The Cloud Service Distribution Problem in Distributed Cloud Networks

Marc Barcelo*, Jaime Llorca†, Antonia M. Tulino†, Narayan Raman†
*Universidad Autonoma de Barcelona, Spain, Email: marc.barcelo@uab.cat
†Bell Labs, Alcatel-Lucent, NJ, USA, Email: {jaime.llorca, a.tulino, narayan.raman}@alcatel-lucent.com

***Abstract*—The cloud service distribution problem (CSDP) is to find the placement of both content and virtual cloud service functions (vCSFs) over a distributed cloud network platform, that meets user requests, satisfies network resource capacities and minimizes overall network cost. We formulate the CSDP as a minimum cost mixed-cast flow problem in which cloud services are represented by a service graph that encodes the relationship between input and output information flows via the virtual functions that create them. As a result, the CSDP can be efficiently formulated using only linear constraints and solved via integer linear programming (ILP). Our solution jointly optimizes the use of compute, storage and transport resources in arbitrary cloud network topologies, and is able to capture flexible service chaining, resource consolidation savings, unicast and multicast delivery, and latency constraints. We further provide conditions for which a relaxed version of the presented ILP leads to optimal polynomial-time solutions. We finally present results for an illustrative sample of cloud services that show the advantage of optimizing the placement of content and vCSFs over a programmable distributed cloud network.**

Index Terms — distributed cloud network, cloud service distribution, software defined networking, network functions virtualization

I. Introduction

Internet traffic will soon be dominated by the consumption of resource and interaction intensive cloud services and applications from an increasing number of resource-limited communication end points. Both end user experience and overall network efficiency will push for a fundamental transformation of today's centralized network and cloud architectures towards a highly distributed converged cloud-network platform, composed of a large number of cloud nodes distributed across an increasingly meshed converged metro-core [1], [2]. In this massively distributed cloud network, and with the introduction of network functions virtualization (NFV) and software defined networking (SDN), virtual cloud service functions (vCSFs) can be dynamically and elastically instantiated over commodity servers at multiple cloud locations close to the end users and interconnected via a programmable network fabric [3]. This way, a cloud network operator can host a variety of services and applications over a common distributed physical infrastructure, reducing both capital and operational expenses, while delivering high quality of experience (QoE). To maximize the benefits of this attractive scenario and enable its sustainable growth, a cloud network operator must be able to deploy cloud services in such a way that the allocation of the associated virtual cloud service functions and the content/information they require minimizes the overall use of the physical resources. In this paper, we introduce the *cloud service distribution problem* (CSDP), as a generalization of the content distribution problem, in which the goal is to find the placement of both content and virtual cloud service functions such that user demands are satisfied with guaranteed QoE and minimum overall network cost.

A. Related Work

The data placement problem, introduced in [4], was generalized to the content distribution problem (CDP) in [5], where the goal is to find the placement and routing of content objects that meets user demands with minimum overall network cost. The CDP is formulated as a min-cost mixed-cast flow problem, shown to admit polynomial-time solutions when the network is allowed to perform intra-session network coding. Recently, motivated by the advent of distributed cloud networking [3], the work in [6] addressed the CDP in virtual content distribution networks, in which the placement and routing of content objects is jointly optimized with the allocation of the required virtual storage and transport resources, essentially combining the CDP and virtual network embedding (VNE) problems. In [7], [8], the authors addressed the problem of placing video processing functions for efficient distribution of next generation video services, in which source video streams can be combined through the network to create personalized streams for the end user, and provided a polynomial-time algorithm for tree networks.

B. Contributions

The CSDP, introduced in this paper, includes and generalizes the problems above via the notion of the *service graph*, which establishes the relationship between content/information objects via the virtual functions that manipulate them. Following the approach of [5] and [6], and via the use of the *service graph*, the CSDP can be efficiently formulated as a minimum cost mixed-cast flow problem with compute, storage and transport resource activation decisions, and solved via (integer) linear programming, as shown in Section III. One of the key aspects of the proposed formulation is the introduction of the *generalized flow conservation* constraints, which assure the conservation of the combined transport, storage and processing flows at every node in the network. Section III further provides conditions for which a relaxed version of the proposed ILP leads to optimal polynomial-time solutions.

978-1-4673-6432-4/15/$31.00 ©2015 IEEE

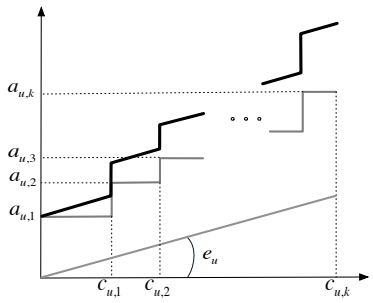


Fig. 1: The cost function of a generic resource at cloud node $u \in \mathcal{V}$, where $c_{u,k}$ and $a_{u,k}$ are the capacity and activation cost of k physical resource units (e.g., k compute servers) at node $u \in \mathcal{V}$, and e_u is the efficiency of the given physical resource at node $u \in \mathcal{V}$.

II. System Model

A. Cloud network model

We consider a cloud network modeled as a directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with $|\mathcal{V}|$ vertices and $|\mathcal{E}|$ edges representing the set of network nodes and links, respectively. In the context of a distributed cloud network, a node represents a distributed cloud location, in which virtual cloud service functions or vCSFs can be instantiated over commercial of the shelf (COTS) compute and storage servers, while an edge represents a virtual link (e.g., IP link) between two cloud locations. A cloud service is implemented as a virtual network slice in $\mathcal{G}$, referred to as a virtual cloud service network or vCSN. A vCSN is hence composed of a set of interconnected vCSFs over the programmable cloud network $\mathcal{G}$.

B. Cost model

We assume three types of physical resources: processing resources, in the form of compute servers; storage resources, in the form storage servers; and transport resources in the form of IP links between cloud locations. The cost associated with the activation and operation of a physical resource is modeled according to the following two components:

- Activation cost: cost of switching on or activating a physical resource and its operation at zero load.
- Operational cost: cost of operating a physical resource as a function of the load on that resource.

Fig.1 illustrates the cost function for a generic resource at node $u \in \mathcal{V}$. Specifically, we denote with c_u^{pr} and c_u^{st} the capacity of a compute server (in bps) and of a storage server (in bits) at cloud node $u \in \mathcal{V}$, respectively, and with $a_{u,k}^{pr}$ and $a_{u,k}^{st}$ the cost associated with the activation of k compute servers and k storage servers at node $u \in \mathcal{V}$, respectively. As illustrated in Fig. 1, typically, $a_{u,k} < ka_u$ to capture the benefits of resource consolidation. Moreover, we denote with e_u^{pr} the efficiency (cost per bps) of the compute servers at node $u \in \mathcal{V}$, and with e_u^{st} the efficiency (cost per bit) of the storage servers at node $u \in \mathcal{V}$. Analogously, c^{tr} denotes the capacity of an IP link (in bps), $a_{vu,k}^{tr}$ the cost associated with the activation of k IP links between nodes v and u, and e_{vu}^{tr} the efficiency (cost per bps) of and IP link between v and u.

C. Cloud service model

We denote with $\mathcal{O}$ the set of information objects or flows that can be stored, processed or transported over the cloud network, and with $\mathcal{P}$ the set of virtual functions that can be implemented over the cloud network. An information object $o \in \mathcal{O}$ may represent: 1) hosted content (e.g., stored media offered by an on-demand content provider), 2) live content (e.g., real-time stream produced by a live-content provider), 3) processed content (e.g., personalized stream resulting from the combination of multiple source streams). An information object $o \in \mathcal{O}$ is, in general, the output of a function $p_o \in \mathcal{P}$ that requires the set of objects $\mathcal{Z}(o)$ as input. For example, in a personalized video streaming application, the final information object requested by the end user is a processed video stream that results from the combination of multiple source video streams via a video processing function.

We represent a cloud service $\phi \in \Omega$ by a rooted graph $\mathcal{T}_\phi = (\mathcal{A}_\phi, \mathcal{O}_\phi)$, referred to as the *service graph*. For any node $o \in \mathcal{O}_\phi \subset \mathcal{O}$, there is a directed edge $(z, o) \in \mathcal{A}_\phi$ for all $z \in \mathcal{Z}(o)$, as shown in Fig. 2. Hence, the set of objects $\mathcal{Z}(o) \subset \mathcal{O}$ required to generate object o via function p_o are represented as the children of o in the service graph $\mathcal{T}_\phi$. In particular, the root of the service graph $r_\phi \in \mathcal{R} \subset \mathcal{O}$ represents the final information object that needs to be delivered to the end user(s). Note that r_ϕ has no outgoing links in $\mathcal{T}_\phi$, as it is not the input to any other object for cloud service ϕ, while the leaves of the rooted graph have no incoming links, as they represent source or hosted information that does not need to be created by a cloud service function. The service graph hence encodes the relationship between all the objects necessary to deliver the final product to the end user(s) via the required cloud functions.

It is important to note that each object $o \in \mathcal{O}$ is uniquely characterized by the pair $(p_o, \mathcal{Z}(o))$. This allows different objects to share the same input, but be created via different functions, or to share the same function, but have different inputs. Also, multiple services may include a same object $o \in \mathcal{O}$ as part of their service graphs. Finally, a cloud service can also be represented by multiple service graphs depending on the granularity of the demands for the service available to the cloud network operator. For example, an operator can optimize the placement of functions for a mobile video service by using as input the average demand for video coming from the operator's mobile customers. In this case, a single service graph with its root representing the general class of mobile video, and its arcs encoding the different functions the source video needs to go through to get delivered to the mobile users, suffices. On the other hand, an operator may want to optimize its deployment by exploiting the specific demands for each video file/class available based on their popularity. In this case, multiple service graphs, each rooted at each of the video files/classes shall be used (see example in Fig. 5c).

We refer to a cloud service as a service implemented in the cloud or, in this case, in a distributed cloud network. A service can be, for example, a network service, such as an

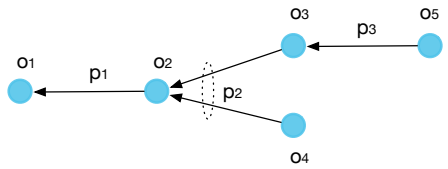


Fig. 2: An example of a service graph, $\mathcal{T}_\phi = (\mathcal{A}_\phi, \mathcal{O}_\phi)$, with $|\mathcal{O}_\phi| = 5$ objects, $|\mathcal{A}_\phi| = 4$ edges, and $|\mathcal{P}| = 3$ virtual functions.

"evolved packet core" service (vEPC), which then requires a set of virtual network functions or VNFs (e.g, vMME, vPGW, vSGW) [3]. But it can also be a commercial application service such as an immersive video service, which requires a set of virtual functions that determine how to combine multiple source video streams to generate final immersive video streams for the end users, for example. In this paper, we refer to the virtual functions of a cloud service, at any level, as virtual cloud service functions or vCSFs.

D. Service requirements

When a user requests a cloud service ϕ, the user is, in essence, requesting the final information object or flow represented by the root of the service graph, $r_\phi \in \mathcal{R}$. We assume that a content object $o \in \mathcal{O}$ may represent a specific content item or flow (e.g., a media file, a video feed from a particular source) or a set of content items or flows (e.g., a set of files in the same popularity class, a set of flows originating at a particular network location). We denote with B_o the size in bits of content object (or set of content objects) $o \in \mathcal{O}$, and with $r_{d,o}$ the average request rate (requests per second) for object $o \in \mathcal{R}$ originating at destination $d \in \mathcal{V}$ (i.e., from the users served by cloud node $d \in \mathcal{V}$). Then, the bit-rate associated to the flow of object $o \in \mathcal{O}$ in bps is given by $\lambda_{d,o} = r_{d,o}B_o$. In the case of object flows not associated to stored objects, but coming from live content sources, we use $\lambda_{d,o}$ to directly denote the bit-rate of live flow $o \in \mathcal{O}$.

We denote with h_{vu} the transport delay associated to link $(v,u) \in \mathcal{E}$ and with h_u the processing delay at cloud node $u \in \mathcal{V}$. We then use $H_{d,o}$ to denote the maximum delay (or maximum number of hops) allowed for the delivery of content object/class $o \in \mathcal{O}$ at destination $d \in \mathcal{V}$, as determined by the given service level agreements (SLAs).

III. Cloud Service Distribution Problem (CSDP)

We formulate the CSDP as a generalized minimum cost mixed-cast flow problem [5], in which processing flows are used to model vCSF placement decisions. In addition, extending the approach in [6], resource activation variables are used to model the allocation of compute, storage, and transport resources to accommodate the respective flows. In particular, the CDSP is characterized by the following flow variables:

- User-object flows: are characterized by a triplet (d,o,z), which indicates the object being carried $z \in \mathcal{O}$, the intended final product $o \in \mathcal{O}$, and the intended destination $d \in \mathcal{V}$. In particular, $f_{vu}^{tr,d,o,z}$, $f_u^{st,d,o,z}$, $f_u^{pr_i,d,o,z}$ and $f_u^{pr_o,d,o,z}$ indicate the amount of object z carried/stored/processed by edge (v,u) or node u for final product o at destination d, respectively. Note that we differentiate between $f_u^{pr_i,d,o,z}$ and $f_u^{pr_o,d,o,z}$, which denote the input and output flows of the processing unit at node u associated with triplet (d,o,z), respectively. Fig. 3 illustrates the network flows associated with a given triplet (d,o,z) at node u, where pr represents the processing unit that hosts virtual functions at node u and $q_u^{d,o,z}$ is a binary demand parameter that satisfies: $q_u^{d,o,z} = 1$ iff $u = d$, $z = o$, and $\lambda_{d,o,z} > 0$; $q_u^{d,o,z} = 0$ otherwise.
- Global flows: f_{vu}^{tr}, f_u^{st}, and f_u^{pr}, determine the total amount of information flow carried, stored, and processed at a given physical node/link, respectively.

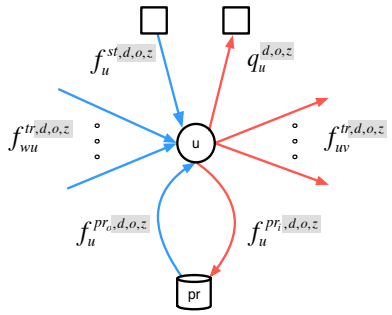


Fig. 3: Generalized flow conservation at node $u \in \mathcal{V}$, where pr represents the processing unit that hosts virtual functions at node u. The incoming flow (in blue) is composed of the incoming transport flows, the storage flow, and the processing flow coming in from the processing unit. The outgoing flow (in red) is composed of the outgoing transport flows, the processing flow leaving node u towards the processing unit, and the demand flow.

A. Objective Function

The objective function is characterized by the global flows and determines the cost associated with the storage, processing, and transport of information over the physical resources of the distributed cloud network.

$$\begin{aligned}\text{minimize} \quad & \sum_{(v,u)\in\mathcal{E}} \left(\sum_{k\in\mathcal{K}_{vu}^{tr}} a_{vu,k}^{tr} \ell_{vu,k}^{tr} + e_{vu}^{tr} f_{vu}^{tr} \right) + \\ & \sum_{u\in\mathcal{V}} \left(\sum_{k\in\mathcal{K}_u^{st}} a_{u,k}^{st} \ell_{u,k}^{st} + e_u^{st} f_u^{st} \right) + \\ & \sum_{u\in\mathcal{V}} \left(\sum_{k\in\mathcal{K}_u^{pr}} a_{u,k}^{pr} \ell_{u,k}^{pr} + e_u^{pr} f_u^{pr} \right). \end{aligned} \tag{1}$$

In (1), $l_{*,k}^*$ are binary resource activation variables that take value 1 if there are k active physical resources of a generic resource at a generic network element, and 0 otherwise; $\mathcal{K}_u^{pr}$ is the set of integers denoting the possible number of active compute servers at node $u \in \mathcal{V}$; $\mathcal{K}_u^{st}$ is the set of integers denoting the possible number of active storage servers at node $u \in \mathcal{V}$; and $\mathcal{K}_{vu}^{tr}$ is the set of integers denoting the possible number of active IP links between v and u.

B. Generalized Flow Conservation Constraints

User-object flows must satisfy demand and flow conservation. By modeling the demand $q_u^{d,o,z}$ as part of the outgoing flow of node $u \in \mathcal{V}$, we can use the following generalized flow conservation constraints:

$$q_u^{d,o,z} + \sum_{w \in \mathcal{N}^+(u)} f_{uw}^{tr,d,o,z} + f_u^{pr_i,d,o,z} = \sum_{v \in \mathcal{N}^-(u)} f_{vu}^{tr,d,o,z} + f_u^{st,d,o,z} + f_u^{pr_o,d,o,z} \quad \forall u,d,o,z. \tag{2}$$

Flow conservation constraints state that the outgoing flow associated with a given triplet (d,o,z) must be equal to the incoming flow for that same triplet, for any node $u \in \mathcal{V}$. As illustrated in Fig. 3, the outgoing flow is composed of the outgoing transport flows, the processing flow leaving node u towards the processing unit, and the demand flow; while the incoming flow is composed of the incoming transport flows, the storage flow, and the processing flow going out of the processing unit. In addition, each processing unit must satisfy the following flow conservation constraint:

$$f_u^{pr_o,d,o,z} \leq f_u^{pr_i,d,o,y} \quad \forall u,d,o,z,y \in \mathcal{Z}(z). \tag{3}$$

This constraint makes sure that in order to have a processed flow z for demand (d,o), a flow associated with each of the input objects required to generate z, $y \in \mathcal{Z}(z)$, for demand (d,o), must be present at the input of the processing unit.

C. Virtual Function Availability Constraints

These constraints allow restricting the set of virtual functions that can be implemented at a given cloud location:

$$f_u^{pr_o,d,o,z} = 0 \quad \forall u,d,o,z,p_z \notin \mathcal{P}_u, \tag{4}$$

where $\mathcal{P}_u \subset \mathcal{P}$ is the set of virtual functions available at node $u \in \mathcal{V}$.

D. Source Constraints

We denote the set of objects that are available in the network as source information and hence input for cloud services as $\mathcal{S} \subset \mathcal{O}$. We define the set $O_u \subset \mathcal{S}$ as the objects permanently hosted by node u as input for cloud services:

$$f_u^{st,d,o,z} = 1 \quad \forall u,d,o,z \in \mathcal{O}_u. \tag{5}$$

In addition, we need to make sure that source objects $\mathcal{S} \subset \mathcal{O}$ are not created in the network:

$$f_u^{pr_o,d,o,z} = 0 \quad \forall u,d,o,z \in \mathcal{S}. \tag{6}$$

Note that objects in $\mathcal{S}$ are allowed to be replicated and stored throughout the cloud network to reduce overall transport costs. Finally, for all other objects $o \in \mathcal{O} \backslash \mathcal{S}$, we need to make sure they are not stored in the network, as they need to be created via their respective virtual functions at the cloud nodes across the network:

$$f_u^{st,d,o,z} = 0 \quad \forall u,d,o,z \notin \mathcal{S}. \tag{7}$$

E. Delay Constraints

We use $\delta_{d,o,z}$ to denote the (local) delay associated with a particular user-object flow (d,o,z), and it is computed as the sum of the delay associated with the transport and processing of (d,o,z) flows, as:

$$\delta_{d,o,z} = \sum_{(v,u) \in \mathcal{E}} f_{vu}^{tr,d,o,z} h_{vu} + \sum_{u \in \mathcal{V}} f^{pr_o,d,o,z} h_u \quad \forall d,o,z. \tag{8}$$

In (8), without loss of generality, we assume that input processing flows have zero delay and capture all processing delay with the output processing flows. We then use $\delta_{d,o,z}^{ag}$ to denote the aggregate delay associated with user-object flow (d,o,z), computed as the sum of the local delay, $\delta_{d,o,z}$, plus the maximum across the aggregate delays of all of its input flows $\delta_{d,o,y}^{ag}, \forall y \in \mathcal{Z}(z)$, as:

$$\delta_{d,o,z}^{ag} = \delta_{d,o,z} \quad \forall d,o,z \in \mathcal{S}, \tag{9}$$

$$\delta_{d,o,z} + \delta_{d,o,y}^{ag} \leq \delta_{d,o,z}^{ag} \quad \forall d,o,z,y \in \mathcal{Z}(z). \tag{10}$$

Finally, the aggregate delay associated with the delivery of final object $o \in \mathcal{O}$ at destination $d \in \mathcal{V}$ is constrained to be no larger than the maximum delay allowed by the SLAs.

$$\delta_{d,o,o}^{ag} \leq H_{d,o} \quad \forall d,o. \tag{11}$$

F. Mixed-cast Constraints

Mixed-cast constraints allow modeling the unicast or multicast nature of storage/compute/transport flows via the corresponding relationship between user-object and global flows. Since a single stored object can be used to satisfy multiple demands, storage flows are said to be multicast. Hence, user-object storage flows for the same object, but for different destinations, are allowed to overlap, as captured by the following constraints:

$$f_u^{st,d,o,z} \leq f_u^{st,z} \quad \forall u,d,o, \tag{12}$$

$$\sum_{z \in \mathcal{O}} f_u^{st,z} B_z = f_u^{st} \quad \forall u. \tag{13}$$

On the other hand, since the use of multicasting cannot be exploited for today's dominant video on-demand services, we generally assume transport and processing flows to be unicast. As such, user-object transport and processing flows cannot overlap and must be added across both objects and destinations, as:

$$\sum_{d \in \mathcal{V}} \sum_{o \in \mathcal{O}} \sum_{z \in \mathcal{O}} f_{vu}^{tr,d,o,z} \lambda_{d,o} B_z = f_{vu}^{tr} \quad \forall (v,u), \tag{14}$$

$$\sum_{d \in \mathcal{V}} \sum_{o \in \mathcal{O}} \sum_{z \in \mathcal{O}} f_u^{pr_o,d,o,z} \lambda_{d,o} B_z \gamma_z = f_u^{pr} \quad \forall u. \tag{15}$$

Note that user-object transport and output processing flows must be sized by both the size of the object B_z and the request rate $\lambda_{d,o}$. In addition, output processing flows are scaled by a factor γ_z that captures the overhead associated with the generation of object z from its input $\mathcal{Z}(z)$ via function p_z.

We remark that in order to capture the use of multicast technologies for the delivery of live streaming applications, for

example, our mixed-cast flow model would require updating (14) and (15) as in (12)-(13).

G. Capacity Constraints

Global flows must satisfy capacity constraints, as:

$$f^{tr}_{vu} \leq \sum_{k \in \mathcal{K}^{tr}_{vu}} c^{tr}_{vu,k} \ell^{tr}_{vu,k} \quad \forall (v,u), \tag{16}$$

$$f^{st}_{u} \leq \sum_{k \in \mathcal{K}^{st}_{u}} c^{st}_{u,k} \ell^{st}_{u,k} \quad \forall u, \tag{17}$$

$$f^{pr}_{u} \leq \sum_{k \in \mathcal{K}^{pr}_{u}} c^{pr}_{u,k} \ell^{pr}_{u,k} \quad \forall u, \tag{18}$$

where the binary activation variables must assure that only one capacity level is activated,

$$\sum_{k \in \mathcal{K}^{tr}_{vu}} \ell^{tr}_{vu,k} \leq 1 \quad \forall (v,u), \tag{19}$$

$$\sum_{k \in \mathcal{K}^{st}_{u}} \ell^{st}_{u,k} \leq 1 \quad \forall u, \tag{20}$$

$$\sum_{k \in \mathcal{K}^{pr}_{u}} \ell^{pr}_{u,k} \leq 1 \quad \forall u. \tag{21}$$

H. Integer/Fractional Constraints

While resource activation variables are always set to be binary,

$$\ell^{tr}_{vu,k}, \ell^{st}_{u,k}, \ell^{pr}_{u,k} \in \{0,1\} \quad \forall (v,u), u, k, \tag{22}$$

user-object flows can either be binary or fractional,

$$f^{tr,d,o,z}_{vu}, f^{st,d,o,z}_{u}, f^{pr_i,d,o,z}_{u}, f^{pr_o,d,o,z}_{u} \in \{0,1\} \text{ or } \in [0,1] \quad \forall (v,u), u, d, o, z, \tag{23}$$

as discussed in the following subsection.

I. Solution and Complexity

The cloud service distribution problem is formulated as a mixed integer linear program (MILP) with objective function (1) and constraints (2)-(23). While the complexity of the CSDP increases exponentially with the number of integer (binary) variables, this number can be significantly reduced based on the following conditions. On one side, as shown in [5], binary flow variables in mixed-cast flow problems can be relaxed if: a) all flows are unicast (e.g., unicast services with fixed storage/sources), b) the network topology is a tree, or c) the network can perform intra-session network coding (e.g., random linear coding). On the other side, as used in lower-bounded facility location problems [9], resource consolidation can be captured by lower-bounding the load on each resource, reducing the number of binary activation variables. Note that this indicates the existence of regimes of practical interest in which the CSDP admits solutions of reduced and even polynomial-time complexity. However, in this first CSDP paper, we focus on presenting preliminary results for the general MILP that also shows how powerful linear programming solvers can be used to provide efficient solutions to integer versions of the problem.

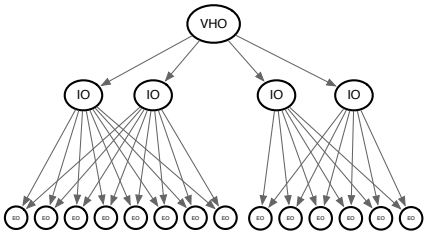


Fig. 4: A 19-node metropolitan area network.

TABLE I: Capacities and Activation Costs

	Capacity per link/server	Activation cost
Transport	100 Mbps	k
Storage	1 GB	$10\log(\mathrm{k}+1)$
Processing	100 Mbps	$a^{pr}\log(\mathrm{k}+1)$

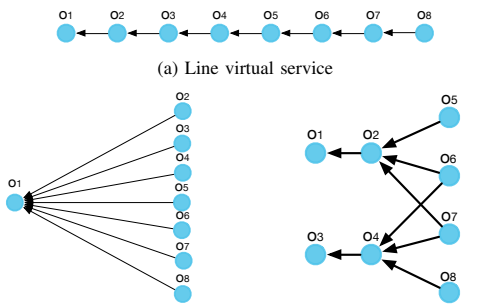
We also remark that by appropriately setting the demand and cost parameters, the proposed model can be used for both i) the dimensioning of the physical infrastructure, based on estimated demands over long time periods, as well as ii) the dimensioning of a particular vCSN, based on shorter time-scale demands for the particular service.

IV. Numerical Results

In this section, we present results obtained from solving the cloud service distribution problem in a sample of illustrative scenarios via the linear programming solver Xpress-MP [11]. We first consider a subset of a real metropolitan area network containing 19 nodes: 1 Video Head Office (VHO), 4 Intermediate Offices (IOs) and 14 End Offices (EOs), as shown in Fig. 4. The EOs aggregate end user requests and represent the destination nodes in our model.

We assume homogeneous transport/storage/processing capacities and costs across the network, following the values in Table I, where k indicates the number of active physical links, compute and storage servers, respectively. For simplicity, we neglect load-dependent costs (typically dominated by activation costs) and normalize all activation costs to the transport cost. We then use a^{pr} as a parameter to analyze the effect of increasing compute costs at cloud locations. In terms of capacity, we assume 100 Mbps IP ports, 100 Mbps compute servers, and 1 GB storage servers. The transport activation cost is assumed proportional to the number of active links/ports, while the storage and processing costs follow a non-decreasing concave function with the number of active physical resources in order to capture the cost benefits of resource consolidation at a particular cloud location. Finally, we assume homogeneous objects' size and processing overhead, with $B_o = 1$GB and $\gamma_o = 1, \forall o \in \mathcal{O}$.

We remark that for this initial set of results, the demand and cost parameters are set to capture an initial sample of scenarios that illustrate the trade-offs between the different cost components. As pointed out in the previous section, our model allows setting up the cost parameters according to the particular scenario of interest, and will depend on the actual objective (physical infrastructure dimensioning, vCSN slice

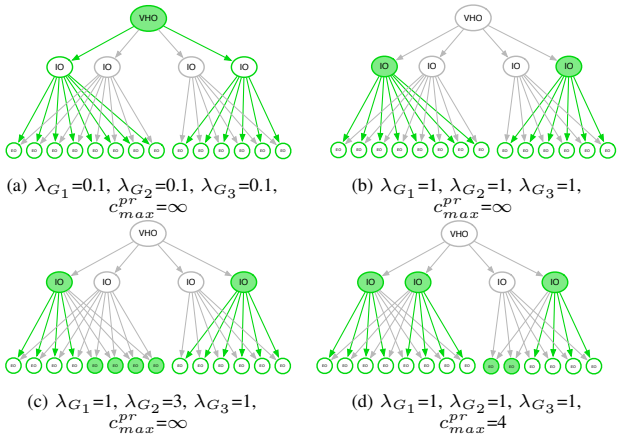


(a) Line virtual service

(b) Star virtual service

(c) Multi-Tree virtual service

Fig. 5: The service graph of a set of illustrative cloud services.



(a) λ_{G_1}=0.1, λ_{G_2}=0.1, λ_{G_3}=0.1, c^{pr}_{max}=∞

(b) λ_{G_1}=1, λ_{G_2}=1, λ_{G_3}=1, c^{pr}_{max}=∞

(c) λ_{G_1}=1, λ_{G_2}=3, λ_{G_3}=1, c^{pr}_{max}=∞

(d) λ_{G_1}=1, λ_{G_2}=1, λ_{G_3}=1, c^{pr}_{max}=4

Fig. 6: vCSN solutions in a 19-node metro network.

dimensioning), business model (pay-per-capacity, pay-per-use, etc), resource specifications, and service requirements.

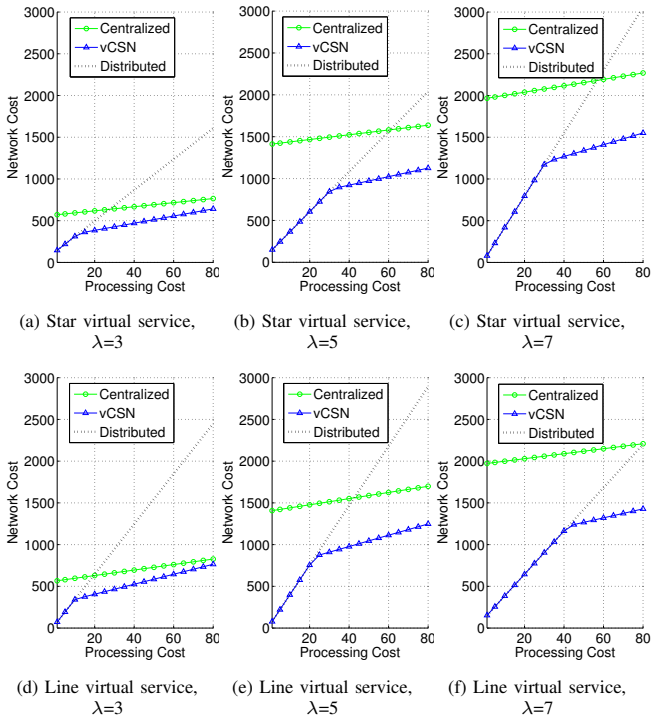
We consider an illustrative set of cloud services, represented by a star, a line, and a multi-tree service graph, as shown ing Fig. 5. Recall that a service graph is a rooted graph, in which the leave nodes represent the input information objects for the cloud service,while the root node represents the final information object that must be delivered to the end user. The line and star graphs are especially important structures, as any cloud service can be represented via a combination of line and star graphs. An example of a line service is a virtual network service (e.g., vEPC) typically defined by a chain of virtual network functions [3]. On the other hand, an interesting example of a star service is an augmented reality service, in which the video stream delivered to the end user results from the combination of multiple source video streams [10].

We analyze the following three solution approaches:

- vCSN: optimal virtual cloud service network solution resulting from solving the proposed ILP when all nodes (VHO, IOs, and EOs) are considered cloud nodes, and hence allowed to host content and virtual functions as needed.
- Centralized: solution obtained by placing all content and virtual functions at the VHO.
- Distributed: solution obtained by placing all content and virtual functions at each EO.

Fig. 6 illustrates the vCSN solution obtained by solving the cloud service distribution problem for the star virtual service in Fig. 5b. The 14 EOs have been partitioned into 3 groups (G_1=1-4, G_2=5-8 and G_3=9-14). We then use λ_{G1}, λ_{G2}, and λ_{G3} to denote the request rate aggregated at the EOs of each group. Observe in Fig. 6a that when demands are low, as expected, all the content and virtual functions are placed at the VHO, reducing compute and storage costs via resource consolidation and reduced content replication. Fig. 6b shows the solution when all demands are increased from $\lambda = 0.1$ to $\lambda = 1$. In this case, cloud resources are consolidated at the IOs in order to reduce the transport costs from the VHO to the IOs. In Fig. 6c the demand of G_2 is increased to $\lambda_{G3} = 3$. As a result, the optimal solution allocates both source content and virtual functions directly at the EOs. Finally, in Fig. 6d, we show the optimal solution when the processing/compute capacity at each cloud location is limited to 40 servers (4 Gbps). Due to this limitation, the demands of G_1 and G_2 have to be satisfied by different IOs. Interestingly, note how in order to satisfy the demand of G_3, the network does not use the respective upstream IO, as in this case the resource consolidation savings of 2 EOs is not sufficient to compensate the savings in transport costs obtained by distributing the content and virtual function to the EOs.

In Fig. 7 we compare the network cost of the line (Fig. 5a) and star (5b) virtual services, for different demand rates and processing costs. First, let's focus on the fully centralized and fully distributed solutions. As expected, the fully distributed approach is more efficient when the processing cost is low compared to the transport cost. As the processing cost increases, the cost savings due the consolidation of compute and storage resources at one cloud location becomes larger than the additional transport cost required to deliver the final information object to the end users, making a fully centralized approach more efficient. Observe that the processing cost for which a centralized approach becomes more efficient than a fully distributed approach differs for the star and line services, being higher for a star service. This can be explained by the fact that a star service requires one virtual function that combines all source information flows into a final flow for the end user, while a line service requires a number of virtual functions (in this case 7) in order to generate the final information flow for the end user. This results in a stronger impact of the resource consolidation savings for the line service, as shown in Fig. 7. Next, observe how the optimal vCSN solution adapts to the network conditions (resource costs) and service requirements (service type and demand rates) in order to minimize the total network cost. If the processing cost is low, the optimal vCSN results in a fully distributed approach. As the processing cost becomes more relevant, virtual functions start moving higher in the network hierarchy to exploit resource consolidation savings, converging into a fully centralized solution for high enough processing cost, as shown in Fig. 7. In the scenario of Fig. 7, for a processing cost equal to the storage cost (a^{pr}=10), and demand rates $\lambda = 3, 5$, and 7, an optimized vCSN solution reduces the



(a) Star virtual service, λ=3 (b) Star virtual service, λ=5 (c) Star virtual service, λ=7

(d) Line virtual service, λ=3 (e) Line virtual service, λ=5 (f) Line virtual service, λ=7

Fig. 7: Total network cost for the delivery of line and star cloud services over the metro network in Fig. 4.

overall network cost compared to a fully centralized approach by a factor of 2.4, 3.6 and 4.8, respectively, for the line service, and by a factor of 2.6, 4 and 5.2 for the star service.

We now present results for the entire metro network under consideration, composed of 120 nodes, including 1 VHO, 20 IOs, and 99 EOs. Int his case, we consider the cloud service in Fig. 5c. This service graph may represent, for example, a video streaming service in which five source video streams $(o_5, \ldots, o_8)$ can be combined to produce two customized or augmented video streams (o_2 and o_4), plus two more video streams resulting from adding advertisement to o_2 and o_4, resulting in o_1 and o_2. We assume demand rates are homogeneous across EOs and are driven by the object popularities $p_{o_1} = 0.2$, $p_{o_2} = 0.4$, $p_{o_3} = 0.1$, and $p_{o_4} = 0.3$. Demand rates are hence given by $\lambda_{o_i} = \lambda p_{o_i}$ for $i \in \{1, 2, 3, 4\}$. In Fig. 8, we plot the overall network cost reduction obtained via an optimized vCSN compared to a fully centralized approach as a function of the request rate λ for different processing cost values. As expected, the cost reduction obtained via vCSN is specially relevant at high request rates and low processing costs. In particular, for a demand rate $\lambda = 1$, vCSN reduces the network cost by a factor of 1.4 for processing cost $a^{pr} = 1$, and by a factor of 2.6 when $a^{pr} = 20$. When the demand rate increases to $\lambda = 5$, the cost reduction factor goes up to 11.5 for processing cost $a^{pr} = 20$.

V. Conclusions

In this paper, we introduce the cloud service distribution problem, where the goal is to find the placement of both content and virtual network functions over a distributed cloud network. We formulate the CSDP as minimum cost flow problem and provide an efficient linear programing formulation that includes and generalizes the content placement and the virtual function placement problems. The virtual cloud service network (vCSN) resulting from the solution to the CSDP jointly optimizes the use of compute, storage and transport resources in arbitrary cloud network topologies, is able to capture arbitrary cloud services via an efficient *service graph* representation, and takes into account resource consolidation savings, unicast and multicast delivery, capacity, and latency constraints. While the generality of the proposed ILP allows capturing a comprehensive set of practical constraints, we also provide promising directions for polynomial-time approximations to the CSDP that are of current investigation.

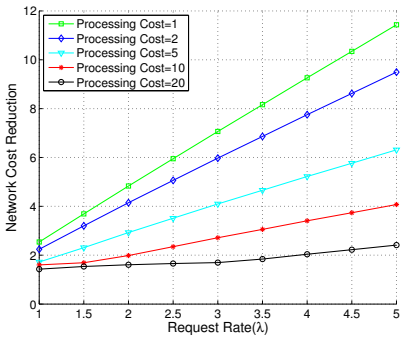


Fig. 8: Network cost reduction of vCSN compared to a fully centralized approach for the delivery of cloud service in Fig. 5c over a 120-node metro network.

Acknowledgement

The authors would like to thank Abdol Saleh and Ben Tang for the very helpful discussions, and the Spanish Ministry of Education for partially supporting this work under FPU grant AP2010-1911.

References

[1] Bell Labs Strategic White Paper, "Metro Network Traffic Growth: An Architecture Impact Study," December 2013.
[2] Marcus Weldon, "Defining the Future of Networks," CommsDay Summit 2014, April 2014.
[3] Alcatel-Lucent Strategic White Paper, "The Programmable Cloud Network - A Primer on SDN and NFV," June 2013.
[4] I.D. Baev, R. Rajaraman, C. Swamy, "Approximation algorithms for data placement problems," *SIAM J. Comput* vol. 38, pp. 1411–1429, 2008.
[5] J. Llorca, A.M. Tulino, "The content distribution problem and its complexity classification," Alcatel-Lucent technical report, 2013.
[6] J. Llorca, C. Sterle, A.M. Tulino, N. Choi, A. Sforza "Joint Contnet-Resource Allocation in Software Defined Virtual CDNs," Alcatel-Lucent technical report, 2014.
[7] J. Llorca, K. Guan, G. Atkinson, D. C. Kilper, "Energy efficient delivery of immersive video centric services," IEEE INFOCOM, March 2012.
[8] J. Llorca, K. Guan, G. Atkinson, D. C. Kilper, "Energy benefit of distributed in-network processing for personalized media service delivery," IEEE ICC, June 2012.
[9] S. Ahmadian, C. Swamy, "Improved Approximation Guarantees for Lower-Bounded Facility Location," CS arXiV, September 2012.
[10] Biocca, Frank, and Mark R. Levy, eds. Communication in the age of virtual reality. Routledge, 1995.
[11] C. Guret, C. Prins, M. Sevaux, "Applications of Optimization with Xpress-MP", 2002.