Towards a Carrier SDN: An example for Elastic Inter-Datacenter Connectivity

L. Velasco[1*], A. Asensio[1], J.L. Berral[1,2], A. Castro[1], V. López[3]

(1) Universitat Politècnica de Catalunya (UPC), Barcelona (Spain), Email: lvelasco@ac.upc.edu
(2) Barcelona Supercomputing Center (BSC), Barcelona (Spain)
(3) Telefónica Investigación y Desarrollo (TID), Madrid (Spain)

Abstract *We propose a network-driven transfer mode for cloud operations in a step towards a carrier SDN. Inter-datacenter connectivity is requested in terms of volume of data and completion time. The SDN controller translates and forwards requests to an active PCE controlling a flexgrid network.*

Introduction

Cisco global cloud index [1] forecasts datacenter (DC) traffic to quadruple over the next years, reaching 554 EB per month by 2016. Two main components of traffic leaving DCs can be distinguished: traffic among DCs (DC2DC) and traffic between DCs and end users (DC2U). The former includes database (DB) synchronization among replicated services and virtual machine (VM) migration to manage the cloud elastically, whilst the latter is associated to applications, such as web, email, video-on-demand, etc.

DC operations are commonly automated using some cloud middleware which applies scheduled-based algorithms to optimize some utility function ensuring quality of service, service availability and operational costs (e.g. energy consumption). The outcome of those algorithms is the set of VMs to be activated or stopped in each DC, or to be migrated from a DC to a remote one, as well as the set of DBs to be synchronized. As a result, the connectivity required between two DCs highly varies along the day, presenting dramatic differences in an hourly time scale as a consequence of the huge amount of raw data being transferred.

To deal with DC2DC traffic variations, the fine optical spectrum granularity and elastic connection capabilities provided by the flexgrid technology [2] can be used. Dynamic and elastic connections can be requested to an Application-Based Network Operations (ABNO) controller [3]. ABNO controller delegates some functions in other entities, such as the provisioning process, which is done by an active stateful PCE [4].

However, competence for network resources could lead to connections capacity being reduced or even blocked at requesting time, resulting in a poor cloud performance. Therefore, cloud middleware can either perform connection request retries, similar to I/O pooling (software-driven I/O) in computers, to increase the bitrate of already established connections or set-up new ones.

Carrier Software Defined Network (SDN)

To provide an abstraction layer to the underlying network, a new stratum on top of the PCE, the carrier SDN, could be deployed. The carrier SDN controller implements a northbound interface to request transfer operations. Those applications' operations are transformed into network connection requests. The northbound interface uses application-oriented semantic, liberating application developers from understanding and dealing with network specifics and complexity.

As an example of that paradigm, in this paper we propose an operation where cloud middleware requests transfers using its native semantic: amount of data to be transferred, DC destination, completion time, etc. The SDN controller is in charge of managing inter-DC connectivity; if not enough resources are available at requesting time, notifications (similar to interruptions in computers) are sent from the PCE to the SDN controller each time specific resources are released. Upon receiving a notification, the SDN controller takes decisions on whether to increase the bitrate associated to a transfer. Therefore, we have effectively moved from pooling to a *network-driven* transfer mode.

Scenario

We consider a scenario where a flexgrid-based network is used to interconnect federated DCs.

A *follow-the-work* strategy for VM migration was implemented; VMs are moved to DCs closer to the users, reducing the user-to-service latency. Cloud scheduling algorithms run periodically taking VM migration decisions. Once the set of VM to be migrated and DB synchronization needs are determined, cloud management performs those operations in collaboration with inter- and intra- DC networks.

Since our scheduling algorithms run periodically each hour, VM migrations are required to be performed within each period. In fact, the shorter the transfer time, the better the offered network

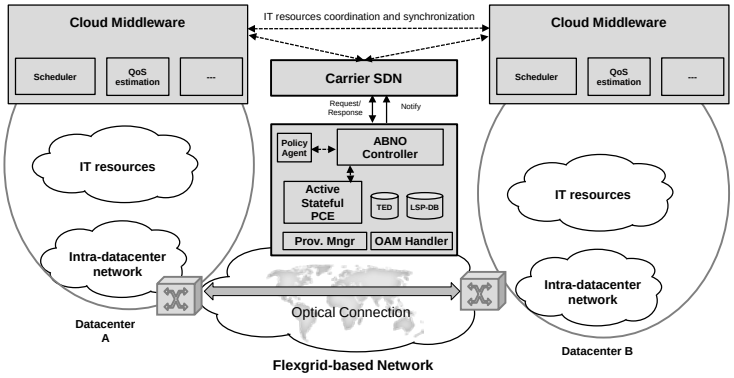

Fig. 1. Network Operating System implementing System Calls for the application layer and request/response and notifications towards the network.

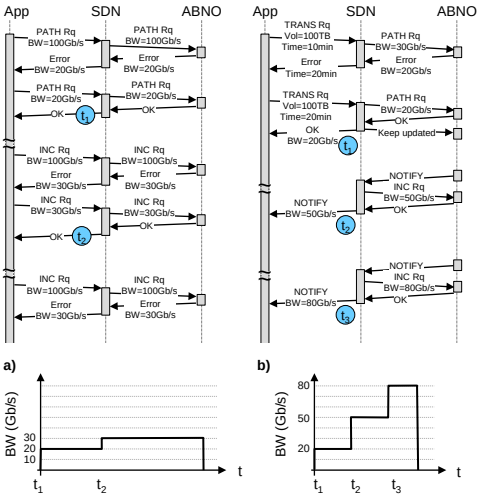

Fig. 2. Software-driven (a) and network-driven (b) models.

service is, by having earlier the VMs in their proper locations.

Carrier SDN is deployed in between the ABNO controller and DC middleware (Fig. 1) and implements two connectivity models: *i*) *application-driven*, and *ii*) *network-driven* model.

In the application-driven model, each local cloud middleware manages connectivity to remote DCs so as to perform VM migration in the shortest total time. The source cloud manager requests Label Switched Paths (LSP) set-up, teardown, as well as elastic operations to the SDN controller, which forwards them to the ABNO controller (Fig. 2a). In this model, the SDN works as a proxy between applications and network. After checking local policies, the ABNO controller forwards the requests to the active stateful PCE, which performs LSP operations on the controlled network.

It is worth noting that, although applications have full control over the connectivity process, physical network resources are shared with a number of clients and LSP set-up and elastic spectrum increments could be blocked as a result of lack of resources in the network. Hence, applications need to implement some sort of periodical retries to increase the allocated bandwidth until reaching the required level. These retries, could impact negatively on the performance of the inter-DC control plane and do not ensure achieving higher bandwidth.

In the network-driven model (Fig. 2b), applications request transferences instead of connectivity. The source cloud manager sends a transfer request to the SDN controller specifying the destination DC, the amount of data that need to be transferred, and the maximum completion time. Upon its reception, the SDN controller requests the ABNO controller to find the greatest spectrum width available, taking into account local policies and current service level agreements (SLA) and sends a response back to the cloud manager with the best completion time. The source cloud manager organizes data transference and sends a new transfer request with the suggested completion time. A new connection is established and its capacity is sent in the response message; in addition, the SDN controller requests ABNO controller to keep it informed upon more resources are left available in the route of that LSP. ABNO controller has access to both the Traffic Engineering Database (TED) and the LSP-DB. Algorithms deployed in the ABNO controller monitor spectrum availability in those physical links. When resource availability allows increasing the allocated bitrate of some LSP, the SDN controller performs elastic spectrum operations so as to ensure committed transfer completion times. Each time the SDN controller modifies bitrate by performing elastic spectrum operations, a notification is sent to the source cloud manager containing new throughput. Cloud manager then optimizes VM migration as a function of the actual throughput while delegating ensuring completion transfer time to the SDN controller.

Illustrative results

For evaluation purposes, we developed scheduling algorithms in an OpenNebula-based cloud middleware emulator. Federated DCs are connected to an ad-hoc event-driven simulator developed in OMNET++. The simulator implements the SDN controller and the flexgrid network with an ABNO controller on the top, as described in Fig. 1. Regarding PCE, the algorithm described in [5] for elastic operations was implemented.

For our experiments, we assume the global 11-node topology depicted in Fig. 3. These

Fig. 3. Global inter-DC topology

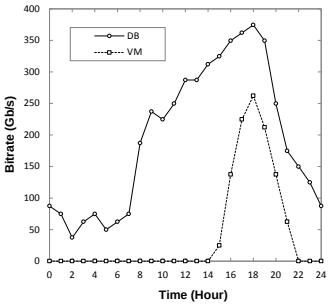

Fig. 4. Req. bitrate in a day

Table 1. Performance results

	Requests (#/h, %success)	Max/Avg Time-to-transfer (min)	
		DBs	VMs
Software-driven	43.1, 53.5%	58.0 / 29.0	54.0 / 28.2
Network-driven	65.3, 100%	28.0 / 22.4	27.0 / 22.2

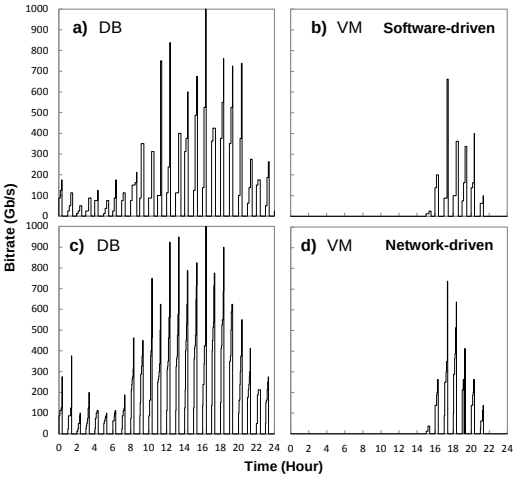

Fig. 5. DC2DC connection bitrate vs. time

locations are used as source for DC2U traffic. In addition, four DCs are strategically placed in Illinois, Spain, India, and Taiwan. DC2DC and DC2U traffic compete for resources in the physical network. We fixed the optical spectrum width to 4 THz, the spectral granularity to 6.25 GHz, the capacity for the ports connecting DCs to 1 Tb/s, the number of VMs to 35,000 with an image size of 5 GB each and we considered 300,000 DBs with a differential image size of 450 MB and a total size of 5 GB each; half the size of Wikipedia [6]. Additionally, TCP, IP, GbE and MPLS headers have been considered.

Fig. 4 shows the required bitrate to migrate VMs and synchronize DBs in 30 minutes. VM migration is performed as a *follow-the-work* strategy and thus, connectivity is used for just during part of the day. In contrast, DB synchronization is performed along the day, although bitrate depends on the amount of data to be transferred, i.e. on users' activity.

Fig. 5a and Fig. 5b depict the assigned bitrate for DB synchronization and VM migration, respectively between two DCs during a 24 hours period when the software-driven model is used. Fig. 5c and Fig. 5d show the assigned bitrate when the network-driven model is used. Software-driven model tends to provide longer transfer times as a result of not obtaining additional bitrate in retries. Note that although the initial bitrate which is assigned at each interval is the same, intervals tend to be narrower using the network-driven model. The reason is that models assigns additional bitrate to the connections as soon as resources are released by other connections.

Table 1 shows the number of required requests messages per hour needed to increase bitrate of connections for the whole scenario. As illustrated, only 50% of those requests succeeded to increase connections' bitrate under the software-driven model, in contrast to 100% reached under the network-driven model. Additionally, Table 1 shows that when using the network-driven model both the maximum and average required time-to-transfer are significantly lower than when the software-driven model is used. The longest transfers could be done in only 28 minutes when the network-driven model was used compared to just under 60 min using the software-driven model. Note that the amount of requested bitrate is the same for both models.

Conclusions

A DC federation scenario has been used as example of a carrier SDN controller implementing a northbound interface with application-oriented semantic. Two connectivity models have been compared. The software-driven model needs periodical retries requesting increase connection's bitrate, which do not translate into immediate bitrate increments and could have a negative impact on the performance of the inter-DC control plane. In contrast, the network-driven model takes advantage of the use of notify messages being able to reduce time-to-transfer remarkably.

Finally, the proposed network-driven model opens the opportunity to network operators to implement policies so as to dynamically manage connections' bitrate of a set of customers and fulfill simultaneously their SLAs.

Acknowledgements

The research leading to these results has received funding from the European Community's Seventh Framework Programme FP7/2007-2013 under grant agreement no. 317999 IDEALIST project and by the MINECO ELASTIC project (TEC2011-27310).

References

1 Cisco, Global Cloud Index (2012)
2 M. Jinno et al., IEEE Commun. Mag., 47 (2009).
3 D. King and A. Farrel, IETF draft, (2013).
4 E. Crabbe, et al. IETF draft, (2012)
5 A. Asensio et al., in Proc. ONDM, (2013)
6 Wikipedia, http://en.wikipedia.org