FedRR – A Federated Resource Reservation Algorithm for Multimedia Services

Jeroen Famaey, Steven Latré, Tim Wauters, and Filip De Turck

Department of Information Technology
Ghent University – IBBT
Gaston Crommenlaan 8/201, B-9050 Gent, Belgium
Email: jeroen.famaey@intec.ugent.be

***Abstract*—The Internet is rapidly evolving towards a multimedia service delivery platform. However, existing Internet-based content delivery approaches have several disadvantages, such as the lack of Quality of Service (QoS) guarantees. Future Internet research has presented several promising ideas to solve the issues related to the current Internet, such as federations across network domains and end-to-end QoS reservations. This paper presents an architecture for the delivery of multimedia content across the Internet, based on these novel principles. It facilitates the collaboration between the stakeholders involved in the content delivery process, allowing them to set up loosely-coupled federations. More specifically, the Federated Resource Reservation (FedRR) algorithm is proposed. It identifies suitable federation partners, selects end-to-end paths between content providers and their customers, and optimally configures intermediary network and infrastructure resources in order to satisfy the requested QoS requirements and minimize delivery costs.**

I. Introduction

Communication networks have been widely adopted for the delivery and consumption of multimedia content. Currently, content is often offered in one of two ways, either directly by the Internet Access Provider (IAP) or across the Internet by an Over-the-top (OTT) content provider. However, both approaches have their distinct disadvantages.

Current telecommunication operators often act as both IAP and content provider for their customers. In addition to providing Internet access, they offer a range of multimedia services, including IP-TV, time-shifted television, video on demand (VoD) and Voice-over-IP (VoIP). As the operator has full control over its own network infrastructure, the Quality of Service (QoS) of the delivered content can be guaranteed. However, the operator needs to compose its own content catalogue, acquiring licenses from a wide range of content distributors. Moreover, it can only serve this content to the relatively small customer-base directly connected to its own network. Therefore, the operator will have to choose between offering only recent and highly popular content or providing an extensive catalogue that caters to all tastes, resulting in a trade-off between licensing costs and customer satisfaction.

The increasing penetration of broadband Internet and the decreasing bandwidth costs for end-users have given rise to a growing number of OTT content providers. They offer their content to users across the world, directly over the Internet. In contrast to telecommunication operators, OTT providers are not limited to a single access network and can potentially reach a huge number of customers. Nevertheless, this approach also has its disadvantages, both for customers and IAPs. As content is delivered over the current best-effort Internet, it is nearly impossible for them to provide QoS guarantees. Additionally, multimedia content is known for having high bandwidth demands. This causes the IAP networks to become more heavily loaded, while they do not share in the profit.

Future Internet research has given rise to the idea of loosely coupled federations [1]. In a federation, several network domains cooperate in order to deliver end-to-end services across the Internet. This paper presents a novel content delivery approach, based on the principle of federations. It aims to solve the aforementioned problems associated with current multimedia content delivery mechanisms. This is facilitated by the Federated Resource Reservation (FedRR) algorithm. It selects a suitable end-to-end route between the content provider and its customers and reserves network and infrastructure resources in the domains along this route. Its goal is to minimize the delivery costs for the content provider, while satisfying the QoS requirements of the customers.

More specifically, the FedRR algorithm is used by the content provider during the negotiation process for the delivery of content with its customers. As input it takes the customer QoS demands (e.g., packet loss, delay and throughput constraints). As output it returns the minimum-cost end-to-end route between the content provider and every customer, consisting of a set of core and edge Internet domains. Additionally, FedRR determines the resources that should be reserved within each of these domains (e.g., a link with a specific bandwidth or server resources for hosting a cache) in order to satisfy the QoS requirements. The algorithm's output can be used by the content provider to unambiguously determine the cost associated with delivering the content. In turn, this cost information can be used by the content provider in the price negotation process to calculate expected revenues.

The current Internet does not support any type of collaboration between the core and edge network domains. However, such federations are expected to become possible in the Future

978-1-4673-0269-2/12/$31.00

Internet [2]. In such a scenario, core Internet domains can take part in the end-to-end content delivery federations, allowing resource reservations to be made on the delivery path and thus improving QoS satisfaction. This allows core domains to share in the revenues in return for offering QoS guarantees. The presented algorithm supports this Future Internet scenario and is capable of making QoS reservations within the Internet core along the delivery paths to satisfy the QoS requested by the IAPs.

The remainder of this paper is structured as follows. Section II describes related work in the fields of federated end-to-end service delivery and QoS negotiations. Section III details the envisioned federated content delivery architecture. The FedRR algorithm, which sets up federated content delivery paths between content providers, cloud providers, IAPs and Internet core domains, is presented in Section IV. Section V thoroughly evaluates the algorithm. Finally, the paper is concluded in Section VI.

II. Related Work

Providing end-to-end QoS guarantees for the delivery of services across the Internet, has been an important topic of study for several years. More recently, interest has shifted from intra-domain QoS-based service provisioning towards inter-domain QoS-aware federations. The latter focuses on the collaboration across network domains to achieve end-to-end QoS guarantees. So far, work has mostly focused on the negotiation of Service Level Agreements (SLAs) and QoS between federation partners.

Several evolutionary approaches have been proposed to support end-to-end QoS on top of the current best-effort Internet. Kumar *et al.* presented the Alliance network model [3]. It allows interconnected Autonomous Systems (AS) to form an alliance, which enables optimal inter-domain path selection and QoS guarantees. Additionally, it is compatible with the Border Gateway Protocol (BGP) and can thus coexist with the current best-effort Internet. More recently, Xiangjiang *et al.* presented a similar approach, also based on BGP [4]. A more revolutionary mechanisms for end-to-end QoS negotiation and path selection was presented by Pouyllau *et al.* [2], [5]. They formulate the problem of finding a QoS satisfying end-to-end path, as a Service Level Specification (SLS) composition problem. It consists of finding a path across network domains, for which the corresponding SLS chain satisfies the requested delay, packet loss and bandwidth. They propose a game theory solution for the negotiations between the domains on the selected path [2]. Recently, a reinforcement learning algorithm was proposed to solve the previously formulated game [5]. In contrast, our work focuses on identifying the edge domains that should take part in the federation and the amount of QoS that should be reserved in each domain along the path in order to minimize costs. As such, the previously described QoS negotiation and path selection mechanisms are complementary to our work and can even be used together.

Another important aspect of setting up end-to-end federations is the negotiation of SLAs. Yan *et al.* presented a multi-agent approach to negotiate QoS for the provisioning of service compositions [6]. Every agent is responsible for provisioning a single service component within the composition. A coordinating agent makes sure the total offered QoS satisfies the requested amount. Additionally, several frameworks and architectures have been proposed to support SLA negotiation between federation partners. Yuanming *et al.* developed a framework for the negotiation of SLAs between service providers, network operators and content providers [7]. More recently, the SLA-based SERViceable Metacomputing Environment (SERVME) was proposed [8]. It consists of a framework and accompanying SLA model, which guide the SLA negotiation process, match providers based on QoS requirements and perform on-demand resource provisioning.

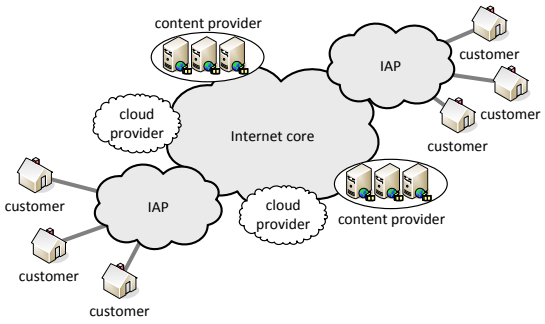

Fig. 1. An overview of the stakeholders involved in the proposed content delivery architecture and their location within the Internet

The negotiation of SLAs, QoS and prices is an important aspect of setting up inter-domain federations. However, before any type of sensible negotiation can be performed, the involved actors need to be able to correctly estimate the costs associated with delivering the requested service at the requested QoS level. Our work focuses on this aspect, and presents an algorithm that allows the stakeholders involved in the delivery of multimedia content to determine the total delivery costs. This, in turn, can be used as initial input for the SLA negotiation process.

III. Federated content delivery architecture

In the envisioned content delivery architecture, the stakeholders involved in the content delivery process can dynamically set up loosely coupled federations. This allows them to guarantee QoS, reduce costs and share revenues, thus combining the advantages of current content delivery approaches.

Figure 1 depicts the stakeholders involved in the envisioned federated content delivery architecture. A potentially large number of *content providers* is spread across the Internet. Their goal is to sell and serve multimedia content to their customers. The *IAPs* offer Internet access and multimedia services, such as Video on Demand, to their customers. However, they do not provide the content for these services themselves, but instead participate in federations with content providers in order to obtain the requested multimedia content. *Cloud providers* offer on-demand infrastructure, platform and software level resources, which dynamically scale with customer needs. The

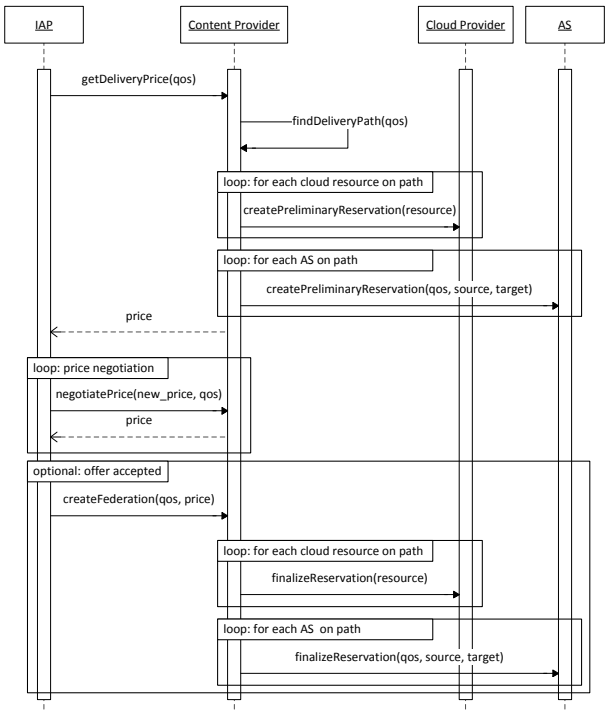

Fig. 2. A sequence diagram showing the process of setting up a federation between the different stakeholders for delivering content between a single content provider and IAP

cloud computing paradigm thus enables content providers to host services, such as proxy caches, throughout the Internet. As such, the considered clouds are assumed to be Platform as a service (PaaS) providers. The Internet core consists of many hierarchically structured *Autonomous Systems* (AS). Their responsibility is to transfer network traffic between edge domains. In the current Internet, the core is merely a best-effort delivery mechanism. Nevertheless, in the Future Internet, federations between edge and core domains are expected to become possible, allowing edge domains to reserve resources along specific paths within the core [2].

The sequence diagram, shown in Figure 2, details the federation set-up process. More specifically, the process consists of the following steps. The IAP initiates the process when one or more of its customers are interested in content offered by the content provider. It contacts the provider and asks a price for delivering content at one or more QoS levels. Such a QoS level description contains a set of constraints for different QoS parameters (e.g., delay, packet loss and bitrate). The content provider executes the FedRR algorithm to find the cheapest delivery path to every IAP for each of the requested QoS levels. The content provider contacts the cloud providers along the selected path to negotiate a preliminary price contract for the requested resources. Additionally, it negotiates a price for the required network QoS levels with core AS domains. The results of the algorithm allow the content provider to calculate the cost per customer request at each QoS level. An initial price, based on this calculated cost and the expected profits, is returned to the IAP. Subsequently, an optional negotiation step takes place, allowing the IAP and content provider to come to an agreement. If they do, the preliminary contracts are finalized and a federation is set up between the domains along the selected path. Otherwise, the IAP must either find another provider that offers the required content or restart the negotiations using another QoS level. The contracts between content providers and IAPs have a finite duration. The procedure to set up a federation should thus be repeated whenever the contract expires. This allows a new price to be negotated, to take into account fluctuations in resource costs.

The remainder of this paper focuses on the algorithm that finds a cost-minimizing delivery path, which makes up an important step of the interactions between the federation stakeholders.

IV. Setting up content delivery paths

The previous section described the process of setting up a federation between the different stakeholders involved in the delivery of multimedia content. An important aspect of this process is finding a suitable delivery path, as it allows the content provider to calculate the total cost associated with delivering its content to an IAP. This cost forms the initial input for the contract negotiation process between the content provider and IAPs. This section introduces the necessary notations and assumptions, gives a formal formulation of the path selection problem and presents FedRR, an algorithm to solve it.

A. Notations & Assumptions

The core Internet consists of a set of Autonomous Systems $\mathcal{A}$, connected to each other in a hierarchical graph structure. They are responsible for routing data and content between the edge domains $\mathcal{D}$. Every edge domain is connected to the Internet through a single AS. A pair of edge domains $(i, j) \in \mathcal{D}$ communicates over a fixed path $p_{i,j} \subseteq \mathcal{A}$. In total, three types of edge domains are involved in the content delivery process: content providers $\mathcal{P}$, Internet Access Providers $\mathcal{S}$ and cloud infrastructure providers $\mathcal{I}$.

In the considered Future Internet scenario, edge domains can make QoS reservations in the Internet core. As such, every AS $a \in \mathcal{A}$ is characterized by a set of QoS cost functions, which model the cost for reserving a specific QoS level. The cost function $C_a^\tau(x)$ returns the cost for sending a single bandwidth unit through AS a with a guaranteed QoS level x of QoS type τ. The total cost for sending a single unit of bandwidth through an AS is defined as the sum of the QoS type cost functions. Every QoS type τ has a minimum τ_a^{min} and maximum τ_a^{max} reservable value within AS a. In this paper, the network QoS types delay and packet loss are considered. Their cost functions are respectively defined as $C_a^\delta(x)$ and $C_a^\pi(x)$. As an example, the total cost for sending a stream with bitrate β through AS a with maximum delay $10ms$ and maximum packet loss 0.01% equals $\left(C_a^\delta(0.01) + C_a^\pi(0.0001)\right) \times \beta$.

Every content provider $p \in \mathcal{P}$ offers a total number of σ_p unique content items, at a bitrate β_p. If a content provider

offers multiple bitrates, it can be modelled as a set of virtual content providers. Consequently, it can be assumed, without loss of generality, that every content provider offers content at only one bitrate. Additionally, a cumulative popularity distribution function $P_p(x)$ is associated with p. This function returns the percentage of content requests for the x most popular content items. Note that although the popularity of individual content items is dynamic over time, the popularity distribution across ranked content items is expected to remain constant [9].

The delivery path selection process is guided by the QoS requested by the IAPs. An IAP $s \in \mathcal{S}$ has a requested delay δ_s, packet loss π_s and bitrate β_s. Additionally, r_s defines the expected average number of simultaneous requests that the IAP s will send for the specified QoS levels. In line with content providers, IAPs that want to negotiate a contract for several QoS levels, can be modelled as multiple virtual IAPs at the same physical location.

Finally, the content provider may decide to deploy proxy caches and bitrate adaptation services throughout the delivery tree. Both services can be deployed in the cloud provider domains. Additionally, proxy caches can also be hosted by the IAPs and bitrate adaptation can be performed by the content provider itself. As such, every edge domain $d \in \mathcal{D}$ has an associated bitrate adaptation cost γ_d^b and caching cost γ_d^c. The bitrate adaptation cost represents the price for changing the bitrate of a single content stream at a specific edge domain, while the caching cost defines the price for storing a single content item in the local proxy cache.

B. Problem formulation

The goal of the presented algorithm is to find the end-to-end path from the content provider to every IAP that minimizes the total cost and satisfies the requested QoS contract. If the offered and requested bitrates differ, a bitrate adaptation service must be deployed along this path. Additionally, the content provider may decide to deploy proxy caches at the domains along the path. Such proxy caches are capable of reducing bandwidth consumption and thus costs by locally storing popular content. It can be easily shown that sharing a proxy cache between multiple IAPs performs more efficiently than disjoint proxy caches of the same combined size. Therefore, instead of finding disjoint paths, the algorithm finds a content delivery tree, where parts of the end-to-end paths overlap. This allows the content provider to deploy shared caches in the overlapping domains, and thus further reduce costs. Note that the proxy cache in a domain can only be shared among IAPs if the bitrate of their content streams is the same within that domain.

Formally, for a content provider $p \in \mathcal{P}$ and the set of IAPs $\mathcal{S}$, the algorithm finds the lowest-cost delivery tree t. This tree contains as its root p and as leaves the IAPs in $\mathcal{S}$. Additionally, it contains zero or more intermediary cloud providers $\mathcal{I}_t$. As such, t consists of the set of edge domains $\mathcal{D}_t = \{p\} \cup \mathcal{S} \cup \mathcal{I}_t$ and edges $\mathcal{E}_t$. The edge $e_{i,j} \in \mathcal{E}_t$ connects the edge domains i and j of t. Every edge e of t is associated with a path p_e of ASes through the Internet core. The set of edges $\mathcal{E}_{t,s} \subseteq \mathcal{E}_t$ represents the path within the tree between the content provider p and IAP s.

The algorithm must decide the amount of QoS that should be reserved in the ASes of every edge $e \in \mathcal{E}_t$. The requested delay in AS a of edge e is defined as $\Delta_{e,a}$, while packet loss is defined as $\Pi_{e,a}$. For every IAP s, the total delay and packet loss along the path cannot exceed the requested values:

$$\forall s \in \mathcal{S}: \quad \sum_{e \in \mathcal{E}_{t,s}} \sum_{a \in p_e} \Delta_{e,a} \leq \delta_s \tag{1}$$

$$\forall s \in \mathcal{S}: \quad \sum_{e \in \mathcal{E}_{t,s}} \sum_{a \in p_e} \Pi_{e,a} \leq \pi_s \tag{2}$$

Moreover, the reserved values cannot exceed the minimum and maximum reservable QoS values within the associated AS:

$$\forall e \in \mathcal{E}_t, \forall a \in p_e: \quad \delta_a^{min} \leq \Delta_{e,a} \leq \delta_a^{max} \tag{3}$$

$$\forall e \in \mathcal{E}_t, \forall a \in p_e: \quad \pi_a^{min} \leq \Pi_{e,a} \leq \pi_a^{max} \tag{4}$$

Additionally, the algorithm must deploy bitrate adaptation services along the delivery paths to assure that the delivered bitrate equals the requested bitrate. For an IAP s, the bitrate adaptation service is deployed within edge domain $d_s^b \in \mathcal{I}_t \cup \{p\}$. The bitrate $\beta_{e,s}$ on edge $e \in \mathcal{E}_{t,s}$ for streams delivered to s equals β_p if e comes before d_s^b and β_s otherwise.

Subsequently, the size of all proxy caches within the tree t is determined. As a cache in a domain $d \in \mathcal{I}_t \cup \mathcal{S}$ is shared among all streams with the same bitrate, only one cache size $c_{d,\beta}$ per bitrate β within each domain d must be determined. The set $\mathcal{B}_d$ contains all the bitrates that pass through domain d. The size of a cache should not be larger than the total number of content items:

$$\forall d \in \mathcal{I}_t \cup \mathcal{S}, \forall \beta \in \mathcal{B}_d: \quad 0 \leq c_{d,\beta} \leq \sigma_p \tag{5}$$

As stated, the goal is to find the tree t, which minimizes the total delivery cost. This cost consists of three components: caching cost, bitrate adaptation cost and transmission cost. The total cost, is the sum of these three. The caching cost equals the resource cost to store the relevant content items in memory or on disk:

$$\sum_{d \in \mathcal{I}_t \cup \mathcal{S}} \sum_{\beta \in \mathcal{B}_d} c_{d,\beta} \times \gamma_d^c \tag{6}$$

The transmission and bitrate adaptation costs depend on the total number of content streams, which in turn depends on the aggregated cache sizes on the paths towards the IAPs in the tree. The aggregated cache size $c_{d,\beta}^{aggr}$ of a domain d for bitrate β is defined as the total number of content items that can be cached on the paths in the tree t from domain d towards all IAPs whose content bitrate is β on the incoming edge of d. The aggregated cache size, can be defined recursively. If the domain d is an IAP s, then it is defined as follows:

$$c_{s,\beta}^{\text{aggr}} = \begin{cases} c_{s,\beta_s} & : \beta = \beta_s \\ 0 & : \beta \neq \beta_s \end{cases} \tag{7}$$

Otherwise, it is recursively calculated based on the aggregated cache size of the child domains of d in the tree t:

$$c^{\text{aggr}}_{d,\beta} = c_{d,\beta} + \frac{\sum_{s \in \mathcal{S}_{d,\beta}} c^{\text{aggr}}_{\text{succ}(d,s),\beta_{e^{\text{src}}_d,s}} \times r_s}{\sum_{s \in \mathcal{S}_{d,\beta}} r_s} \tag{8}$$

with $\mathcal{S}_{d,\beta}$ the set of IAPs whose streams have bitrate β on the incoming edge of domain d in t. The variable $\text{succ}\,(d,s)$ represents the successor of d on the delivery path towards IAP s, while e^{src}_d is defined as the edge of tree t that has domain d as its source.

The transmission cost equals the cost for sending content to the IAPs under the reserved QoS values. Using the previously described definition of aggregated cache size, it can be calculated as follows:

$$\begin{aligned} &\sum_{d \in \mathcal{I}_t \cup \mathcal{S}} \sum_{\beta \in \mathcal{B}_d} \sum_{s \in \mathcal{S}_{d,\beta}} \sum_{a \in p_{e^{\text{dst}}_d}} \left(1 - P_p\left(c^{\text{aggr}}_{d,\beta}\right)\right) \\ &\times \beta \times r_s \times \left(C^{\delta}_a\left(\Delta_{e^{\text{dst}}_d,a}\right) + C^{\pi}_a\left(\Pi_{e^{\text{dst}}_d,a}\right)\right) \end{aligned} \tag{9}$$

with e^{dst}_d the edge of tree t that has domain d as its destination.

Finally, the bitrate adaptation cost is defined as the total resource cost for adapting the bitrate of all content streams to the requested bitrate. As the service uses a specific amount of resources per stream it needs to adapt, the total cost depends on the number of content streams on the outgoing edge of domain d on which the service is deployed. It is calculated as follows:

$$\sum_{s \in \mathcal{S}} \left(1 - P_p\left(c^{\text{aggr}}_{\text{succ}(d^b_s,s),\beta_s}\right)\right) \times r_s \times \gamma^b_{d^b_s} \tag{10}$$

C. FedRR algorithm

This section presents FedRR, a novel heuristic for solving the previously formulated optimization problem. The problem consists of a set of decision variables (i.e., $\Delta_{e,a}$, $\Pi_{e,a}$, d^b_s and $c_{d,\beta}$), constraints (i.e., Equations 1, 2, 3, 4 and 5) and an objective function (i.e., the sum of Equations 6, 9 and 10). The goal of the algorithm is to find the values of the decision variables, that minimize the objective function, while satisfying the constraints. As Equation 9 contains products of decision variables, the presented model is not linear and the presented optimization problem is thus a Non-Linear Programming (NLP) problem [10]. In general, NLP problems are computationally complex and very difficult to solve [11]. As such, FedRR splits the problem in several sub-problems, which are solved sequentially. The resulting sub-problems are Linear Programming (LP) problems, which can be more easily solved using existing LP solvers. Note that these problems are only linear if the functions C^{δ}_a, C^{π}_a and P_p are linear. However, a wide range of non-linear functions can be approximated by piecewise linear functions, which result in an approximated linear model.

The algorithm solves the presented resource allocation problem in three steps. First, it generates a set of candidate delivery trees. Second, for every tree it reserves suitable QoS along the edges, determining optimal values for the decision variables $\Delta_{e,a}$ and $\Pi_{e,a}$. Third, for every tree, it deploys bitrate adaptation and cache services at suitable locations, determining optimal values for the decision variables d^b_s and $c_{d,\beta}$. The remainder of this section describes these three steps in more detail.

1) End-to-end tree generation: The goal of this step is to select a subset of all possible delivery trees, to be used as input for the rest of the algorithm. The total number of candidate delivery trees grows exponentially with the size of $\mathcal{I}$. In order to limit the computational complexity of the algorithm, only trees with zero or one intermediary cloud providers are considered in this paper. In total, the algorithm thus considers $|\mathcal{I}| + 1$ delivery trees.

2) Network resource allocation: Subsequently, for every candidate delivery tree generated in the first step, the algorithm determines suitable values for the decision variables $\Delta_{e,a}$ and $\Pi_{e,a}$. This problem can be modelled as an LP problem and thus solved using LP solving algorithms, such as the simplex method [12]. The constraints correspond to Equations 1–4. The objective function minimizes the total QoS reservation cost of the tree. However, as the location of caches and bitrate adaptation services has not yet been decided, it cannot take into account bitrates and number of streams on each tree edge. As such, the objective function becomes:

$$\min \sum_{e \in \mathcal{E}_t} \sum_{a \in p_e} C^{\delta}_a\left(\Delta_{e,a}\right) + C^{\pi}_a\left(\Pi_{e,a}\right) \tag{11}$$

3) Infrastructure resource allocation: Once the delay and packet loss reservations for each AS along the tree's edges have been decided, the algorithm determines where to deploy bitrate adaptation and caching services. Additionally, it determines the optimal size of every deployed proxy cache. As the value of the decision variables $\Delta_{e,a}$ and $\Pi_{e,a}$ is fixed at this point, Equation 9 no longer contains any products of decision variables. As such, it becomes a linear function.

This step consists of two parts. First, a set of possible locations for the bitrate adaptation services are selected. Second, for every generated set of locations, the corresponding LP problem is solved. In the end, the combination with the lowest total cost, is selected.

The total number of possible combinations for the deployment of bitrate adaptation services grows exponentially as a function of the size of $\mathcal{I}$ and $\mathcal{S}$. Therefore, we present a heuristic that considers a subset of all deployment schemes. If the bitrate adaptation service of an IAP is deployed near the root of the delivery tree, bandwidth consumption per stream is obviously decreased. On the other hand, if it is deployed near the IAP, there will be more opportunities for cache sharing, thus decreasing the total number of streams. Additionally, IAPs with a lower requested bitrate result in a relatively higher gain when their adaptation service is deployed near the tree's root. Based on these observations, deployment schemes are generated as follows. The first considered scheme deploys the adaptation service for every IAP on the parent of that IAP in the delivery tree. In the next scheme, the location of the service of the IAP with the lowest requested bitrate is moved up one

position towards the tree's source. This is repeated until the service of each IAP has moved up one position. The process then starts again with the IAP with the lowest requested bitrate, until all bitrate adaptation services are located at the tree's root (i.e., the content provider).

Subsequently, for every generated deployment scheme of bitrate adaptation services, the optimal cache sizes are determined. The decision variables $c_{d,\beta}$, which represent cache sizes, are integers, resulting in an Integer Linear Program (ILP). An ILP model is more difficult to solve than LP. As such, the variables are relaxed to continuous values. The final solution is obtained by rounding the values of $c_{d,\beta}$ to the nearest integers. The constraints are described in Equation 5 and as an objective it minimizes the sum of Equations 6, 9 and 10.

V. Results & discussion

The goal of this section is twofold. First, scalability of the FedRR algorithm is evaluated. Second, the algorithm's output is used to determine the feasibility of the proposed novel federated content delivery architecture. The usefulness of intermediary cloud providers and cache sharing are studied. Additionally, the effects of the ratio between the different types of costs associated with end-to-end content delivery are evaluated.

The presented results were obtained from a Java-based implementation of the algorithm. The LP optimization problems were solved using the Java version of CPLEX 12.3[1]. All tests were performed on a computer with one Dual-Core AMD Opteron 2212 processor and 4 GiB RAM memory, running the GNU/Linux Debian 5.0 operating system. Finally, all depicted results are averaged over 30 iterations, with the error bars showing the standard error of the mean.

A. Evaluation setup

The core Internet topology used throughout the evaluations, was generated using the ReaSE topology generator [13]. It consists of 250 ASes, including 45 transit domains and 205 stub domains. The ReaSE parameters P and Δ were left at the default values 0.4 and 0.04. During each iteration, a stub AS was randomly selected for every edge domain, through which it was connected to the rest of the network. The AS path $p_{i,j}$ between every pair of edge domains (i, j) was set to the shortest-hop path. The minimum and maximum reservable delay were set to 0.1ms and 100ms, while the minimum and maximum reservable packet loss were set to 0.01% and 1% for all ASes. A linear function was used to model the QoS cost functions C_a^{δ} and C_a^{π}, with a cost qc to reserve the minimum (i.e., best) QoS value and a cost of 0 to reserve the maximum (i.e., worst) QoS value. The value of parameter qc thus directly influences the transmission cost per stream.

All results are presented from the point of view of a single Video on Demand content provider, which offers 5000 content items at a 15 Mbps bitrate. As a popularity distribution, a linear approximation of the Zipf-mandelbrot distribution, with parameters $\alpha = 0.9$ and $q = 1$, is used. This distribution has been shown to be realistic for modelling the popularity of multimedia content [14]. The IAPs request content with maximum delay 250ms, maximum packet loss 0.1% and a bitrate randomly selected from the set $\{5, 10, 15\}$ Mbps. The parameter ni defines the total number of IAPs, while nc defines the number of clouds in an experiment. The transcoding cost is fixed at 10 units per stream, while the caching cost per content item varies and is defined by the parameter cc.

[1] http://www.ibm.com/software/integration/optimization/cplex-optimizer/

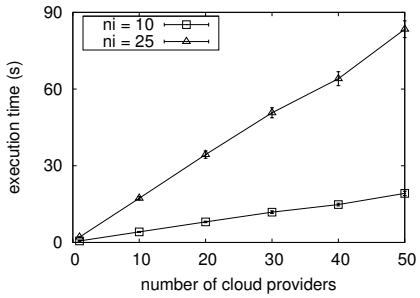

Fig. 3. The execution time of the algorithm as a function of the number of cloud providers ($qc = 5$, $cc = 100$)

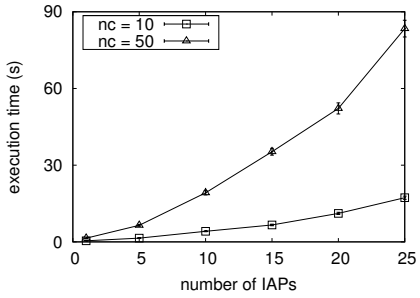

Fig. 4. The execution time of the algorithm as a function of the number of IAPs ($qc = 5$, $cc = 100$)

B. Scalability

The presented algorithm's computational complexity is strongly influenced by the number of considered cloud providers nc and IAPs ni. This section evaluates the effect of these two parameters on the execution time of the algorithm. The parameters qc and cc were fixed at 5 and 100 units respectively, as they do not influence execution time. Figure 3 depicts the execution time of the algorithm as a function of number of cloud providers nc, while Figure 4 plots it as a function of number of IAPs ni.

As shown in Figure 3 the algorithm clearly scales linearly as a function of the number of cloud providers. This is due to the fact that adding additional cloud providers to the problem, does not increase the complexity of the LP problems themselves, but merely the amount of them that need to be solved.

In contrast, as depicted in Figure 4, ni has a higher impact on execution time. Adding additional IAPs to the input increases the complexity of the LP problems. It has

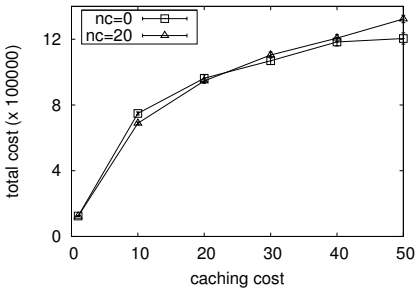

Fig. 5. The total delivery cost as a function of the relative caching cost ($qc = 10$, $ni = 25$)

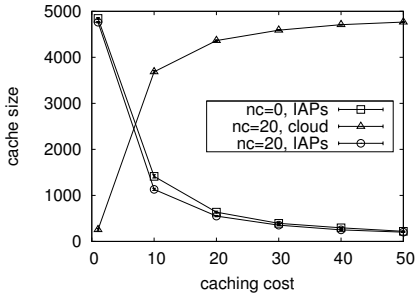

Fig. 6. The average cache size of the intermediary cloud provider and IAPs as a function of the relative caching cost ($qc = 10$, $ni = 25$)

been shown that LP problems can be solved in polynomial time [15]. Therefore, the algorithm does not scale linearly in terms of ni, but still polynomially.

C. Usefulness of intermediary cloud providers

The proposed content delivery architecture allows content providers to set up federations with cloud providers, in order to deploy proxy caches across the Internet. In contrast to caches deployed within the IAP domains, cloud-based caching supports sharing of cache space across IAPs. Additionally, cloud resources are cheap and can dynamically scale in accordance with demand. On the other hand, deploying proxy caches within the access domain is known to be expensive [16]. In addition to these qualitative advantages, this section explores the quantitative advantages of using cloud providers in content delivery federations.

Routing content streams via intermediary cloud providers is expected to increase the transmission costs, of individual streams. However, as they support cache sharing across IAPs, it is also expected the total number of streams will be reduced. This cost trade-off is influenced by the relative ratio between QoS reservation and caching cost. This section explores the usefulness of intermediary cloud providers as a function of the relative caching cost cc. Figures 5 and 6 depict the total delivery cost and average cache size of the optimal delivery tree with an intermediary cloud provider (i.e., $nc = 20$) as compared to without an intermediary cloud provider (i.e., $nc = 0$).

Figure 5 compares the total delivery cost, for a delivery tree with and without an intermediary cloud provider. The figure shows that the solution with intermediary cloud provider actually achieves a lower cost when the cost of caching is relatively low compared to the cost of QoS reservations (i.e., $cc \leq 30$). Once the caching cost becomes relatively large (i.e., $cc \geq 50$), using an intermediary cloud provider becomes too expensive. This behaviour clearly illustrates the trade-off between cost reduction due to cache sharing and the increased QoS reservation cost when using an intermediary cloud provider. When the caching cost is low, the cost reduction due to cache sharing has the upper hand. Once the relative caching cost becomes too high, the gain is offset.

The average cache size of the optimal delivery tree in the intermediary cloud provider and IAPs is shown in Figure 6. As expected, the size of the caches in the IAP domains is inversely proportional to the caching cost. As caching becomes more expensive, QoS reservations become relatively cheaper and the algorithm chooses to reduce IAP caches. In contrast, when an intermediary cloud provider is used, the algorithm does choose to increase the cloud cache size as the caching costs increase. This is due to the fact that an intermediary cloud cache is more cost effective, as it can be shared across many IAPs. Nevertheless, once the relative caching cost surpasses a specific threshold, the optimal cache size is expected to decline again, as the costs of a shared cache start to outweigh the reduction in transmission costs. Finally, the results show that, if the caching cost is relatively low (i.e., $cc \leq 30$), the introduction of an intermediary cache located within the cloud reduces the required cache size in the IAP domains.

In summary, it can be concluded that including intermediary cloud providers in the content delivery tree can reduce costs, as long as the cost of caching is relatively low compared to the transmission cost. Not only was the total delivery cost up to 8% lower in the evaluated scenario, the required size of IAP caches was also reduced up to 20%. In the presented scenario, the deployment of an IAP cache was assumed to have the same cost as a cloud-based proxy cache. However, an IAP cache is known to be expensive [16], while cloud resources are expected to become cheaper in the future. This would further increase the usefulness of the cloud-based approach.

D. Cache sharing

A major advantage of deploying proxy caches inside the cloud, is the opportunity to share caches among several IAPs. This allows the cache to be more effectively used, reducing the required resources, while increasing the cache efficiency. The effects of cache sharing are evaluated, by studying the cost and cache size as a function of a growing number of IAPs included in the content delivery tree. As the amount of IAPs in the tree is increased, the cloud cache can be shared among a larger number of customers, consequently increasing cache efficiency. The results are depicted in Figures 7 and 8, which respectively show the total delivery cost per IAP and the average size of cloud and IAP caches.

Figure 7 shows that the average delivery cost per IAP decreases significantly as the number of IAPs grows. This is

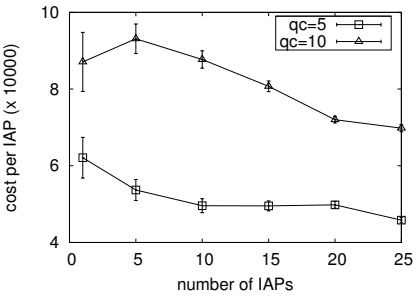

Fig. 7. The cost per IAP as a function of the number of IAPs ($cc = 100$, $nc = 20$)

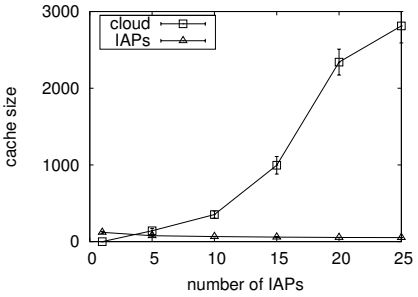

Fig. 8. The average cache size of the intermediary cloud provider and IAPs as a function of the number of IAPs ($qc = 5$, $cc = 100$, $nc = 20$)

especially true if the relative QoS reservation cost becomes larger. Additionally, Figure 8 shows that an increase in number of IAPs induces an increase in the average cloud cache size and a decrease in the average IAP cache size. As such, as the number of served IAP domains grows, it becomes more cost-effective to cache inside the cloud. These observations allow us to conclude that cache sharing results in a significant cost reduction, and becomes more effective as the served customer base grows.

VI. Conclusion

This paper presents a novel architecture for setting up end-to-end federations to deliver QoS-constrained multimedia content. It tackles the problems related to current content delivery approaches, by allowing the different stakeholders involved in the multimedia service delivery process to set up loosely coupled federations. This allows them to collaborate in order to improve QoS and increase profits. Additionally a mathematical model, which describes the problem of identifying suitable federation partners and reserving the relevant network and infrastructure resources, is formally defined. Finally, an algorithm, called FedRR, was introduced to solve the presented model. It employs mathematical optimization techniques to determine the parameter values that minimize the total content delivery cost, while satisfying the requested QoS values.

The merits of the devised federated content delivery architecture were explored in detail, based on simulation results obtained from a prototype implementation. First, FedRR was shown to execute in polynomial time complexity as a function of the model's parameters. Second, the use of cloud providers, in order to deploy intermediary proxy caches along the delivery paths, was investigated. Results showed that a single intermediary cache is capable of reducing the total delivery cost, as well as the required cache size in the access domains. Additionally, the intermediary cache can be shared among many customers, spread across the Internet. This further reduces the delivery cost per customer, as the customer base grows.

Acknowledgment

Jeroen Famaey is funded by the Institute for the Promotion of Innovation by Science and Technology in Flanders (IWT). Steven Latré and Tim Wauters are funded by the Fund for Scientific Research Flanders (FWO). The research leading to these results was partially performed within the context of the FP7 OCEAN project and received funding from the European Union's Seventh Framework Programme ([FP7/2007-2013]) under grant agreement number 248775.

References

[1] M. Serrano, S. van der Meer, V. Holum, J. Murphy, and J. Strassner, "Federation, a matter of autonomic management in the Future Internet," in *12th IEEE/IFIP Network Operations and Management Symposium (NOMS)*, 2010, pp. 845–849.

[2] H. Pouyllau and R. Douville, "End-to-end QoS negotiation in network federations," in *12th IEEE/IFIP Network Operations and Management Symposium – Workshops (NOMS)*, 2010, pp. 173–176.

[3] N. Kumar and G. Saraph, "End-to-end QoS in interdomain routing," in *2nd International Conference on Networking and Services*, 2006.

[4] H. Xiangjiang, Z. Peidong, C. Kaiyu, and G. Zhenghu, "AS alliance in inter-domain routing," in *22nd International Conference on Information Networking and Applications – Workshops (AINAW)*, 2008, pp. 151–156.

[5] H. Pouyllau and G. Carofiglio, "Inter-carrier SLA negotiation using Q-learning," *Telecommunication Systems*, 2011.

[6] J. Yan, R. Kowalczyk, J. Lin, M. B. Chhetri, S. K. Goh, and J. Zhang, "Autonomous service level agreement negotiation for service composition provision," *Future Generation Computer Systems*, vol. 23, pp. 748–759, 2007.

[7] C. Yuanming, W. Wendong, G. Xiangyang, and Q. Xirong, "Initiator-domain-based sla negotiation for inter-domain qos-service provisioning," in *4th International Conference on Networking and Services*, 2008.

[8] P. Rubach and M. Sobolewski, "Dynamic SLA negotiation in autonomic federated environments," in *On the Move to Meaningful Internet Systems*, 2009.

[9] Z. Avramova, S. Wittevrongel, H. Bruneel, and D. De Vleeschauwer, "Analysis and modeling of video popularity evolution in various online video content systems: Power-law versus exponential decay," in *1st International Conference on Evolving Internet*, 2009, pp. 95–100.

[10] M. S. Bazaraa, H. D. Sherali, and C. M. Shetty, *Nonlinear Programming: Theory and Algorithms*. Wiley-Interscience, 2006.

[11] M. Bellare and P. Rogaway, "The complexity of approximating a nonlinear program," *Mathematical Programming*, vol. 69, no. 1, pp. 429–441, 1995.

[12] M. J. Todd, "The many facets of linear programming," *Mathematical Programming*, vol. 91, no. 3, pp. 417–436, 2002.

[13] T. Gamer and M. Scharf, "Realistic simulation environments for ip-based networks," in *1st International Conference on Simulation Tools and Techniques for Communications, Networks and Systems (Simutools)*, 2008.

[14] W. Tang, Y. Fu, L. Cherkasova, and A. Vahdat, "Modeling and generating realistic streaming media server workloads," *Computer Networks*, vol. 51, pp. 336–356, 2007.

[15] N. Karmarkar, "A new polynomial time algorithm for linear programming," *Combinatorica*, vol. 4, no. 4, pp. 373–395, 1984.

[16] T. Monath, M. Kind, T. Heger, M. Schlesinger, and J. Aznar, "Economical analysis of experience-optimized service delivery," in *9th Conference on Telecommunications, Internet and Media Techno Economics (CTTE)*, 2010.