Resource discovery and allocation for federated virtualized infrastructures

Pittaras, C.; Papagianni, C.; Leivadeas, A.; Grosso, P.; van der Ham, J.; Papavassiliou, S.

DOI
10.1016/j.future.2014.01.003

Publication date
2015

Document Version
Final published version

Published in
Future Generation Computer Systems

License
Article 25fa Dutch Copyright Act (https://www.openaccess.nl/en/in-the-netherlands/you-share-we-take-care)

Link to publication

Citation for published version (APA):

General rights
It is not permitted to download or to forward/distribute the text or part of it without the consent of the author(s) and/or copyright holder(s), other than for strictly personal, individual use, unless the work is under an open content license (like Creative Commons).

Disclaimer/Complaints regulations
If you believe that digital publication of certain material infringes any of your rights or (privacy) interests, please let the Library know, stating your reasons. In case of a legitimate complaint, the Library will make the material inaccessible and/or remove it from the website. Please Ask the Library: https://uba.uva.nl/en/contact, or a letter to: Library of the University of Amsterdam, Secretariat, Singel 426, 1012 WP Amsterdam, The Netherlands. You will be contacted as soon as possible.
Resource discovery and allocation for federated virtualized infrastructures

C. Pittaras\textsuperscript{a}, C. Papagianni\textsuperscript{b,*}, A. Leivadeas\textsuperscript{b}, P. Grosso\textsuperscript{a}, J. van der Ham\textsuperscript{a}, S. Papavassiliou\textsuperscript{b}

\textsuperscript{a} System and Network Engineering research group, University of Amsterdam, The Netherlands
\textsuperscript{b} NETwork Management and Optimal DEsign Laboratory, National Technical University of Athens, Greece

\textbf{HIGHLIGHTS}

- Networking innovations over virtualized infrastructures—NOVI federation architecture.
- Resource discovery and allocation over multi-domain virtualized infrastructures.
- Prototype of a distributed, semantic-based resource discovery and mapping framework.
- A semantic aware virtual network embedding algorithm.
- The approach is evaluated via simulation against non-semantic aware solutions.

\textbf{ARTICLE INFO}

Article history:
Received 1 August 2013
Received in revised form 6 November 2013
Accepted 18 January 2014
Available online 1 February 2014

Keywords:
Network virtualization
Distributed systems
Resource discovery
Resource allocation
Virtual network embedding

\textbf{ABSTRACT}

The European Union Project Networking Innovations Over Virtualized Infrastructures (NOVI) set out to design and implement a modular data, control and management plane federation architecture, leading to an integrated experimental prototype mounted on interconnected European Future Internet testbeds. In this paper we present the components of this architecture, responsible for resource discovery and mapping of virtual topologies over a federated multi-domain network virtualization environment. We subsequently introduce a method for the efficient mapping of user requests for virtual networks onto a substrate infrastructure, adopting a semantic-based approach to address the problem. The efficiency of the proposed scheme is evaluated via simulation and critically compared against common non-semantic-based solutions.

© 2014 Elsevier B.V. All rights reserved.

1. Introduction

Over the last decade, network virtualization has been introduced as the means to overcome the ossification of the Internet. Large-scale research experimental infrastructures, providing sharing mechanisms, have been widely deployed, enabling Future Internet (FI) experimental validation in realistic testing environments. What is more, network virtualization is considered an inherent component of a polymorphic ecosystem promoting a flexible architectural FI design.

Two initiatives have emerged as main promoters of research on the FI research experimental infrastructures: the Global Environment for Network Innovations (GENI) in the United States and the Future Internet Research and Experimentation (FIRE) in Europe. The EU Project NOVI [1], Networking Innovations Over Virtualized Infrastructures, an experimentally driven research project under the FIRE initiative, was set out to create a blueprint of FI federated infrastructures, by designing and prototyping a service portfolio based on combined virtualized facilities from virtualized infrastructures. NOVI architecture has been prototyped as a proof of concept of the proposed design and deployed over Federated E-infrastructure Dedicated to European Researchers Innovating in Computing network Architectures (FEDERICA) [2] and a private deployment of the PlanetLab [3] infrastructures.

In this paper we present the prototyped components of the NOVI architecture, responsible for resource discovery and mapping over a federated multi-domain Network Virtualization...
Environment (NVE). On one hand, the problem of mapping virtual networks (VNs) onto a (multi-domain) substrate network is the main resource allocation challenge in network virtualization, commonly referred to as (inter-domain) Virtual Network Embedding (VNE) problem or Virtual Network Mapping problem. On the other hand resource discovery in the context of an NVE remains still an unexplored research area [4], despite its large impact on the subsequent resource allocation phase. NOVI introduces a distributed, semantic-based resource discovery and VN mapping framework over the NOVI-federated testbeds. The framework is enabled by the NOVI Information Model (NOVI IM) [5] that provides abstractions and semantics of federated virtualized platforms. Semantic technologies facilitate reasoning when selecting resources and services that improves the precision of resource discovery. The semantic-based VNE approach is presented in detail in the study. Due to the small scale of the substrate testbeds used in NOVI, its efficiency is illustrated in a simulation environment that allows for a flexible and comparative performance evaluation.

The rest of the paper is organized as follows. Section 2 summarizes work on resource discovery and allocation in the context of a virtualized environment. Section 3 provides a high level description of the NOVI architecture, identifies the components responsible for distributed resource discovery and mapping and provides a thorough description of the proposed resource discovery and allocation framework. The semantic-based VNE approach is described in Section 4 and is compared against non-semantic aware heuristics, provided by the authors in [6], in Section 5. Finally, Section 6 concludes the paper.

2. Related work

Inter-domain VNE can be basically broken down to the following sub-problems [7]: (i) selecting the appropriate testbed to embed a segment or the entire VN request (VN partitioning) and (ii) solving the resulting distinct VNE problems within each single administrative domain. VN partitioning requires that testbeds advertise their resources to an appropriate resource discovery framework and the VN request graph is split among testbeds (sub) optimally based on advertised costs.

2.1. Resource discovery and VN partitioning

Publications focusing on resource discovery for multi-domain virtualized infrastructures are sparse [4]. A resource discovery method based on the conceptual clustering technique has been proposed [8]. Substrate resources are described in terms of functional and non-functional attributes. Functional attributes define properties of the resources e.g., resource type, Operating System (OS) etc., while non-functional attributes specify criteria and constraints including dynamic features like node and link capacity. Authors in [9] use the same framework as in [8] but propose a hierarchical organization framework of resource discovery data.

In most studies, advertised resource provisioning costs are randomly generated [7,10]. The cost of links spanning multiple domains is assumed to be an order of magnitude higher than the average intra-domain cost. However, dynamic information about resources is not included due to the induced real time monitoring overhead. Belbekkouche et al. [4] pinpoint the necessity to make information on non-functional attributes available at the Service Provider (SP) level. In the context of a networked cloud environment (NCE), authors in [11] defined advertised costs based on resources (VM/network) availability within the networked cloud. The cost of an inter-cloud virtual link is defined as a function of the transit network providers it traverses. Following a similar approach, in the previous work done by the authors [12], advertised costs include resource availability, in terms of scarcity of the resource and the average utilization over a time window.

NOVI follows a semantic-based approach for resource discovery. NOVI’s resource discovery framework operates on a distributed set of semantic repositories located on the NOVI service layer of every testbed in the federation [13]. Advertised costs include functional and non-functional attributes, incorporating monitoring mechanisms to acquire aggregated information about each virtualized platform [12].

The problem of VN graph partitioning among several virtualized infrastructures is NP-Hard [7], VN partitioning has been presented in the literature within the context of an NCE [11,12] or an NVE [7,9]. Within the context of NOVI, the Iterated Local Search meta-heuristic has been adopted and prototyped, due to its intrinsic simplicity and documented efficiency in overcoming time performance/scalability issues [12].

2.2. Virtual network embedding

The problem of assigning virtual nodes and link to a substrate network without violating capacity constraints within a single domain can be reduced to the NP-hard multi-way separator problem [14]. The VN embedding problem is quite challenging, due to finite node and link resource constraints, admission control, and the on-line nature of VN requests. The problem remains computationally intractable even if some conditions are relaxed (i.e., all the requests are known in advance). Several approaches have been followed to deal with the complexity and challenges related to the VNE problem. Extensive literature reviews are provided in the following studies [15,16,4], including comprehensive overviews of the main challenges and diverse aspects of VNE, highlighting existing approaches and emerging requirements.

The NOVI service layer abstracts the physical substrate to a semantic graph. Based on the assumption that experimenters describe the attributes of their resource requirements explicitly, a semantic-based resource mapping approach is followed in NOVI where SPARQL (SPARQL Protocol and RDF Query Language) endpoints enable fast and easy access to substrate resource information.

2.3. Resource discovery and allocation in other projects

The GENI and the FIRE initiatives include a number of projects that face the challenge of resource discovery and allocation. GENI encompasses a number of control frameworks and testbeds e.g., Emulab, ProtoGENI, PlanetLab, Open Resource Control Architecture (ORCA), etc. Slice-based Federation Architecture (SFA) [17] is used as a common API, providing a common method for advertising resources and servicing user requests over multiple administrative domains. The central data structure used by SFA is the Resource Specification (RSpec) [18]. It is used as an interchange format for GENI platforms to advertise substrate resources or describe allocated resources via an appropriate manifest. In addition it enables the user to request resources by providing a complete mapping between requested and substrate resources. However, the ORCA [19] framework is capable of supporting mapping of requested to substrate resources [11]. Specifically topology partitioning is enabled by integrating a minimum k-cut algorithm followed by sub-graph isomorphism. Virtual topology embedding within a single cloud site is facilitated by NEuca [19]. ProtoGENI also defines a slice embedding service without however providing additional details on the service implementation [20]. The slice-based federation architecture is the most common also in the FIRE initiative for the discovery, reservation and provisioning tasks. Within FIRE the use of semantic-based resource descriptions is considered for federated testbeds along with exploiting the expressiveness of semantic technologies for VN embedding [21].
3. Resource discovery and allocation in NOVI

3.1. NOVI architecture: an overview

NOVI was set out to create a federation layer on top of existing network and computing FI platforms. Fig. 1 illustrates the NOVI federation approach, introduced in [22]. Utilizing NOVI’s GUI, experimenters are able to request a slice of virtualized resources, spanning through the entire set of virtualized platforms that are members of the NOVI federation. Following the GENI [23] definition, a slice is a set of virtualized resources connected together to provide a single virtual testbed for an experimenter. NOVI’s service layer provides the set of services for control and management of such slices in the NOVI federation.

Following a bottom up description, at the lower layer heterogeneous platforms contain the virtual resources to be allocated to slices. The NSwitch, NOVI’s Distributed Virtual Switch, enables data plane connectivity between virtual resources belonging to a federated slice. The middle layer provides basic control and management federation capabilities across testbeds: NOVI prototype implementation builds on top of SFA [24]. The top layer, referred to as the NOVI Service Layer, implements NOVI control and management services that offer advanced capabilities to the federation users.

Each peer in the NOVI architecture is essentially a virtualized platform deploying the NOVI Service Layer. The two proposed components for distributed resource discovery and allocation, namely Resource Information Service (RIS) and Intelligent Resource Mapping (IRM), are components in the NOVI Service Layer. NOVI peers form an overlay network that enables each platform (i) to discover resources in each others’ domains (RIS); (ii) assist them to decide which platform will fulfill a VN request, or part of it, in a cost-efficient way (IRM) and (iii) embed the VN (partial) request in the underlying platform (RIS/IRM).

In the NOVI project a single Information Model (IM) is used [5]. This is required since NOVI federates heterogeneous testbeds, each of which can have different kinds of resources, and often different ways of representing their resources as well. The IM has been created using the Web Ontology Language (OWL) [25]. The IRM service utilizes the NOVI IM to (i) identify functional and non-functional characteristics of the requested resources and (ii) express mapping information. RIS leverages the NOVI IM to store, annotate semantically and query information for each testbed.

3.2. Resource information service

The RIS (Fig. 2) maintains the NOVI database (NOVI DB) which contains information about the NOVI peer’s facility (physical infrastructure, virtual topologies, monitoring values, users and policy information). RIS uses the Resource Description Framework (RDF) contexts to isolate the information in the database. For instance, the physical substrate information and the virtual topologies are stored with unique contexts. RIS uses a pull mechanism to retrieve the substrate information from each testbed, via a custom component (i.e., Request Handler in Fig. 1) that translates the testbed’s native information model (e.g. PlanetLab Resource Specification) to NOVI IM. RIS uses the Policy Service to get authorization information for users and resources. RIS also uses the Monitoring Service (Fig. 1) to retrieve information on non-functional characteristics of the resources. To reduce the overhead of RIS—Monitoring Service communication, a local monitoring cache is maintained on each NOVI peer. The Sesametriple store [26] is used to create the NOVI DB. Incoming requests (e.g., VN request) to the RIS are essentially Java objects. Alibaba library [27] is used for the translation between RDF triples and Java objects.
3.3. Intelligent resource mapping

The JAVA-based IRM (Fig. 2) orchestrates the creation of a new slice or the update of an existing one. In the case of a VN request, where mapping information is included in the description, along with configuration/authorization information regarding the slice/user, the IRM gathers information from the RIS service regarding requested resources availability/authorization rights. If requested substrate resources are available and the user is authorized to reserve them, the slice reservation process continues via the NOVI service layer. In case mapping information is not included in the request, the IRM orchestrates the NOVI resource discovery and the VN-partitioning process. The embedding-〈Testbed〉 component (e.g., embedding-PlanetLab) implements the embedding-api, the interface to the intra-domain VNE algorithms for executing distributed VNE on the selected set of underlying testbeds.

3.4. Resource discovery and VN mapping workflow

The discovery and mapping process in NOVI is executed in two phases: the resource discovery and VN Partitioning phase and the intra-domain VNE phase.

3.4.1. Resource discovery and VN partitioning

The experimenter submits the VN request at the NOVI layer instance deployed on top of the platform where he/she is registered as a user. The request includes functional and non-functional requirements related to the requested resources. The local IRM service triggers distributed semantic discovery in the federation (Fig. 3(a)). The local RIS contacts peer services, forwarding the requested VN description. Each RIS that belongs to a NOVI-enabled platform (i) analyzes the incoming VN request, grouping virtual resources based on their functional characteristics and (ii) identifies the set of substrate resources capable of provisioning each one of them. The latter involves dynamically created SPARQL queries, denoted as matching queries, at the local triple store. Listing 1 shows a simple matching query, which searches for servers with a specific functional attribute: in this case Intel 80386 microprocessor. In the following, RIS retrieves non-functional attributes (e.g. link utilization, free memory) by aggregated queries, expressed in the RDF format, to the local monitoring component. The distributed RIS advertises partitioning cost information per platform, based on (i) the cardinality of the matching set for each requested resource and (ii) the acquired non-functional characteristics of the resources [12].

```sql
PREFIX im:<http://fp7-novi.eu/im.owl#> PREFIX rdf:<http://www.w3.org/1999/02/22-rdf-syntax-ns#> PREFIX xsd:<http://www.w3.org/2001/XMLSchema#>
SELECT ?var1 FROM im:testbedSubstrateContexts where {
  ?var1 rdf:type im:Node .
  ?var1 im:hardwareType ?var1hwType .
  FILTER regex(str(?var1hwType), "i386", "i")
}
```

Listing 1: SPARQL matching query example.

The IRM service that initiated the resource discovery process retrieves the advertised costs and subsequently determines the most cost-efficient way to assign virtual resources to platforms-members of the NOVI federation. To deal with the inherent complexity and scalability of the VN partitioning problem across multiple administrative domains, a testbed-independent request partitioning approach with the use of Iterated Local Search (ILS) meta-heuristic is adopted. Performance evaluation via simulation demonstrates the efficiency of the algorithm over a large number of requests and number of platforms in the federation, with minimum computation time [12].

3.4.2. Virtual network embedding

The VNE phase is executed in a distributed way, over each platform that is selected during VN partitioning (Fig. 3(b)). Each peer’s IRM service is responsible for embedding the received partial request (VN sub-graph) to its substrate network. Solving the VNE mandates synchronous communication to the local RIS service, to gather information for available substrate resources. In the case that a semantic-based approach is followed for VNE, this process involves matching requirements posed by the experimenter.
Testbed providers may follow different approaches based on the resources of each testbed and the supported services to the end user. In the case of PlanetLab, where slices are collections of virtual machines, solving the VNE resumes to virtual-to-physical node mapping. Specifically a semantic-based load-aware greedy node mapping approach is prototyped in IRM, involving node mapping queries executed in RIS. A generic description of the semantic-based approach for VN mapping is described in the following section. For embedding VN requests on the FEDERICA substrate, two algorithms have been prototyped: (i) the heuristic proposed in [6] denoted as Networked Cloud Mapping (NCM) and (ii) a Greedy Node Mapping followed by a Shortest Path Algorithm for the link mapping phase (G-SP) [28].

4. A semantic-based approach on VN mapping

Substrate network and VN request

A VN request is modeled as a weighted undirected graph denoted by $G^V = (N^V, E^V)$ where $N^V$ represents the set of virtual nodes and $E^V$ the set of virtual links. Similarly, the substrate network is modeled as a weighted undirected graph $G^S = (N^S, E^S)$. Every node is associated with a set $A$ of functional attributes including its type e.g. $A = \{\text{nodetype, OS, virtEnv, etc.}\}$ so that $n^S_x \in N^S, X \in \{V, S\}$. Moreover each node $n^V_x \in N^V$ is attributed with an explicit set $B$ of non-functional attributes, denoted as capacities, $c_b(n^V_x), b \in B, n^S_x \in N^S, X \in \{V, S\}$ (e.g., CPU capacity, memory, storage capacity, etc.). The vector of capacities for each node is denoted as $c(n^S_x)$. Moreover, every edge $(n^V_x, m^V_x) \in E^V, \forall n^S_x, m^S_x \in N^S, X \in \{V, S\}$ is associated with a link of bandwidth capacity $bw(n^S_x, m^S_x)$.

Fig. 4(i) shows a substrate network. Routers, physical and logical are depicted as circles while servers/VMs as rectangles. The numbers included in the shapes indicate (i) the ID for both routers and servers and (ii) available/requested logical routers or CPU computational capacity. The numbers over the links represent available/requested bandwidths.

Node mapping

Mapping determines the allocation of physical resources to the VN request. Node Mapping within a testbed is denoted as:

$$M^N: N^V \rightarrow N^S$$

where

$$M^N(n^V_x) \in V^S_A, V^S_A = \{n^S_y \in N^S : A = A', c_b(n^V_x) \leq c_b(n^S_y), \forall b \in B\}$$
and \( C_b(n^V_\mathcal{S}) \) refers to the remaining capacity of the substrate node \( n^V_\mathcal{S} \). That is:

\[
C_b(n^V_\mathcal{S}) = C_b(n^V_\mathcal{S}) - \sum_{M^V_\mathcal{P}, M^N_\mathcal{P} = n^V_\mathcal{S}} C_b(M^V_\mathcal{P}).
\]

Expressing anti collocation constraints for the request, each virtual node of the VN must be assigned to a different substrate node. Therefore:

\[
M^N(n^V_\mathcal{S}) = M^N(m^V_\mathcal{P}) \iff n^V_\mathcal{S} \equiv m^V_\mathcal{P}.
\]

Link mapping

On the other hand, every virtual link can be mapped to a single substrate path \( P^\mathcal{S} \) for non-bifurcated routing or a set of substrate paths \( P^\mathcal{S} \), when traffic is split in multiple routes among the mapping solutions of the virtual link end points. Under the assumption that flow bifurcation is enabled, link mapping is denoted as:

\[
M^V : E^V \mapsto P^\mathcal{S},
\]

where \( E^V(n^V_\mathcal{S}, m^V_\mathcal{P}) \in P^\mathcal{S}(M^N(n^V_\mathcal{S}), M^N(m^V_\mathcal{P})) \) where,

\[
P^\mathcal{S}(M^N(n^V_\mathcal{S}), M^N(m^V_\mathcal{P})) = \{ P^\mathcal{S} \in E^V(n^V_\mathcal{S}, m^V_\mathcal{P}), bw(n^V_\mathcal{S}, m^V_\mathcal{P}) \leq bw(P^\mathcal{S}) \}.
\]

The available bandwidth capacity of a substrate path \( bw(P^\mathcal{S}) \) is restricted to the available capacity of the most loaded link in the path.

\[
bw(P^\mathcal{S}) = \min_{\forall (u^V, v^V)} \mathcal{B}\mathcal{W}(u^V, v^V)
\]

\[
\mathcal{B}\mathcal{W}(u^V, v^V) = bw(u^V, v^V) - \sum_{\forall (u^V', v^V') \in M^V} bw(u^V', v^V').
\]

Based on the node and link mapping notation, the VN request graph \( G^V \) can be mapped to a subset of the substrate nodes and paths denoted as:

\[
M^E : G^V \mapsto G^\mathcal{S} \cup M^E.
\]

The set of feasible mappings is denoted as \( \Omega(M^E) \).

**Semantic-based greedy mapping**

The pseudo-code for the Semantic-Based Greedy VN Mapping algorithm is provided in Algorithm 1. Quality of Service (QoS) related parameters are also taken into consideration. Specifically, the goal is to provide an upper bound for the number of hops for a virtual link mapped on a substrate path, denoted as MaxHop. In essence, reducing the number of hops along a traffic route between two communication nodes is a common practice to provide QoS enhancements with regard to latency. Initially, based on the incoming VN request \( G^V \) and MaxHop, a SPARQL query is formulated and executed on the tested provider’s local triple store that holds information regarding the substrate topology \( G^\mathcal{S} \), including functional and non-functional attributes of physical resources. The output of the SPARQL query is the set of feasible mapping \( \Omega(M^E) \). In the case that \( \Omega(M^E) \neq \emptyset \), the mapping that minimizes the embedding cost [12,6] is selected.

As an example, the VN request in Fig. 4 is formulated as the query provided in Listing 2. Imposed constraints include (i) capacity constraints on resources as defined in the node and link mapping paragraphs and (ii) maximum path length for link mapping.

Topology mapping queries are constructed automatically based on the virtual topology. The \texttt{SELECT} part of the query identifies the variables: one per requested node and a set of variables for each virtual link, with cardinality equal to the maximum number of hops. With regard to a virtual link, a disjunction of clauses is used using the \texttt{UNION}. Each clause defines a path with specific length, including capacity constraints for the substrate links in the path.

Algorithm 1 Semantic-based greedy VN mapping

\[\text{Input: } G^V, \text{ MaxHops} \] 

\[\text{Output: Mapping } \]

\[\text{MinEmbeddingCost } \leftarrow \infty \]

\[\text{Links[i] } \leftarrow \text{sortAscendingCapacity}(E^V) \]

\[\text{setCurrentMaxHops(Links, 1)} \]

\[\text{currentMaxHops } \leftarrow 1 \]

\[\text{linkIndex } \leftarrow 0 \]

\[\text{while currentMaxHops } \leq \text{MaxHops} \text{ do} \]

\[\text{q } \leftarrow \text{query.generateQuery}(G^V, \text{Links)} \]

\[\text{M^C } \leftarrow \text{executeQuery}(G^V, \text{q)} \]

\[\text{if } \text{M^C } \neq \emptyset \text{ then} \]

\[\text{Mapping } \leftarrow \text{M^C} \{ \text{return the first one, that you find} \} \]

\[\text{MinEmbeddingCost } \leftarrow \sum_{\forall (u^V, v^V) \in P^\mathcal{S}} bw(P^\mathcal{S}) \]

\[\text{return } \text{Mapping} \]

\[\text{end if} \]

\[\text{for } i = \text{linkIndex } \rightarrow |E^V| - 1 \text{ do} \]

\[\text{Links[i].currentMaxHops } \leftarrow \text{Links[i].currentMaxHops } + 1 \]

\[\text{currentMaxHops } \leftarrow \text{Links[i].currentMaxHops} \]

\[\text{if } i = |E^V| - 1 \text{ then} \]

\[\text{linkIndex } \leftarrow 0 \]

\[\text{exitForLoop} \]

\[\text{end if} \]

\[\text{if } \text{Links[i+1].capacity } \neq \text{Links[i].capacity} \text{ then} \]

\[\text{linkIndex } \leftarrow i + 1 \]

\[\text{exitForLoop} \]

\[\text{end if} \]

\[\text{end for} \]

\[\text{end while} \]

\[\text{return } \emptyset \{ \text{no mapping was found} \} \]

For instance, for a maximum number of two hops, each virtual link has two clauses, the first one determines one-hop paths while the second one defines two-hop paths. For the latter, (i) intermediate nodes are defined with no capacity constraints and (ii) we set filters, making sure that the path is comprised of distinct substrate nodes. The sets of clauses of all virtual links are set together using conjunction.

In order to optimize the query execution time, the virtual nodes along with their requirements are defined as parts of the virtual link that interconnects them. The query results into four possible mappings: \( \Omega(M^E) = \{ [A \rightarrow 3, B \rightarrow 9, (A, B) \rightarrow (3,9)], [A \rightarrow 4, B \rightarrow 9, (A, B) \rightarrow (4,3), (3,9)], [A \rightarrow 2, B \rightarrow 9, (A, B) \rightarrow (2,3), (3,9)], [A \rightarrow 1, B \rightarrow 9, (A, B) \rightarrow (1,3), (3,9)] \} \).

To further speed up the process, the \texttt{query.generateNextQuery()} function increases gradually the maximum number of hops variable, bounded by the MaxHop value, but only for the virtual link (or links) with the lower capacity requirements. If all links have \( \text{hops } = \text{currentMaxHops} \), then it increases the \text{currentMaxHops} by one. The goal is to break down the query defined in Listing 2 into smaller ones that run far more efficiently. The rationale behind the pruning process is that (i) mappings with shorter substrate paths are more cost efficient and (ii) corresponding queries are faster due to the restricted search-space.

5. Evaluation and results

In this section, the efficiency of the proposed approach is evaluated via simulation. A discrete event java-based simulator CVI-Sim (SIMulator for Controlling Virtual Infrastructures) was used for that purpose. CVISim [29,6] was implemented to provide an extendable simulation environment that will enable us to evaluate the performance and the efficiency of VN embedding approaches. The semantic-based greedy VN mapping (SVNM) approach is compared against non-NCM and G-SP.
The experimentation setup is aligned with commonly used environments in the literature (e.g., [6,12]). Regarding the available substrate network, two types of nodes have been used, servers and routers. Each node is characterized from its type, its operating system and its virtualization environment. Moreover the nodes have different non-functional characteristics based on their type e.g., {CPU computational capacity, memory, storage} for servers and {number of available logical router instances} for routers. The available CPU capacity per server and available bandwidth capacity per substrate link are uniformly distributed in the interval [50,100]. These present unit values. Similarly, available storage and memory capacities are also defined as real numbers uniformly distributed between [50,100]. A maximum of 15 available logical routers can be instantiated on every physical router. Substrate topologies are randomly generated as partial mesh topologies, while each substrate is comprised of 25 nodes. The probability of generating a specific type of node is 80% for servers and 20% for routers.

The requested CPU capacity is uniformly distributed in the interval [0,20] for every requested virtual machine and requested bandwidth is uniformly distributed in the interval [0,50]. Requested storage and memory capacities are also defined as real numbers uniformly distributed between [0,20]. One logical router corresponds to each requested node with routing capabilities. The number of virtual nodes per request is randomly selected by a uniform distribution between 2 and 8 with 40% connectivity.

MySQLim:<http://www.w3.org/1999/02/22-rdf-syntax-ns#> PREFIX
rdfs:<http://www.w3.org/2001/XMLSchema#>

SELECT DISTINCT ?node0 ?node1 ?link0 WHERE {
  ?node0 rdf:type im:Node .
  ?node0 im:hardwareType "router" .
  ?node0 im:hasAvailableLogicalRouters ?node0LogRout .
  FILTER (xsd:integer(?node0LogRout) >= xsd:integer(1)) .
  ?node0 im:hasOutboundInterface im:isSource ?link0 .
  ?link0 rdf:type im:Link .
  FILTER (xsd:float(?link0acapacity) >= xsd:float(9.0)) .
  FILTER (xsd:integer(?node0LogRout) >= xsd:integer(1)) .
  ?link0im:hasAvailableCapacity ?link0acapacity .
  ?link0im:isSink im:hasInboundInterface ?node0 .
  ?node0 im:hardwareType "server" .
  ?node0 im:hasComponent ?node0_cpu .
  ?node0_cpu rdf:type im:CPU .
  ?node0_cpu im:hasAvailableCores ?node0_cpu_avalCores .
  FILTER (xsd:integer(?node0_cpu_avalCores) >= xsd:integer(9)) .
  ?node0_cpu im:hasAvailableLogicalRouters ?node0_cpu_avalLogicalRouters .
  FILTER (xsd:integer(?node0_cpu_avalLogicalRouters) >= xsd:integer(9)) .
  ?node1 rdf:type im:Node .
  ?node1 im:hardwareType "router" .
  ?node1 im:hasAvailableLogicalRouters ?node1LogRout .
  FILTER (xsd:integer(?node1LogRout) >= xsd:integer(1)) .
  ?node1 im:hasOutboundInterface im:isSource ?link0 .
  ?link0 rdf:type im:Link .
  FILTER (xsd:float(?link0acapacity) >= xsd:float(9.0)) .
  FILTER (xsd:integer(?node1LogRout) >= xsd:integer(1)) .
  ?link0im:hasAvailableCapacity ?link0acapacity .
  ?link0im:isSink im:hasInboundInterface ?node1 .
  ?node1 im:hardwareType "server" .
  ?node1 im:hasComponent ?node1_cpu .
  ?node1_cpu rdf:type im:CPU .
  ?node1_cpu im:hasAvailableCores ?node1_cpu_avalCores .
  FILTER (xsd:integer(?node1_cpu_avalCores) >= xsd:integer(9)) .
  FILTER (xsd:integer(?node1_cpu_avalCores) >= xsd:integer(9)) .
  ?node1_cpu im:hasAvailableLogicalRouters ?node1_cpu_avalLogicalRouters .
  FILTER (xsd:integer(?node1_cpu_avalLogicalRouters) >= xsd:integer(9)) .
  FILTER (xsd:integer(?node1_cpu_avalLogicalRouters) >= xsd:integer(9)) .
}

Listing 2: SPARQL mapping query example.

5.1. Evaluation setup and metrics

The experimentation setup is aligned with commonly used environments in the literature (e.g., [6,12]). Regarding the available substrate network, two types of nodes have been used, servers and routers. Each node is characterized from its type, its operating system and its virtualization environment. Moreover the nodes have different non-functional characteristics based on their type e.g., {CPU computational capacity, memory, storage} for servers and {number of available logical router instances} for routers. The available CPU capacity per server and available bandwidth capacity per substrate link are uniformly distributed in the interval [50,100]. These present unit values. Similarly, available storage and memory capacities are also defined as real numbers uniformly distributed between [50,100]. A maximum of 15 available logical routers can be instantiated on every physical router. Substrate topologies are randomly generated as partial mesh topologies, while each substrate is comprised of 25 nodes. The probability of generating a specific type of node is 80% for servers and 20% for routers.

The requested CPU capacity is uniformly distributed in the interval [0,20] for every requested virtual machine and requested bandwidth is uniformly distributed in the interval [0,50]. Requested storage and memory capacities are also defined as real numbers uniformly distributed between [0,20]. One logical router corresponds to each requested node with routing capabilities. The number of virtual nodes per request is randomly selected by a uniform distribution between 2 and 8 with 40% connectivity. The probability of generating a specific type of node is 90% for VMs and 10% for virtual routers.

VN requests arrive according to a Poisson process with a varying rate (4 requests per 100 time units). Each of them is assumed to have exponentially distributed lifetime with an average of 1000 time units. Each simulation is executed for 1000 requests and repeated for 5 iterations.

In order to quantify the performance of these approaches we use the metrics presented in [6]: (i) the embedding cost reflects the cost for allocating substrate resources, (ii) the average percentage of accepted VN requests and (iii) the average number of hops per virtual link.

5.2. Numerical results and comparison

Fig. 5(a)–(c) present the behavior of the three algorithms under consideration, namely SVN-M, NCM and G-SP, on the aforementioned metrics as a function of time. The basic observation is that the average hop number (Fig. 5(c)) is decreased for SVN-M as opposed to NCM that also aims to reduce the length of the substrate paths that are mapped to virtual links [6] and the G-SP, where the node and link mapping phases are consecutive (uncoordinated VNE). NCM maps successively more requests than G-SP (Fig. 5(a)), increasing the utilization of substrate resources. That leads to an increase in the hop count, due to the load balancing scheme enforced by NCM [6], since it selects balanced, yet more distant resources in the node mapping phase. In most of the cases, a reduction of hops can implicitly be translated to an improvement in the provided QoS parameters (e.g., delay) [30], especially when load balancing is also taken into consideration in the resource allocation process. The decrease in the substrate path length is also supported by the lower embedding cost of the SVN-M in comparison to the G-SP and NCM, as depicted in Fig. 5(b). There is a tradeoff between the number of hops on the adopted path(s) and the ability of accepting incoming requests [6], a fact witnessed by all three algorithms.

The evaluation results demonstrate the benefits of the proposed approach. Exploiting semantic annotations, the testbed operator has the ability to control with precision the parameters of the substrate resources allocation process (e.g., substrate path length), in order to efficiently support user slice requests and adjust its operational objectives (e.g., QoS provisioning).

6. Conclusions

Resource discovery and allocation are key challenges to support network virtualization. In this paper we report NOVI’s contributions on resource discovery and mapping (embedding) of user virtual slices into substrate resources, spanning multiple administrative domains. NOVI’s prototyped resource discovery and allocation framework is thoroughly described. We exploit semantic web technology to solve interoperability issues between infrastructure providers (e.g., PlanetLab and FEDERICA) and use the NOVI IM to exchange information between them. The ability to abstract information about the capabilities of an infrastructure provider allows for different levels of abstraction, either for security, business or scalability reasons. In addition, we study the virtual resource allocation problem for network virtualization environments and provide an appropriate semantic-based heuristic approach to address the problem. Breaking down the inter-domain virtual network mapping problem to virtual network partitioning followed by virtual network embedding within each contributing domain (i) respects the requirements of infrastructure providers for controlling the information that they disclose regarding their infrastructure while (ii) enhances the scalability of the embedding process.
The virtual network embedding problem has been addressed considering wired layer 2 (or layer 3) networks and infrastructures. Extending the proposed framework to take into account new and more dynamic, heterogeneous environments and infrastructures (e.g., wireless or software defined networks) presents additional challenges in optimizing their specific contexts, e.g., efficient flowspace allocation among virtual networks over an OpenFlow Facility or the existence of mobile nodes on a wireless environment. Moreover including the time axis as an additional parameter at the embedding process will enable scheduling and common use of resources (e.g. wireless nodes) that do not follow the resource-sharing model of virtualized infrastructures. The development of such a holistic framework is of high research and practical importance and is part of our current and future work.

Acknowledgments

This work has been partially supported by the European Commission, 7th Framework Programme for Research and Technological Development, Capacities, Grant No. 257867—NOVI. We would like to thank A. Monje and C. Cervello-Pastor for their contribution to the NOVI IRM service.

References


works, algorithms, semantic web and resource discovery.


Federation for FIRE. http://www.fed4fire.eu.


OWL Web Ontology Language. http://www.w3.org/TR/owl-ref/.


Chariklis Pittaras graduated from the School of Electrical and Computer Engineering of National Technical University of Athens in 2008. He received his MSc in Advanced Computing from Imperial College London in 2009. He worked on EU FP7 research project NOVI and Dutch national project COMMIT, during his work at the System and Network Engineering Research Group at the University of Amsterdam, from 2011 until 2013. Currently he is working as Analyst/Programmer at Hellenic Bank in Cyprus, at the department of Research and Development of Information Technology and Systems. His research is in the area of distributed networks, algorithms, semantic web and resource discovery.

Dr. Chrysa Papagianni is a Senior R&D Associate at the Network Management and Optimal Design Laboratory (NETMODE), National Technical University of Athens (NTUA), Athens, Greece, since October 2010. She obtained her Diploma and the PhD degree in Electrical and Computer Engineering, both from NTUA in 2003 and 2009 respectively. Her research interests lie in the area of computer networks with emphasis on cloud computing, virtualization, network optimization and management, and service provisioning.

Aris Leivadeas received the Diploma degree in electrical and computer engineering from the University of Patras, Greece (2008), and the MSc degree in mobile and personal communications from King’s College London, UK (2009). He has been a member of the Network Management and Optimal Design Laboratory (NETMODE), National Technical University of Athens, Athens, Greece, since October 2010, where he is currently working towards his Ph.D. His research interests include cloud computing, wired/wireless virtualization and optimization.

Dr. Paula Grosso is assistant professor in the System and Network Engineering research group at the Universiteit van Amsterdam. She is the lead researcher of the group activities in the field of optical networking, distributed infrastructure information modeling and Green IT activities. She is leading a work-package in the CineGrid Amsterdam project, leads the UVA activities in GigAppRout Research on Project networks to develop network models and control plane for lambda networks and topology handling. Her research interests are provisioning and design of hybrid network models for lambda services; development of information models for hybrid multi-domain multi-layer networks, design of network lightpaths architecture for 4K and beyond Digital cinemas. She participated in the EU NOVI project. She co-chaired the NML-WG (Network Markup Language Working Group) and leads the GreenSonar efforts in GLIF (Global Lambda Integrated Facility) and OGF – Open Grid Forum.

Dr. Jeroen van der Ham received his MSc in Artificial Intelligence from Utrecht University in 2002, his MSc in System and Network Engineering in 2004, and his Ph.D in 2010 at the University of Amsterdam on the topic of “Semantic descriptions of complex computer networks”. He is currently working as a lecturer and researcher at the System and Network Engineering research group at the University of Amsterdam. His research interests are in semantic descriptions of networks and (virtualised) resources, but also the ethics of computer science and security research in the connected world.

Dr. Symeon Papavassiliou is currently an Associate Professor with the Faculty of Electrical and Computer Engineering Department, National Technical University of Athens. He received the Diploma in Electrical and Computer Engineering from the National Technical University of Athens, Greece, in 1990, and the MSc and PhD degrees in Electrical Engineering from Polytechnic University, Brooklyn, NY, USA, in 1992 and 1996 respectively. From 1995 to 1999, Dr. Papavassiliou was a senior technical staff member at AT&T Laboratories in Middletown, New Jersey, and in August 1999 he joined the Electrical and Computer Engineering Department at the New Jersey Institute of Technology (NJIT), where he was an Associate Professor until 2004. Dr. Papavassiliou was awarded the Best Paper Award in IEEE INFOCOM’94, the AT&T Division Recognition and Achievement Award in 1997, the National Science Foundation (NSF) Career Award in 2003, and the Best Paper Award in IEEE WCNC 2012. Dr. Papavassiliou has an established record of publications in his field of expertise, with more than two hundred technical journal and conference published papers. His main research interests lie in the area of computer and communication networks with emphasis on wireless communications, resource allocation and optimization, network virtualization and performance analysis and evaluation of stochastic systems.