

The P2P-PCE architectures use the Backward Recursive Path Computation (BRPC) algorithm [23] to compute paths in a multi-domain scenario. The PCEs are provided with a pre-configured domain chain (based on agreements between infrastructure operators) and then create a Virtual Shortest Path Tree (VSPT) from the destination to the source. The VSPT is initialized from the ending node in the destination domain and is extended to all border nodes which are connected to the upstream domain in the provided domain chain. The process is repeated recursively by each domain up until the source domain, which can compute the optimal end-to-end path. A comparison between PCE based path computation techniques and signaling based path computation schemes such as RSVP and RSVP-with-crankback is presented in [24]. The use of this technique is limited for dynamic anycast path computations: multiple domain paths need to be considered (there are multiple IT end points spread over multiple domains) which would result in the concurrent computation of multiple VSPTs, where the path segment computation is not coordinated from a central entity and cannot be optimized.

2.2.3. Hierarchical PCE

Another approach to compute multi-domain paths in the network, which is used in our proposal for the NCP+, is to have a Hierarchical PCE (H-PCE) architecture which is similar to the routing area hierarchy as proposed in the Private Network-to-Network Interface (P-NNI) [25]. In H-PCE, a parent PCE is in charge of coordinating the end-to-end path computation, through multiple node-to-node intra-domain requests to its child PCEs located along the candidate inter-domain path. In H-PCE, (i) the domain path is not required to be pre-configured since it can be dynamically computed by the parent PCE itself, (ii) no critical information is required for inter-domain LSP calculation and (iii) no sharing of intra-domain information (topology, policies, etc.) with other domains is necessary. Consequently, H-PCE suits the anycast routing model

perfectly. We note that there are several ongoing projects which employ H-PCE as a key element for the path calculation in both management and control plane architectures: ONE [26], MAINS [27] and STRONGEST [28] are some of these H-PCE related projects which are pushing PCE based architectures, extending PCEP protocol and pushing the standardization in several forums. However, the use of H-PCE for anycast path computation in multi-domain infrastructures comprising both network domains and IT end points has not yet been investigated.

2.3. Topology aggregation Techniques

In H-PCE provisioning, the child topologies are abstracted into a single aggregated topology. This mechanism is mainly used to overcome scalability and confidentiality issues, which are inherent in the multi-domain scenario. Previous works that addressed the aggregation problem provided the following taxonomy [29]: (i) *Single Node* aggregation where a domain is represented by a single node, (ii) *Star* aggregation where the domain is characterized by a star and (iii) *Full Mesh* (*FM*) aggregation where the domain is represented by a full mesh topology of its border nodes.

The work described in [30] compares the single node aggregation with *FM* and investigates two wavelength selection schemes, showing a notable reduction in light path blocking using *FM*. The authors of [31] have compared the three topology abstraction mechanisms, in terms of blocking probability, network load and inter-domain connection cost. Apart from these aggregation schemes, they also propose a hybrid form where, depending on the ratio of border nodes to the total nodes of a domain, *FM* or *Star* aggregation is used. Results confirm that independently of traffic intensity, single node aggregation leads to an intolerable service blocking. The *Star* mechanism performs better then single node, but cannot achieve the same efficiency as *FM*. In high load scenario's, all aggregation schemes achieve the same amount of blocking, while *FM* is still able to achieve a higher channel occupation. Based on these works we have opted to use *FM* and *Star* as possible candidates for the aggregation techniques for the integrated network and IT infrastructure of the NCP+. We finally refer to a comprehensive survey [20] of inter-domain peering and provisioning solutions.

3. NCP+ architectural model

3.1. General overview

To date, GMPLS [11] is one of the de facto control planes, widely applied in today's access to backbone networks. The GMPLS control plane is divided into two components: (i) the signaling protocol (i.e., RSVP-TE [32]) used to reserve network resources along a path and establish connections in the transport network and (ii) the routing protocol (i.e., OSPF-TE [33]) used to announce the resource capabilities and availabilities in the network. The general NCP+ architecture is depicted in Fig. 1. We organize the IT resources in multiple "IT domains", called IT Sites (IT-S). Each IT-S includes different types of IT

resources that are locally controlled through an IT Manager (IT-M) (e.g., OpenNebula[1]) and we assume that all the IT resources belonging to a single IT-S are connected to a single network domain. The IT-M is the entity in charge of the management of the IT resources and is able to interact with a new component, called the NIPS Client. It is responsible for triggering the procedures for provisioning the network resources associated to the cloud service. In particular, the NIPS Client translates the description of the requested service to a set of requirements compliant with the Service Level Agreements (SLAs) established between the operator and its customer. These requirements are propagated to the NIPS Server, which acts as a centralized service access point for each network domain on the NCP+ side and triggers the necessary actions to establish the connectivity service (path computation, signaling, etc.). Note however, that the IT-M and the GMPLS control plane remain responsible for the final "configuration" of the resources in their own scope: the IT-M generates the commands to configure the IT resources, while the NCP+ manages the commands on the network side. This principle is based on the fundamental requirement to limit the impact required by the integration of the NCP+ functionalities on existing IT-Ms, which will facilitate the adoption of the solution in already deployed IT environments.

3.2. NIPS Client and NIPS Server

The interaction between the IT-S and NCP+ takes place in the NIPS User-To-Network-Interface (NIPS UNI) [34]) as shown in Fig. 2, which is a REpresentational State Transfer (REST) service-to-network interface based on HTTP between the NIPS client and the NIPS Server. The NIPS UNI enables the NIPS Client to request enhanced transport network connectivity services, receive notifications about the status of the established services and advertise the capability and availability of the local IT resources. This approach limits the impact on existing cloud middleware, since it only requires the introduction of a specific client to request transport network connections. The complexity of the network side protocols, for signaling and routing, is completely transparent for the IT-M, since it is entirely managed within the NCP+. On the NIPS server side, the REST messages are translated into the related network protocol messages (e.g., for signalling). The NIPS Server implements a UNI-Client (UNI-C) to interact with the transport network GMPLS controllers through their UNI-Network (UNI-N) component. Fig. 2 shows that the NIPS server, implementing the UNI-C, serves as a sort of proxy between the NIPS client and the UNI-N of the GMPLS controller. Consequently, all the service requests issued by the NIPS Client over the NIPS UNI are translated into RSVP-TE messages and propagated to the UNI-N of the corresponding GMPLS controller (i.e., the ingress node where the IT-S is attached to). For the advertisement of IT resources towards the NCP+, the NIPS server implements a Path Computation Client (PCC) to push the received information into the associated routing

[1] http://opennebula.org/

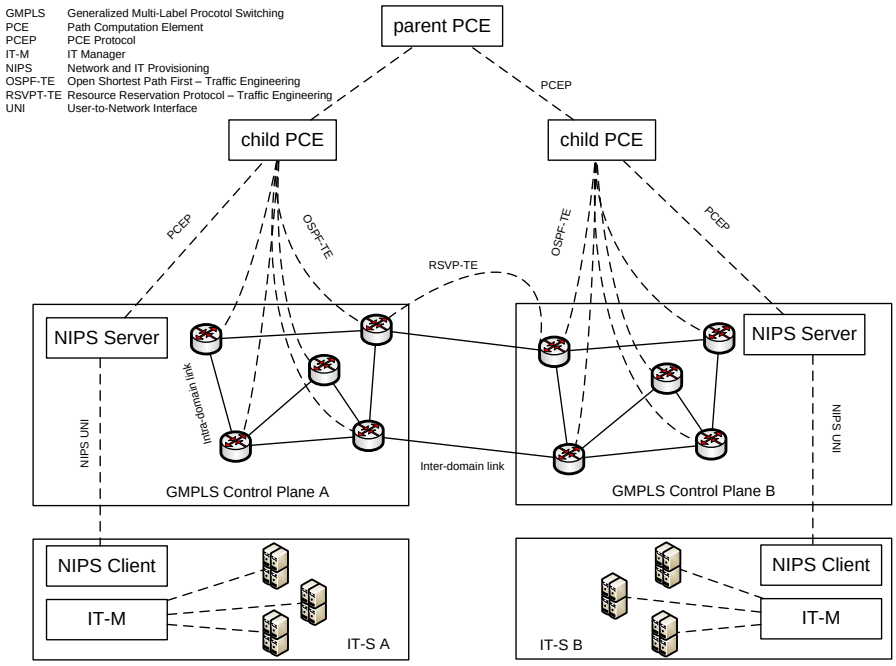


Figure 1: Overview of the modules of the proposed NCP+. The dotted lines represent modules interacting with each other, using the stated protocol or interface.

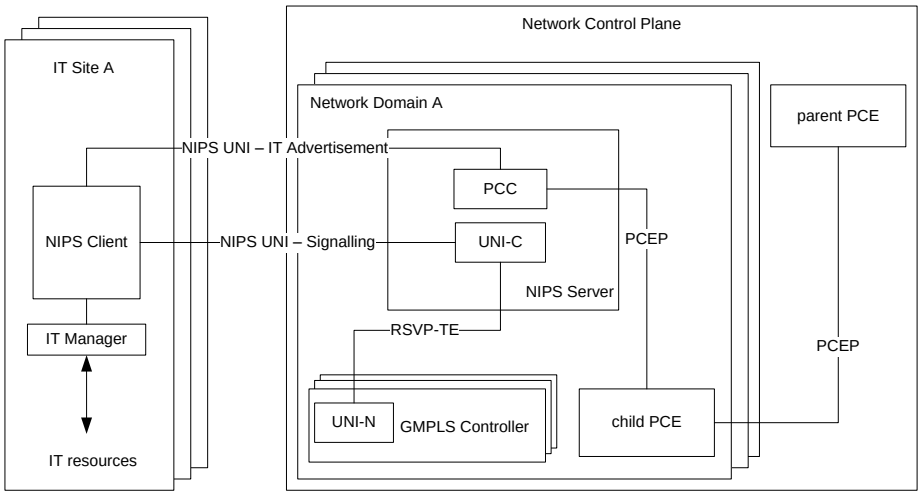


Figure 2: Interfaces and signaling between different modules in the NCP+

controller (i.e., the child PCE) on the NCP+.

3.3. IT resource advertisement

In order to enable inter-domain IT-aware path computation, an advertisement mechanism has to be defined to propagate the IT resource availabilities within each network domain (i.e., at the child PCE level) and at the parent PCE. In the specific case of anycast service provisioning, the choice of the IT end points of the connectivity service is made by the parent PCE. In the NCP+ architecture shown in Fig. 1 we propose the PCEP Notify message to update IT resource information: the child PCE collects IT advertisements in the form of PCEP Notify messages sent by the PCC of the NIPS Server. The same mechanism is in place between the child PCE and the parent PCE. With that information in place, the parent PCE can take into account not only network Traffic Engineering (TE) parameters, but also additional attributes describing capabilities and availabilities of the IT resources. On the other hand, the network parameters are sent using the conventional OSPF-TE protocol from the GMPLS controllers to the child PCE, and from this child PCE to the parent PCE.

3.3.1. PCEP notify protocol extension

In the NCP+, advertisements about IT resources capabilities and availabilities at the IT-S are notified from the NIPS server to the PCE(s) through PCEP Notify (PCNtf) messages properly extended with a set of new Type-Length-Values (TLV). When the NIPS Server receives a new IT advertisement over the NIPS UNI, the PCC implemented within the NIPS Server generates a new PCNtf message that is sent to the child PCE responsible for path computa-

Name	Descritpion
`QUERY`	sent from a parent PCE server to a child PCE and used to query all the IT resource information stored at the child PCE.
`UPDATE`	sent between modules which need to update information for a new or modified IT resource.
`DELETE`	sent between modules which need to delete an existing IT resource.

Table 1: New notification values used in the extensions of the PCEP protocol.

Name	Description
`IT TLV`	provides generic information about the overall advertisement (see Figure 3), including the identifier of the originating IT site, the type of advertisement (i.e., resource add, remove, or update), and its trigger (e.g., synchronization, resource failure, resource modification from administration, etc.).
`Storage TLV`	describes the parameters associated with storage resources (e.g., Storage Size).
`Server TLV`	describes the parameters associated with a server (e.g., Operation System, Memory).

Table 2: New TLV values for the PCEP extention.

tion within the local domain, which in turn forwards the received PCNtf to the parent PCE.

The extended PCNtf messages for IT advertisement are compliant with the generic format described in [35]. The following new value for for the Notification Type (NT) is defined: `IT resource information (0x04)`. For the Notification Values (NV) we have defined three new values, listed in Table 1. The description of the IT resources is included in the optional TLV, using three new top-level TLVs, shown in Table 2. The Storage and Server TLVs are structured in further sub-TLVs (details see [36]). Each of them describes specific characteristics of the resource and can be structured in sub-TLVs themselves. This approach provides a flexible mechanism to notify a partial description of the resources, fundamental in case of resources updates involving a limited set of parameters or when the IT operator restricts the range of information to be disclosed to the network operator.

4. Path Computation

The overall procedure for a NIPS request calculation is performed as follows (i) first an IT-S is selected, on the basis of the specific scheduling algorithm (which are described in Section 4.3) and (ii) later on a path to this IT-S is chosen (using a specific routing algorithm described in Section 4.2). Note that there is no backtracking, i.e., if the IT-S selected in step (i) later on is no longer reachable (because of a congested network) in step (ii), the request is blocked.

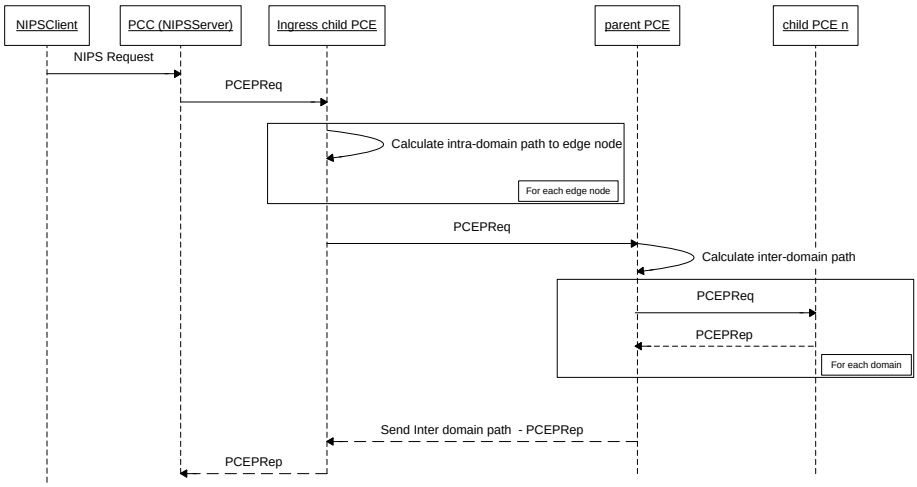
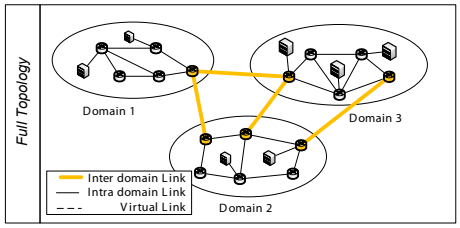
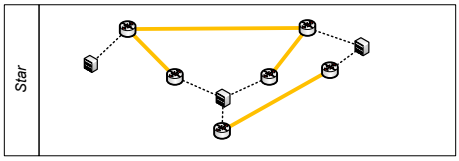


Figure 3: Path computation sequence diagram for H-PCE in NCP+

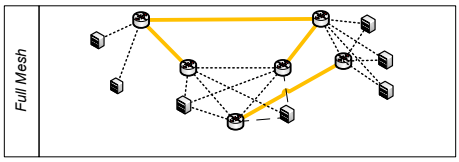
The path computation function (the routing) is a key feature to automate efficient provisioning of both computing resources and network connectivity tailored to the service needs. As already indicated, the NCP+ adopts a hierarchical architecture where a parent PCE is in charge of coordinating the end-to-end path computation through multiple intra-domain requests to its child PCEs. In our H-PCE model there is one centralized point (the parent PCE) maintaining a global view (without internal details) of all the domains, including the attached IT-S. These domains in turn all have a child PCE server to compute intra-domain paths. As can been seen in Fig. 3, the end-to-end path computation is triggered by a NIPS service request sent by the NIPS client. Upon receiving a NIPS request, the NIPS server notifies its collocated PCC, which sends a path computation request (PCReq) to the child PCE that is located in the same domain. This initially computes paths from the requesting node to all of the domain edge nodes after which it sends a PCReq message to its parent PCE. The parent PCE computes a candidate inter-domain path according to its own higher-level topology. The related child PCEs are asked to compute the candidate edge-to-edge path segments which are then combined into the resulting end-to-end path and returned to the ingress child PCE which notifies the PCC of the NIPS Server. There is an observation to be made here: the ingress child PCE initially computes paths to all the border nodes. The motivation is that the parent PCE computes a path with the first node on that path always being a border node, without any connectivity information on reaching that border node from the requesting source node. Hence, the child PCE provides this information in its request to the parent PCE, which in turn is able to choose its first border node based on correct intra-domain availability information.



(a) Original topology.



(b) Star abstraction.



(c) Full Mesh abstraction.

Figure 4: The different abstraction methodes.

4.1. Topology abstraction

The parent PCE has an aggregated or abstracted view of the topology, without any specifics, of each of its child domains. This has two main reasons: (i) confidentiality, as by aggregating, the actual topology of the domains is hidden from the other domains and (ii) scalability, because the end-to-end path computation problem is decomposed and solved over a reduced number of nodes.

In this paper we will discuss and compare two proposals for aggregation schemes: Full Mesh (*FM*) and *Star* aggregation. These aggregation policies are enhanced versions of existing network topology abstractions [20, 29] that only consider network resources: we also incorporate representation of IT resources. In what follows we describe these aggregation techniques and show how to extend them in order for the parent PCE to perform anycast path computations.

In the following section we consider a WDM network $G = (V, E, DC)$ where V is the set of nodes, E the set of edges and DC the set of IT-S. The network is divided into D domains $G^i = \left(V^i, E^i, DC^i\right)$ where domain G^i comprises $\left|V^i\right|$ nodes, $\left|E^i\right|$ edges and $\left|DC^i\right|$ IT end points. For each domain we also denote $B^i \subseteq V^i$ as the set of border nodes.

4.1.1. Full Mesh Abstraction

The aggregated FM topology (see Fig. 4) is a transformation of the original topology where the i^{th} domain $G^i = \left(V^i, E^i, DC^i\right)$ is transformed into a graph

$G^i_* = \left(V^i_*, E^i_*, DC^i_*\right)$ containing the border nodes $b \in B^i$ and the IT end points $dc \in DC^i$. The border nodes in that subgraph are connected by a full mesh ($\forall k, l \in V^i_* : k \neq l$ create virtual edge $e(k, l)$) and every data center node is connected to every network node ($\forall k \in DC^i_*, \forall l \in V_i$ create virtual edge $e(k, l)$). The inter-domain links are then copied into the aggregated topology. The virtual links are assigned a cost, computed by the child PCE responsible for that domain. The child PCE computes a path p between all node pairs and calculates its physical length together with a wavelength availability bitmap which indicates if wavelength λ is available on all links along path p. The bitmap ω_p for a path p is computed in eq. 1 (b_l is the wavelength bitmap for link l).

$$\omega_p = \bigwedge_{l \in p} b_l \tag{1}$$

This information is sent in a Label State Update (LSU) message from the child PCE to the parent PCE (as part of OSPF-TE). As IT resources are not abstracted, IT information is copied and sent from the child PCE to the parent PCE using a PCEP notify message.

The advantage of this aggregation mechanism is that we have a fairly accurate view of the topology and consequently we can compute paths using this detailed wavelength availability. Conversely, the topology can become quite big ($n^2 - n$ links per domain comprising n border nodes and IT resources), which limits the scalability advantage of using a H-PCE method and increases the time needed for path computation.

4.1.2. Star Abstraction

When transforming a domain into a *Star* topology, as depicted in Fig. 4, a graph is created that comprises all the border nodes connected to a single virtual node, i.e. $\forall G^i = \left(V^i, E^i, DC^i\right)$ create n^i_* and connect it to every $b \in B^i$. This n^i_* will not only serve as a connection point for the border nodes, but is also the representation of an aggregated IT endpoint in that domain. All inter-domain links are then copied into the topology.

Assigning link weights for the *Star* aggregation is more complex than for *FM*, as all possible paths between a certain pair of border or IT nodes, are abstracted into a single two-link path in the *Star* topology. We will compare three approaches to compute the availability ω_l (number of available wavelengths) and a length metric $|l|$ for each virtual link l with one of the border nodes as a source. Links with the virtual node as a source have no special meaning and are always available with length metric 0, which means that the length of a path from a border node to any other border node of the same domain is always the same.

1. *Bin*ary: each virtual link in the aggregated topology receives a binary availability and unit length. The link metric is set to unavailable (i.e. it

has infinte weight) when the network it abstracts lacks resources to set up additional traffic between the link's border node and the IT-S node.

$$\omega_l = \begin{cases} 1 & \text{if available} \\ \infty & \text{if unavailable} \end{cases} \tag{2}$$

$$|l| = 1 \tag{3}$$

2. *Avg*: for each border node $b \in B_i$, we calculcate a set S_b of intra-domain paths to every other border domain node and IT-S. The number of available wavelengths for the virtual link l connecting b with to the central node is then calculated as the average number of free wavelengths from the availability maps from the calculated paths (computed as in eq. 1) while the length is the average of the path lengths from the paths $p \in S_b$. (We denote the number of available wavelengths in a bitmap ω_p as ω_p^λ and the length of path p as $|p|$).

$$\omega_l = \frac{1}{|S_b|} \cdot \sum_{p \in S_b} \omega_p^\lambda \tag{4}$$

$$|l| = \frac{1}{|S_b|} \cdot \sum_{p \in S_b} |p| \tag{5}$$

3. *Max*: for each link l, connecting border node b with the central node, we calculate the same set S_b as for Avg, but now ω_l equals ω_p^λ for path $p \in S_b$ with the highest number of available wavelengths, while the length $|l|$ corresponds to the actual length of that respective path.

$$\omega_l = \max_{p \in S_b} \left(\omega_p^\lambda\right) \tag{6}$$

$$|l| = |p| : \max_{p \in S_b} \left(\omega_p^\lambda\right) \tag{7}$$

With *Star* abstraction, the resulting aggregated topology is considerably smaller ($2n$ if n is the number of border nodes), improving the scalability of H-PCE and minimizing the path computation time at the parent PCE. Moreover, employing Bin limits the label state updates (LSU) to updates for the inter-domain links. However, the drawback is that computed paths may be suboptimal, leading to a potentially higher blocking ratio.

4.2. Routing Algorithms

In this paper we consider three metrics, described in Table 3, used by a shortest path routing algorithm (e.g., Dijkstra), employed by the PCE child and parent modules. (We denote the weight for link l as $\pi(l)$).

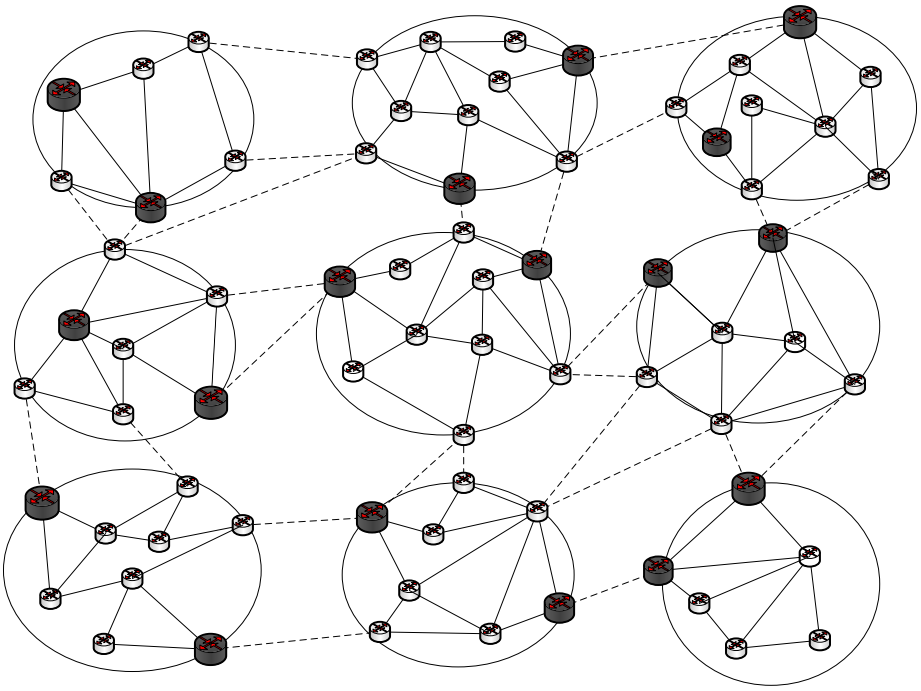


Figure 5: The topology considered for the simulations, taken from [38]. The dark grey nodes represent the nodes that have an attached IT-S.

4.3. Scheduling Algorithms

We consider a number of IT-S scheduling strategies listed in Table Table 4, which we apply for *Star* and *FM* aggregation. These scheduling mechanisms use the exact information of the IT-S for the *FM* abstraction. *Star* however, aggregates a domain's IT information into one virtual node, for which we average all the information per domain (current processing load, maximum available capacity, etc.) and send it to the PCE child.

5. Simulation Results

The performance of the different aggregation mechanisms and routing algorithms is evaluated by simulation. We have built a simulation environment based on OMNeT[2], which is fully described in [39]. A 9 domain, 72 node optical network has been considered, with 18 data centers as shown in Fig. 5. There are 38 border nodes and 96 bidirectional links (of which 24 inter domain links). Each intra-domain and inter-domain link accommodates 32 and 64 wavelengths respectively and we consider a network with wavelength conversion. Hence, the

[2]http://www.omnetpp.org/

Name	Description
SP	Physical length Shortest Path routing: $\pi(l) = \|l\|$
AV	For each link l use its current fraction of used wavelengths: $\pi(l) = \frac{\omega_l}{\omega_{total}}$ where ω_{total} represents the total number of wavelengths of link l and ω_l the number of active wavelengths. AV forces links with a higher load to be less likely used.
AV-L	For each link l we use $\pi(l) = \|l\| \times \frac{\omega_l}{\omega_{total}}$. This way, the algorithm favours shorter paths with high availability.

Table 3: Routing metric used by the shortest path algorithm in the parent and child PCE modules.

Symbol	Description
L-max	Schedule to the IT-S with the highest current load, concentrating requests at the same location as much as possible. This algorithm is used, e.g., when IT energy minimization is of concern [37].
L-Min	Choose the IT-S with the lowest current load, performing IT load balancing.
Closest	We schedule to the IT-S which is closest in terms of the employed link metrics (SP, AV or AV-L).
Random	We randomly select an IT-S for benchmarking purposes.

Table 4: Scheduling Mechanisms

effect of resource fragmentation on blocking in our use case is not present, as blocking only occurs when there is no more free network or IT capacity. Each data center has 500 servers. A request originating at a source site asks for a certain amount of servers that need to be reserved at a destination site of choice, and one unit of bandwidth (i.e., a wavelength) between the source site and the chosen destination site. In order to accommodate a request, a destination site needs to be chosen with enough available capacity and a path needs to be computed between the source site and the destination site. The numbers shown in the graphs are averages of 20 simulations with a different seed. The 95% cofidence intervals are very small, so we have opted not to draw them to make the graphs more clear. We stopped simulation after 200.000 requests have been processed. In our simulations, the requests are generated using a Poisson process (with exponentially distributed arrival and service rate) and are scheduled following one of the scheduling mechanisms described in Section 4.3. We apply a uniform traffic profile, where each node in every domain has the same arrival rate. We only choose among destination sites that can accommodate the requested capacity; if there are multiple equivalent IT-S (e.g., *L-max* where two IT-S have the same load), we choose the IT-S which is closest (in terms of the metrics used by the routing algorithm).

We have divided the results in two main categories: (i) the *network-intensive scenario* where there is always enough IT capacity (i.e., each request only requires one server), thus blocking only occurs because of lack of network resources (called network blocking) and (ii) the *computing-intensive scenario* where the number of requested servers is significant (15 servers), hence blocking occurs because of either lack of network or IT resources (the latter is called IT blocking).

We first investigate the different aggregation schemes separately, trying to find the best

- scheduling technique (Section 5.1),
- routing algorithm (Section 5.2),
- and network information abstraction methods, (Section 5.3)

for both *Star* and *FM*. Subsequently we compare these *Star* and *FM* strategies on service blocking, end-to-end setup time and network control plane load in Section 5.4.

We do this for:

- the *network-intensive scenario*, and
- the *computing-intensive scenario*.

5.1. Scheduling algorithm (network-intensive scenario)

We want to find the best scheduling algorithm terms of service blocking. Fig. 6a and Fig. 6b show this blocking for *FM* and *Star* aggregation respectively. First, we notice that the best scheduling policy (for both *FM* and *Star*) is *Closest*

as expected: we schedule to the nearest IT-S minimizing the required number of network resources and as there is always enough IT capacity, blocking due to a lack of IT resources never occurs. Secondly, we notice that either IT load balancing (*L-Min*) or concentrating requests in one location (*L-max*), leads to a significant service degradation. This is also reflected in Fig. 6c and Fig. 6d, which show the average network load (defined as the ratio of the number of active and the total number of wavelengths). *Closest* requires the least amount of network resources (as shorter paths are chosen) and *L-Min* and *L-max* require about 24% and 43% more network resources in the *FM* case. We note that these qualitative conclusions remain (*Closest* is always best) for all routing algorithms and network abstraction methods.

Lastly we note that for *Star* aggregation, *L-max* has substantially high network blocking values. The reason is that *Star* aggregation uses abstracted information for its virtual links. Initially, a domain is able to accommodate request and hence one IT-S is chosen to do this. As more and more requests are scheduled to that IT-S, there is a point where no path can be found between one of the domain's border nodes and that IT-S. So, although still having enough IT capacity, the IT-S is unable to serve any more request as it has become unreachable in the real topology, while the abstracted topology tells otherwise. The metrics used for the *Star* abstraction cannot reflect this unavailability as one virtual link abstracts multiple paths. Hence, the scheduling mechanism chooses a reachable IT-S in the abstracted topology, which is unreachable in the actual topology and the request is blocked when an intra-domain path needs to be found by the child PCE responsable. As no requests are provisioned, the network load is also low (see Fig. 6d). We note that this effect also happens for the *computing-intensive scenario* and have chosen not to show the results for *Star- L-max* for the *computing-intensive scenario*.

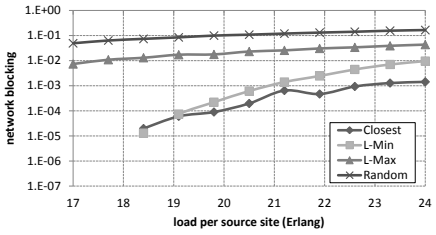
5.2. Routing Algorithms (network-intensive scenario)

5.2.1. FM Routing algorithms

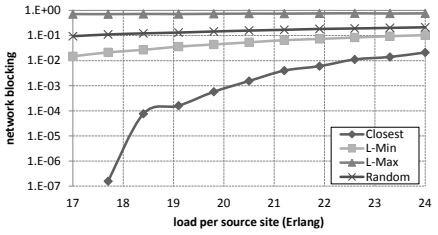
The impact of routing algorithms on service blocking depends on the scheduling strategy. For scheduling strategies where the destination site remains constant for a certain period (i.e., *Closest* and *L-max*), the difference in service blocking among varying routing algorithms is minor, which means that optimal routing amounts to the choice of a shortest path. We do not show the blocking figures, because of space limitations. For the scheduling strategies where the destination choice varies more over time (i.e., *L-Min* and *Random*), we notice a subtle distinction between routing algorithms. Fig. 7a and Fig. 7b show that the network load balancing algorithm (AV routing) minimizes blocking, closely followed by AV-L and SP. Consequently, when the choice of destination IT-S is more dynamic during a certain period, the wavelength availability information is exploited to find better paths, lowering the network blocking.

5.2.2. Star routing algorithms

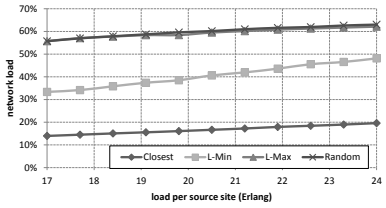
The distinction between routing algorithms becomes apparent when applied in *Star* aggregation. We show the blocking in Fig. 8a and Fig. 8b for *Clos-*



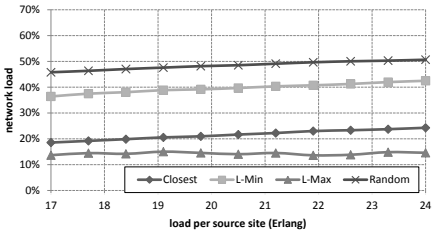
(a) Network blocking for *FM*.



(b) Network blocking for *Star*.

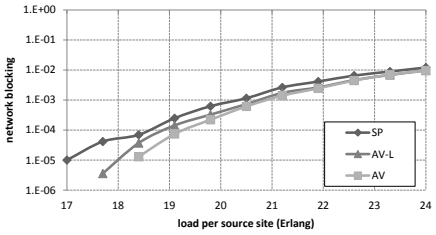


(c) Network load for *FM*.

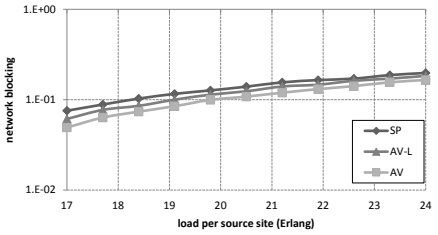


(d) Network load for *Star*.

Figure 6: Network blocking and network load figures for *Star* and *FM* aggregation, using AV as routing algorithm and *Avg* as information abstraction method for *Star* for the *network-intensive scenario*. *Closest* scheduling minimizes network blocking and network resource load. The relative position of the scheduling mechanisms remain the same for routing algorithms or network abstraction methods.

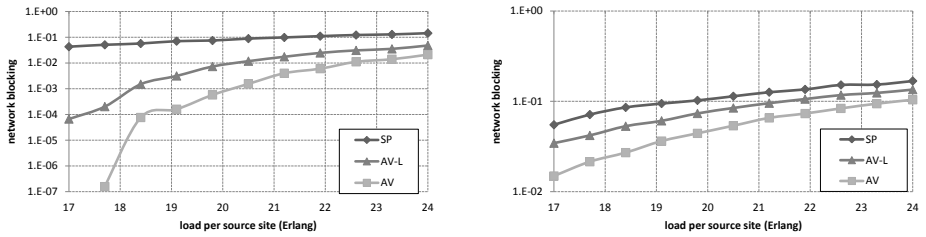


(a) *L-Min*



(b) *Random*.

Figure 7: Network blocking figures for *L-Min* and *Random* for *FM* abstraction for the *network-intensive scenario*. We can see that AV routing is preferred, but that differences are subtle.



(a) *Closest* (b) *L-Min*.

Figure 8: Network blocking figures for *Closest* and *L-Min* scheduling for *Star* abstraction (using *Avg* as information aggregation) for the *network-intensive scenario*. The differences between routing algorithms are clearly defined: AV has the best performance, followed by AV-L and SP.

est and *L-Min* respectively using *Avg* information abstraction (but conclusions also apply for *Random* and *L-max* and other information abstraction methods). Incorporating the aggregated network availability information per border domain node into the routing algorithm (i.e., AV and AV-L routing) optimizes the network blocking: (i) the inter-domain chain can be chosen according to the abstracted wavelength availability information and (ii) the intra-domain paths can be optimized using the exact wavelength availability information.

We also note that AV-L never outperforms AV: the wavelength availability information suffices to find the optimal choice for paths, while the information on the exact distance of the paths does not play a crucial role.

5.3. Information aggregation (network-intensive scenario)

In Fig. 9 we compare network blocking figures between three methods to abstract the domain's information : (i) *Bin*, (ii) *Avg* and (iii) Max. Note that we only include the graph for AV routing with *Closest* scheduling, but conclusions qualitatively apply for the other routing/scheduling strategies. We see that the best way to abstract the information, is averaging the network information. Max is unable to achieve the same network blocking as *Avg*: choosing information from one representative path as abstraction method cannot attain the same service blocking as an aggregated representation of the whole domain.

We also note that *Closest* scheduling with *Bin* as abstraction method, leads to pure intra-domain scheduling and routing: all requests are scheduled to one of the available IT-S in the same domain. The distance from the requesting source to the domain's virtual node (abstract IT-S) is either one or unavailable. Distances to another domain's virtual node would require a distance larger than one, and are consequenlty never chosen. This observation confirms the need for inter-domain scheduling: a sheduling and routing strategy which is able to perform inter-domain scheduling and routing (e.g., AV-*Closest* with *Avg* as abstraction schedules 62% intra-domain and 38% inter-domain), outperforms in

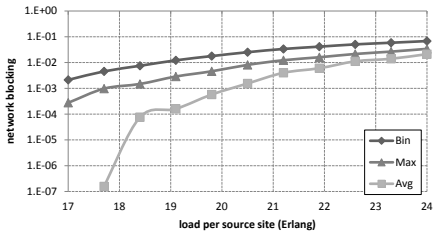


Figure 9: Comparison of network blocking for the different network information abstraction methods for Star abstraction with, *Closest* scheduling and AV routing

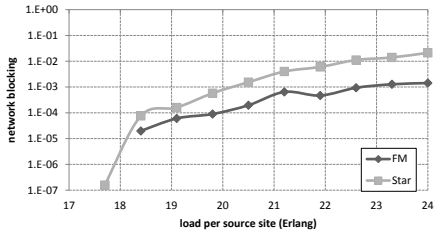
terms of blocking every possible routing and scheduling strategy which schedules requests only intra-domain.

5.4. FM vs Star for the network-intensive scenario

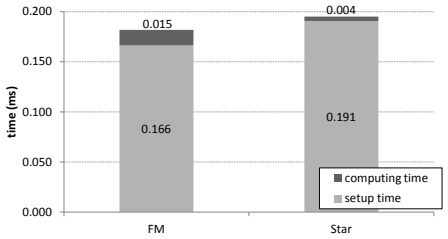
We compare performance metrics for *FM* with those from *Star* aggregation. Above, we have concluded that AV routing together with *Closest* scheduling achieves minimum service blocking (together with *Avg* abstraction for *Star*). Hence we compare those strategies on service blocking, end-to-end setup time and network control plane load in Fig. 10. The setup time has three contributions:

1. Time required to exchange control messages.
2. Time required to compute the path
3. Time required to configure the optical devices. For this we have used 50ms, which is the worst case scenario for mirco-eletromechanical system (MEMS) cross-connects configurations [40].

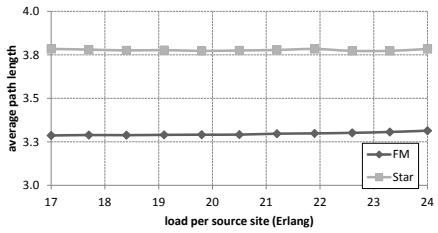
As expected, Fig. 10a shows that the increased connectivity information for each border node seems to be beneficial as *FM* has a lower blocking probability than *Star* (between 2 and 14 times smaller). The time to compute a path however, is much smaller for *Star* then *FM*, which can be seen in Fig. 10b: *Star* computes its paths about 3.5 times faster. However, *Star* computes suboptimal paths which are about 10% longer than the paths computed by *FM*. Hence, the decrease in computation time is outweighed by the extra time needed to set up the longer path: *FM* is able to set up a path in about 6% less time. In addition, these longer paths also increase the number of the OSPF Label State Updates (LSU) that need to be exchanged, which increases complexity for the network control plane: on average 24% more LSU messages are needed (see Fig. 10d). Concluding, (i) the increased service blocking ratio, (ii) the longer paths which are computed, (iii) the associated longer setup times and, (iv) the increased network control plane load turn the *Star* aggregation as a redundant technique for a *network-intensive scenario*. The scalability motivation (reduced number of virtual links in the aggregated topology) is nullified as the time needed to set up a path is still larger than the more complex *FM* technique.



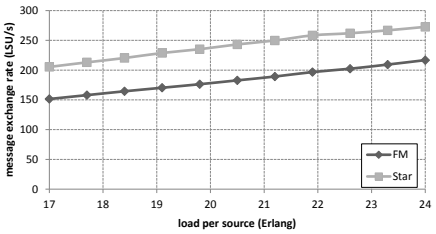
(a) Network blocking between for *FM/ Star*.



(b) end-to-end setup and path computation time for *FM/ Star*.



(c) Average path length for *FM/ Star*.



(d) LSU message exchange rate between the parent PCE and child PCE modules for *FM/ Star*.

Figure 10: This figure compares the network blocking, end-to-end setup times, average path length and the average number of LSU messages exchanged between the parent PCE and its children for AV-*Closest* (and *Avg* information abstraction for *Star*). *FM* leads to lower network blocking, shorter paths, lower end-to-end setup times and a reduction in network control plane load for the *network-intensive scenario*.

5.5. Computing-intensive scenario

In the *computing-intensive scenario*, we increase the number of servers demanded per request from 1 to 15. Here, situations may occur where there is not enough IT capacity and hence both network and IT blocking may occur. Our simulations point out that some of the conclusions drawn for the *network-intensive scenario*, also apply here: (i) for both *Star* and *FM*, AV routing is the best routing mechanism, (ii) *Closest* is still the best scheduling strategy for *Star*, (iii) *Avg* information abstraction for *Star* is still the best strategy and (iv) inter-domain scheduling shows to decrease service blocking compared to a scenario where only intra-domain scheduling is performed.

However, the relation between scheduling strategies changes for *FM*, which can be observed in Fig. 11, where we show the amount of blocking due to insufficient network (network blocking, Fig. 11a) and IT resources (IT blocking, Fig. 11b). The sum is the total blocking.

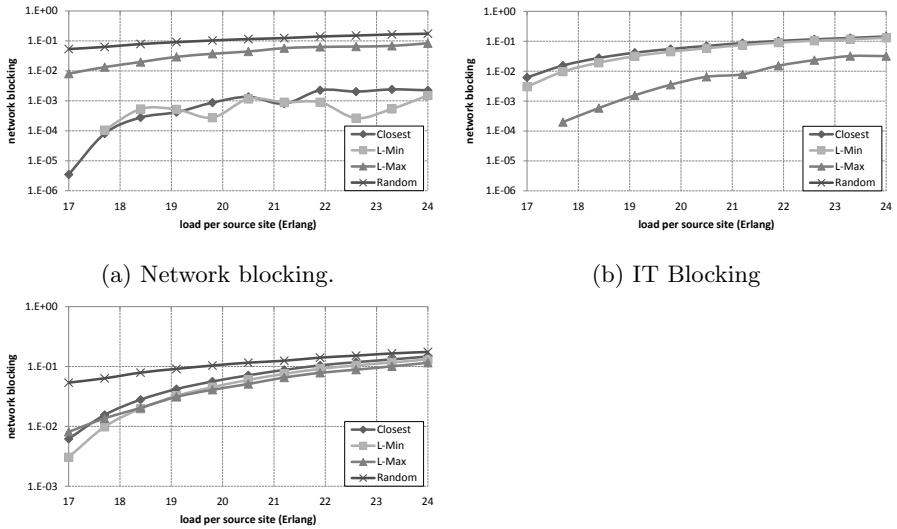
We see in Fig. 11c that up to 18.5 Erlang, *L-Min* achieves lowest total blocking, but for higher loads *L-max* is best. This is attributed to the fact, that in a low load scenario *L-max* computes longer paths to distant IT-S, leading to a higher network blocking ratio (see Fig. 11a). Hence, *L-max* achieves a lower computational resource load than *L-Min* and *Closest* (requests are blocked) and consequently, the moment where *L-Min* and *L-max* attain the same IT resource load is different (e.g., *L-max* reaches an average IT resource load of 87% at 24 Erlang, while *L-Min* reaches this at 20 Erlang). Since in higher load scenarios, *L-max* has more available IT capacity, it attains lower IT blocking and because the contribution of IT blocking to the total blocking is dominant, it also attains lower total blocking.

When comparing *Star* and *FM* in terms of service blocking in Fig. 12a, we see that *Star* with *Closest* scheduling has about the same service blocking figures as *FM* with *Closest* scheduling. Indeed, *Closest* scheduling is not the best strategy for *FM* (as it achieves high IT blocking values). When comparing *Star-Closest* (best choice for *Star* aggregation) with *FM* with *L-max* scheduling however, we see that *FM* is again able to lower its service blocking ratio noticeably. Nevertheless, this intelligent scheduling of *FM* comes at a price, as shown in Fig. 12. To achieve the decrease in service blocking, *FM- L-max* computes longer paths (as shown in Fig. 12c) which comes at two extra costs: (i) a longer setup time (longer paths) for *FM- L-max* with higher computing time leads to a higher end-to-end setup time (see Fig. 12b) and (ii) more network control plane load as more LSU messages need to be sent to the parent PCE module.

Comparing *FM- Closest* with *Star- Closest*, we observe that they attain about the same blocking ratio. *FM* however, is able to reduce the path length compared to *Star* which balances out the time required to compute the path: when enforing *Closest* scheduling *Star* has only 2.21 % faster end-to-end setup times. Consequently, when fast setup times are required, the operator has two options: (1) run *FM- L-max* and *Star- Closest* in parallel and choose which abstraction method to choose based on the setup time requirements or (2) run *FM- L-max* and schedule using the *Closest* strategy when the setup time requirements are important.

	network-intensive scenario	***computing-intensive scenario***
Star	• *Closest* scheduling • AV routing • AVG abstraction	• *Closest* scheduling • AV routing • AVG abstraction
FM	• *Closest* scheduling • AV/SP routing	• *L-max* scheduling • AV routing
Star vs. *FM*	• *FM* computes better paths in terms of service blocking • Due to suboptimal paths, the reduction in computation time of *Star* is nullified and *FM* has faster setup times.	• *FM- L-max* reduces service blocking the most • *FM- L-max* has higher setup times than *Star Closest* and *FM- Closest*, which correspond in service blocking and setup time.

Table 5: This table summarizes the best scheduling, routing and information abstraction techniques per aggregation method and scenario in terms of service blocking. The last line sums up the conclusions for the comparison between *FM* and *Star* per scenario.



(a) Network blocking.

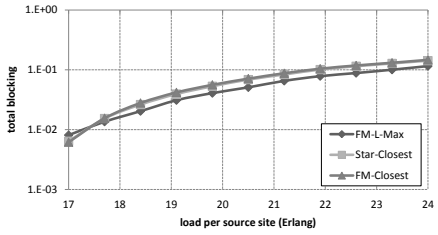
(b) IT Blocking

(c) Total blocking

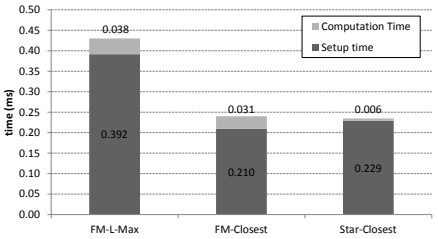
Figure 11: The network, IT and total blocking for *FM* with AV routing. Although *Closest* and *L-Min* achieve the minimal network blocking, IT blocking is relatively high. *L-max* is able to reduce the IT blocking penalty, which is reflected in the total blocking figures.

6. Conclusion

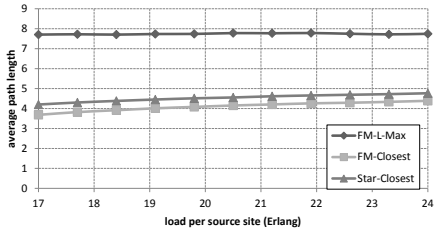
Today, we observe an evolution to network-based service offerings, where applications are pushed further into the network and increasingly rely on interworking of many distributed components: the cloud computing paradigm. Given the increasing adoption of the cloud ideas (cf. IaaS, PaaS, SaaS), in both the consumer, business and even academic spaces, service providers are confronted with more stringent and/or demanding network requirements (in terms of bandwidth, latency) as well as the need to incorporate IT resources in their offering. Therefore, it becomes essential that the service management system is able to manage both network and IT resources in a coordinated way. This paper proposes a set of extensions to the well know Generalized Multi-Protocol Label Switching (GMPLS) and Path Computation Element (PCE)-based Network Control Plane for an optical, multi-domain routing scenario. The NCP+ is aware of the data centers attached to its network and is able to compute anycast paths to one of these IT end points. Our proposed NCP+ architecture provides protocol extensions to disseminate IT information, and includes enhanced topology information aggregation schemes and joint network and IT resource selection and allocation policies. We have evaluated these schemes and policies using simulation with general conclusions summarized in Table 5. We have demonstrated that for a scenario where applications have very strict network but flexible IT requirements, *FM* abstraction performs best in terms of service blocking, end-to-end setup times and added network control plane load.



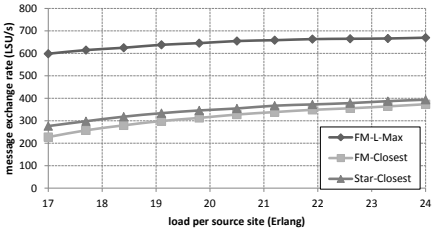
(a) Comparing network blocking between *FM* and *Star*, for *Closest* and *L-max*.



(b) Comparing the end-to-end setup time for *Star* and *FM*.



(c) Comparing the path lengths for *Star* and *FM*.



(d) Amount of LSU messages exchanged between the parent and child PCE modules, for *Star* and *FM*.

Figure 12: Comparing *FM* and *Star* on total blocking, end-to-end setup time, average path length and network control plane load with *Closest* scheduling, AV routing and *Avg* information for the *computing-intensive scenario*. *FM* still achieves lower total blocking, but cannot achieve lower end-to-end setup times then *Star*.

However, in a scenario where IT requirements are dominant, running both algorithms in parallel could lead to an improvement in service blocking and end-to-end setup time. Future work includes an investigation in an adaptive approach: either using *Star* or *FM* depending on the IT vs. network load.

7. Acknowledgement

J. Buysse was supported by a PhD grant of the Flemish government Agency for Innovation by Science and Technology (IWT). M. De Leenheer is supported as post-doctoral fellow of the Research Foundation Flanders (FWO-Vl.). This work was carried out using the Stevin Supercomputer Infrastructure at Ghent University, funded by Ghent University, the Hercules Foundation and the Flemish Government - EWI department. The research work has been supported by the GEYSERS project (www.geysers.eu), funded by the European Community's Seventh Framework Programme (FP7/2007-2013),under grant agreement no. 248657 and by Ghent University through GOA Optical Grids (grant 01G01506).

8. References

[1] C. Develder, M. De Leenheer, B. Dhoedt, M. Pickavet, D. Colle, F. De Turck, P. Demeester, Optical networks for grid and cloud computing applications, Proc. IEEE 100 (4) (2012) 1149–1167.

[2] H. Urs, B. Luiz Andre, The Datacenter as a Computer: An Introduction to the Design of Warehouse-Scale Machines, Vol. 6 of Synthesis lectures in computer architecture, Morgan and Claypool, 2009.

[3] D. Verchere, Cloud computing over telecom networks, in: Proc. of. Optical Fiber Commun. Conf. and Exposition and the Nat. Fiber Optic Engineers Conf. (OFC/NFOEC), Los Angeles, USA, 2011, pp. 1–3.

[4] L. Garber, Converged infrastructure: Addressing the efficiency challenge, IEEE Computer 45 (8) (2012) 17–20.

[5] J. Buysse, M. De Leenheer, B. Dhoedt, C. Develder, Providing resiliency for optical grids by exploiting relocation: A dimensioning study based on ilp, Comput. Commun. 34 (12) (2011) 1389–1398.

[6] K. Walkowiak, Anycasting in connection-oriented computer networks: models, algorithms and results, International Journal of Applied Mathematics and Computer Science 20 (1) (2010) 207–220.

[7] B. Bathula, J. Plante, V. Vokkarane, Crosstalk-aware anycast routing and wavelength assignment in optical WDM networks, in: Proc. IEEE 4th Int. Symp. on Advanced Netw. and Telecom. Systems (ANTS), Mumbai, India, 2010, pp. 94–96.

[8] J. Buysse, C. Cavdar, M. De Leenheer, B. Dhoedt, C. Develder, Improving energy efficiency in optical cloud networks by exploiting anycast routing, in: Proc. of Asia Commun. and Photonics Conf., Vol. 8310, Shangai, China, 2011, pp. 1–6.

[9] J. Buysse, K. Georgakilas, A. Tzanakaki, M. De Leenheer, B. Dhoedt, C. Develder, P. Demeester, Calculating the minimum bounds of energy consumption for cloud networks, in: Proc. of IEEE Int. Conf. Comp. Commun. and Networks (ICCCN), Maui, USA, 2011, pp. 1–7.

[10] K. Bhaskaran, J. Triay, V. Vokkarane, Dynamic anycast routing and wavelength assignment in WDM networks using ant colony optimization (ACO), in: Proc. IEEE Int. Conf. on Commun. (ICC), Kyoto, Japan, 2011, pp. 1–6.

[11] E. Mannie, IETF, RFC 3945 : Generalized multi-protocol label switching (GMPLS) architecture, Tech. rep., Network Working Group (Oct. 2004).

[12] A. Farrel, J.-P. Vasseur, J. Ash, IETF, RFC 4655: A path computation element PCE-based architecture, Tech. rep., Network Working Group (Aug. 2006).

[13] S. Figuerola, N. Ciulli, M. De Leenheerc, Y. Demchenko, W. Zieglere, A. Binczewski, PHOSPHORUS: Single-step on-demand services across multi-domain networks for e-science, in: Proc. European Conf. and Exhibition on Optical Commun. (ECOC), Berlin, Germany, 2007, pp. 1–16.

[14] N. Ciulli, G. Carrozzo, G. Giorgi, G. Zervas, E. Escalona, Y. Qin, R. Nejabati, D. Simeonidou, F. Callegati, A. Campi, W. Cerroni, B. Belter, A. Binczewski, M. Stroinski, A. Tzanakaki, G. Markidis, Architectural approaches for the integration of the service plane and control plane in optical networks, Optical Switching and Networking 5 (23) (2008) 94– 106.

[15] T. Metsch, Open cloud computing interface - use cases and requirements for a cloud API, Tech. rep., Open Grid Forum (16 Sep. 2009).

[16] S. Andreozzi, M. Sgaravatto, M. C. Vistoli, Sharing a conceptual model of grid resources and services, Computing Resource Repository (2003) 1–4.

[17] J. MacLaren, HARC: The highly-available resource co-allocator, in: R. Meersman, Z. Tari (Eds.), On the Move to Meaningful Internet Systems 2007: CoopIS, DOA, ODBASE, GADA, and IS, Vol. 4804 of Lecture Notes in Computer Science, Springer Berlin Heidelberg, 2007, pp. 1385–1402.

[18] J. MacLaren, Co-allocation of compute and network resources using HARC, in: Proc. Lighting the Blue Touchpaper for UK e-Science: Closing Conference of the ESLEA Project, Edinburgh, UK, 2007, pp. 1–5.

[19] S. R. Thorpe, L. Battestilli, G. Karmous-Edwards, A. Hutanu, J. MacLaren, J. Mambretti, J. H. Moore, K. S. Sundar, Y. Xin, A. Takefusa,

M. Hayashi, A. Hirano, S. Okamoto, T. Kudoh, T. Miyamoto, Y. Tsukishima, T. Otani, H. Nakada, H. Tanaka, A. Taniguchi, Y. Sameshima, M. Jinno, G-lambda and enlightened: wrapped in middleware co-allocating compute and network resources across Japan and the US, in: Proc. Int. Conf. on Netw. for grid applications, Lyon, France, 2007, pp. 5:1–5:8.

[20] M. Chamania, A. Jukan, A survey of inter-domain peering and provisioning solutions for the next generation optical networks, IEEE Commun. Surveys and Tutorials 11 1 (2009) 33–51.

[21] J. Vasseur, A. Ayyangar, R. a. Zhang, IETF, RFC 5152 : A per-domain path computation method for establishing inter-domain traffic engineering (TE) label switched paths (LSPs), Tech. rep., Networking Working Group (Feb. 2008).

[22] A. Farrel, A. Satyanarayana, A. Iwata, N. Fujita, G. Ash, IETF, RFC 4920 - crankback signaling extensions for MPLS and GMPLS RSV, Tech. rep., Network Working Group (Jul. 2007).

[23] J. Vasseur, R. Zhang, N. Bitarn, J. Le Roux, IETF, RFC 5441: A backward-recursive PCE-based computation procedure to compute shortest constrained inter-domain traffic engineering label switched paths., Tech. rep., Network Working Group (9 Apr. 2009).

[24] S. Dasgupta, J. de Oliveira, J.-P. Vasseur, Path computation element based architecture for interdomain MPLS/GMPLS traffic engineering: Overview and performance, Network Architectures, Management, and Applications 21 (2007) 38–45.

[25] S. Sanchez-Lopez, J. Sole-Pareta, J. Comellas, J. Soldatos, G. Kylafas, M. Jaeger, PNNI-based control plane for automatically switched optical networks, IEEE Journal of Lightwave Technology 21 (2003) 2673–2682.

[26] L. Contreras, V. Lopez, O. De Dios, A. Tovar, F. Munoz, A. Azanon, J. Fernandez-Palacios, J. Folgueira, Toward cloud-ready transport networks, IEEE Communications Magazine 50 (9) (2012) 48–55.

[27] J. Triay, S. G. Zervas, C. Cervelló-Pastor, D. Simeonidou, GMPLS/PCE/OBST architectures for guaranteed sub-wavelength mesh metro network services, in: Optical Fiber Commun. Conf. (OFC), Los Angeles, USA, 2011, pp. 1–3.

[28] F. Paolucci, O. G. de Dios, R. Casellas, S. Duhovnikov, P. Castoldi, R. Munoz, R. Martinez, Experimenting hierarchical PCE architecture in a distributed multi-platform control plane testbed, in: Proc. Optical Fiber Commun. Conf. (OFC), Los Angeles, USA, 2012, pp. 1–3.

[29] S. Uludag, K.-S. Lui, K. Nahrstedt, G. Brewster, Analysis of topology aggregation techniques for QoS routing, ACM Comput. Surv. 39 (3) (2007) 7–38.

[30] Q. Liu, M. Kok, N. Ghani, V. Muthalaly, M. Wang, Inter-domain provisioning in DWDM networks, in: Proc. IEEE Global Telecommun. Conf. (GLOBECOM), San Francisco, CA, USA, 2006, pp. 1–6.

[31] G. Maier, C. Busca, A. Pattavina, Multi-domain routing techniques with topology aggregation in ASON networks, in: Proc. Int. Conf. on Optical Netw. Design and Modeling (ONDM), Vilanova i la Geltru, Spain, 2008, pp. 1–6.

[32] D. Awduche, L. Berger, D. Gan, T. Li, V. Srinivasan, G. Swallow, IETF, RFC 3209 : RSVP-TE: Extensions to RSVP for LSP tunnels, Tech. rep., Network Working Group (Dec. 2001).

[33] J. Moy, IETF, RFC 2328: OSPF version 2, Tech. rep., Network Working Group (Apr. 1998).

[34] S. Shew, J. Sadler, User network interface (UNI) 2.0 signaling specification, Tech. rep., Optical Internetworking Forum (Feb. 2008).

[35] J. Vasseur, J. L. Le Roux, IETF, RFC 5440 :path computation element (PCE) communication protocol (PCEP), Tech. rep., Network Working Group (Mar. 2009).

[36] N. Ciulli, G. Landi, F. Salvestrini, D. Parniewicz, X. Chen, G. Buffa, P. Donadio, Y. Demchenko, C. Ngo, J. Jimenez, A. Tovar De Duenas, C. Garcia Argos, P. Robinson, A. Manfredi, P. Drozda, Brzozowski, J. Ferrer Riera, S. Spadaro, E. Lopez, E. Escalona, M. Antoniak-Lewandowska, L. Drzewiecki, A. Tzanakaki, K. Georgakilas, J. Buysse, Deliverable D4.1 GMPLS+/PCE+ control plane architecture, Tech. rep., GEYSERS Project (Nov. 2010).

[37] J. Buysse, K. Georgakilas, M. De Leenheer, B. Dhoedt, C. Develder, Energy-efficient resource provisioning algorithms for optical clouds, submitted to Journal of Optical Communications and Networks (July 2012).

[38] S. Shang, N. Hua, L. Wang, R. Lu, X. Zheng, H. Zhang, A hierarchical path computation element (PCE)-based k-random-paths routing algorithm in multi-domain WDM networks, Optical Switching and Networking 8 (4) (2011) 235–241.

[39] M. De Leenheer, J. Buysse, K. Mets, B. Dhoedt, C. Develder, Design and implementation of a simulation environment for network virtualization, in: Proc. 16th IEEE Int. Workshop Computer Aided Modeling, Analysis and Design of Commun. Links and Netw. (CAMAD), Kyoto, Japan, 2011, pp. 87–91.

[40] A. Olkhovets, P. Phanaphat, N. C., D. Shin, C. Lichtenwalner, M. Kozhevnikov, J. Kim, Performance of an optical switch based on 3-D MEMS crossconnect, IEEE Photonics Technology Letters 16 (3) (2004) 780–782.