Journal of Network and Computer Applications 56 (2015) 124–136

Contents lists available at ScienceDirect

Journal of Network and Computer Applications

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

Distributed and scalable embedding of virtual networks

Michael Till Beck [a,*], Andreas Fischer [b], Juan Felipe Botero [c], Claudia Linnhoff-Popien [a], Hermann de Meer [b]

[a] Ludwig-Maximilians-Universität München, Germany
[b] Universität Passau, Germany
[c] Universidad de Antioquia, Colombia

ARTICLE INFO

Article history:
Received 8 December 2014
Received in revised form
15 May 2015
Accepted 1 June 2015
Available online 8 July 2015

Keywords:
Virtual Network Embedding
Distributed algorithm
Framework

ABSTRACT

Network virtualization is widely regarded as a key technology for the Future Internet, enabling the deployment of new network protocols without changing dissimilar hardware devices. This leads to the problem of mapping virtual demands to physical resources, known as Virtual Network Embedding (VNE). Current VNE algorithms do not scale with respect to the substrate network size. Therefore, these algorithms are not applicable in large-scale scenarios where virtual networks have to be embedded in a timely manner.

This paper discusses DPVNE, a *Distributed* and *Generic* VNE framework: It runs cost-oriented centralized embedding algorithms in a distributed way, spreading workload across the substrate network instead of concentrating it on one single node (as centralized algorithms do). Several state-of-the-art algorithms were evaluated running inside the DPVNE framework. Results show that DPVNE leads to runtime improvements in large-scale scenarios and embedding results are kept comparable.

1. Introduction

Virtualization enables infrastructure providers to rent network resources to multiple tenants, ensuring that shared resources are isolated and, thus, do not interfere with each other. Instantiation of virtual infrastructures has to take care of not only the computing requirements of virtual machines, but also the communication requirements between them. For example, a cloud-based web infrastructure may consist of a load balancer connected to a number of web servers, which are themselves connected to database servers. Neglecting the communication requirements may lead to starvation of the traffic between the respective virtual machines, and thus violation of Service Level Agreements.

Virtual Network Embedding (VNE) provides the mechanisms to optimally assign both communication and computation resources according to constraints dictated by virtual infrastructures. Each virtual resource has to be assigned to one or several substrate resources. Substrate resources, however, are limited in capacity, restraining the number of virtual resources that can be hosted by a certain substrate resource. This fact immediately raises two questions: First, can a given set of virtual network requests be mapped to a given substrate network? Second, what is the optimal way to map a virtual network to a substrate network, i.e., which substrate resources should be used to host the virtual resources? In this context, optimality can be defined, e.g., as minimizing the amount of substrate resources that are allocated to the virtual networks (commonly known as *embedding cost*), leaving space for future requests that can be embedded onto the substrate network later on. Optimal VNE is $\mathcal{NP}$–hard (Fischer et al., 2013). Unfortunately, current VNE algorithms do not scale in large-scale networks due to the fact that solving the VNE problem is a CPU-intense task. Most VNE algorithms perform this task in one centralized node, concentrating computational workload on this node. Even for networks with 50 nodes, many conventional VNE algorithms take several minutes to perform a single embedding. For larger network topologies, the situation gets worse quickly.

Thus, in scenarios where the embedding has to be performed in a timely manner, runtime of these algorithms quickly becomes a critical factor. Such scenarios can be found in, e.g., shared cloud infrastructures where frequently arriving virtual network requests can put serious stress on the system. Furthermore, in testbed environments like Future Internet testbeds, various network configurations are evaluated and, thus, the embedding of virtual networks needs to be solved frequently.

As a remedy to this situation, this paper discusses a generic and distributed VNE framework, DPVNE, presented first at IEEE ICC 2013 (Beck et al., 2013). The framework is capable of executing VNE algorithms in a distributed manner by partitioning the substrate network into several smaller network parts. To increase

* Corresponding author.
E-mail addresses: michael.beck@ifi.lmu.de (M.T. Beck), andreas.fischer@uni-passau.de (A. Fischer), juanf.botero@udea.edu.co (J.F. Botero), linnhoff@ifi.lmu.de (C. Linnhoff-Popien), demeer@uni-passau.de (H. de Meer).

http://dx.doi.org/10.1016/j.jnca.2015.06.012
1084-8045/

scalability, DPVNE aims to spread computational load to multiple nodes instead of concentrating it on one single embedder node. The paper presented in ICC introduced the general concept of DPVNE and analyzed first results in the context of small network topologies (100 nodes). In contrast, this paper elaborates on the generality and scalability of the DPVNE framework:

1. It is shown that the DPVNE framework is generic: Several cost-optimizing VNE algorithms are evaluated and run within the DPVNE framework. Previously, DPVNE was evaluated with only one VNE algorithm.
2. Second, this paper discusses the communication protocol for the substrate nodes performing the VNE and analyzes message overhead.
3. Third, DPVNE is analyzed in the context of large substrate network topologies and it is shown that DPVNE scales very well in these scenarios: DPVNE leads to significant runtime improvements, while embedding results still remain comparable to original results.

2. Background and related work

This section shortly explains the VNE problem and describes properties of centralized and distributed embedding approaches.

2.1. The VNE problem

VNE is performed by assigning appropriate resources in the substrate network for each virtual resource. To this end, node and link mapping has to be performed. In the node mapping stage, each virtual node is assigned to an available substrate node. Moreover, a single substrate node can host several virtual nodes. In the link mapping stage, a virtual link between virtual nodes v and w is mapped to a set of consecutive substrate links (substrate path) connecting the substrate hosts of v and w. An individual substrate link may then be part of one or several virtual links. This is depicted in Fig. 1: two Virtual Network Requests (VNRs) with three nodes each are embedded on a substrate network with four nodes and five links. The VNE problem becomes $\mathcal{NP}$−hard when substrate nodes and links have finite resources which are reserved by virtual node and links (in particular node mapping is directly linked to the well-known bin-packing problem). In the figure finite resources are indicated with node and link weights.

It can be seen that a substrate node is capable of hosting several virtual nodes as long as resources are not exceeded. A virtual link can either directly correspond to a physical link. This is the case for VNR1. If no direct link exists or all resources on a direct link are spent, it can also span a path in the substrate network (cf. the right link of VNR2). Such a situation introduces a "hidden hop" on the virtual link (Botero et al., 2012a). One can also see that there may be multiple solutions to the problem. In the given example, the node at the top which is part of VNR2 might also be mapped to the right node in the substrate network, thus reducing the load on the top node in the substrate network.

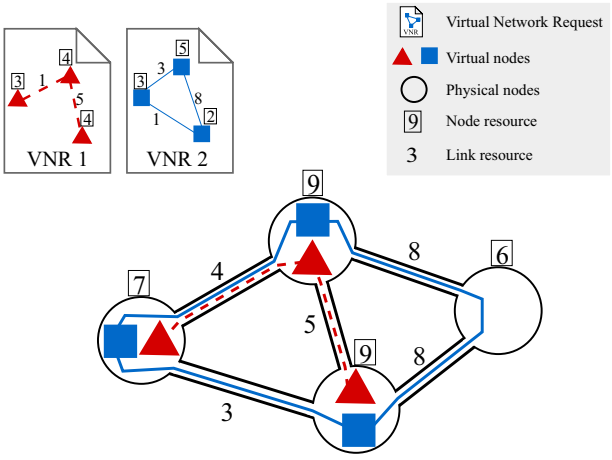

Fig. 1. VNE of two VNRs onto one substrate network.

For the sake of brevity, the notation presented in Fischer et al. (2013) will be used here. In particular, N will denote the set substrate nodes, whereas L will denote the set of substrate links. The substrate network is then represented by the tuple (N, L) of substrate nodes and links. Likewise, (N^i, L^i) will represent the i-th VNR (VNR^i) with its respective virtual nodes N^i and virtual links L^i.

2.2. Centralized VNE algorithms

Due to the $\mathcal{NP}$−hardness of the VNE problem, exact approaches are not scalable and are only applicable in small network scenarios (Houidi et al., 2011; Botero et al., 2012b; Botero and Hesselbach, 2013). Therefore, current solutions are based on heuristics (Zhu and Ammar, 2006; Yu et al., 2008; Butt et al., 2010; Fajjari et al., 2011a; Rahman et al., 2010; Cheng et al., 2011; Botero et al., 2013) or metaheuristics (Fajjari et al., 2011b; Zhang et al., 2011, 2013; Cheng et al., 2012). Many VNE algorithms try to maximize the number of embedded VNRs by reducing embedding costs (Chowdhury et al., 2009; Lischka and Karl, 2009; Fajjari et al., 2011b; Cheng et al., 2011).

All those previous approaches share the following characteristics: they rely on a *centralized embedder node* to perform the VNE and, therefore, VNE is performed by only one embedder node. Being centralized, the embedder node benefits from full knowledge of the substrate network to compute a near-optimal VNE result without any message overhead. However, they share the following shortcomings:

- *Poor scalability*: Current centralized VNE algorithms do not scale for large substrate network topologies. Since the complexity of the VNE problem increases with the network size, runtime of current VNE algorithms increases significantly with increasing network size. One notable approach that aims to improve scalability is presented in Fuerst et al. (2013): Authors evaluate the performance of an algorithm that pre-partitions virtual networks in order to reduce problem size. In contrast to DPVNE, this approach is a centralized approach, whereas DPVNE aims at distributing computational workload between multiple nodes. Furthermore, DPVNE hierarchically partitions the substrate network in order to increase scalability of VNE algorithms in large-scale substrate network topologies, whereas the approach presented in Fuerst et al. (2013) aims to reduce complexity of embedding large-scale virtual network topologies.
- *Delay to embed simultaneous requests*: When multiple VNRs arrive simultaneously, the requests are attended sequentially by the centralized embedder node, resulting in high utilization of the embedder node. Thus, this introduces queuing delay. The time for a single embedding is on the order of seconds to minutes (depending on the size of the VNR and the substrate network). For highly dynamic environments this might be unsuitable. In contrast, distributed approaches aim to spread computational load among multiple embedder nodes.

Besides centralized VNE algorithms, some distributed algorithms have been proposed so far, which will be discussed in the following subsection.