2015 IEEE 7th International Conference on Cloud Computing Technology and Science

Enhancing Real-Time Applications by means of Multi-Tier Cloud Federations

Vamis Xhagjika, Leandro Navarro
Universitat Politècnica de Catalunya
Barcelona, Spain
Email: {xhagjika, leandro}@ac.upc.edu

Vladimir Vlassov
Royal Institute of Technology
Stockholm, Sweden
Email: vladv@kth.se

***Abstract*—The evolution of Cloud Computing beyond the frontier of a single datacenter is justified as a mean to enhance various system architecture aspects like: cost, geo-locality, energy efficiency and structural properties. Orchestration of different cloud providers in order to ensure better Quality of Service (QoS) is refered to as Multi-Cloud. One such multi-cloud federation model is Community Clouds, a federation concept in which various micro-clouds are owned by different communities of individuals contributing cloud resources. Community Clouds provide high locality to services towards the user by bringing the Cloud in the edges of the network, and in some cases as an integral part of the access network. Another such type of multi-cloud federation is Structured Cloud Federation for Internet Service Providers and Telephony Providers, composed of geo-distributed micro-clouds placed on the edges of the network and a centralized datacenter cloud on the core network. Services deployed in the micro-clouds have augmented locality, while services on the datacenter cloud enhanced performance. In this work we identify how these approaches to multi-cloud, through the ability to deploy services close to the final user, can be exploited to enhance existing real-time streaming applications. Furthermore a novel multi-cloud overlay algorithm is introduced that provides both latency and backbone traffic optimization for latency critical existing and novel services (concretely focusing on stream computation, live streaming).**

***Keywords*—cloud computing, live streaming, multi-cloud, structured cloud federation, community clouds, federation overlay**

I. Introduction

The advent and advance of Cloud technology led to a growth in service complexity. Commodity computing, through dynamic resource allocation, enables services to modify their structure at runtime in order to comply with QoS agreements between provider and cloud user. Such software governed growth, in service complexity, encompasses and dictates substantial growth in all the layers of the Cloud stack from Infrastructure as a Service (IaaS) to Platform as a Service (PaaS) and Software as a Service (SaaS).

Service complexity and the need to overcome cloud computing limitations (vendor-lockin, performance, geo-locality, resource availability) [1] [2] [3] has contributed to the move of such services into Cloud Federations or Multi-Clouds. In this work we analyze and provide new mechanism for multi-cloud federations, that we call *Structured Cloud Federation* for Telephony and ISP providers and federations of Community Clouds, that improve existing Live Streaming services through augmented service locality leading to better and more predictable overall application performance. This paper focuses particularly on how such distributed architecture can optimize live streaming applications. Such applications provide a hot topic on present access networks, with examples like WhatssApp [4] (live messaging) reaching millions of active connections, and live video streaming with global consumer Internet video traffic rising to 80 percent of all consumer Internet traffic by 2019 [5].

Cloud federations of highly distributed nature like the one considered in this paper, provide good candidates to cloud enhancements of existing live streaming application techniques. The contributions of this work are as follows.

1) First, we define **a Multi-Tier Cloud Federation Model** that allows to enhance existing Cloud-based application as well as enables new classes of applications, such as latency-sensitive real-time edge services that need to be located in geographic proximity of mobile devices in order to provide required QoS and improve end-user experience with the services.
2) Second, we propose **a novel algorithm for Cloud Federation Construction and Maintenance** that self-organizes the clouds in a minimum latency configuration, from source to destinations, including also a fallback and high scalability mechanism.
3) Finally, we **evaluate and demonstrate the benefits that can be achieved from deploying real-time applications and services on a multi-tier cloud federation**, by considering an enhanced live streaming application as a use-case.

This paper is organized as follows. Section II provides an introduction to the basic concepts needed to understand the various entities of the system model. Section III describes the basic concepts needed to understand and motivate this work by providing necessary state-of-the-art information. In Section IV, we describe an overlay based cloud federation model that codifies the augmented locality approach. Section V presents overlay construction and maintenance algorithms. In Section VI, we describe enhanced applications, that benefit by the use of this architecture. We conclude and discuss future work in Section VII.

II. Stream computing applications

The targeted services for optimization, are general stream computing applications, where a stream of data needs to be processed in a distributed fashion. Live streaming applications, like WhatsApp [4] live messaging, live video with message

978-1-4673-9560-1/15 $31.00 © 2015 IEEE
DOI 10.1109/CloudCom.2015.69

interaction Periscope [6] or live multi-party interactive communications like Skype [7], should provide a service and communication time lower than the human perception time in order to provide a, as close as possible, natural communication medium. The human reaction time can be accounted at the order of 150ms [8], and anything more than that would create highly unnatural interactions. Live Streaming services could be subject to other Service Level Agreement (SLA) related concerns in order to support third party application orchestration, thus not only human reaction time. One such example could be live feedback from the viewers of the live stream, in other words live interaction of the viewers with the content being streamed.

A. System model

The system model is composed of access networks providing highly geo-distributed clouds, backbone and public internet clouds. A high level view of such model is shown in Fig. 1 with all the related components. Each access network provider enables service deployment in any of the distributed clouds and the backbone cloud. The clouds inside the same provider are interconnected with high performance links and the providers have high performance links to the internet clouds or direct high performance links to a limited set of neighbouring providers. In this work we provide a way to build a self-organized structure between such clouds, in order for the clients to fetch the live stream directly from the local clouds, the closest cloud, minimizing both backbone traffic and stream latency. The introduction of a self-organized structure between the clouds eliminates the possibility of central management becoming a central point of load and failure.

A relevant research question in this case is using the architecture to have a distributed multi-cloud algorithm that produces a good enough approximation of the shortest path between the source of the stream and the receivers, and at the same time minimizing payload duplication along the backbone. In developing such algorithm we discovered other properties that benefit the application model of live streaming. Such properties are client churn decoupling, enhanced support for client mobility and on-the-fly scaling of the resources to meet load demand.

B. Cloud Federations

Cloud federations as introduced in [9], [10] define a collection of cooperating Clouds, managed by the same or different administrative authorities, in order to provide a set of common services or applications. The nature of interaction between the clouds defines the federation type, thus differentiating between Cross-Cloud, Cloud Federation (providers cooperate in the federation) and Multi-Cloud (provider has no idea of the federation, based on client middleware) [10]. The rest of the paper will use the term cloud federation, but no limits are imposed on the federated entities, as such the terms can be used interchangeably. In this work we consider highly distributed cloud federations like those described in [11] and [12]. To generalize, a fabric of highly distributed Micro-Clouds are used in conjunction with datacenter grade Internet Clouds in order to provide distributed services. The key idea of such cloud federations is the ability to have augmented service locality through the micro-cloud fabric (ability to deploy services on the edges of the network) and a stable backbone through private or public cloud to provide performance oriented resources. Such federation provides to the services the possibility to trade-off between performance and locality and in some cases the best of both worlds, when services are composed of locality components and performance components. Thus these cloud federations are needed in order to have a highly scalable and highly distributed architecture, with augmented locality. In some cases due to the scale of the distributed environment having a one cloud solution may not even be an option, as the system traffic would greatly influence the client traffic, with which it shares the access network. In such cases self-managed micro-clouds are the only option in order to have a distributed cloud infrastructure.

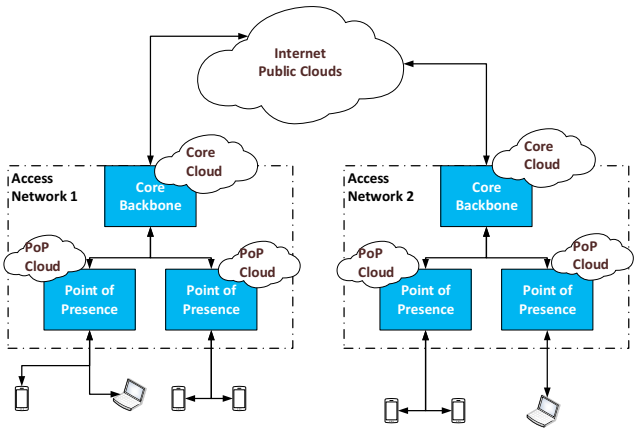

Fig. 1. Extended Cloud Infrastructure and Cloud Federation Model

C. Stream Computing (Live Streaming)

Crucial use cases are enhancements to real-time applications achieved through the use of highly distributed cloud federations as previously described. The applications that will be discussed, namely Live Video Streaming, are to be treated as a case study for such technology and as a small sample of application types that can be enhanced through these architectures and techniques. Live Streaming refers to streaming of live video channels by means of HTTP communication, where not only the quality of the stream is the issue (network bandwidth) but also the latency of the stream. Real life examples of such case are Web TV or IPTV and live event multicasting where the latency profile of the stream should be restrained by a minimum QoS.

D. Overlays

The concept of overlay is that of creating a structure on top of an existing physical interconnection between resources. Overlay structures may be constructed in such a way to encapsulate system properties that enhance applications. Such overlays are common in publish/subscribe systems and in P2P systems where peers are organized through a distributed overlay. The presented system design makes use of an overlay construct in order to build a locality enhanced, cloud enabled, environment in order to optimize service latencies. In Section IV a federation overlay is constructed in order to optimize latency of real time applications such as Live video Streaming.

III. State-of-the-Art Review

This work is based on previous studies in the field of Peer-to-Peer (P2P) live streaming, and enhances these approaches by creating a new hybrid architecture where the clouds build a self-regulated structure to enhance latency. Other preceding systems [13] [14] [15] based on P2P overlays try to minimize mostly bandwidth as in general peers have a limited bandwidth to dedicate to the stream. Differently in case of cloud federations such restrain remains valid but the available bandwidth is in orders of magnitudes higher and does not pose a limiting factor. In general the bandwidth of the backbone is build to match the access network bandwidth for the clients, as such by construct the limiting factor remains latency.

Gradient based approaches to P2P live streaming like Sepidar [13], GradientTv [14] and GLive [15] are used to build a gradient overlay between peers in order to optimize bandwidth and also provide incentive mechanisms in order to deal with free-riders. Such approaches do not include the notion of cloud and also do not minimize the traffic on the communication backbone. By depending only on the existing resources of the peers, no new resources can be allocated to meet load changes. Our approach uses the same overlay type for the clouds in order to build a locality aware federation that can scale to meet load requirements and is not influenced by churn or startup delay. Effectively the cloud federation overlay is not coupled to user churn.

In our architecture cloud resources for different streams can be reused as in AnySee [16], in order to efficiently distribute the load. Cliquestream[17] and Climber[18] try to promote peers to a set of peers and to super peers respectively. This approach is still limited to bandwidth optimization, is unable to scale beyond the resources of the participating peers, and give no guarantees on latency SLAs.

The approach introduced in this work is also based on previous work in high performance Content Delivery Networks (CDNs). The approach described in the workings of Akamai CDNs [19] presents a tree based delivery system with statically pre-allocated nodes. In this work, apart from envisioning dynamically allocated resources, we also organize the servers in a self-organized hierarchy with no central point of failure. By using the overlay, the naming scheme does not need to be centralized as the resources can use distributed discovery in the overlay. In Akamai's case DNS is used as a naming scheme but this introduces a lot of overhead to manage such scheme and also modifies DNS to serve as real-time consensus.

Other approaches [20] [21] [22] try to organize super peers on top of CDN server, but they still focus on a Tree like or DHT base architecture in order to build a overlay of the CDNs, and also deploy only statically pre-allocated resources. Our approach permits to scale-up the system both vertically and horizontally in order to meet clients QoS and lower costs by providing a dynamic cloud federation. The self regulating overlay mechanism provides a more dynamic and locality aware environment that both Tree and DHT based routing.

IV. Federation Model

The federation model developed in this section is a self-organized cross-cloud service overlay, that enables enhanced locality cloud-based services. The cloud federation mechanism embeds in the federation structure the concept of locality. A software level overlay is built and maintained between the clouds in order to provide minimized stream latency. In order to give a clear picture of the proposed solution, we define some requirements that the federation should conform to. Following such federation requirements the remaining subsection of this chapter presents an architecture blue print.

A. Federation and System Model

The system model presented in this section clearly defines the cloud federation nature and system components. A high-level view of the considered federation and system components is shown in Fig. 2. As previously described we assume the presence of Micro-Clouds (PoP Clouds) in the access networks of various providers and also a datacenter grade Core Cloud, in such AS backbone. PoP Clouds are physically interconnected to a small number of physically close micro-clouds and also to the Core Clouds with a high performance link backbone. The Core Clouds posses high-performance links to a small number of neighboring Core Clouds, to all the PoP Clouds of the AS and also to the various Internet Clouds.

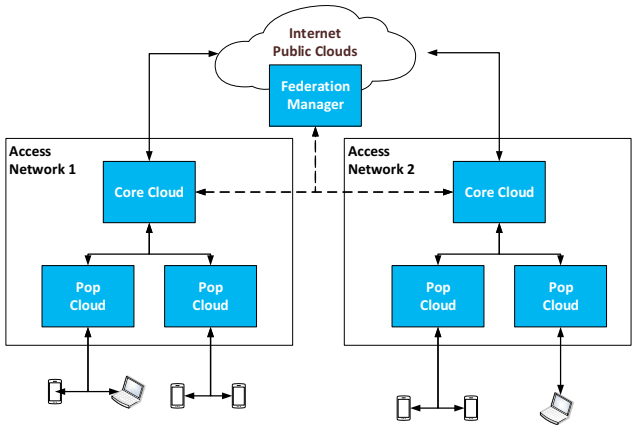

Fig. 2. Layered Federation Model

As for the FM introduced in the following subsection, it is placed in the Public Clouds in order to have ease of visibility to the other components of the proposed federation model. The proposed self-organized algorithm builds a locality aware overlay based on distance from the source PoP Cloud toward all the members of the live streaming multicast. The assumption regarding the source of the stream, is that the source client, uploads the stream to the local PoP Cloud which effectively becomes the source cloud toward which the other clouds try to build a proximity aware overlay.

In this first version of the system design we assume the setup is being used for a single stream, in order to focus more on the overlay construction mechanism and leaving to future work additional concerns such as resource allocation within the clouds and multiple streams. The overlay is build on top of the physical setup shown in Fig. 2, and all the resources inside the clouds use the overlay setup for communication. This simplifications of the system view enable the study of the system as a graph of which the edges are Wheel graphs with the Core Clouds as center and in turn the Core Clouds

and the Internet Clouds form a random interconnected graph (Fig. 5).

B. Federation Components

Enhanced locality is a crucial part of the system design as such will form the base of the system design. The architecture is composed of a geo-distributed decentralized cloud infrastructure built of fully functional cloud deployments. In case of Community Clouds, such micro clouds are provided by the community and provide resources on a cooperative basis or on pay-as-you-go plan. In case of ISP and Telephony clouds, the decentralized infrastructure is managed by one provider or collaborating providers as a paid service.

The underlying cross-cloud network infrastructure should provide a uniform network access to the micro clouds such as public IP addressing or VPN access for cross-cloud communications. The clouds should be accessible to all other clouds and clients of the service. IP is chosen as the best inter-connectivity protocol as it is the de facto standard for internet applications, and if is fairly supported by any live streaming application.

Hence forth the decentralized cloud infrastructure will be referred to as micro-cloud fabric or micro-clouds. Such denomination does not imply anything about the performance of the clouds. The micro-cloud can be anything between a single machine cloud deployment to a highly geo-distributed datacenter. The name implies only that the resources available are quantitatively less then that of a public or private backbone cloud.

In order to enhance this cloud infrastructure with a more stable backbone and the ability to cross connect clouds in different regions a private or public cloud datacenter is introduced in the design. At worst on geo-distributed resource exhaustion, the system falls back to an Internet model by using the private or public cloud, until new local resources are available. A central entity manages resource allocation and hosts the Federation Manager (FM). Such manager is in charge of monitoring and de/allocating resources in the federation in order to scale the services. The FM manager is going to be used solely for resource de/allocation for the clouds taking part in the federation, while the overlay structure implements the discovery algorithm for the clients of the service. By doing so the FM does not become a bottleneck for the system and client resource discovery, distributing such service to the clouds.

A naming scheme or cloud transaction provider is needed in order for dynamic resource discovery in the cloud federations. Such provider could be a P2P gossiping algorithm, where the clouds periodically update resource availability, or on the other hand such service could be provided by a third party resource market place.

In our solution the cloud overlay will use the gossip based protocol for resource discovery or control statements from client toward the service or for cloud-to-cloud service data. As for cloud resource allocation, the centralized FM entity will allocate resources based on maintenance cost. The FM in our case will be the source datacenter cloud or a cloud in the public domain. We choose a gossip based algorithm as it alleviates the load of managing mobility of clients through a central entity and also it provides a good distributed approximation to the optimal shortest path from the source to the clients.

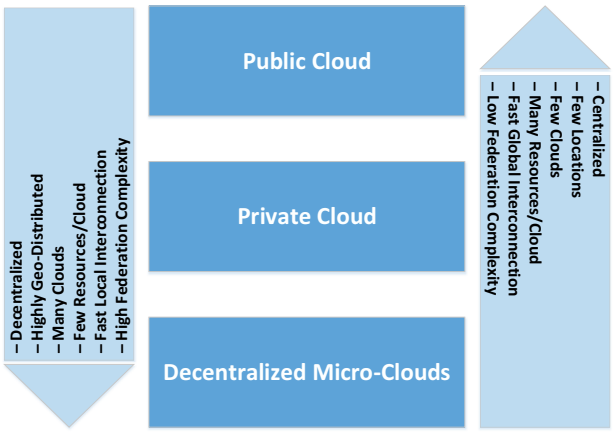

Fig. 3. Layered Federation Model Properties

More properties of such tiered cloud federation are shown in Fig. 3, as we see the more we distribute the clouds the more we gain in locality but loose in resource availability. The federation overhead is higher in the lower tiers, as higher decentralization implies greater complexity in managing the cloud services.

Having discussed the main system and federation components we move on to the description of the federation dynamics as a factor of service implementation.

C. Federation Dynamics

In this work we provide a dynamic federation overlay between the clouds in order to have a loosely coupled federation. The federation model mimics the application restraints in order to provide the best available resource locality for live stream computing. Providing a common interface and a tightly coupled federation between the service and the resources, the application enhancements are encapsulated in the structure of the federation and as such management overhead is minimized.

The discovery service provides the clouds with the available resources that can be scheduled and the federation can allocate. Each application then builds its own federation overlay on top of the federated resources in order to optimize the runtime. As such for the live streaming scenario we developed a hybrid federation where the federation is built as an ”Gradient” variant overlay.

The system as shown in Fig. 3 is composed of three service layers, in a multi-tier federation model. The upper layers of the federation model provide service stability, more centralization, enhanced performance. By moving from the upper layers to the lower layers we make compromises while gaining on important factors. The lower layers of the federation provide enhanced locality for services, and are more geo-distributed and sparse, but provide lower service performance and higher coordination cost. Fig. 3 shows more properties and trade-offs of having the service in a specific layer.

A similar architecture of structured federation architecture is introduced in our previous work [12], where services can move between the various layers of the cloud federation in order to trade between cost, stability and locality. We continue

building on the concepts introduced in that work in order to optimize real time streaming applications.

V. General Architecture

This section provides an architecture of the proposed solution for live streaming by use of a cloud federation overlay. The solution decouples user churn from the architecture, enhancing stream latencies and also decreasing backbone traffic for the different AS-s involved in a stream session.

The cloud architecture is based on a self-regulated "Gradient" structure between the clouds built by using a distance metrics from the source of the stream. In our experiments such distance metric is path length from the source. Fig. 4 shows a possible evolution of the system and the organization of the clouds back-end in the gradient configuration.

We chose a Gradient topology as it provides several benefits in the live streaming scenario. As a first benefit, the distance from the source cloud is cumulatively calculated as the algorithm proceeds and at the same time the gradient is consolidated. On every cloud executing the algorithm in rounds, the best neighbors in terms of distance to source and distance from the node to such neighbor clouds are selected. Each round of the algorithm, creates a better distance approximation and every round neighbors are guaranteed never to give a worst distance metric than the previous round.

Another benefit is that the similarity set of the gradient for each node, provides also a fallback mechanism for the cloud backbones, when the closest cloud fails then the stream can be fetched by the second next closest cloud and so on. A third benefit provided by such approach is that mobility of the clients is taken care of locally to all the entities of the system. The management of the locality is achieved by gossiping to neighbors and updating the proximity view metrics fetching the stream from the similarity view.

A Gradient overlay structures the clouds in layers based on the distance that the proximity function calculates from the source of the stream. In our case the proximity metric can be latency or number of hops, as an approximation of the end-to-end latency parameter. In this work the latency is considered as seen by the end-to-end propagation latency from the micro-clouds, clouds, or clients, and the forwarding latency is included in the end-to-end measurement. This is justified by the fact that the inter-connectivity between the micro-clouds and backbone clouds are optimized by construction of the access network, and as such the end-to-end latency is a more adequate measure.

The clouds from different AS-s which are closer to the source receive directly the stream from the source while the other clouds further down the hierarchy use these clouds as forwarders. By doing so we lower packet duplication that would result from having each cloud fetch the content from the source cloud, or across peers in a P2P system without topology information codified in the system design. Effectively implementing a multi-cast protocol across different AS-s.

The cloud federation back-end is constructed in self-organized tiers, while the clients can use such back-end as a multiple source to receive the stream, and as far as mobility goes in the contest of this application, when the user moves

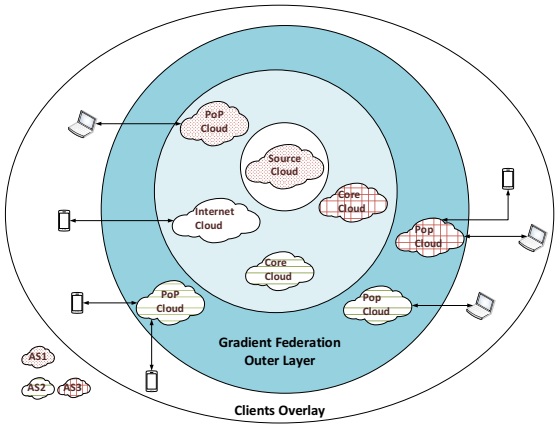

Fig. 4. Layered Federation Model Instance

the metrics of proximity get updated and the user fetches the stream from the newly discovered close-by cloud. In order to allocate the resources needed between the different clouds, as mentioned previously, a centralized FM is used. Such federation manager will be introduced in the remainder of this section, and is responsible for scaling the application by allocating/releasing resources in a dynamic fashion. This manager is necessary for accounting reasons, but also as a cost allocation entity for the federation. The manager is not concerned with functional concerns of the actual streaming.

A. Software Architecture

The key component of the system design is the Federation Manager. Such component de/allocates cross-cloud resources in order to scale the system to meet client demand. Accounting for costs and resources is attributed to the FM. If such task would be distributed, the decision making would over complicate management of the system and tracking of costs. When any of the clouds is exhausting their resources they notify the FM to scale the system either horizontally (scale vm parameters for that particular instance) or vertically (allocate more vm-s in the AS in order to handle the load).

The FM and each cloud in the federation can map hosts to AS-s by means of prefix matching and also by querying the involved AS clouds. Since the federation is built by paying for resources, such information is crucial and should be provided by the clouds on which resources can be allocated. The routing information can be also derived by matching prefixes to a known IP block allocation. More precise info can be given by the specific AS cloud provider with the finality of selling the resources.

Clouds participating in the live streaming overlay are bootstrapped by the FM. When new cloud resources are allocated the FM provides a initial seed list of other clouds part of the system, from which the cloud can start integrating itself in the gradient overlay. The virtual servers reorganize themselves on a proximity based gradient in order to provide a locality aware spanning architecture. A high level view of the system is shown in Fig. 4, and as shown the cloud resources are organized in a gradient. The clients connect to the architecture gradient back-end transparently and update their proximity measure to ensure that they are getting the stream from the closest source.

For scaling purposes the clouds simply monitor the rate of incoming clients and if such rate is equal or higher than the setup speed of new resources, notifies the FM to allocate new resources and starts admission control on resource exhaustion. The clients keep an updated list of servers ordered by cloud proximity to itself. As such when admission control is operating the clients simply try different clouds until a cloud with available resources is found. More advanced and efficient allocation and scheduling policies will be considered in future related work.

B. Overlay Construction, Maintenance and Resource Scaling

Algorithm. 1 presents the overlay construction and maintenance operation for the cloud federation service oriented architecture. Each cloud runs a cloud virtual infrastructure manager (VIM) which in turns execute the following algorithm to maintain a local view of the neighboring clouds closest to the source cloud and to itself. Constantly the clouds exchange their views in order to derive a stable gradient topology, where the center of the gradient is the source and the farthest from the source a cloud is, the furthest it is placed in the gradient.

Algorithm 1 Overlay Construction

Input: L^k View ordered by tuple (proximity to source cloud, proximity to cloud executing algorithm, cloud ID)
Input: k view length
Input: tr_window transmission window length
Output: D^k New view of neighbors
Output: $dist\_to\_src$ Cloud distance to source cloud
Initialisation :
1: $round\_partner$ = $random\_sample(L^k, 1)$
2: S = $push\_to\_query(round\_partner,\ L_{1-tr_window})$
3: D = L
Measure neighbor sample distances
4: **for** $i = 1$ to k **do**
5: $S_i.distance\_to$ = $measure\_distance(i)$
6: **end for**
Update View
7: **for** $cloud$ in S **do**
8: if $closer\_to\_source(cloud, D)$
9: $D.append(S)$
10: **end for**
Sort View by Tuple(proximity to source cloud, proximity to cloud executing algorithm, cloud ID)
11: D=$trim(sort(D), k)$
12: $dist\_to\_src$ = $measure\_distance(D_0) + D_0.dist\_to\_src$
13: **return** D, $dist\_to\_src$

The Virtual Cloud Managers of each cloud periodically executes Algorithm. 1, a gossip based algorithm, in order to build the cross-cloud communication network for the live streaming model described in this work. The VIM at every moment in time maintains a local view of the whole system that is based on the clouds that enhance the locality measure compared to the distance from the source that the current cloud has. Based on the "Gradient" gossip algorithm, the view consist of a set of clouds (similarity set) that have a better utility function then the actual cloud running the algorithm and the criteria for new clouds to join the similarity set is as follows: the candidate cloud to enter the similarity set, needs to improve or at least not deteriorate the utility function over all the other clouds in the set. This is translated into the fact of having better triangular distance between the actual cloud and the source (not worst proximity to either the source and the actual cloud running the algorithm).

Line 1-3 respectively pick a random cloud VIM partner to exchange views with, sends the local view to the partner with a request to exchange views, and create a copy of the view to process during the algorithm. Proceeding on lines 4-6 the cloud receives the best representatives of the view of the partner, and measure the distance between itself and the clouds in such view. The following lines 7-10 check if any of the new clouds takes us closer to the source and updates the similarity set, the view of the system. Finally terminating the algorithm in lines 11-13 the new view of the neighboring clouds is sorted and trimmed to match the k parameter dictated by the system, and updates the proximity measure for the cloud executing the algorithm toward the source by using the best candidate, the first element of the sorted view. The order of the elements in the similarity set is generated by calculating for each element a tuple composed of: Triangular distance to source through this neighbor, distance to this neighbor and last the cloud ID. This order shows the preferences the algorithm implements toward the forwarding cloud.

The initial view of the system is obtained by the clouds running the algorithm directly by the central FM, which acts as a bootstrap server. In the basic implementation of the system, the FM allocates resources in a greedy fashion in order to satisfy all clients with local connection. A problem of the P2P live streaming gossip based algorithms is the inability to limit or foresee the depth of the spanning tree or of the formed graph in general and as such no strict guarantees can be made on latency. The following subsection introduces a theorem which dictates that for this system model, such limits do exist.

The resources of a multi-cloud federation have predictable availability based on SLA-s contracted with the cloud providers. In case of cloud failure the algorithm can be modified to simply remove such cloud from the proximity list of neighbors and run with the reduced neighbor view.

C. Proof of limited spanning depth

The key idea in order to enhance latency in the gradient overlay is not only the "Greedy" approach used on building the overlay by means of locality, the other idea is a limited height of the overlay levels. In order to prove that the latency growth is limited we present the following theorem. By construction Access Networks interchange data through special point of presents (different from the clients points of presence described in this work), that provide high bandwidth low-latency cross-AS connectivity. This network topology enables simplified network control, accounting and management. Thus the shortest and fastest path between two hosts in different AS-s is to pass through the Core Networks. If the edges of the two AS-s could be able to exchange data between client PoP-s the data path would be significantly slower and bandwidth limited.

Definition 1. *Core Network Clouds, by construction, can only be interconnected directly through each other or through a Public Cloud.*

Theorem 1. *Worst case, overlay latency and depth spread,*

is bounded by the largest pair-wise distance between Core Clouds.

Proof: We prove this theorem by considering the possibilities of the paths connecting two random hosts in the overlay. Considering any two hosts H_a and H_b and a possible path connecting the two in the overlay $p_{(a,b)}$. Such path is of the form $p_{(a,x)}...p_{(x_n,b)}$. Let us consider the types of possible intermediary paths, elements H_{x_i} and $H_{x_{i+1}}$ can be of types Core-to-Core, PoP-to-PoP or PoP-to-Cloud. As stated in Definition. 1, by crossing towards a core network intermediary, by construction of the system, we are guaranteed that the chosen path can be a candidate best proximity based possible path. Cross-AS crossings of the type Core-to-Core, and PoP-to-Core, comply with system design and could be accepted as part of a best path between two different AS hosts. In case of PoP-to-PoP intermediary links, let us suppose by absurd that the shortest path constructed between two AS-es is eventually only made of PoP-to-PoP links. By means of this assumption we have a best proximity wise path between Cross-AS hosts that is made of only PoP-to-PoP links. This implies that there exists a path between two AS not crossing their Core Clouds, that is the best possible path between such hosts, which contradicts Definition 1, and proves that a chain of forwarders based on proximity can't grow more than the best alternative path that crosses a Core Cloud link. As such in a worst case scenario, the maximum pairwise distance of Core Clouds is the maximum possible chain length of inter-PoP chaining. ■

Having discussed in details the overlay algorithm and the implications of such system design we move on to the evaluation of a simulation of the proposed design.

VI. Case study of enhanced streaming algorithm

A. Enhanced live streaming algorithm evaluation

In this section we provide some results based on simulation to support the usage of such novel cloud technology. The provided results validate the system construction as described in this work, by constructing a solid cross-cloud overlay that minimizes latency and has limited overlay diameter or depth, as such providing enhanced service locality for live streaming. For these experiments we implemented a simulation of the

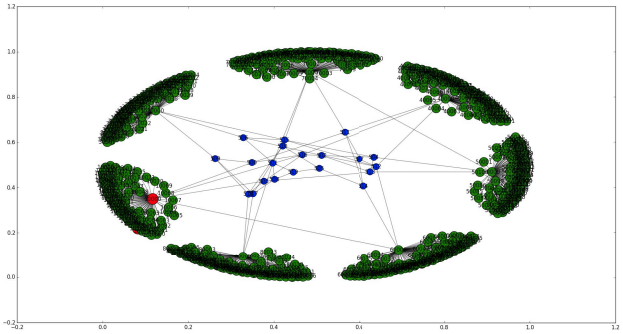

Fig. 5. Simulated multi-cloud infrastructure

cross-cloud distributed algorithm. In order to have more meaningful results the proximity measure used in this evaluation is number of hops. Such metric can be easily translated to real latency estimation based on latency measurements on these specific networks (Community Networks and ISP and Telephony access network) introduced in Section. II.

A generalized view of the real access network infrastructure and used in these simulations is shown in Fig. 5. Based on the described model, we have a number of geo-distributed clouds that are grouped and interconnected in a "Wheel" graph with the respective Core Clouds as the center of the graph. Internet clouds are simulated by a connected random graph. The Core Clouds are then randomly connected in between themselves and the Internet Clouds graph. The composed graph as shown and previously discussed in the system model has at the core the Internet Clouds and Core Clouds and at the edges the PoP Clouds. In Fig. 5 the clusters represent the access networks while the random graph in the center represent the Internet Clouds. This simulation model provides a good approximation of the system model introduced in this work. The first exper-

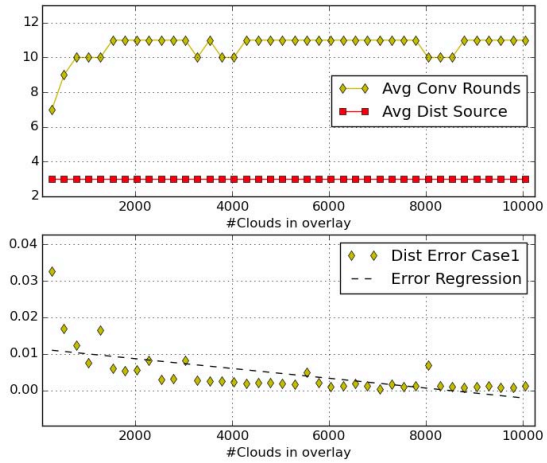

Fig. 6. Case1: Convergence for Scaling Micro-Clouds

iment is conducted by fixing the random graph representing the Internet Clouds and uniformly increasing the number of local micro-clouds. As we can see from Figure. 6 the growth of the micro-cloud fabric does not influence much both the conversion rate and the average distance to the source, further more the error between the overlay estimation and the optimal proximity based paths is minimal. In this second experiment the dimension of the micro-cloud fabric is kept unchanged and the dimension of the public cloud back-end is increased. With this experiment we study the impact of the overhead of such growth in the overlay algorithm. As predicted the growth of the Internet back-end impacts lightly the convergence rate and the distance to the source. In Fig.7 we can see that an addition of 400 public clouds impacts lightly the conversion rate and the distance to source. Nevertheless this number would be far smaller in reality as the global cloud providers are in a far smaller numbers. In general by having a small number of public providers and a large number of local clouds as in the first case Fig.6 the system can scale without performance penalty. This in itself provides also a good property of the system, as local growth provides better performance and very little management or performance penalty. In this second case

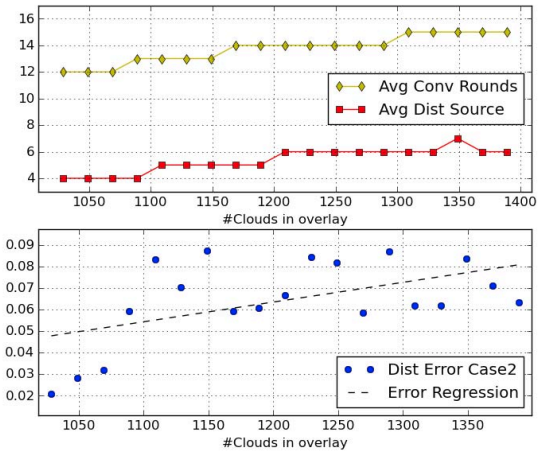

Fig. 7. Case2: Convergence for Scaling Public Cloud Fabric

as well the errors are really negligible and for this architecture the distributed algorithm produces a pretty good approximation of the optimal proximity aware path.

VII. Conclusion and Future Work

To summarize in this work we presented a multi-cloud self-organized algorithm to optimize stream latency for live streaming on massive scale. The proposed algorithm builds an overlay that approximates the optimal proximity based solution up to 0.02 of standard error.

The clients actively request the stream from the closest cloud transparently and are insured persistent presence of the stream either through local or internet clouds. This architecture enables better support for client mobility and scales both vertically and horizontally providing a decoupling of the streaming backend from the client churn and providing a more stable backend for the service.

Future work will be focused on open problems toward the finalization of the proposed architecture, such as resource allocation in the clouds involved in the system, resource provisioning and price vs performance trade-offs toward a complete scalable solution.

Acknowledgement

This work was supported in part by the Erasmus Mundus Joint Doctorate in Distributed Computing (EMJD-DC) funded by the Education, Audiovisual and Culture Executive Agency (EACEA) of the European Commission under the FPA 2012-0030, and by the Spanish government under contract TIN2013-47245-C2-1-R.

References

[1] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, I. Stoica, and M. Zaharia, "A view of cloud computing," *Commun. ACM*, vol. 53, no. 4, pp. 50–58, Apr. 2010. [Online]. Available: http://doi.acm.org/10.1145/1721654.1721672

[2] A. Khosravi, S. K. Garg, and R. Buyya, "Energy and carbon-efficient placement of virtual machines in distributed cloud data centers," in *Euro-Par 2013 Parallel Processing*. Springer, 2013, pp. 317–328.

[3] S. Nedevschi, S. Ratnasamy, J. Padhye, and I. Reserch, "Hot data centers vs. cool peers." in *HotPower*, 2008.

[4] P. Olson, "WhatsApp Hits 600 Million Active Users, Founder Says," *Forbes, retrieved October*, 2014.

[5] C. Forecast. (2014, Dec.) Cisco Visual Networking Index: Forecast and Methodology, 20142019. [Online]. Available: http://www.cisco.com/c/en/us/solutions/collateral/service-provider/ipngnipnextgenerationnetwork/white_paper_c11481360.html

[6] Periscope. (2015, Dec.) Live streaming company. [Online]. Available: http://www.periscope.com/

[7] Skype. (2015, Dec.) Live communications. [Online]. Available: http://www.skype.com/

[8] M. G. Saslow, "Effects of components of displacement-step stimuli upon latency for saccadic eye movement." *Journal of the Optical Society of America*, vol. 57, no. 8, pp. 1024–1029, 1967.

[9] D. Petcu, "Consuming Resources and Services from Multiple Clouds," *Journal of Grid Computing*, vol. 12, no. 2, pp. 321–345, Jan. 2014. [Online]. Available: http://link.springer.com/10.1007/s10723-013-9290-3

[10] A. N. Toosi, R. N. Calheiros, and R. Buyya, "Interconnected Cloud Computing Environments," *ACM Computing Surveys*, vol. 47, no. 1, pp. 1–47, May 2014. [Online]. Available: http://dl.acm.org/citation.cfm?id=2620784.2593512

[11] R. Baig, J. Dowling, P. Escrich, F. Freitag, A. Moll, L. Navarro, E. Pietrosemoli, R. Pueyo, V. Vlassov, M. Zennaro, and R. Meseguer, "Deploying Clouds in the Guifi Community Network," *14th IFIP/IEEE Symposium on Integrated Network and Service Management (IM 2015)*, May 2015.

[12] V. Xhagjika, V. Vlassov, M. Molin, and S. Toma, "Structured cloud federation for carrier and isp infrastructure," in *Cloud Networking (CloudNet), 2014 IEEE 3rd International Conference on*, Oct 2014, pp. 20–26.

[13] A. H. Payberah, F. Rahimian, S. Haridi, and J. Dowling, "Sepidar: Incentivized market-based P2P live-streaming on the Gradient overlay network," *Proceedings - 2010 IEEE International Symposium on Multimedia, ISM 2010*, pp. 1–8, 2010.

[14] A. H. Payberah, J. Dowling, F. Rahimian, and S. Haridi, "GradienTv: Market-based P2P live media streaming on the gradient overlay," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 6115 LNCS, pp. 212–225, 2010.

[15] A. H. Payberah, J. Dowling, and S. Haridi, "GLive: The Gradient overlay as a market maker for mesh-based P2P live streaming," *Proceedings - 2011 10th International Symposium on Parallel and Distributed Computing, ISPDC 2011*, no. i, pp. 153–162, 2011.

[16] X. Liao, H. Jin, Y. Liu, L. M. Ni, D. Deng, C. W. Bay, and H. Kong, "AnySee: Peer-to-Peer Live Streaming," vol. 00, no. c, 2006.

[17] S. Asaduzzaman, Y. Qiao, and G. Bochmann, "Cliquestream: An efficient and fault-resilient live streaming network on a clustered peer-to-peer overlay," in *Peer-to-Peer Computing , 2008. P2P '08. Eighth International Conference on*, Sept 2008, pp. 269–278.

[18] K. Park, S. Pack, and T. . Kwon, "An adaptive peer-to-peer live streaming system with incentives for resilience," *Computer Networks*, vol. 54, no. 8, pp. 1316–1327, 2010. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1389128609003521

[19] L. Kontothanassis, R. Sitaraman, J. Wein, D. Hong, R. Kleinberg, B. Mancuso, D. Shaw, and D. Stodolsky, "A transport layer for live streaming in a content delivery network," *Proceedings of the IEEE*, vol. 92, no. 9, pp. 1408–1419, 2004.

[20] Y. Liu, H. Yin, G. Zhu, and X. Liu, "Peer-assisted content delivery network for live streaming: Architecture and practice," *Proceedings of the 2008 IEEE International Conference on Networking, Architecture, and Storage - IEEE NAS 2008*, pp. 149–150, 2008.

[21] T. N. Kim, S. Jeon, and Y. Kim, "A CDN-P2P hybrid architecture with content/location awareness for live streaming service networks," *Proceedings of the International Symposium on Consumer Electronics, ISCE*, pp. 438–441, 2011.

[22] Z. Zhuang and C. Guo, "Optimizing CDN infrastructure for live streaming with constrained server chaining," *Proceedings - 9th IEEE International Symposium on Parallel and Distributed Processing with Applications, ISPA 2011*, pp. 183–188, 2011.