Technology-aware multi-domain multi-layer routing

Iqbal, Farabi; van der Ham, Jeroen; Kuipers, Fernando

DOI
10.1016/j.comcom.2015.01.010

Publication date
2015

Document Version
Accepted author manuscript

Published in
Computer Communications

Technology-Aware Multi-Domain Multi-Layer Routing

Farabi Iqbal[a], Jeroen van der Ham[b], Fernando Kuipers[a]

[a] *Network Architectures and Services, Delft University of Technology, Mekelweg 4, 2628 CD Delft, The Netherlands*
[b] *Universiteit van Amsterdam, Science Park 107, 1098 XG Amsterdam, The Netherlands*

Abstract

Transporting Big Data requires high-speed connections between end-hosts. Research and educational networks typically are state-of-the-art networks that facilitate such high-speed user-created network connections, possibly spanning multiple domains. However, there are many different high-speed optical data plane standards and implementations, and vendors do not always create compatible data plane implementations. These technology incompatibilities may prevent direct communication between domains and therefore complicate the configuration of connections. However, some domains may have adaptation capabilities that can lift the technology incompatibility constraint in establishing paths between incompatible domains. Within this context, we address two problems, namely: (1) how to model the technology incompatibilities of multi-domain multi-layer networks, and (2) how to optimally establish paths in such networks. We introduce the inclusion of the information of the supported technologies and adaptation capabilities of each domain and inter-domain link in our model. We subsequently propose technology-aware routing algorithms for finding the shortest feasible path in a multi-domain multi-layer network.

Keywords: optical network, technology incompatibility, technology adaptation

1. Introduction

Many different scientific research projects are now producing Big Data. For example, the fields of physics and astronomy have traditionally been the largest producers of data with projects such as the Large Hadron Collider [1], the Sloan Digital Sky Survey [2] or the planned Square Kilometer Array [3] and the Large Synoptic Survey Telescope [4]. We now see that other fields, such as biology and medical research, are also producing and transporting large data sets. These data sets are often shared between different institutes, within countries, but also across the globe. Most countries have their own National Research and Education Network (NREN) for providing high-speed connections between universities and research institutes within their country. For instance, the Dutch NREN is called SURFnet [5]. NRENs can be considered as a catalyst of collaboration between research partners in their prospective countries. Currently, as became evident in a project with SURFnet, one of the main problems faced by NRENs is how to cooperate and pool their resources for setting up *international* lightpaths to fulfill the ever-increasing worldwide research needs of scientific equipment sharing, data distribution, cloud computing, etc. An example of a worldwide NRENs cooperation is the Global Lambda Integrated Facility (GLIF) [6] initiative.

Traditionally, NRENs are interconnected by inter-domain links between their border nodes. In the recent years, GLIF has taken the initiative to propose the use of optical exchanges as open and neutral interconnection points between NRENs, as illustrated in Figure 1. Figure 1 consists of several administrative domains, e.g., NRENs and optical exchanges, where an administrative domain is defined as a network under the control of a single network administrator. Optical exchanges, e.g., the NetherLight [7] are points of presence where all NRENs that are connected to them can communicate with each other. Optical exchanges may also be connected to other optical exchanges. Ideally, the optical exchanges can adapt their client technologies transparently without any restrictions (e.g., client identities, content type or size).

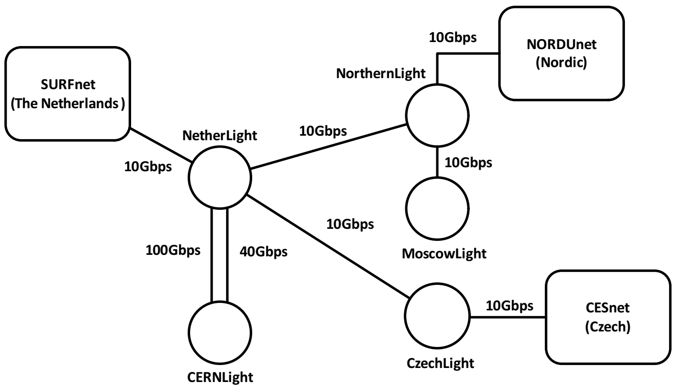


Figure 1: Example of a multi-domain network.

Multi-domain routing is under the jurisdiction of several standardization bodies, such as the ITU Telecommunication Standardization Sector (ITU-T), the Internet Engineering Task Force (IETF), and the Open Grid Forum (OGF). Though their focus varies, all of them have proposed standards related to the multi-domain networking, namely the ITU-T G.8080/Y.1304 as

Email addresses: `M.A.F.Iqbal@tudelft.nl` (Farabi Iqbal), `vdham@uva.nl` (Jeroen van der Ham), `F.A.Kuipers@tudelft.nl` (Fernando Kuipers)

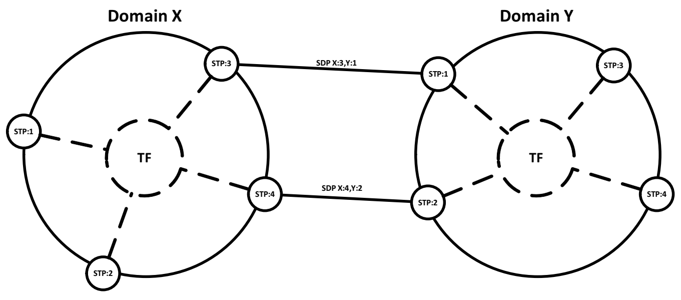


Figure 2: An NSI multi-domain network topology.

a telecommunication standard, the Path Computation Element (PCE) framework (e.g., IETF RFC4655) as an internet standard, and the Network Service Framework (NSF) (e.g., OGF GFP173) as a grid standard.

In the ITU-T recommendation G.8080/Y.1304 [8], an architecture framework referred to as the Automatically Switched Optical Network (ASON) was proposed for a more intelligent optical network operation. The framework introduces a logical architecture of three planes, the transport plane (i.e., data plane), the control plane and the management plane. The framework also encompass the notion of domain, inter-domain links, and several routing approaches.

The IETF RFC4655 [9] aims to decouple the routing function from the control plane such that a dedicated routing component referred to as the Path Computation Element (PCE) is used instead to find more advanced paths, e.g., impairment-aware paths, multi-domain-paths, and multi-layer paths. The PCE architecture can be either centralized or distributed. Multiple PCEs work together via the use of the PCE protocol (PCEP). The standard covers inter-domain routing, intra-domain routing, and inter-layer routing. Munoz et al. [10] provided a good overview of the PCE functionality.

The OGF GFP173 [11] proposes the Network Service Interface (NSI) protocol [12] for domains to cooperate in servicing multi-domain connection requests. NSI has been implemented by various research partners of GLIF, e.g., AutoBAHN by GÉANT, G-Lambda/A by AIST, G-Lambda/K by KDDI R&D Labs, DynamicKL by KISTI, OpenNSA by NORDUnet, OSCARS by ESnet and BoD by SURFnet [13]. Each domain is associated with a software-based management system referred to as the Network Service Agent (NSA). Multiple NSAs work collectively to establish, maintain, and terminate multi-domain connections spanning their domains. Domains are interconnected at their Service Termination Points (STPs), which represent ports on a switch, border nodes, or specific VLANs on a port as illustrated in Figure 2. A grouping of two STPs is referred to as a Service Demarcation Point (SDP). Unlike the IETF PCE framework, the OGF NSF has not yet define any specific standard for multi-domain routing.

Administrators usually build and upgrade their domain according to their preferences for vendors and technologies. These preferences could be based on capital expenditure, equipment availability, maintenance ease, etc. The wide selection of vendors and technologies leads to no de-facto standard in building domains, rendering possible technology incompatibilities between domains. Technology incompatibilities can occur in the data plane, which contains a number of switches interconnected by physical interfaces. A path between two domains is possible only if they support at least a similar technology, can adapt between the technology incompatibilities, or if there is another domain with suitable technology adaptation capability between them. Hence, routing between domains is not a trivial task. Examples of technology incompatibilities are:

Architecture incompatibilities (e.g., IP over WDM [14], SONET/SDH over WDM [15], EoS over WDM [16], or Ethernet over WDM [17]) imply the needs for common lowest-layer technology and adaptation feasibility to upper layers.

Switching type incompatibilities (e.g., wavelength, waveband and fibre channel at layer 1, Ethernet, Fast Distributed Data Interface (FDDI) and cell switching (ATM) at layer 2, (Generalized) Multi-Protocol Label Switching and Internet Protocol (IP) at layer 3) can exist at various layers.

Interface incompatibilities (e.g., 1 GE Ethernet can be encapsulated into VC-3-21v SDH, VC-4-7v SDH, STS-1-24c SONET, or STS-3c-7v SONET) imply possible adaptation and deadaptation problems [18].

Rate incompatibilities (e.g., 1, 10, 40, or 100 Gbps) imply the need for data-rate conversion.

Wavelength incompatibilities (e.g., 850, 1310 or 1550 nm) imply the need for wavelength conversion.

Since the notion of technology-aware multi-domain multi-layer routing is not yet fully addressed in both IETF PCE framework and OGF NSF, and vendor interoperability issues remain an open research [19], we address this problem in this paper. First, we propose a generic network model that incorporates technology incompatibilities and scales well with the increase of graph size and number of technology incompatibilities. Our network model is applicable for use in modeling variety of technology incompatibilities that can occur in multi-domain multi-layer networks. Our network model would also be a useful addition to existing multi-domain standards, and existing technology representation approaches (e.g. NML [20]). Secondly, we propose exact and heuristic algorithms to find technology-aware loopless path from a source node to a destination node in networks with technology incompatibilities. Although triggered by a realistic problem in the NREN community, our work applies to multi-domain multi-layer networks in general.

The remainder of this paper is organized as follows. Section 2 gives an overview of related work and highlights our contributions. In Section 3, we introduce our network model and give some application examples. In Section 4, we define the problem formally, for which routing algorithms are proposed in Section 5. We present a simulative performance analysis of our algorithms in Section 6, and conclude in Section 7.

Table 1: Related work.

Authors	Network Model	Encapsulation Order	Node Looping	Routing Algorithm
Chlamtac et al. [21]	Wavelength graph	No	Yes	Polynomial algorithm
Jabbari et al. [22]	Channel graph	No	No	A variant of Yen's algorithm
Kuipers and Dijkstra [18]	Device-based, layer-based, and stack-based graphs	Yes	Yes (with bandwidth constraint)	A variant of BFS, exact algorithm and heuristic
Shirazipour and Pierre [23]	Simple graph with technology information	Yes	No	Yen's algorithm, and optimized by a Binary Integer Program (BIP)
Lamali et al. [24]	Push Down Automaton (PDA)	Yes	Yes	Polynomial algorithm based on PDA
This paper	Simple graph with technology information	Yes	No	Exact algorithm with look-ahead function and heuristics

2. Related Work

In a network with limited wavelength conversion, only a subset of nodes can convert between wavelengths. A path between two distinct nodes is *feasible*[1] if the wavelength of the path is continuous, or if appropriate wavelength conversion is conducted along the path. Chlamtac et al. [21] modeled wavelength incompatibilities by introducing a wavelength graph of NW nodes. The graph contains N columns and W rows, where N is the number of nodes in the original network, and W is the number of wavelengths. Link existence between nodes depends on the wavelength availability (horizontal links), and the wavelength conversion (vertical nodes). Though their work focuses on the intra-domain routing, their model can also be applied to multi-domain networks.

The ITU-T ASON framework does not include any specific control plane protocol, since it was meant to be a generic architectural framework. In the IETF RFC3945 [25], a control plane protocol suite referred to as the Generalized MultiProtocol Label Switching (GMPLS) [26] was proposed to support multi-layer applications that consist of different types of switching technologies. A GMPLS node may support several types of switching technologies, e.g., Packet Switch Capable (PSC), Layer 2 Switch Capable (L2SC), Time Division Multiplex Capable (TDM), Lambda Switch Capable (LSC), and Fiber Switch Capable (FSC). A connection may traverse multiple nodes with different switching technologies by the nesting of Label Switched Paths (LSPs). The order of nesting is PSC, L2SC, TDM, LSC and FSC. [22, 23, 27] have studied modeling and routing under GMPLS switching incompatibilities.

Jabbari et al. [22] proposed a channel graph of $\sum_{e\in\mathcal{E}} t_e$ nodes, where $\mathcal{E}$ is the set of links and, t_e is the number of switching types supported at link e. Each node in the graph corresponds to a switching type supported by a link. Link existence between nodes depends on the switching capability (horizontal links), and switching adaptation (vertical links). They used a variant of Yen's algorithm [28] as the routing algorithm. Their work was later extended in [27]. However, their solution does not consider the encapsulation order, which is important to ensure proper decapsulation. For example, if technology t_1 is encapsulated in technology t_2, and later in technology t_3, decapsulation of t_3 is required before t_2 to get t_1 back.

Shirazipour and Pierre [23] have used a graph of N nodes and E links as the model. Each link has a number of possible switching types, and each node has an adaptation function between the switching types. Similarly to [22], they have also used Yen's algorithm, and later optimized the returned solution by a Binary Integer Program (BIP).

GMPLS has a limited concept of adaptation [29], which may lead to technology adaptation and deadaptation complications as highlighted by [18, 29]. Responses to GMPLS are also mixed. For example, while [22, 27] have deemed it promising, Das et al. in [30] have argued that GMPLS is completely unusable as an intelligent unified control plane for various technologies in wide-area networks. Instead of GMPLS, Lamali et al. [24] have considered the Pseudo-Wire architecture, and have proposed a language-based Push Down Automaton (PDA) model. Each protocol is represented by an alphabet, and an adaptation function between alphabets is maintained at each domain. They also developed a polynomial routing algorithm based on the PDA. Their work has been extended in [31].

Kuipers and Dijkstra [18] proposed three methods to model technology incompatibilities. Device-based, where there are N devices, with links if two devices are connected. Layer-based, where there are N devices and L technology layers. Each node corresponds to a device that is aware of its technology layer. Links are either physical links or adaptation capability between technology layers. Stack-based, of at most NL nodes, where N is the number of devices, and L is the number of technology layers. Contrary to the layer-based model, each technology incompatibility is modeled using a different layer. Hence, there are no parallel links in this model. Nodes connected by a horizontal link in a layer can communicate directly without needing

[1]A feasible path faces no technology incompatibility.

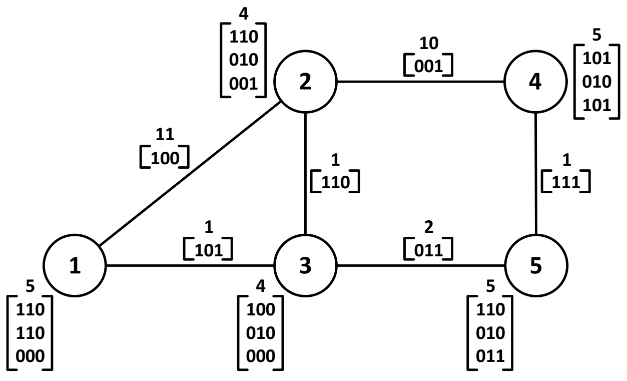
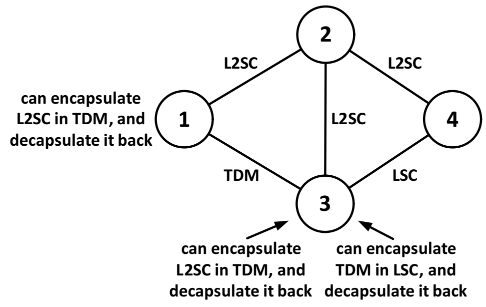
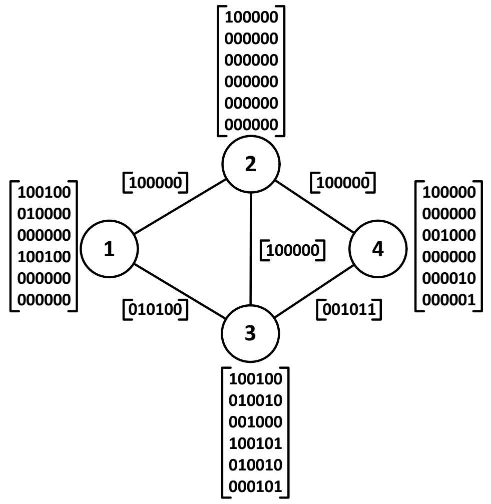


Figure 3: Our proposed model.



2
L2SC
L2SC
can encapsulate L2SC in TDM, and decapsulate it back
1
L2SC
4
TDM
LSC
3
can encapsulate L2SC in TDM, and decapsulate it back
can encapsulate TDM in LSC, and decapsulate it back

(a) GMPLS switching incompatibilities



(b) Our network model

Figure 4: Modeling of GMPLS switching incompatibilities.

any technology adaptation, while nodes connected by a vertical link in different layers can adapt between the technologies represented by the corresponding layers. The layer-based model is unidirectional, while the device-based and stack-based models are bidirectional. A variant of BFS [32] was proposed for the layer-based model, and an exact algorithm and a heuristic were proposed for the stack-based model. The layer-based model was later implemented in [33]. The problem of [18] is NP-complete, because of the imposed bandwidth constraint when traversing a node multiple times. If the bandwidth constraint in [18] were relaxed while looping is still allowed, the problem will reduce to a polynomial complexity as in [21, 24].

Similar to [18], we consider the more broad term of adaptations instead of confining to encapsulations. Contrary to the work of [21, 18, 24], which allow a connection to traverse a node multiple times, we allow only *simple*[2] paths as [22, 23]. Although confining to a simple path may lower the chance of finding feasible paths, utilizing only simple paths would lower the signaling complexity between domains in servicing connections. Our model scales well with the increase of graph size and number of technology incompatibilities, while the graph transformation approaches of [21, 22, 18, 27] may not.

3. Network Model

We propose that each domain n be characterized with a single positive additive weight γ_n and a binary technology (and technology encapsulation) matrix X_n, while each inter-domain link (u, v) is characterized by a single positive additive weight ℓ_{uv} and a binary technology (and technology encapsulation) vector Y_{uv}. γ_n can be assumed as the largest intra-domain shortest path cost between any of the STPs of the domain n. X_n represents the technology (and technology encapsulations) supported by domain n, while Y_{uv} represents the technologies (and technology encapsulations) supported by inter-domain link (u, v). Domains can only support a finite set of technologies due to the limitations of network components. From this finite set of technologies, only some adaptations might be possible, e.g., domains can never adapt to or from unsupported technologies and the technology adaptations may or may not be reciprocal. Inter-domain links have no technology adaptation capabilities.

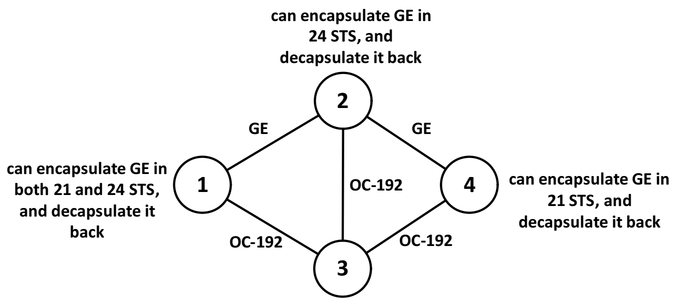
For example, in Figure 3, inter-domain link $(1, 3)$ with a weight of 1 supports technology 1 and 3 ($Y_{1,3}[1] = Y_{1,3}[3] = 1$), while domain 1 with a weight of 5 supports technology 1 and 2 ($X_1[1][1] = X_1[2][2] = 1$), and can adapt technology 1 to 2 or vice versa ($X_1[1][2] = X_1[2][1] = 1$). One denotes that the technology is supported (or can be adapted to/from) and zero, otherwise. Domains 1 and 3 can communicate directly using technology 1. Instead of using binary values to denote technology adaptation capabilities, we could also use technology adaptation cost as the values. In the remainder of the paper, when presenting our algorithms, we will confine to binary values.

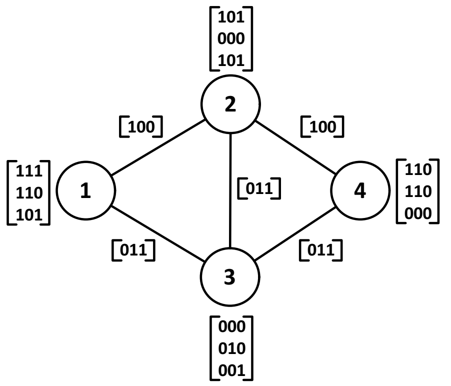
3.1. Application Examples

We provide a generic approach to model technology incompatibilities in a multi-domain multi-layer network. Our model is applicable for various application scenarios. We proceed with two application examples of our model.

In the case of GMPLS switching incompatibilities [22, 23, 27], a domain may support one or more GMPLS switching technologies discussed in Section 2, and some domain may be able to encapsulate and decapsulate between the switching technologies, as illustrated in Figure 4. To model the network, six distinct technology representations t_i are needed:

[2]A simple path has no repeating domains.


can encapsulate GE in 24 STS, and decapsulate it back
can encapsulate GE in both 21 and 24 STS, and decapsulate it back
can encapsulate GE in 21 STS, and decapsulate it back
GE
GE
OC-192
OC-192
OC-192
1
2
3
4
(a) SONET (de)adaptation incompatibilities



(b) Our network model

Figure 5: Modeling of SONET incompatibilities.

1. Technology 1 : L2SC
2. Technology 2 : TDM
3. Technology 3: LSC
4. Encapsulation 4 : L2SC encapsulated in TDM
5. Encapsulation 5 : TDM encapsulated in LSC
6. Encapsulation 6 : L2SC encapsulated in TDM encapsulated in LSC

Though originally only three switching technologies were considered, we need to also consider the encapsulation order such that proper decapsulation can be made. To explain Figure 4, we take domain 3 as an example. Domain 3 supports L2SC, TDM and LSC, so $X_3[1][1] = X_3[2][2] = X_3[3][3] = 1$. Supporting t_2 would imply the support of t_4, and supporting t_3 would imply the support of t_5 and t_6 as well. Hence, $X_3[4][4] = X_3[5][5] = X_3[6][6] = 1$. Since domain 3 can encapsulate L2SC in TDM, decapsulate L2SC from TDM, encapsulate TDM in LSC, and decapsulate TDM from LSC, $X_3[1][4] = X_3[4][1] = X_3[2][5] = X_3[4][6] = X_3[5][2] = X_3[6][4] = 1$.

Another example would be the SONET (de)adaptation incompatibilities studied in [18, 29]. Considering that 1 Gigabit/second Ethernet may be encapsulated in either 21 SONET STS channels or in 24 SONET STS channels as illustrated in Figure 5, three distinct technology representations are needed:

1. Technology 1 : Ethernet
2. Encapsulation 2 : Ethernet in 21 STS
3. Encapsulation 3 : Ethernet in 24 STS

To explain Figure 5, we take domain 1 as an example. Domain 1 supports both GE and SONET, so $X_1[1][1] = X_1[2][2] = X_1[3][3] = 1$. In this scenario, supporting SONET would imply the support of t_2, and t_3. Since domain 1 can encapsulate GE in both 21 STS and 24 STS and decapsulate it back, $X_1[1][2] = X_1[1][3] = X_1[2][1] = X_1[3][1] = 1$.

Based on the given two examples, we showed that our model is indeed generic and can be applied to model various types of technology incompatibilities. However, particular insights on the distinct technology representations are needed such that proper technology representations are considered. This can vary on a case-to-case basis. Many multi-layer and hybrid devices exist, which already solve some of the incompatibilities mentioned in this subsection. These devices are exactly what makes current multi-layer networking possible, and also so complex.

4. Problem Formulation

Problem 1. *Technology-Aware Shortest Path (TASP) problem: Consider an undirected graph $G = (\mathcal{N}, \mathcal{E}, \mathcal{T})$ consisting of a set $\mathcal{N}$ of N domains, a set $\mathcal{E}$ of E inter-domain links, and a set $\mathcal{T}$ of T incompatible technologies. Each domain $n \in \mathcal{N}$ is characterized by a single positive additive weight γ_n and a binary technology matrix X_n, while each inter-domain link $(u, v) \in \mathcal{E}$ is characterized by a single positive additive weight ℓ_{uv} and a binary technology vector Y_{uv}. Find a simple feasible path from a source domain s to a destination domain d such that the total path weight is minimized.*

To prove that the TASP problem is NP-hard, we show that any instance of the NP-hard Min-Sum Disjoint Paths (MSDP) problem [34] can be transformed in polynomial time to an instance of the TASP problem. Though the MSDP problem was intended for intra-domain routing, we refer to it in a multi-domain context.

Problem 2. *Min-Sum Disjoint Paths (MSDP): Given a graph $G = (\mathcal{N}, \mathcal{E})$, and k source-destination domain pairs $(s_1, d_1), ..., (s_k, d_k)$ with each $s_i, d_i \in \mathcal{N}$. Find k disjoint paths to connect all the source-destination domain pairs with minimized total length.*

Figure 6 illustrates the transformation of an instance of the MSDP problem to an instance of the TASP problem. We assume that each source-destination domain pairs (s_i, d_i) from the original graph supports only technology i. Thus, domains s_{2i} and s_{2i+1} with $1 \leq i \leq \lfloor \frac{k-1}{2} \rfloor$ have incompatible technologies and cannot communicate directly. Similarly, domains d_{2i-1} and d_{2i} with $1 \leq i \leq \lceil \frac{k-1}{2} \rceil$ also have incompatible technologies and cannot communicate directly. All other domains and inter-domain links from the original graph support all technologies.

We add a new domain x that supports all technologies, connecting x to s_1 by an inter-domain link supporting technology 1. For each domain pair (s_{2i}, s_{2i+1}), we add an adaptation domain a_{2i} that can adapt technology $2i$ to technology $2i+1$. Each domain a_{2i} is connected to s_{2i} by an inter-domain link supporting technology $2i$, and to s_{2i+1} by an inter-domain link supporting technology $2i + 1$. Thus, the domain pairs can now

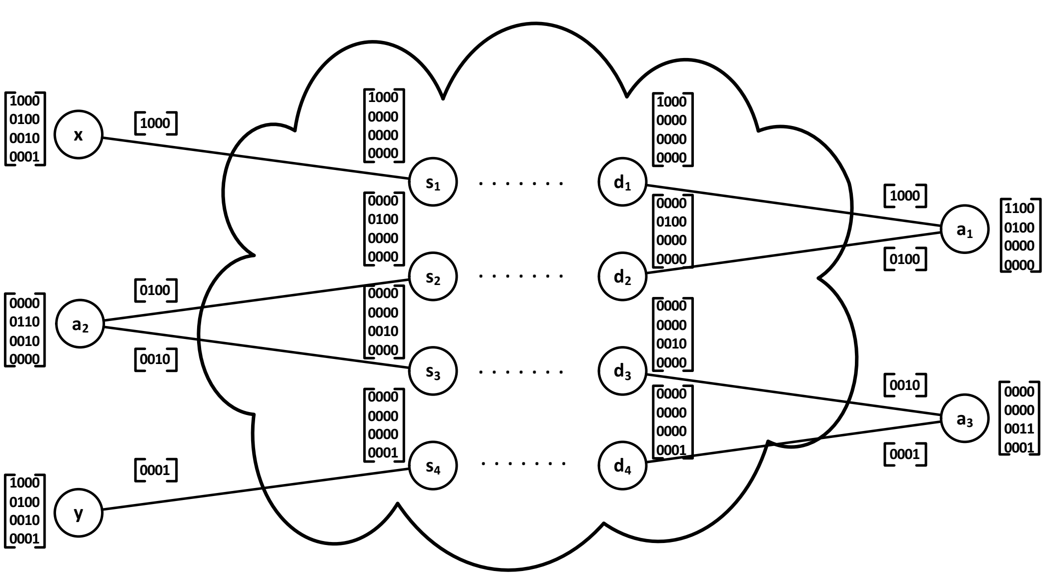


Figure 6: Transformation of an MSDP instance to an TASP instance.

communicate. Similarly, for each domain pair (d_{2i-1}, d_{2i}), we add an adaptation domain a_{2i-1} that can adapt technology $2i-1$ to technology $2i$. Each domain a_{2i-1} is connected to domain d_{2i-1} by an inter-domain link supporting technology $2i-1$, and to domain d_{2i} by an inter-domain link supporting technology $2i$. Either domain s_k or d_k with no connection to any of the new domains will be connected to a new destination domain y that supports all technologies by an inter-domain link supporting technology k.

A solution to both the MSDP and TASP problems exists if a simple feasible path exists from domain x to domain y. According to [34], the unweighted MSDP problem (i.e., all links have weight 1) is hard to approximate within $\Omega(E^{1-\epsilon})$ for any constant $\epsilon > 0$. Since a fully polynomial-time approximation scheme for the TASP problem is unlikely to exist, we focus on developing exact and heuristic algorithms.

5. Routing Algorithms

We propose Algorithm 1, which we refer to as the Exact Technology-Aware Routing Algorithm ($ETARA$), to solve the TASP problem. A feasible path does not succumb to technology incompatibilities. For example, although domains 1 and 3 can communicate directly using technology 1, and domains 3 and 5 can communicate directly using technologies 2 and 3 in Figure 3, $P_1 = 1 - 3 - 5$ of weight 17 is unfeasible since domain 3 cannot adapt technologies 1 to technology 2. The actual shortest simple feasible path from domain 1 to domain 5 is $P_2 = 1_1 - 2_2 - 3_2 - 5_2$ of weight 32. Although P_2 is longer than P_1, technology 1 can be adapted to technology 2 at domain 2 before proceeding to domain 5 through domain 3. Path $P_3 = 1_1 - 3_1 - 2_2 - 3_2 - 5_2$ of weight 27 is ignored since it is not a simple path.

$ETARA$ implements a k-shortest paths approach [35] by maintaining a list of feasible subpaths at each intermediate domain. To reduce the number of subpaths maintained, we prune out subpaths that use unnecessary technology adaptation. For example, path $P_4 = 1_1 - 3_1 - 2_2$, which adapts technology 1 to 2 at domain 2 is pruned, since path $P_5 = 1_1 - 3_2 - 2_2$ does not use technology adaptation at domain 2. However, if the identical subpaths use different technologies, e.g., $P_6 = 1_1 - 2_1 - 3_1$ and $P_7 = 1_1 - 2_2 - 3_2$, both of them will be kept since further subpath extension might need to use either of them. The subpaths of the shortest feasible path may not necessarily be shortest paths themselves. For instance, P_2 uses subpath $P_8 = 1_1 - 2_2$ with a weight of 20, which is longer than subpath P_4 of weight 15. Using subpath P_4 instead of P_8 would however lead to a path P_3 that contains loops. The shortest feasible path may also be unidirectional, and may not be simply redirected to find the shortest feasible path in the reversed direction, due to unidirectional technology adaptations of domains.

5.1. Pseudocodes

$ETARA$ uses algorithms SPT in line 1 and $LOOK-AHEAD$ in line 15 to reduce its search space and improving its running time. We will explain the pseudocode starting from the SPT.

SPT functions to compute a shortest feasible paths tree Z rooted at the destination domain. SPT may not span all domains since it opts only for the best subpath that may lead to the shortest feasible path and ignores subpaths with higher weight. SPT first initializes the tentative weight D'_{nt} and predecessor π'_{nt} of all entries n_t (i.e., domain and technology pairs) in lines 1-3. For each technology supported by the destination domain, the tentative weight of the corresponding entry D'_{dt} is updated with the weight of the destination domain γ_d in line 6, and inserted into the queue Q' in line 7. While Q' is not empty, the entry u_t with the lowest tentative weight is extracted in line 9. SPT checks if the current subpath of u_t can be extended to its adjacent domains in line 10. For each subpath extension, SPT ensures the feasibility of subpath in line 12, the subpath has no unnecessary technology adaptation in lines 15-16, and the

Algorithm 1 ETARA(G, s, d)

```
 1: $Z \leftarrow$ SPT$(G, d)$
 2: for each $n \in \mathcal{N}$
 3:   for each $t \in \mathcal{T}$
 4:     $C_{nt} \leftarrow 0$
 5: $\Delta \leftarrow \infty$
 6: for each $t \in \mathcal{T}$
 7:   if $X_s[t][t] = 1$
 8:     $k \leftarrow ++C_{st}, D_{stk} \leftarrow \gamma_s, \pi_{stk} \leftarrow$ NIL
 9:     INSERT$(Q, s_{tk}, D_{stk}, \pi_{stk})$
10: while $Q \neq \phi$
11:   $u_{tk} \leftarrow$ EXTRACT-MIN$(Q)$
12:   if $u = d$
13:     return solution
14:   if $D_{utk} < \Delta$
15:     LOOK-AHEAD$(u_{tk}, \Delta, Z)$
16:     for each $v_x \in adj[u_t]$
17:       GO $\leftarrow$ FALSE
18:       if $X_v[t][x] = Y_{uv}[t] = 1$
19:         if $D_{utk} + \ell_{uv} + \gamma_v < \Delta$
20:           GO $\leftarrow$ TRUE
21:         if $x \neq t$ and $!(X_u[x][x] = 0$ or $Y_{uv}[x] = 0$ or
            $C_{ux} = 0)$
22:           GO $\leftarrow$ FALSE
23:         $a_{bc} \leftarrow u_{tk}$
24:         while $a_{bc} \neq NIL$ and GO = TRUE
25:           if $a = v$
26:             GO $\leftarrow$ FALSE
27:           $a_{bc} \leftarrow \pi_{abc}$
28:         if GO = TRUE
29:           $y \leftarrow ++C_{vx}$ /* $y \leq k_{max}$ for $kTARA$ */
30:           $D_{vxy} \leftarrow D_{utk} + \ell_{uv} + \gamma_v$
31:           $\pi_{vxy} \leftarrow u_{tk}$
32:           INSERT$(Q, v_{xy}, D_{vxy}, \pi_{vxy})$
```

Algorithm 2 SPT(G, d)

```
 1: for each $n \in \mathcal{N}$
 2:   for each $t \in \mathcal{T}$
 3:     $D'_{nt} \leftarrow \infty, \pi'_{nt} \leftarrow NIL$
 4: for each $t \in \mathcal{T}$
 5:   if $X_d[t][t] = 1$
 6:     $D'_{dt} \leftarrow \gamma_d$
 7:     INSERT$(Q', d_t, D'_{dt}, \pi'_{dt})$
 8: while $Q' \neq \phi$
 9:   $u_t \leftarrow$ EXTRACT-MIN$(Q')$
10:   for each $v_x \in adj[u_t]$
11:     GO $\leftarrow$ FALSE
12:     if $X_v[t][x] = X_v[x][t] = Y_{uv}[t] = Y_{vu}[x] = 1$
13:       if $D'_{ut} + \ell_{uv} + \gamma_v < D'_{vx}$
14:         GO $\leftarrow$ TRUE
15:       if $x \neq t$ and $!(X_u[x][x] = 0$ or $Y_{uv}[x] = 0)$
16:         GO $\leftarrow$ FALSE
17:       $a_b \leftarrow u_t$
18:       while $a_b \neq NIL$ and GO = TRUE
19:         if $a = v$
20:           GO $\leftarrow$ FALSE
21:         $a_b \leftarrow \pi'_{ab}$
22:       if GO = TRUE
23:         $D'_{vx} \leftarrow D'_{ut} + \ell_{uv} + \gamma_v$
24:         $\pi'_{vx} \leftarrow u_t$
25:         INSERT$(Q', v_x, D'_{vx}, \pi'_{vx})$
```

Algorithm 3 LOOK-AHEAD$(u_{t,k}, \Delta, Z)$

```
 1: if $u_t \in Z$
 2:   GO $\leftarrow$ TRUE, $a_b \leftarrow u_t$
 3:   while $\pi'_{ab} \neq$NIL and GO = TRUE
 4:     $e_f \leftarrow \pi'_{ab}, p_{tk} \leftarrow u_{tk}$
 5:     while $p_{tk} \neq$ NIL
 6:       if $p = e$
 7:         GO $\leftarrow$ FALSE
 8:       $p_{tk} \leftarrow \pi_{ptk}$
 9:     $a_b \leftarrow e_f$
10:   if $D_{utk} + D'_{ut} - \gamma_u < \Delta$ and GO = TRUE /*FTARA stops*/
11:     $\Delta \leftarrow D_{utk} + D'_{ut} - \gamma_u$
```

subpath is simple in lines 17-21. If a subpath extension is feasible, the corresponding entry v_x is updated in lines 23-24, and inserted to Q' in line 25.

After utilizing SPT, $ETARA$ proceeds by initializing all the entries counter C_{nt} of each domain and technology pair to zero in lines 2-4. $ETARA$ maintains k subpath entries for each domain and technology pair. k could grow exponentially (albeit bounded by the maximum number of possible simple paths between two domains). The optimum weight Δ is set to infinity since no feasible path has been found so far in line 5. For each technology supported by the source domain, the tentative weight of the corresponding entry D_{stk} is updated with the weight of the source domain γ_s, its predecessor entry π_{stk} set to empty in line 8, and inserted into the queue Q in line 9. While Q is not empty, the entry u_t with the lowest tentative weight D_{utk} is extracted in line 11. If u_t is the destination domain, the optimum path P is returned. Else, if D_{utk} is lower than Δ, $ETARA$ proceeds with algorithm $LOOK-AHEAD$ in line 15 to tighten the value of Δ. We will explain $LOOK-AHEAD$ in the next paragraph. Then, $ETARA$ checks if the current subpath of the u_{tk} can be extended to its adjacent domains in line 16. $ETARA$ ensures the feasibility of the subpath in line 18, the subpath tentative weight is less than Δ in lines 19-20, the subpath has no unnecessary technology adaptation in lines 21-22, and the subpath is simple in lines 23-27. If a subpath extension is feasible, the corresponding entry v_{xy} is updated in lines 28-31, and inserted into Q in line 32.

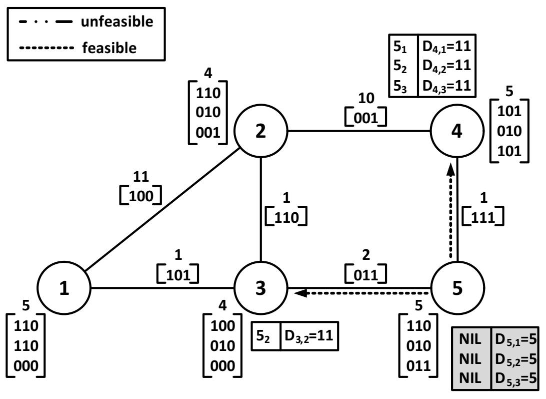
$LOOK-AHEAD$ functions to tighten the value of Δ such that any subpath extension with higher tentative weight can be ignored. $LOOK-AHEAD$ uses the shortest feasible path tree Z returned by SPT while doing so. For each extracted entry u_{tk} of Q, $LOOK-AHEAD$ checks whether u_t is a part of Z in line 1. $LOOK-AHEAD$ confirms that the predecessors of the u_{tk} do not coincide with any of the entries in the branch of u_t in Z, in lines 2-9. If so, $LOOK-AHEAD$ tightens Δ to the weight of the stitched end-to-end feasible path. Although a feasible path has been found by $LOOK-AHEAD$, $ETARA$ will still proceed since this might not be the shortest feasible path from s to d. However, if only a feasible path is required (not necessarily the shortest feasible path), one could terminate with the path as the solution. We call this variant the Feasible Technology-Aware Routing Algorithm ($FTARA$).

By limiting the maximum number of maintained entries k_{max} for each domain and technology pair (similarly to the approach of [36] or that of a k-shortest path algorithm [35]), a heuristic form of $ETARA$, which we refer to as (k) Technology-Aware Routing Algorithm ($kTARA$), is obtained. $kTARA$ is heuristic, because we do not know upfront how big to choose k_{max} to find an exact result (while $ETARA$ automatically adapts to the appropriate value). Hence, if we choose k in $kTARA$ smaller than the k_{max} used by $ETARA$ on the same instance, $kTARA$ will fail to find the optimal feasible path.

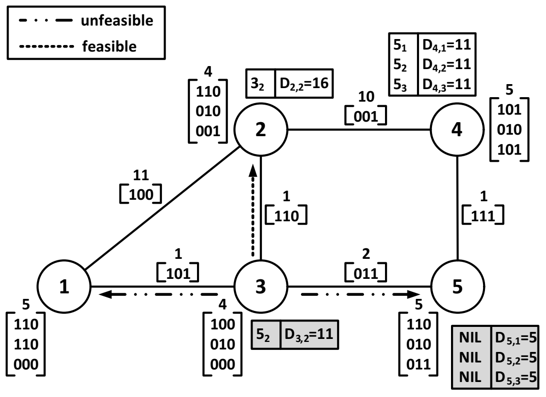
5.2. Illustrative Example

Consider the problem of finding the shortest simple feasible path from domain 1 to domain 5 in the network shown in Figure 3. $ETARA$ starts by invoking SPT. Since the destination domain 5 supports t_1, t_2, and t_3, the three valid entries (i.e., domain-technology pair) 5_1, 5_2, and 5_3 of domain 5 will have a tentative weight of $\gamma_d = 5$, $D'_{5,1} = D'_{5,2} = D'_{5,3} = 5$. The entries are inserted into the queue Q'. Then, the three entries are extracted from Q' one by one. Since domain 5 can communicate with domain 3 using technology 2, and with domain 4 using all technologies, $D'_{3,2}$, $D'_{4,1}$, $D'_{4,2}$ and $D'_{4,3}$ are relaxed, added to Q' and their predecessor is set, as illustrated in Figure 7a. Then, entry 3_2 with $D'_{3,2} = 11$ is extracted from Q. Path $3_2 - 1_2$ is not feasible since inter-domain link (2, 1) does not support technology 2, and path $3_2 - 5_2$ contains a loop. However, domain 3 can communicate with domain 2 using technology 2, as illustrated in Figure 7b. After that, as illustrated in Figure 7c, no further subpath extensions are feasible or optimal.

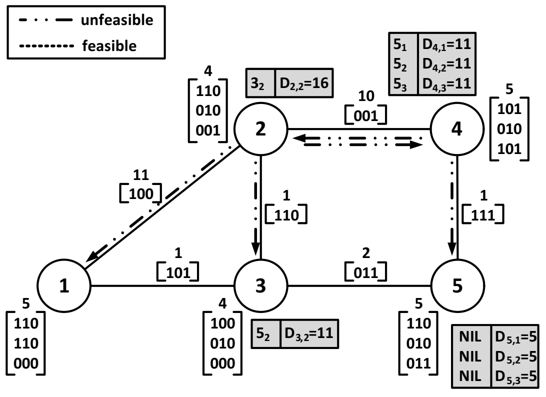
After utilizing SPT, $ETARA$ proceeds by initializing all the valid entries (i.e., domain-technology-number pair) of the source domain, $1_{1,1}$ and $1_{2,1}$, updating their tentative distance, $D_{1,1,1} = D_{1,2,1} = 5$, and inserting them into the queue Q as illustrated in Figure 8a. Unlike SPT, a maximum number of k_{max} entries could be maintained by $ETARA$ for each domain-technology pair. The two entries are then extracted from Q one by one while checking whether the subpath could be extended to domains 2 or 3. Since domain 1 can communicate with domain 2 using technology 1, with possible adaptation to technology 2, and with domain 3 using technology 1, entries $2_{1,1}$, $2_{2,1}$ and $3_{1,1}$ are inserted into Q, as illustrated in Figure 8a. Δ remains at infinity since these entries are not in Z. Then, entry $3_{1,1}$ with $D_{ntk} = 10$ is extracted from Q', which is also not part of Z. Path $1_1 - 3_1 - 5_1$ is not feasible due to technology restriction of inter-domain link (3, 5), and paths $1_1 - 3_1 - 1_1$ and $1_1 - 3_1 - 1_2$ contain a loop. However, domain 3 can communicate with domain 2 using technology 1 and further adapt it to technology 2 as in Figure 8b. Then, entries $2_{1,2}$ and $2_{2,2}$ with $D_{ntk} = 15$ are extracted from Q as illustrated in Figure 8c. Paths from domain 3 to domains 1, 2, and 4 are not feasible due to looping or



(a)



(b)



(c)

Figure 7: Illustrative example of SPT.

technology incompatibility. Although entry 3_2 exists in Z, the existence of domain 2 among its predecessors when checked by $LOOK-AHEAD$ prevents $ETARA$ from updating Δ. When entry $3_{2,2}$ is extracted from Q in Figure 8d, Algorithm 3 confirms that the predecessors of the entry do not coincide with any of the entries in the branch of $3_{2,2}$ in Z. Hence, $ETARA$ tightens Δ to 32, the weight of the stitched feasible path $1_1 - 2_2 - 3_2 - 5_2$. Then, entries $3_{1,2}$ and $3_{2,1}$ are extracted. However, all subpath extensions from them are not feasible or non-optimal. Hence, $ETARA$ terminates and the shortest feasible path of weight 32 can be traced back from the earlier stitched path.

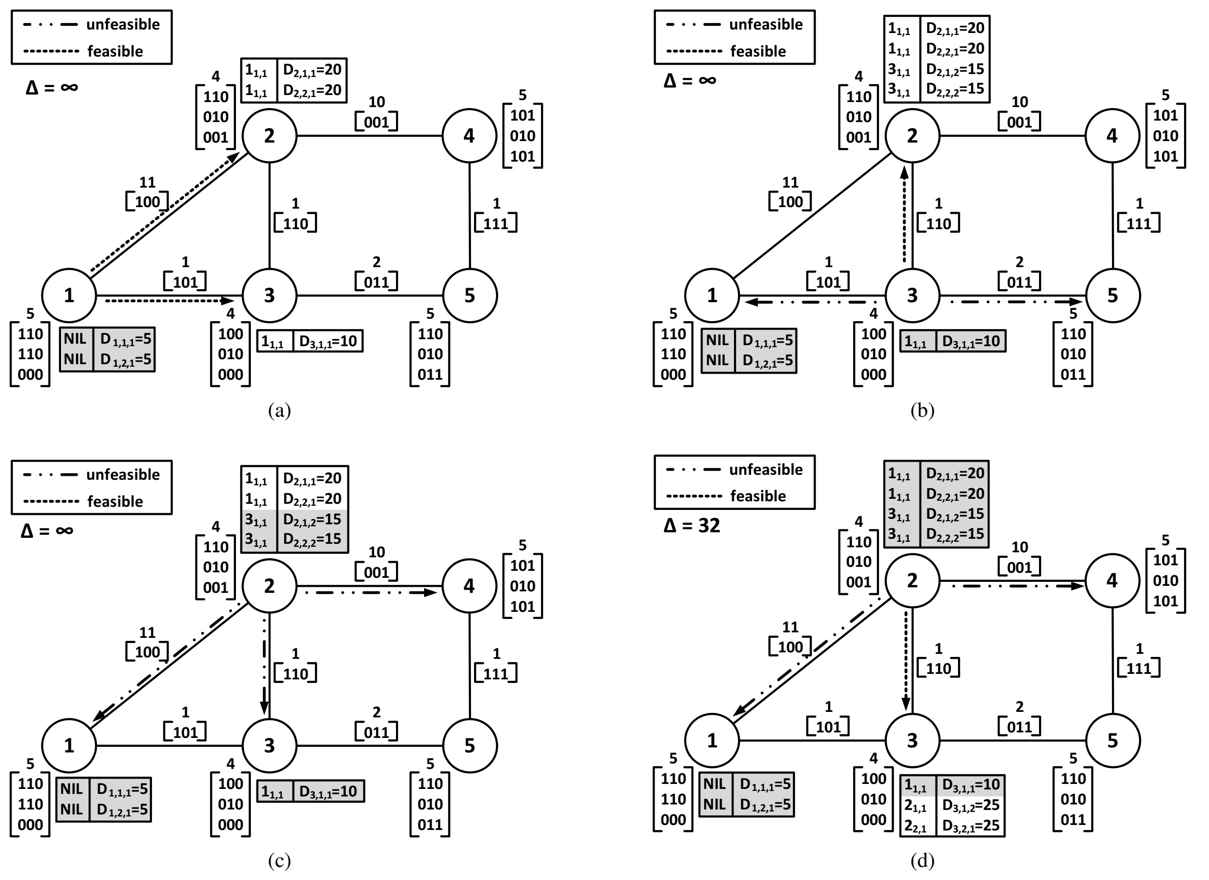


Figure 8: Illustrative example of *ETARA*.

5.3. Time complexity

The initialization phase in lines 1-9 of *ETARA* has a time complexity of $O(NT \log(NT) + NET)$. The extract procedure in line 11 takes at most $O(k_{max}NT \log(k_{max}NT))$ time, since Q contains at most $k_{max}NT$ entries. Algorithm 3 in line 15 takes at most $O(N^2)$ time. The for loop in line 16 takes at most $O(k_{max}ET)$ time, since it is invoked at most $k_{max}T$ times for each side of each inter-domain link. Lines 17-32 take at most $O(N)$ time. Summing up all the contributions, the worst-case complexity of *ETARA* is $O(k_{max}NT \log(k_{max}NT) + k_{max}NET + N^2)$. k_{max} can grow exponentially with the input, implying that *ETARA* has an exponential running time. However, when k_{max} is bounded, as in *kTARA*, the complexity is polynomial.

5.4. Correctness proof

A brute force approach would consider all possible subpath extensions from s to d, which can be time and memory consuming. In order to make the searching process more efficient, *ETARA* prunes out all subpath extensions that is unfeasible, have loops, use unnecessary technology adaptations or with $D_{ntk} > \Delta$. Our search-space reduction will never remove any subpath of the shortest feasible path, if it exists. If a subpath violates the technology continuity constraint or has loops, it can never be a part of the shortest feasible path and thus is safe to be ignored. If a domain could be reached directly using a technology, then it is safe to ignore any subpaths with the exact domain sequence so far that use the technology adaptation capability at the domain to adapt it to the technology. Whenever a feasible subpath is found by stitching the current subpath with the branch containing the current entry in Z, its D_{ntk} is compared with Δ. If D_{ntk} is lower, then Δ is updated to D_{ntk}. If D_{ntk} is higher, the feasible subpath could safely be ignored since another shorter subpath with the weight of Δ has already been found. We do not need to consider any extracted entry or subpath extension with $D_{ntk} > \Delta$. Upon termination, *ETARA* will always finds the solution by retracing the shortest feasible path, if it exists. *ETARA* is thus guaranteed to be exact. When limiting k_{max} using *kTARA*, exactness can no longer be guaranteed. If we stop once a feasible path is found using *FTARA*, the feasible path may not be the shortest feasible path.

6. Simulations

We study the performance of our algorithms in four network topologies, namely the Erdős-Rényi random network [37], the lattice network, the Waxman network [38], and the GÉANT network [39]. For the random network, the probability of an inter-domain link existence is reflected by $\frac{logN}{N}$. The lattice network may resemble the inner core of an ultra-long-reach optical data plane systems [40]. We use a square lattice network of $n \times n$ dimension where $n = \sqrt{N}$. The Waxman network is frequently used to model communication networks [41] due to its unique property of decaying link existence over distance. In the Waxman network, the domains are uniformly positioned in the plane, and the inter-domain link existence is reflected by $ae^{\frac{\ell_{uv}}{b\beta}}$, where β is the maximum distance between any two domains in the plane, and we set $a = 0.1$ and $b = 0.5$. The GÉANT network is a realistic pan-European network interconnecting multiple countries. We refer to [42] for the GÉANT network topology.

We choose that a domain has a random weight in the range of 0.1 to 0.3, while an inter-domain link has a random weight in the range of 0.2 to 0.5 (except for the Waxman network, where the link's weight depends on the coordinates between the endpoints). No self-loops or parallel inter-domain links are allowed. The probability that a domain or an inter-domain link supports a technology is reflected by p and $\sqrt{p}$ respectively. The probability that a domain supports a unidirectional technology adaptation is p to the power of two, p^2, and the probability that a domain supports a bidirectional technology adaptation is p to the power of three, p^3. A domain can only adapt between its supported technologies. We compare the performance of $ETARA$, $1TARA$, $FTARA$ and the classical Dijkstra's algorithm [43] in finding the shortest feasible path from a random s to a random d, while varying several network characteristics. All simulation results are averaged over a hundred thousand runs. In all simulations, we terminate when the number of entries processed exceeded a million, since a solution might not exist in the randomly generated multi-domain networks.

6.1. Effect of Network Size

Figure 9 illustrates the performance of $ETARA$, $1TARA$, $FTARA$, and Dijkstra's algorithm as a function of N. The optimality ratio reflects how often the algorithm was able to retrieve the shortest feasible path. Since $ETARA$ is exact, an optimality ratio for $ETARA$ below one indicates that in some instances, no feasible path existed. $ETARA$ performs best in finding the optimal feasible path while $1TARA$ comes second due to the limitation of the number of maintained subpaths at each domain. The Dijkstra's algorithm performs badly because the probability that the shortest path being also the shortest feasible path decreases as the network size increases. $FTARA$ has lower optimality ratio since it terminates whenever a feasible path is found, even though the feasible path may be sub-optimal. If only a feasible path is needed, $FTARA$ performs similarly to $ETARA$, as indicated by the feasibility ratio.

6.2. Effect of the Number of Technologies

Figure 10 illustrates the optimality ratio of $ETARA$, $1TARA$, $FTARA$, and Dijkstra's algorithm as a function of T. As the number of technologies increases, the optimality ratio for finding the optimal path increases for both $ETARA$ and $1TARA$, since we assumed an identical probability of technology existence for all technologies. The performance of $FTARA$ dropped back after a certain number of T is reached. When the number of possible feasible paths increases due to the increase of T, the probability that $FTARA$ terminates whenever a sub-optimal feasible path is found increases. Having higher number of technologies with similar p increases the chance of technology continuity from s to d, thus increasing the performance of Dijkstra's algorithm. Different results may be observed if each technology has a different probability of existence.

6.3. Effect of the Probability of Technology Existence

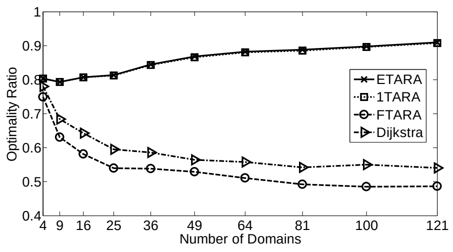
Figure 11 illustrates the optimality ratio of $ETARA$, $1TARA$, $FTARA$, and Dijkstra's algorithm as a function of the technology probability p. We notice an improved performance as p increases. Vice versa, when p decreases, the multi-domain networks are more likely to break into islands of technologies, reducing the optimality ratio.

6.4. Running Time Comparison

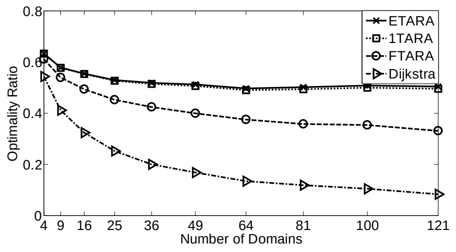
Figure 12 plots the average running time per feasible request for $ETARA$, $FTARA$, and $1TARA$. When $ETARA$ is able to find an optimal feasible path, then $FTARA$ also is guaranteed to find a (not necessarily feasible) path. $1TARA$ guarantees to only return feasible paths, but it does not guarantee to find one if it exists. Generally, the number of entries processed from Q' increases as N increases because more subpaths need to be considered. With the increase of the number of subpaths due to the increase of N, T or p, the benefit of using Algorithm 3 increases. $ETARA$ has highest running time, while $1TARA$ will becomes faster than $FTARA$ as the network average path length becomes longer. Although $FTARA$ can be faster than $1TARA$, $FTARA$ seldom returns the most optimal path, as shown earlier in Figure 9. An important advantage of $kTARA$ over $ETARA$ and $FTARA$ is that it always terminates fast, even when no solution in the multi-domain network exists, while $ETARA$ and $FTARA$ may continue searching for a considerable time to come to that conclusion. Since $kTARA$ is fast (even in absence of a feasible path) and often finds the shortest feasible path, it is our recommended algorithm for multi-domain routing with technology incompatibilities.

7. Conclusion

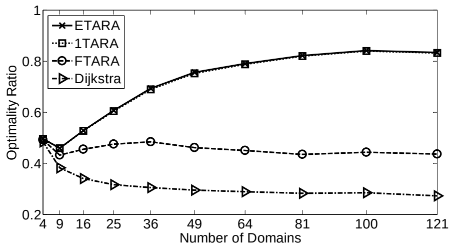
In this paper, we have studied the problem of finding paths in multi-domain multi-layer optical networks with technology incompatibilities. We have proposed a technology representation consisting of a technology matrix at each domain and a technology vector at each inter-domain link. In combination with costs (which could represent available bandwidth, monetary costs, impairment values, etc.) assigned to domains and inter-domain links, the technology matrices and vectors allow for flexibility in including also conversion costs, (different adaptation) policies, etc. We subsequently proposed an exact path-finding algorithm $ETARA$ and heuristic $kTARA$ to compute a technology-aware shortest feasible path from a source domain to a destination domain. The algorithms can be easily modified to take different objective functions (e.g., maximizing bandwidth) or QoS constraints into account. For future work, our conceptual contributions/algorithms in dealing with technology



(a) Erdős-Rényi

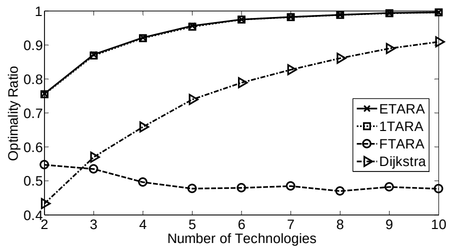


(b) Lattice

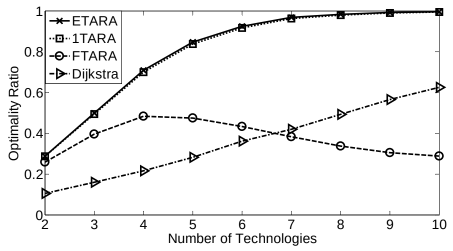


(c) Waxman

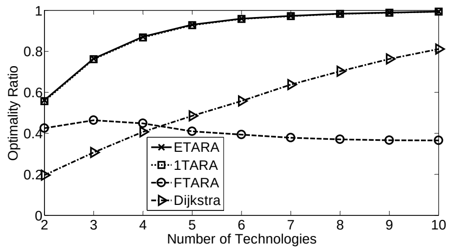
Figure 9: Effect of N on the optimality and feasibility ratio ($T = 3$ and $p = 0.6$).



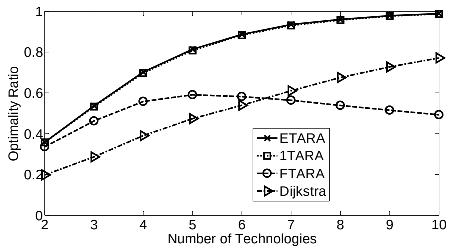
(a) Erdős-Rényi



(b) Lattice



(c) Waxman



(d) GÉANT

Figure 10: Effect of T on the optimality ratio ($N = 49$ (23 for GÉANT network) and $p = 0.6$).

incompatibilities could also be helpful in the Software-Defined Networking (SDN) context as well.

Acknowledgement

This research was supported by the GigaPort 2012 and 2013 Research on Networks project coordinated by SURFnet.

8. Reference

[1] [Online]. Available: http://home.web.cern.ch/about/accelerators/large-hadron-collider

[2] [Online]. Available: http://www.sdss.org/

[3] [Online]. Available: https://www.skatelescope.org/

[4] [Online]. Available: http://www.lsst.org/lsst/

[5] [Online]. Available: http://www.surf.nl/en/about-surf/subsidiaries/surfnet

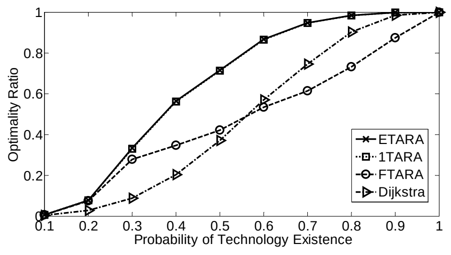
[6] [Online]. Available: http://www.glif.is/

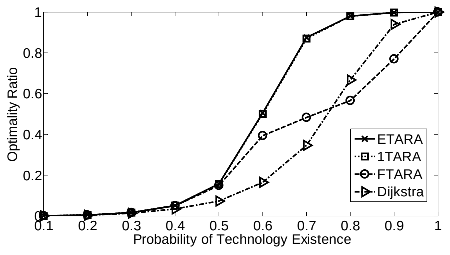
[7] [Online]. Available: http://www.surf.nl/en/knowledge-and-innovation/innovationprojects/2009/netherlight/about-netherlight/index.html

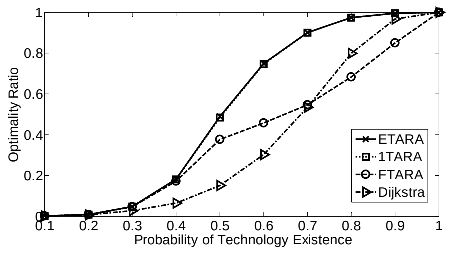
[8] "Architecture for the automatically switched optical network (ASON)," *ITU-T G. 8080/Y. 1304*, 2001.

[9] A. Farrel, J. Vasseur, and J. Ash, "A path computation element (pce)-based architecture, rfc 4655," 2006.

[10] R. Muñoz, R. Casellas, R. Martínez, and R. Vilalta, "Pce: What is it, how


(a) Erdős-Rényi


(b) Lattice


(c) Waxman

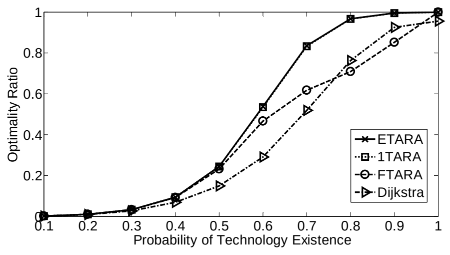
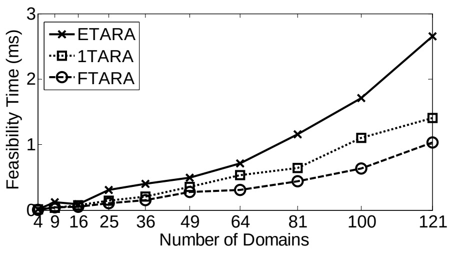

(d) GÉANT

Figure 11: Effect of p on the optimality ratio ($N = 49$ (23 for GÉANT network) and $T = 3$).


(a) Erdős-Rényi

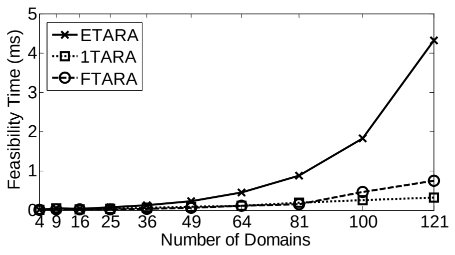

(b) Lattice

Figure 12: Effect of N on the running time ($T = 3$ and $p = 0.6$).

does it work and what are its limitations?" *IEEE. J. Lightwave Technol.*, vol. 32, no. 4, pp. 528–543, 2014.

[11] G. Roberts, T. Kudoh, I. Monga, J. Sobieski, and J. Vollbrecht, "Network services framework v1.0," Open Grid Forum, 2010. [Online]. Available: http://www.ogf.org/documents/GFD.173.pdf

[12] G. Roberts, T. Kudoh, I. Monga, J. Sobieski, J. MacAuley, and C. Guok, "NSI connection service protocol v2.0," GFD-R-P.212, Open Grid Forum, 2014. [Online]. Available: http://www.ogf.org/documents/GFD.212.pdf

[13] [Online]. Available: http://www.glif.is/publications/press/20131203.html

[14] S. Yang and F. Kuipers, "Energy-aware path selection for scheduled lightpaths in IP-over-WDM networks," in *IEEE Annu. Symp. on Commun. and Vehicular Technology (SCVT'11)*.

[15] A. L. Chiu and E. H. Modiano, "Traffic grooming algorithms for reducing electronic multiplexing costs in WDM ring networks," *J. Lightw. Technol.*, vol. 18, 2000.

[16] F. Iqbal, S. Yang, and F. Kuipers, "Energy considerations in EoS-over-WDM network configuration," in *IEEE Sustainable Internet and ICT for Sustainability (SustainIT'13)*, 2013.

[17] A. Hadjiantonis, M. A. Ali, H. Chamas, W. Bjorkman, S. Elby, A. Khalil, G. Ellinas, and N. Ghani, "Evolution to a converged layer 1, 2 in a global-scale, native ethernet over WDM-based optical networking architecture," *IEEE J. Sel. Areas Commun.*, vol. 25, pp. 1048–1058, 2007.

[18] F. Kuipers and F. Dijkstra, "Path selection in multi-layer networks," *Comput. Commun.*, vol. 32, no. 1, pp. 78–85, 2009.

[19] A. Martinez, M. Yannuzzi, V. Lopez, D. Lopez, W. Ramirez, R. Serral-Gracia, X. Masip-Bruin, M. Maciejewski, and J. Altmann, "Network management challenges and trends in multi-layer and multi-vendor settings for carrier-grade networks," *IEEE Commun. Surveys Tutorials*, 2014.

[20] J. van der Ham, F. Dijkstra, R. Łapacz, and J. Zurawski, "Network markup language base schema version 1," GFD-R-P.206, Open Grid Forum, May 2013. [Online]. Available: http://www.ogf.org/documents/GFD.206.pdf

[21] I. Chlamtac, A. Farago, and T. Zhang, "Lightpath (wavelength) routing in large WDM networks," *IEEE J. Sel. Areas Commun.*, vol. 14, no. 5, pp. 909–913, 1996.

[22] B. Jabbari and G. Shujia, "On constraints for path computation in multi-layer switched networks," *IEICE Trans. on Commun.*, vol. 90, no. 8, pp. 1922–1927, 2007.

[23] M. Shirazipour and S. Pierre, "Multi-layer/multi-region path computation with adaptation capability constraints," in *IEEE Global Telecommunications Conf. (GLOBECOM'10)*, 2010.

[24] M. L. Lamali, H. Pouyllau, and D. Barth, "Path computation in multi-layer multi-domain networks," in *IFIP/TC6 Networking Conf. (NETWORKING'12)*, 2012, pp. 421–433.

[25] E. Mannie, Ed., "Generalized multi-protocol label switching (GMPLS) architecture," *IETF RFC 3945*, 2004.

[26] D. Papadimitriou, M. Vigoureux, K. Shiomoto, D. Brungard, and J. Le Roux, "Generalized MPLS (GMPLS) protocol extensions for multi-layer and multi-region networks (MLN/MRN)," *IETF RFC 6001*, 2010.

[27] S. Gong and B. Jabbari, "Optimal and efficient end-to-end path computation in multi-layer networks," in *IEEE Int. Conf. on Commun. (ICC'08)*, 2008, pp. 5767–5771.

[28] J. Y. Yen, “Finding the k shortest loopless paths in a network,” *Manag. Sci.*, vol. 17, no. 11, pp. 712–716, 1971.

[29] F. Dijkstra, B. Andree, K. Koymans, J. van der Ham, P. Grosso, and C. de Laat, “A multi-layer network model based on ITU-T G. 805,” *Comp. Netw.*, vol. 52, no. 10, pp. 1927–1937, 2008.

[30] S. Das, G. Parulkar, and N. McKeown, “Why OpenFlow/SDN can succeed where GMPLS failed,” in *European Conf. Exhibit. on Opt. Commun. (ECOC’12)*, 2012.

[31] M. L. Lamali, H. Pouyllau, and D. Barth, “Path computation in multi-layer multi-domain networks: A language theoretic approach,” *Comp. Commun.*, vol. 36, no. 5, pp. 589–599, 2013.

[32] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to algorithms*. The MIT press, 2001.

[33] L. Xu, F. Dijkstra, D. Marchal, A. Taal, P. Grosso, and C. De Laat, “A declarative approach to multi-layer path finding based on semantic network descriptions,” in *IEEE Int. Conf. on Opt. Netw. Des. and Modeling (ONDM’09)*, 2009.

[34] P. Zhang and W. Zhao, “On the complexity and approximation of the min-sum and min-max disjoint paths problems,” in *Proc. Int. Conf. on Combinatorics, Algorithms, Probabilistic and Experimental Methodologies*, 2007, pp. 70–81.

[35] E. I. Chong, S. Maddila, and S. Morley, “On finding single-source single-destination k shortest paths,” *J. Comput. Inform.*, vol. 95, pp. 40–47, 1995.

[36] P. Van Mieghem and F. Kuipers, “Concepts of exact QoS routing algorithms,” *IEEE/ACM Trans. Netw.*, vol. 16, no. 9, pp. 851–864, 2004.

[37] P. Erdős and A. Rényi, “On random graphs,” *Publicationes Mathematicae Debrecen*, vol. 6, pp. 290–297, 1959.

[38] B. M. Waxman, “Routing of multipoint connections,” *IEEE J. Sel. Areas Commun.*, vol. 6, no. 9, pp. 1617–1622, 1988.

[39] [Online]. Available: http://www.geant.net/

[40] A. R. Moral, P. Bonenfant, M. Krishnaswamy, and O. In, “The optical Internet: architectures and protocols for the global infrastructure of tomorrow,” *IEEE/ACM Trans. Netw.*, vol. 39, no. 7, pp. 152–159, 2001.

[41] P. Van Mieghem, “Paths in the simple random graph and the waxman graph,” *Probability in the Engineering and Informational Sciences*, vol. 15, no. 04, pp. 535–555, 2001.

[42] A. Bianzino, L. Chiaraviglio, and M. Mellia, “Grida: a green distributed algorithm for backbone networks,” in *IEEE Online Conf. on Green Commun. (GreenCom’11)*, 2011, pp. 113–119.

[43] E. W. Dijkstra, “A note on two problems in connexion with graphs,” *Numerische Mathematik*, pp. 269–271, 1959.