A Dynamic Pricing Algorithm for a Network of Virtual Resources.

Bram Naudts*, Mario Flores†, Rashid Mijumbi‡, Sofie Verbrugge*, Joan Serrat† and Didier Colle*
*Department of Information Technology
Ghent University, Ghent, Belgium
Email: {bram.naudts,sofie.verbrugge,didier.colle}@intec.ugent.be
†Network Engineering Department
Universitat Politècnica de Catalunya, Barcelona, Spain
Email: {mario.flores@entel,serrat@tsc}.upc.edu
‡Telecommunications Software and Systems Group
Waterford Institute of Technology, Waterford, Ireland
Email: rmijumbi@tssg.org

***Abstract*—A service function chain (SFC) is an ordered combination of abstract network functions (e.g. network address translation, a firewall, etc.) that together define a network service (e.g. video-on-demand). In an SDN/NFV based architecture, SFCs are composed of virtual network functions that need to be mapped to physical network components. Since the mapping of a SFC may be possible by multiple competing infrastructure providers (InPs), price will be a key differentiating factor. The pricing algorithm is therefore essential towards revenue management, yet current static pricing approaches suffer from several limitations. Among others, they do not consider the characteristics of the requests or the current state of the physical network. Using historical data, market data and the current state of the physical network we investigate whether it is possible to increase total revenue of an InP compared to traditional static pricing approaches. This paper, proposes a dynamic pricing algorithm to determine (1) at which utilization level it is rewarding to charge a higher price for a particular resource and (2) the alternative price that should be charged.**

Our simulation results for 8 different setups show that the proposed heuristic outperforms a static pricing approach significantly (by 8-85 percent points for the considered scenarios). As a consequence, the proposed approach can be considered as an alternative for static pricing approaches. Still, it is unclear how the total revenue of an InP is affected when multiple or all competitors use a dynamic pricing algorithm, this will therefore remain the focus of future work.

I. Introduction

Telecommunication networks are composed of a variety of network elements. Two categories of network elements can be broadly distinguished: (1) those that are part of the infrastructure or transport network with as primary goal packet forwarding and (2) those that are primarily deployed for purposes other than packet forwarding. Switches and routers are well-known examples of the first category. Middleboxes, also called network appliances or network functions (NFs), form the second category. These network elements are connected or chained in a certain way in order to achieve the desired overall functionality or service that the network is designed to provide. In this context, a service function chain (SFC) defines an abstract set of NFs and their ordering constraints that must be applied to packets and/or frames [1]. SFCs have traditionally been realized via the deployment of physical proprietary devices and equipment for each NF. These physical NFs need to be deployed in a strict chain and/or order that must be reflected in the network topology and in the localization of service elements [2]. This approach has certain drawbacks such as a high degree of complexity and inflexibility and heavy dependence on specialized, expensive hardware.

Network function virtualization (NFV) has been proposed as a way to address these challenges by enabling dynamic construction and management of SFCs. The main idea of NFV is the decoupling of physical network equipment from the functions that run on them. This way, a given service can be decomposed into a set of NFs, which could be implemented in software running on virtualized physical network equipment. This type of implementation of a NF is referred to as a virtual(ized) network function (VNF). In addition, software-defined networking (SDN) proposeds to move management functions out of forwarding hardware into controller software to simplify provisioning and reconfiguration of SFCs.

The decomposition of the SFC into NFs is referred to as service decomposition. By decomposing a SFC into elementary NFs, a number of benefits can be realized. First, re-usable elementary blocks are developed. Second, new and more complex services can be realized from these elementary blocks and third, the detailed implementations of these NFs can be abstracted. Figure 1 depicts an example service decomposition. The SFC is decomposed into three NFs (NF1, NF2 and NF3), NF2 is decomposed to NF4 and NF5, etc. Once a SFC has been decomposed into (V)NFs, they can be embedded by an infrastructure provider (InP).

InPs own, control and manage those physical resources. They offer these (virtualized) resources to third parties who embed SFCs on the substrate resources. In return for the provided resources, the InP charges a fee. Typically, a static pricing approach is used to calculate that fee. We argue that this approach leads to lower revenues than possible. Given that the possibility exists to change the price on the spot, the

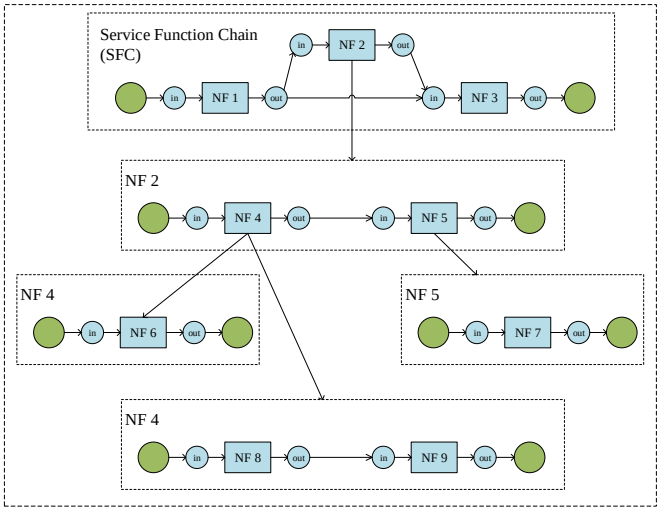

Fig. 1. Example of service decomposition process

InP is able to dynamically change its pricing policy. With a higher price, an operator can get a higher profit margin but the operator may also lose (future) business to a competitor. An important observation in that respect is that the resources are perishable. Non utilized resources generate no revenue. As such a careful trade-off has to be made between supply and demand.

Therefore, our contribution is to propose a dynamic pricing algorithm which varies the price of individual substrate resources over time in such a way that the total revenue of the InP is higher than that of an equivalent competitor applying a static pricing approach.

The remainder of this paper is organized as follows. Section II provides a brief overview of related work on the Virtual Nework Embedding Problem (VNEP) and dynamic pricing algorithms. Section III introduces the stakeholders and provides a detailed description of the problem. In section IV, the proposed algorithm is introduced. The performance evaluation results are reported and discussed in section V. Finally, section VI concludes the paper.

II. Related work

By now, the VNEP is a well studied problem and has multiple application domains. Closely related to the VNEP is the assignment of virtual private networks (VPNs) in a shared provider topology (e.g. [3] and [4]) and the network test-bed mapping problem (e.g. [5]). We refer the interested reader to [6] for a survey of VNE algorithms. Initial studies on placement of VNFs and VNF chains in both InP and optical networks are presented in [7], [8], [9], [10], [11], [12] and our own work [13].

In the field of revenue management, our work is related to the online pricing literature that deals with instantaneous demand dynamics and the adjustment of prices on the spot. Dynamic pricing has become an active field of the revenue management literature, with successful realworld applications in industries such as travel, fashion, and so on [14], [15], [16]. Closely related to our work, revenue management has also been applied to the field of cloud computing, [17], [18], [19], [20], [21]. For cloud providers, unlike other fields, revenue not only depends on the (unknown) number of customers, but also on the (unknown) duration of usage. As such, not only arrival rates but also service times are stochastic. In those works, resources are however considered as interchangeable. When embedding VNFs, the customer will however typically have a set of requirements (e.g. delay, location, etc.) which make them hardly interchangeable. The authors of [22] summarize several situations where the physical resource is not interchangeable for the placement of certain functions:

1) **Efficiency:** VNFs that exchange a lot of data may want to be positioned close to one another (e.g., within the same datacenter, or even on the same physical host).
2) **Resilience:** In order to improve resilience in case a failure occurs in one of the datacenters, the same VNF may be embedded across multiple datacenters.
3) **Legislation:** Hosting VNFs in certain countries due to legislative restrictions may be avoided.
4) **Privacy:** the user might not want the traffic to pass through certain domains due to privacy concerns.
5) **Economic:** for economic reasons (e.g., peering agreements) the placement of functions in certain domains may be promoted or avoided.

The most related work to ours are [23] in which the negotiation process in a multi-domain environment is considered and [24] in which an auction based pricing strategy is used. These approaches are however dependent on a specific VNE algorithm, do not take into account the pricing strategy of competitors or they wait for a certain time to be able to batch a set of requests.

This paper advances the state-of-the-art by proposing a revenue management mechanism for non-interchangeable, perishable resources. The proposed dynamic pricing algorithm can be applied to price requests instantaneously and independent of the chosen VNE algorithm. It assumes a competitive market with price-sensitive customers and knowledge of the competitor's pricing. We detail the dynamic pricing algorithm and validate it via simulation.

III. Problem Description

A number of stakeholders are involved in the realization of an SDN/NFV-driven architecture for realization of service function chaining.

Ecosystem roles. On the left side of Figure 2, the most relevant ecosystem roles are represented. These roles are accomplished by the actors that actively participate in the exchange of value. Most actors will perform more than one role at the same time. For example, traditional ISPs fulfill the role of InP, virtual service infrastructure provider (VSIP) and service provider (SP).

Users. Users, i.e. end/enterprise users, retail or over-the-top providers, request and consume a diverse range of services. In general, users have no strong opinion about how the service

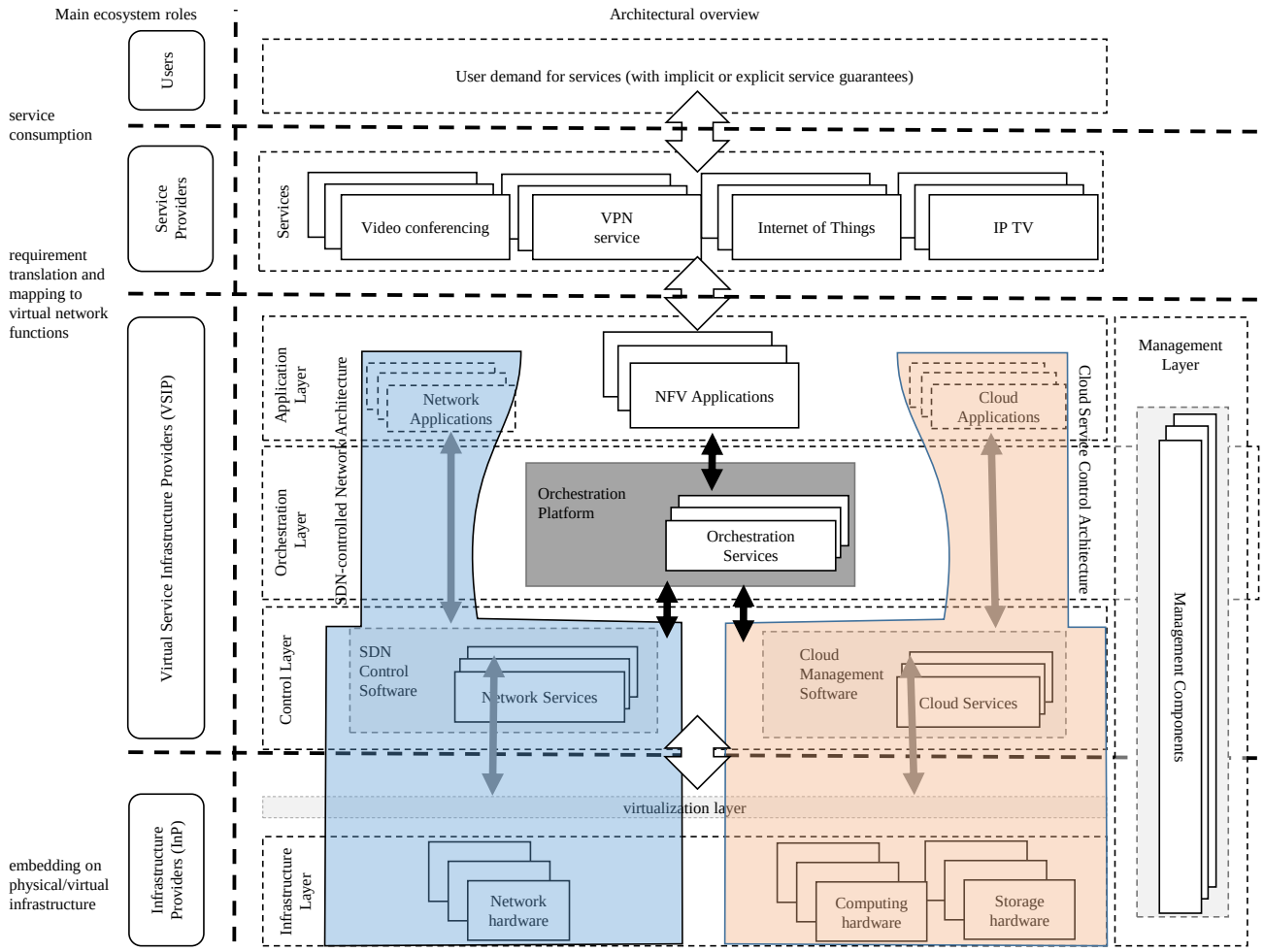

Fig. 2. Network and Cloud control architectures

is delivered as long as their quality of experience expectations are satisfied.

Service providers (SPs). SPs accommodate the service demand from users by offering one or multiple services including over-the-top services and X-play services (e.g. triple play). The SP realizes the offered services on a (virtualized) infrastructure via the deployment of a SFC of VNFs. These SFCs are next handed to a VSIP who will try to map the SFC on the resources provided by the InP.

Virtual service infrastructure providers (VSIPs). VSIPs deliver virtual service infrastructure to SPs, thereby meeting particular service level requirements by combining physical network and cloud resources owned by the InP into a service infrastructure that meets particular SLA requirements [25].

A VNE algorithm should determine if the NFs and their connections in the SFCs can be mapped to the infrastructure. To realize this, two control architectures are used to bring together the areas: (1) the software-driven[1] control of communication networks, and (2) the control of cloud (service) platforms. Both control architectures are depicted in the architectural overview of Figure 2 which is based on [26].

The orchestration platform has a complete view on available networking as well as on computing and storage resources and is used for services that require a combination of these resources. The orchestration components are able to make an informed decision on which infrastructure should be used. The provisioning process itself can then be further delegated to the already existing network and cloud control platforms which manage the virtualized physical resources (of the InP). The request of a VSIP to embed a SFC on the infrastructure provided by a InP is referred to as a virtual network request (VNR).

Infrastructure providers (InPs). InPs own and maintain the

[1]In the context of this article we focus on SDN-controlled networks, although traditional distributed routing protocols could also be considered as the control layer of communication networks.

physical infrastructure and run the virtualization environments. By virtualizing the infrastructure, they open up their resources to remote parties for deploying VNRs. The reusable physical resources comprise all possible resource options (computing, storage and networking) and they span the entire service delivery chain from the end-user gateway and set-top-box over the access, aggregation and core network up to the cloud.

Negotiation process. The negotiation process considers the interaction between the SPs, VSIPs and the InPs. Their main objective is to maximize their own profit. Over time SPs send service request (Serv. Req.) to the VSIPs who try to embed the request on the virtualized substrate resources of an InP.

Since provisioning of the SFCs may be possible by multiple InPs, the VSIPs can use the competition between different InPs to negotiate the best price for a SFC. VSIPs will as such use cost information (the prices charged by competing InPs) to cost-optimally solve the VNE problem. A service request (Serv. Req.) is sent by the VSIP to request for a mapping of a given SFC to the InPs. The request contains information about the amount of requested virtual resources and the duration. After receiving a SR, each InP attempts to perform a mapping. If the mapping is successful, the InP replies with a mapping proposal (MP) to the VSIP giving details of the mapping such as the price. If the mapping fails, the InP sends a mapping failed (MF) message. After receiving a MP, the VSIP replies with an accept proposal (AP) to the InP with the best offer and with a reject proposal (RP) to all other InPs. The negotiation process is summarized in Figure 3.

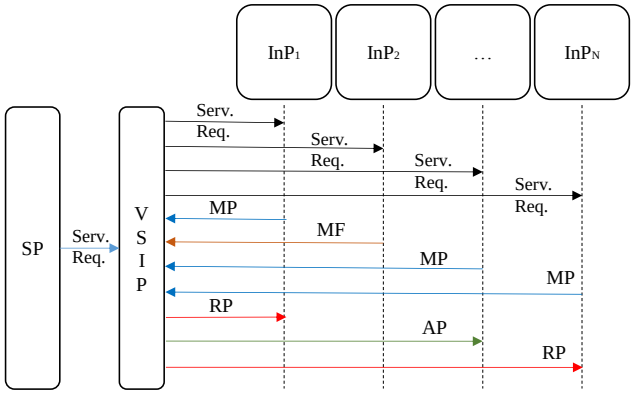

Fig. 3. Overview of the negotiation process

Objective. Our objective is to increase the total revenue of the InP from a population of price sensitive customers (VSIPs). We therefore propose a revenue management mechanism based on a dynamic pricing algorithm. This dynamic pricing algorithm requires a set of inputs. Our assumption is that the InP records information about its own substrate resources (e.g. available and total resources, historic data about provided services per substrate resource) and about the pricing of its competitors (which we assume can be obtained). We do not assume that the InP has information about the used VNE algorithm by the VSIP, nor about the state of the competitor's substrate resources or its cost structure.

IV. Proposed dynamic pricing algorithm

As discussed in the previous section, the customer (VSIP) favors the InP who is able to map the VNR at the lowest price. When we assume that n ($n \in [1, N]$) InPs provide an offer for VNR v ($v \in [1, V]$) at a price $P_{n,v}$, the user's preferred offering for VNR v (PO_v) is as such the minimum of the offers of the different InPs (Equation 1) with $P_{n,v}$ the price of InP n for VNR v.

$$PO_v = min(P_{1,v}, P_{2,v}, P_{3,v}, \ldots, P_{N,v}) \tag{1}$$

Determining $P_{n,v}$. $P_{n,v}$ is typically composed of the units of substrate resource i ($i \in [1, I]$) demanded by VNR v ($R_{i,v}$), and the price charged per unit of the substrate resources i (P_i).

$$P_{n,v} = \sum_{i=1}^{I} R_{i,v} \times P_i \tag{2}$$

In static pricing, the parameter P_i of Equation 2 is a constant over time ($P_{i,s}$). In dynamic pricing, P_i evolves over time and as such the price per substrate resource i will be different per VNR ($P_{i,d}$). The goal of this research is to find an algorithm that is able to determine a $P_{i,d}$ with which an InP that applies the proposed dynamic pricing approach can obtain a higher total revenue in comparison to an identical, competing InP which applies a static pricing approach.

The general idea of our proposal is to vary the price charged for individual resources over time based on the utilization level of the resource and the demand for a resource. To do so, the algorithm takes into account the characteristics of substrate resources:

- **capacity.** not all substrate resources have the same capacity, a large data center may have tens of thousands of servers while a node at the edge of the network may only have a few.
- **demand.** some resources may be popular such as a node which is well connected while demand for other resources may be low
- **service time.** the average service time may vary across resources
- **revenue.** the average revenue accrued per time unit for an embedded VNR may vary across resources

Similarly, the characteristics of VNRs are taken into account:

- **requested resources.** the set of substrate resources varies between VNRs. For example, a VNR may require resources at the core of the network while another VNR may require resources close to the network edge.
- **requested capacity of a resource.** two VNRs may request the same set of resources but each a different capacity
- **size.** a VNR can be small compared to other VNRs that span large parts of the physical network

Based on this information, two mechanism are used to increase total revenue: (1) if the utilization level is low (i.e. ample storage space, computing power or bandwidth available), there will be adequate capacity to embed several VNRs as they arrive one after the other. As such, the price of substrate resource i ($P_{i,d}$) is set at a price that is very competitive to attract demand. (2) if the utilization level of a resource is high, inadequate capacity may be available to serve future requests and the InP will no longer be able to embed each VNR (e.g. only 9 out of every 10 VNRs, or less). As such, we wish to attract VNRs that provide high revenue per unit of the substrate resource with a high utilization level (or increase $P_{i,d}$ for VNRs that otherwise provide a low revenue per unit of the substrate resource).

This second mechanism is illustrated in Figure 4. Let us consider two VNRs, VNR1 and 2, which arrive one after the other to be embedded on a substrate network. For simplicity, we only take into account node resources and assume that the price per unit of a substrate resource is the same for all resources and equal to 1. VNR1 needs two different node resources with a capacity of 1 and 2. VNR2 need 3 with a capacity of 1, 3 and 3. The mapping of each VNR to the substrate network is illustrated in Figure 4. The substrate network has one substrate resource with just 2 units left available (black circle) while all other substrate resources have ample capacity (9 units, white circles). When VNR1 is embedded, all of the remaining units of the resource with the lowest remaining capacity (black circle) are used by that request. This leaves no room for additional requests (such as VNR2). In addition, the revenue for embedding VNR1 is relatively low because only 1 additional resource is used by the VNR per time unit. As such, a total revenue of 3 per time unit is generated by VNR1. VNR2 on the other hand, requires just 1 of the remaining units of the resource with the lowest remaining capacity, leaving 1 unit of that resource available for future VNRs. Also, the total revenue of VNR2 is higher as 6 additional units are required (3 from each node). A total revenue of 7 per time unit is generated by VNR2. As such the ratio of the total revenue divided by the units requested of the resource with low remaining capacity is 1.5 for VNR 1 compared to 7 for VNR 2. Clearly, revenue can be increased by rejecting VNR1 (or request a higher price for it) and accepting VNR2.

To take these two mechanisms into account we propose a heuristic which is the dynamic pricing approach described in algorithm 1. It tackles two challenges: (1) at which threshold should we start to charge more for a substrate resource (a resource which is above this threshold is referred to as a constrained resource) and (2) which price should we charge per unit of the constrained resource. The used parameters are summarized in Table I and the proposed dynamic pricing approach in algorithm 1. The algorithm itself is detailed step-by-step below.

Determining $P_{i,d}$. Instead of using Equation 2 for determining P_v, the price of the VNR is determined by multiplication of the dynamic price of the constrained substrate resource

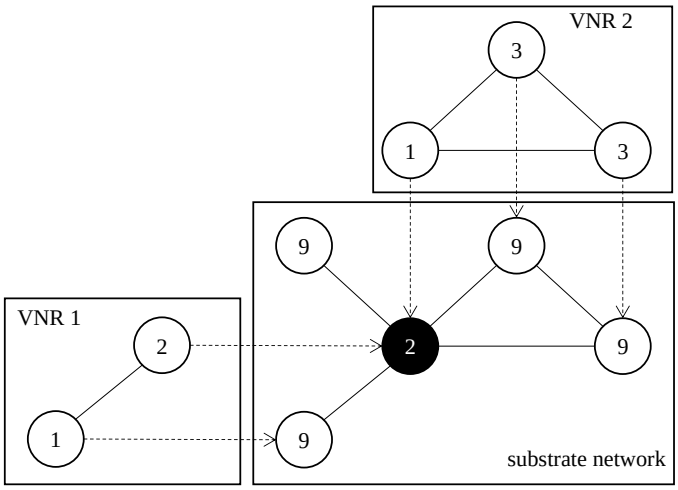

Fig. 4. Embedding of two VNRs on a substrate network with one node with low remaining capacity. The requested or available capacity of each node is indicated inside its circle.

TABLE I
LIST OF PARAMETERS AND THEIR SYMBOLS

parameter	symbol
infrastructure provider (InP)	n, $n \in [1, N]$
virtual network request (VNR)	v, $v \in [1, V]$
substrate resource (SR)	i, $i \in [1, I]$
constrained SR	c, $c \in [1, C]$
acceptance level of substrate resource i	a_i
price of InP n for VNR v	$P_{n,v}$
preferred offering for VNR v	PO_v
price of SR i, static, dynamic	$P_i, P_{i,s}, P_{i,d}$
price of VNR v, static,dynamic	P_v,$P_{v,s}$,$P_{v,d}$
units requested of substrate resource i by VNR v	$R_{i,v}$
duration of VNR v	D_v
expected revenue, at acceptance level a	$E(Rev)$,$E(Rev_a)$
the revenue per unit of SR i for VNR v	$RU_{i,v}$
the average revenue per unit of substrate resource i at the current level of acceptance	$\bar{RU}_{i,a}$
blocking probability of SR i at the current level of acceptance	$P_{b,i,a}$
the system ingress load	E
the mean arrival rate λ of SR i at acceptance level a	$\bar{\lambda}_{i,a}$
the mean service time of SR i	$\bar{s}_i$
the number of servers available for SR i at acceptance level a	$c_{i,a}$
overall acceptance level	A
discount rate	δ

i ($P_{i,d}$) with the number of units requested of the constrained resource by VNR v ($R_{i,v}$). However, when P_v is lower than the P_v obtained by applying equation 1, the latter P_v is used (Equation 3).

$$P_v = max(P_{i,d} \times R_{i,v}, Eq.1 \times (1-\delta)), \forall i \in \{1, I\} \quad (3)$$

In this pricing scheme, $P_{i,d}$ is dynamic and equal to $RU_{i,v=(1-a)\times V+1}$. $RU_{i,v=(1-a)\times V+1}$ is the revenue per unit of substrate resource i for VNR v for the acceptance level a with the highest expected revenue ($E(rev)$). The level of acceptance refers to the number of VNRs that will be priced

Algorithm 1 Dynamic pricing algorithm

while VNR arrive **do**
  map offer
  if offer can be mapped **then**
    gather historic data: PO_v and $R_{i,v}$
    for all $i = 1$ to I **do**
      for all $v = 1$ to V **do**
        calculate $RU_{i,v}$ via Eq. 5
      end for
    end for
    order historic data in ascending order based on $RU_{i,v}$
    for all $i = 1$ to I **do**
      for all $a = \frac{v}{V}$ with $v \in [1, V]$ **do**
        calculate $\bar{RU}_{i,a}$ via Eq. 7
        calculate $c_{i,a}$ via Eq. 9
        calculate $P_{b,i,a}$ via Eq. 8
        calculate $E(Rev_a)$ via Eq. 6
      end for
    end for
    obtain PO_v via Eq. 1
    for all $i = 1$ to I **do**
      select a corresponding with max $E(Rev_a)$
      calculate $P_{i,d}$ via Eq. 4
      calculate $P_{v,d}$ via Eq. 3
      if $P_{v,d} > PO_v$ **then**
        add a_i to list of constrained resources
      end if
    end for
    if multiple constrained resources **then**
      apply algorithm 2 or 3
    end if
    for all $i = 1$ to I **do**
      calculate $P_{i,d}$ via Eq. 4
      calculate $P_{v,d}$ via Eq. 3
      save highest $P_{v,d}$ to P_v
    end for
    send offer to broker with price P_v
    if offer accepted **then**
      embed VNR
    end if
  else if VNR cannot be mapped **then**
    reject VNR
  end if
end while

at an attractive price. For example, an acceptance level of 80% means that 8 out of every 10 VNRs will be priced attractively in respect to the competitors price while the other 2 will be priced at a premium (for which, the client will likely choose for a competitor unless there is no better option). It is also the threshold at which a resource is considered as constrained. Equation 4 is used to calculate $RU_{i,v=(1-a)\times V+1}$.

$$P_{i,d} = RU_{i,v=(1-a)\times V+1} = \frac{PO_v}{R_{i,v}} \tag{4}$$

Determining a. To determine the acceptance level a with the highest $E(rev)$, a database composed of historic data of all VNRs that have been mapped on substrate resource i is maintained. It contains two data fields: (1) the total price charged for VNR v (PO_v) and (2) the units requested of substrate resource i in VNR v ($R_{i,v}$). The ratio $RU_{i,v}$, the revenue per unit of substrate resource i for VNR v, is first calculated for each of the entries according to Equation 5.

$$RU_{i,v} = \frac{PO_v}{R_{i,v}} \tag{5}$$

The database is next ordered in ascending order based on this ratio. Once ordered, the acceptance level a with the highest $E(rev)$ can be found based on two factors: (1) the average revenue per unit of substrate resource i used at the current level of acceptance ($\bar{RU}_{i,a}$) and (2) the blocking probability of substrate resource i at the current level of acceptance ($P_{b,i,a}$).

$$E(Rev_a) = \bar{RU}_{i,a} \times (1 - P_{b,i,a}) \times a \tag{6}$$

Once $E(Rev_a)$ is determined for all a, the acceptance level a that corresponds with the the highest $E(Rev)$ is chosen from the ordered list and used in Equation 4.

Determining $\bar{RU}_{i,a}$. The ordered database is used to calculate $\bar{RU}_{i,a}$. $\bar{RU}_{i,a}$ is then determined according to Equation 7.

$$\bar{RU}_{i,a} = \frac{\sum_{v=(1-a)\times V+1}^{V} \frac{PO_v}{R_{i,v}}}{(V - v + 1)}, v \in [1, V], a = \frac{v}{V} \tag{7}$$

Determining $P_{b,i,a}$. To calculate $P_{b,i,a}$, the VNRs arriving at a substrate resource are modeled as a $M/M/c/K$ queuing system (Kendall's notation). This is a queuing system that needs to satisfy the conditions that arrivals form a single queue and arrive according to a Poisson process (M, memoryless), that service times are exponentially distributed (second M), that there are c servers which serve from the front of the queue and a buffer capacity of K (including those in service). In an $M/M/c/K$ queue only K customers can queue at any one time (including those in service). Any further arrivals to the queue are considered lost for service. In this case, a substrate resource is either able to map a VNR or not. As such, the buffer size K is zero ($c = K$, $M/M/c/c$). The Erlang B formula (also known as the Erlang loss formula), can be used to calculate the blocking probability that describes the probability of losses for a group of identical parallel resources (Equation 8). In Equation 8, $c_{i,a}$ is the number of servers available for substrate resource i at acceptance level a and E is the systems ingress load in erlang which is calculated as the mean arrival rate λ of substrate resource i at acceptance level a ($\bar{\lambda}_{i,a}$) multiplied by the mean service time of substrate resource i ($\bar{s}_i$).

$$P_{b,i,a} = \frac{\frac{E^{c_{i,a}}}{c_{i,a}!}}{\sum_{j=0}^{c_{i,a}} \frac{E^j}{j!}} \tag{8}$$

$\bar{\lambda}_{i,a}$ is calculated by multiplying the historic arrival rate (the number of VNRs that arrive per time unit) with the acceptance level a. $\bar{s}_i$ is calculated as the mean service time of all VNRs that are mapped and use substrate resource i. To estimate $c_{i,a}$, the ordered database of historic data which is maintained per substrate resource is used in combination with the units available of the substrate resource i considered (A_i). $c_{i,a}$ can be calculated via Equation 9.

$$c_{i,a} = \lfloor \frac{\sum_{v=(1-a)\times V+1}^{V} \frac{A_i}{D_v}}{(V-v+1)} \rfloor, v \in [1,V], a = \frac{v}{V} \quad (9)$$

Once $c_{i,a}$ is known, $P_{b,i,a}$ can be determined via Equation 8. Once $\bar{RU}_{i,a}$ and $P_{b,i,a}$ are known, $E(Rev_a)$ can be calculated via Equation 6, etc.

Multiple constrained resources. When using the algorithm described above to price a VNR, multiple resources may be constrained instead of just one. In that situation, the algorithm calculates the (dynamic) price of the VNR as the highest of the prices of all constrained resources (Equation 3). As such, for all but one of the constrained resources, a higher price will be charged than the price that corresponds with the acceptance level of the constrained resources (Equation 4). This would not be a problem if the same set of resources would always be involved in a VNR and if the same resource would always determine the highest price. This is however not the case as each VNR may request a different set of resources and over time different resources will be constrained or not (e.g. after a VNR has terminated, an otherwise constrained resource may become non-constrained). As a consequence of the higher price, the actual acceptance level of individual constrained substrate resource i (a_i) will be lower than the acceptance level we would like to obtain. Similarly, the acceptance level of the combined resources for VNR v A_v will be lower than the acceptance level of each individual constrained substrate resource i (a_i). As a result, fewer VNRs will be attracted than initially hoped. We propose two alternative approaches to obtain a better balance between the A_v and a_i.

In the first approach, the acceptance level of individual resources a_i is scaled up to bring the combined acceptance level A_v closer to the original acceptance level of individual resources a_i. Algorithm 2 summarizes this approach.

The drawback of the scaling approach is that it assumes no correlation between the constrained resources. This is clearly untrue when VNRs exist that have similar requirements. We therefore also propose an alternative approach. In this second approach, we do not scale the the acceptance level of individual resources a_i, instead the dynamic price is set to the average of the prices obtained by using Equation 4 for each of the constrained resources. Algorithm 3 summarizes this approach.

Algorithm 2 Scale availability

```
Ā_v ← (Σ_{i=1}^{C} a_i) / C
A_v* ← Π_{i=1}^{C} a_i
while Ā_v ≠ A_v* do
  S = ∛(Ā_v / A_v*)
  for all i such that 1 ≤ i ≤ C do
    a_i = min(1, S × a_i)
    if a_i > 1 then
      remove a_i from the list of constrained resources
    end if
  end for
  A_v* ← Π_{i=1}^{C} a_i
end while
```

Algorithm 3 Average of constrained resources

```
c = 0, P_v = 0
for all i = 1 to I do
  if a_i = 1 then
    calculate P_{i,d} via Equation 4
    P_v = P_v + P_{i,d}, c = c + 1
  end if
end for
P_v = P_v / c
```

V. Performance evaluation

The focus of our evaluations is on quantifying the benefit of the proposed revenue management algorithm in terms of total revenue. To ensure a fair comparison we model two identical InPs (i.e. same network topology and substrate capacity) who compete against each other. The first InP uses the dynamic pricing algorithm while the second uses a static pricing algorithm.

A. Simulation setup.

We compare different simulation setups: (1) different inter-arrival rates ($\frac{1}{\lambda}$) and (2) different capacity requested per virtual node and link. The scenarios are summarized in Table II. Each setup is simulated for $20,000$ VNR arrivals. In the dynamic pricing algorithm, a discount δ of 5% is given (Equation 3).

TABLE II
Overview of the simulation input parameters

scenario	substrate node capacity	virtual node capacity	substrate link capacity	virtual link capacity	$\frac{1}{\lambda}$
1	100-200	10-20	200-400	16-40	1
2	100-200	10-20	200-400	16-40	2
3	100-200	10-20	200-400	16-40	3
4	100-200	10-20	200-400	16-40	5
5	100-200	25-50	200-400	40-100	1
6	100-200	25-50	200-400	40-100	2
7	100-200	25-50	200-400	40-100	3
8	100-200	25-50	200-400	40-100	5

Substrate network. The substrate network is modeled as an undirected graph. The infrastructure consists of nodes connected via links. Each node has certain capacity in terms of computation, memory and/or storage, each link has a certain capacity in terms of bandwidth and has a certain delay. The substrate network used in the simulations has 25 nodes and 75 links. The minimum and maximum capacity (e.g. available

storage capacity) of each substrate node and link is given in Table II.

Virtual network request. Each VNR is represented as a directed graph to support the dependency between elementary NFs. The NFs are represented as nodes connected via directed links in the graph. Each NF has certain requirements in terms of computation, memory and/or storage and links connecting different NFs should meet certain requirement in terms of maximum allowed delay and bandwidth. The virtual networks used in the simulation have a maximum of 7 nodes and 12 links. The minimum and maximum capacity of each virtual node and link is given in Table II. The average service time is 100.

Virtual network embedding algorithm. The VNE algorithm is an implementation of a link-based multi-commodity flow formulation of the one-shot virtual network embedding [27] in CPLEX 12.6.

Negotiation process. The VNRs are awarded to the InP according to the negotiation process depicted in Figure 3 (we assume 2 InPs). The result of the negotiation process is classified in 5 categories: (1) F, the VNR cannot be mapped to either InP (e.g. the set of requirements cannot be met, inadequate substrate resources, etc.), (2) M1, only the first InP is able to map the request (e.g. the second InP has inadequate substrate resources), (3) M2, only the second InP is able to map the request, (4) P1, both InPs are able to map and InP 1 has the best offer (i.e. InP 1 offers a lower price) and (5) P2, both InPs are able to map and InP 2 has the best offer.

B. Results of the dynamic pricing algorithm with scaling of availability when multiple resources are constrained

We report the embedding results for each of the scenarios in Table III. As can be expected, when demand for substrate resources is high, e.g. due to a low interarrival time (requests follow each other fast) or/and when VNRs demand a large share of the substrate resources (request use a large portion of the available substrate resources), the number of failed mappings is large and vice versa. Also, the number of VNRs that are won by the first provider by undercutting the competitor's price (P1) decreases when demand is high until both are more or less equal for very high levels of demand (e.g. simulation 5). This can be understood as (1) only very few requests can be mapped on the substrate network of both InPs and (2) the VNRs that receive a discount from InP 1 will be limited to those VNRs that have a high payoff per unit of the constrained resource. In general we can observe that for very low levels of demand (e.g. simulation 4), the dynamic pricing algorithm will attract as much requests as possible by offering a discount (quantity over quality). For very high levels of demand (e.g. simulation 1), only a small share of all VNRs will retrieve a discount (typically those with a high revenue per constrained resource) while many VNR offers include a premium to protect the constrained resource from otherwise low value requests. As a result, the InP which applies static pricing (InP2) may embed more VNRs than the InP using the dynamic pricing algorithm. As we illustrate below, this will not negatively impact the total revenue because the revenue per VNR is significantly higher when applying the dynamic pricing algorithm (quality over quantity). For normal levels of demand (e.g. simulation 2 and 3) the dynamic pricing algorithm will carefully balance quality and quantity by changing dynamically over time the price charged for a VNR based on the current utilization levels of the substrate resources involved.

TABLE III
EMBEDDING RESULTS FOR EACH SIMULATION SETUP FOR THE DYNAMIC PRICING ALGORITHM WITH SCALING.

simulation	M1	M2	P1	P2	F	total InP1	total InP2
1	15%	22%	13%	10%	40%	28%	32%
2	2%	26%	44%	18%	10%	46%	44%
3	0%	17%	68%	13%	2%	68%	30%
4	0%	7%	83%	9%	0%	83%	17%
5	9%	9%	1%	1%	80%	10%	10%
6	13%	17%	8%	4%	59%	21%	20%
7	15%	23%	13%	4%	44%	29%	27%
8	5%	28%	45%	13%	9%	50%	41%

When we focus on the total number of VNRs that each InP has obtained (its market share) it is clear that when demand is relatively slow (e.g. simulation 4), the first provider obtains the highest market share and also the highest node utilization and link utilization (Figure 5). This is reached by systematically undercutting the price of its competitors for those VNRs that are considered as valuable and results in a higher total revenue (Figure 6). It is however less straightforward that InP1 is able to reach a higher total revenue than InP2 when its market share is lower (e.g. simulation 1). To clarify this we need to take into account the node and link utilization rates. These are higher even though the market share of the first InP is lower. The proposed revenue management model is able to obtain this result by pricing VNRs that have a high revenue per unit of the constrained substrate resources lower than its competitors while demanding a premium for those VNRs that have a low revenue per unit of the constrained substrate resources (Equation 3). The impact of this decision is further clarified in Table IV which presents the average revenue per VNR for each embedding result (100% represents the highest obtained average revenue for a particular simulation, the other percentages are relative to the highest obtained average revenue). By focusing on high value requests, the InP is able to increase its revenue per VNR. To do so, the InP needs to use its constrained resources optimally (certain substrate nodes and links) and at the same time reach a higher utilization rate for those resources that have a lower demand (e.g. substrate nodes with ample capacity).

C. Results of the dynamic pricing algorithm with averaged out availabilities when multiple resources are constrained

The results discussed above use the scaling approach (Algorithm 2) to handle the situation in which multiple resources are constrained at the same time. It scales the acceptance level of resource i up to reflect that for different VNRs, different substrate resources are the most constrained. As a result of this

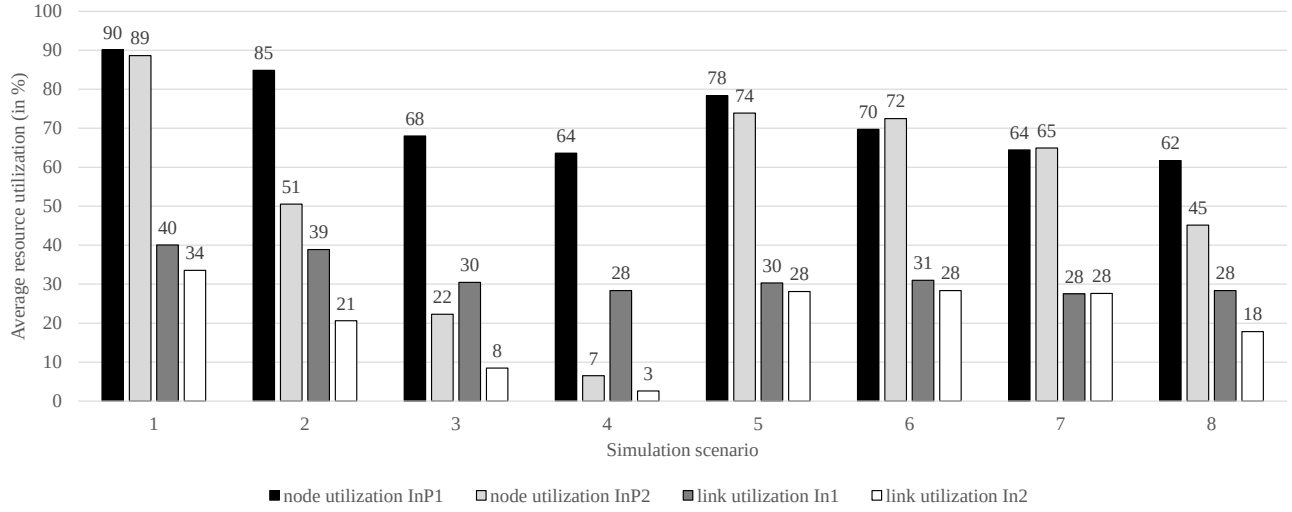

Fig. 5. Average node and link utilization per simulation and per provider for the dynamic pricing algorithm with scaling.

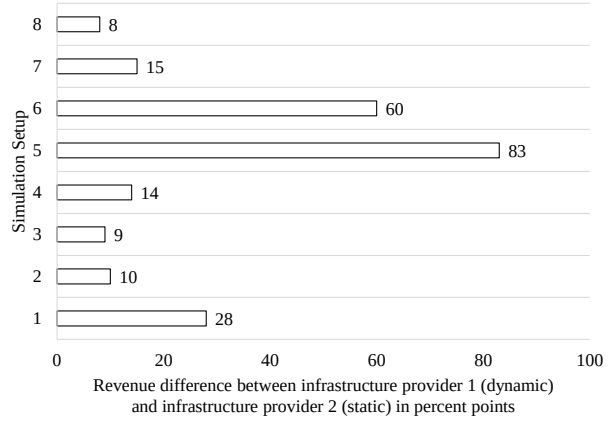

Fig. 6. Percent points difference in revenue of InP1 (dynamic) compared to InP2 (static) per simulation for the dynamic pricing algorithm with scaling.

TABLE IV
COMPARISON OF AVERAGE REVENUE PER VNR FOR THE DYNAMIC PRICING ALGORITHM WITH SCALING.

simulation	average revenue P1	average revenue P2	average revenue M1	average revenue M2
1	93%	50%	100%	92%
2	98%	60%	100%	99%
3	100%	62%	96%	98%
4	100%	62%	98%	99%
5	100%	65%	99%	89%
6	92%	64%	100%	95%
7	97%	70%	100%	97%
8	100%	72%	99%	72%

correction, the acceptance level of each resource will be closer to the optimal acceptance level. However, with this approach, each substrate resource is considered as being independent from other substrate resources. In a VNR this is not the case (as they form a network of connected resources), similarly a relationship between the substrate resource on which VNRs are embedded can be expected (e.g. between the utilization level of a node and its connected links). As a result, the scaling approach can be further improved. Algorithm 3 is proposed as an alternative. It uses the average price of a VNR for all constrained resources instead of using the highest price. This approach indirectly takes into account the level of interdependence between resources. Table V provides an overview of the embedding results for the simulations as described in Table II.

TABLE V
EMBEDDING RESULTS FOR EACH SIMULATION SETUP FOR THE DYNAMIC PRICING ALGORITHM WITH AVERAGES.

simulation	M1	M2	P1	P2	F	total InP1	total InP2
1	14%	19%	17%	13%	37%	31%	32%
2	1%	21%	49%	21%	8%	50%	42%
3	0%	16%	71%	12%	1%	71%	28%
4	0%	5%	85%	10%	0%	85%	15%
5	9%	10%	2%	1%	78%	11%	11%
6	10%	13%	13%	7%	57%	23%	20%
7	15%	22%	17%	4%	42%	32%	26%
8	4%	28%	50%	9%	9%	54%	37%

Table V indicates that by applying this approach, an increase of the percentage of VNRs that are embedded by InP1 (M1+P1) and a decrease of the percentage that are embedded by InP2 (M2+P2) or neither InP (F) compared to the results for Algoirthm 2 presented in Table III. The increase for InP1 is mainly explained by an increase in the number of VNRs that are embedded after winning based on offering the best price ($P1$). On the other hand the number of VNRs that could not be embedded is reduced (F) as well as the number of VNRs that could only be embedded by a single InP ($M1$ and $M2$).

Table VI shows that the relative difference in the average price per VNR has increased between InP1 and InP2. In particular for those VNRs that could only be mapped to a

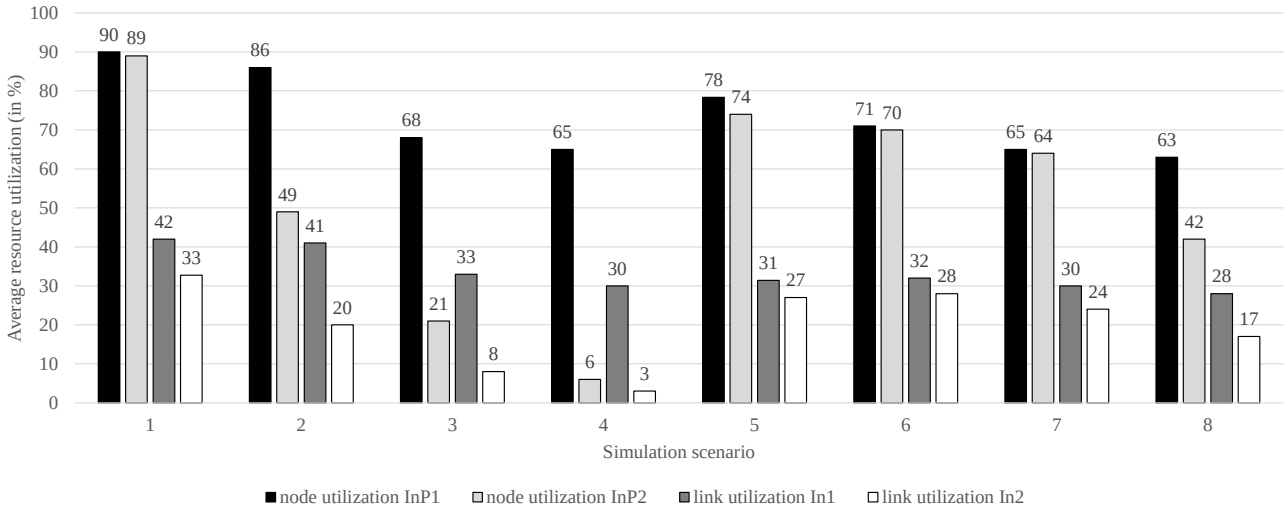

Fig. 7. Average node and link utilization per simulation and per provider for the dynamic pricing algorithm with averages.

single InP (M1 versus M2), the delta for those requests that could be embedded by both InPs remains stable.

TABLE VI
COMPARISON OF AVERAGE REVENUE PER VNR FOR THE DYNAMIC PRICING ALGORITHM WITH AVERAGES.

simulation	average revenue P1	average revenue P2	average revenue M1	average revenue M2
1	93%	62%	100%	91%
2	89%	66%	100%	92%
3	100%	62%	97%	96%
4	100%	62%	98%	98%
5	97%	61%	100%	86%
6	96%	56%	100%	98%
7	98%	73%	100%	89%
8	100%	71%	98%	69%

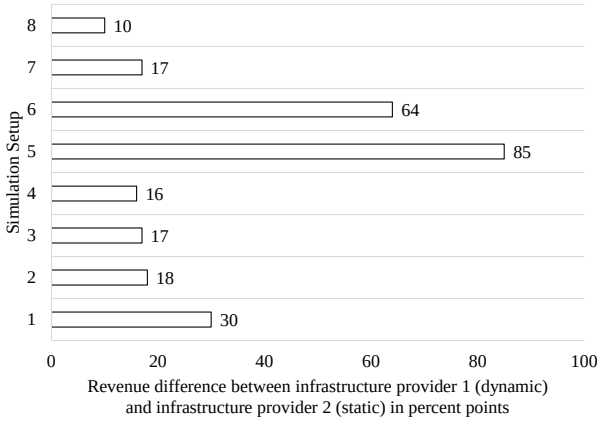

Fig. 8. Percent points difference in revenue of InP1 (dynamic) compared to InP2 (static) per simulation for the dynamic pricing algorithm with averages.

As a consequence of the increased number of VNRs that could be embedded by the first InP and the increase in the relative difference in the average revenue per VNR between InP1 and InP2, we would expect an increase in the total revenue as well as in the node and link utilization levels. Figures 7 and 8 confirm these expectations. As such, we have shown via simulation that the approach with averages (Algorithm 3) outperforms the approach with scaling (Algorithm 2) as it is able to handle multiple constrained resources that are interrelated.

VI. CONCLUSION AND OUTLOOK

This article discusses how pricing can increase the total revenue of an InP in a competitive market with price-sensitive customers. Two different pricing approaches are analyzed: a traditional static pricing approach and the proposed dynamic pricing approach.

The proposed approach is a heuristic which is able to increase the total revenue of the InP compared to a static pricing approach by pricing resources differently over time. To determine the appropriate price, a combination of market data, historic data and the current state of the substrate network is used. A two-fold strategy is followed: (1) when the utilization of a particular substrate resource is low, VNRs are attracted by setting the price below that of competitors and (2) when the utilization of a particular substrate resource is high, VNRs that provide a high revenue per unit of the substrate resource are attracted by proposing a competitive price while low value VNRs are only embedded if a premium (compared to the static price) is paid. The proposed algorithm tackles the two key challenges to apply this strategy: (1) determination of the level at which the utilization of a resource is considered as high and (2) determination of the price that needs to be charged for a particular resource depending on the current utilization level of that resource. The dynamic pricing algorithm has been validated via simulations and outperforms a static pricing approach significantly (by 8-85 percent points for the considered scenarios).

Although the advantages of a dynamic pricing approach can be observed through this paper, there are still many issues that could be of interest for future research. For example, it is unclear how the total revenue of an InP is affected when multiple or all competing InP use a dynamic pricing algorithm, this will therefore remain the focus of our future research work.

ACKNOWLEDGMENT

This work is partly funded by the FP7 UNIFY (grant. no. 619609) and FLAMINGO, a Network of Excellence project (grant. no. 318488), supported by the European Commission under its Seventh Framework Programme.

REFERENCES

[1] P. Quinn and T. Nadeau, "Service function chaining problem statement," *draft-ietf-sfc-problem-statement-10*, 2014.

[2] R. Mijumbi, J. Serrat, J. L. Gorricho, N. Bouten, F. D. Turck, and R. Boutaba, "Network function virtualization: State-of-the-art and research challenges," *IEEE Communications Surveys Tutorials*, vol. 18, no. 1, pp. 236–262, Firstquarter 2016.

[3] A. Gupta, J. Kleinberg, A. Kumar, R. Rastogi, and B. Yener, "Provisioning a virtual private network: a network design problem for multicommodity flow," in *Proceedings of the thirty-third annual ACM symposium on Theory of computing*. ACM, 2001, pp. 389–398.

[4] N. G. Duffield, P. Goyal, A. Greenberg, P. Mishra, K. Ramakrishnan, and J. E. Van der Merwe, "Resource management with hoses: point-to-cloud services for virtual private networks," *Networking, IEEE/ACM Transactions on*, vol. 10, no. 5, pp. 679–692, 2002.

[5] R. Ricci, C. Alfeld, and J. Lepreau, "A solver for the network testbed mapping problem," *ACM SIGCOMM Computer Communication Review*, vol. 33, no. 2, pp. 65–81, 2003.

[6] A. Fischer, J. F. Botero, M. Till Beck, H. De Meer, and X. Hesselbach, "Virtual network embedding: A survey," *Communications Surveys & Tutorials, IEEE*, vol. 15, no. 4, pp. 1888–1906, 2013.

[7] S. Mehraghdam, M. Keller, and H. Karl, "Specifying and placing chains of virtual network functions," in *Cloud Networking (CloudNet), 2014 IEEE 3rd International Conference on*. IEEE, 2014, pp. 7–13.

[8] M. C. Luizelli, L. R. Bays, L. S. Buriol, M. P. Barcellos, and L. P. Gaspary, "Piecing together the nfv provisioning puzzle: Efficient placement and chaining of virtual network functions," *IFIP/IEEE IM*, 2015.

[9] H. Moens and F. De Turck, "Vnf-p: A model for efficient placement of virtualized network functions," in *Network and Service Management (CNSM), 2014 10th International Conference on*. IEEE, 2014, pp. 418–423.

[10] S. Sahhaf, W. Tavernier, M. Rost, S. Schmid, D. Colle, M. Pickavet, and P. Demeester, "Network service chaining with optimized network function embedding supporting service decompositions," *Computer Networks*, 2015.

[11] M. Xia, M. Shirazipour, Y. Zhang, H. Green, and A. Takacs, "Network function placement for nfv chaining in packet/optical datacenters," *Journal of Lightwave Technology*, vol. 33, no. 8, pp. 1565–1570, 2014.

[12] M. F. Bari, S. R. Chowdhury, R. Ahmed, and R. Boutaba, "On orchestrating virtual network functions," http://arxiv.org/abs/1503.06377, 2015, [Online; accessed 5-November-2015].

[13] R. Mijumbi, J. Serrat, J.-L. Gorricho, N. Bouten, F. De Turck, and S. Davy, "Design and evaluation of algorithms for mapping and scheduling of virtual network functions," in *Network Softwarization (NetSoft 2015), 2015 IEEE Conference on*. IEEE, 2015, pp. 1–9.

[14] G. Bitran and R. Caldentey, "An overview of pricing models for revenue management," *Manufacturing & Service Operations Management*, vol. 5, no. 3, pp. 203–229, 2003.

[15] M. K. Geraghty and E. Johnson, "Revenue management saves national car rental," *Interfaces*, vol. 27, no. 1, pp. 107–127, 1997.

[16] B. C. Smith, J. F. Leimkuhler, and R. M. Darrow, "Yield management at american airlines," *interfaces*, vol. 22, no. 1, pp. 8–31, 1992.

[17] B. Javadi, R. K. Thulasiram, and R. Buyya, "Statistical modeling of spot instance prices in public cloud environments," in *Utility and Cloud Computing (UCC), 2011 Fourth IEEE International Conference on*. IEEE, 2011, pp. 219–228.

[18] H. Wang, Q. Jing, R. Chen, B. He, Z. Qian, and L. Zhou, "Distributed systems meet economics: pricing in the cloud," in *Proceedings of the 2nd USENIX conference on Hot topics in cloud computing*. USENIX Association, 2010, pp. 6–6.

[19] V. Kantere, D. Dash, G. Francois, S. Kyriakopoulou, and A. Ailamaki, "Optimal service pricing for a cloud cache," *Knowledge and Data Engineering, IEEE Transactions on*, vol. 23, no. 9, pp. 1345–1358, 2011.

[20] H. Xu and B. Li, "A study of pricing for cloud resources," *ACM SIGMETRICS Performance Evaluation Review*, vol. 40, no. 4, pp. 3–12, 2013.

[21] ——, "Dynamic cloud pricing for revenue maximization," *Cloud Computing, IEEE Transactions on*, vol. 1, no. 2, pp. 158–171, 2013.

[22] N. Bouten, M. Claeys, R. Mijumbi, J. Serrat, J. Famaey, S. Latré, and F. De Turck, "Semantic validation of affinity constrained service function chain requests," in *Network Softwarization (NetSoft 2016), 2016 IEEE Conference on*. IEEE, 2016, pp. 1–9.

[23] R. Mijumbi, J.-L. Gorricho, J. Serrat, J. Rubio-Loyola, and R. Aguero, "Survivability-oriented negotiation algorithms for multi-domain virtual networks," in *Network and Service Management (CNSM), 2014 10th International Conference on*. IEEE, 2014, pp. 276–279.

[24] A. Jarray and A. Karmouch, "Vcg auction-based approach for efficient virtual network embedding," in *Integrated Network Management (IM 2013), 2013 IFIP/IEEE International Symposium on*. IEEE, 2013, pp. 609–615.

[25] S. Latre, J. Famaey, F. De Turck, and P. Demeester, "The fluid internet: service-centric management of a virtualized future internet," *Communications Magazine, IEEE*, vol. 52, no. 1, pp. 140–148, 2014.

[26] W. Tavernier, B. Naudts, D. Colle, M. Pickavet, and S. Verbrugge, "Can open-source projects (re-) shape the sdn/nfv-driven telecommunication market?" *it-Information Technology*, vol. 57, no. 5, pp. 267–276, 2015.

[27] R. Mijumbi, J. Serrat, J.-L. Gorricho, and R. Boutaba, "A path generation approach to embedding of virtual networks," *Network and Service Management, IEEE Transactions on*, vol. 12, no. 3, pp. 334–348, 2015.