Effective Resource Control Strategies using OpenFlow in Cloud Data Center

Davide Adami, Barbara Martini
CNIT, Pisa, Italy
({davide.adami, barbara.martini}@cnit.it)

Molka Gharbaoui, Piero Castoldi
Scuola Superiore Sant'Anna, Pisa, Italy
({m.gharbaoui, castoldi}@sssup.it)

Gianni Antichi, Stefano Giordano
Department of Information Engineering
University of Pisa, Pisa, Italy
((gianni.antichi, stefano.giordano))@iet.unipi.it)

***Abstract*—The increasing popularity of cloud computing applications and the advances in virtualization software technologies (i.e., Virtual Machine deployment) have driven Data Center infrastructures toward a greater complexity and workload dynamicity. Thus, a unified control and management of computing and network resources would be required for assuring proper traffic performances in high volatile virtual machine deployments. This paper introduces a novel resource control platform for virtualized DC environments aimed at optimizing virtual machine placement on physical servers also considering traffic load across links in order to limit oversubscription-related problems. To this purpose OpenFlow statistics are elaborated for their distinguished features that well suit virtualized environments. Two novel algorithms have been conceived and compared by simulations for evaluating the effectiveness of proposed traffic-aware VM placement strategies.**

Keywords-cloud computing; data center; OpenFlow; traffic engineering

I. INTRODUCTION

In the last years, cloud computing has emerged as a new service delivery paradigm offering on-demand computing resources to the general public as a result of a virtualization process of physical resources collected in large Data Centers, i.e., Cloud DCs. Thanks to advances in virtualization software technologies, i.e., Virtual Machine (VM) deployment, cloud service consumers may be offered a subscription-based access to infrastructures, platforms or applications, referred to as Infrastructure as a Service (IaaS), Platform as a Service (PaaS) or Software as a Service (SaaS), respectively [1]. Virtualization technologies allow cloud service providers to apply consolidation procedures within Cloud DCs and, consequently, to achieve a more efficient use of physical servers as well as economies of scales. On the other hand, virtualization technologies pose new challenges in the overall management of the DC infrastructure especially in VMs provisioning and in their proper interoperability. In fact, thanks to more agile procedures for the set-up of new applications (i.e., snapshot, cloning of VMs), provisioning or reconfiguration operations may occur much more frequently in virtualized DC environments than in legacy DCs. Definitely, the high volatility of VMs in Cloud DC requires that the intra-DC network infrastructure becomes more flexible, agile and integrated with server and storage infrastructure [2]. In this paper, a novel OpenFlow-based Virtualization-aware Networking (OFVN) platform is introduced for enhancing the Cloud DC architecture with advanced Traffic Engineering (TE) capabilities that jointly consider IT and network resource requirements to overcome the typical replica of control and management functions characterizing traditional multi-layered DC architectures.

Specifically, this work provides and compares different approaches for jointly selecting proper IT and network resources while providing a viable solution for addressing VM placement across IT servers. In fact, traffic load across network links is taken into account during the IT server selection for VM placement by elaborating flow-based OpenFlow (OF) statistics [4]. Indeed, OF can play a key role due to its distinguished features that well suit virtualized environments, i.e., flow-based network resource control capabilities combined with high programmability of routing/switching functions that allow for the enforcement of per-flow routing strategies during VM provisioning. Traffic statistics available from OF-based switches at both per-port level and per-flow level are used for selecting the proper server required to host the VM. This approach allows for satisfying higher rate of VM allocation requests while avoiding possible congestions due to the oversubscription of network links and minimizing network latency. Two algorithms have been conceived and compared for evaluating the effectiveness of different combined selection strategies of a double set of resources, where the choice of a network resource (i.e., switch, router) affects the possible choices of IT resource (CPU capabilities across servers) and vice versa. Simulation results highlight that resource (physical server, OF switches) number and selection order have, as expected, the most relevant impact on the performance of the algorithms.

The rest of the paper is organized as follows. Section II provides an overview on state-of-art of existing solutions and research efforts, while Section III highlights the general features of the OFVN architecture. Section IV describes the algorithms for the joint selection of IT and network resources, whereas Section V compares and discusses the performance of proposed selection algorithms. Finally, section VI provides some concluding remarks.

978-3-901882-50-0

II. STATE OF ART OF CLOUD DC

A number of research efforts have focused on novel resource management solutions for virtualized environments aiming at optimized VM placement across servers. In this context, noteworthy initiatives include both commercial tools for capacity planning, (e.g., VMware Capacity Planner [5], IBM WebSphere [7]) and research works [6][8][9] aimed at finding optimal solutions for VM allocation across servers. The common target is to minimize the number of required servers considering constraints imposed by server capacity, i.e., CPU and amount of memory. However, they do not consider constraints deriving from the traffic load across links due to oversubscription. In order to avoid congestions and minimize the overall traffic latency, a comprehensive approach for VM placement that also considers network traffic is expected.

Network resource usage constraints (e.g., link bandwidth) are considered by [10][11][12] that propose mathematical formulation of optimal VM placement decision process also taking into account network traffic in terms of either topology constraints (e.g., interconnection network infrastructure) or VM bandwidth usage among application components (i.e., VM traffic patterns). Due to NP-hard complexity, they rely on approximation algorithms, i.e., greedy algorithms, for generating a viable solution considering DC scale. Nonetheless, such solutions are obtained once the whole set of VM placement requests, DC network and computing resource constraints (i.e., link capacity, switch interconnections, server capacity) are given. Instead, a viable solution should consider a scenario in which the pattern of incoming VM placement requests is not given in advance due to the high-volatile scenario of virtualized environments as this work actually does.

Authors in [13] actually propose a runtime system for dynamic mapping of VMs onto a virtualized DC server infrastructure considering both application-level constraints, i.e., redundancy requirements, as well as network-level constraints, e.g., VM communication patterns. However, such work focuses only on the description of the algorithm for the partition of the graph across DC infrastructure that addresses aforementioned constraints while any performance evaluation is missing. Moreover, the network traffic should also be considered not only static but dynamic set of constraints. Thus, in our work real-time statistics are exploited about traffic crossing DC switches on both per-port basis and per-flow basis to optimize VM placement across servers while avoiding possible bottlenecks due to oversubscription of the inter-rack links. To this purpose distinguished features of OF are exploited along with its inherit capability to program single flow treatment in a flexible and agile way.

The usability of OF within DC is also a pretty unexplored research area. To the best of our knowledge, just a couple of recent works argue on the use of OF in DC [14] [15]. However, none of the above research works consider the use of OF in the context of VM placement procedures for optimizing network traffic performance while addressing high rate of VM placement across DC infrastructure as this work actually does.

III. OFVN ARCHITECTURE: FEATURES AND FUNCTIONS

Current Cloud DC infrastructures consist of multiple racks that host the servers and are interconnected through enterprise-level switches [16][17]. The interconnection network usually has a canonical fat-tree 2-Tier or 3-Tier topology (see Figure 1). The servers (usually up to 48 in the form of blades) are accommodated into racks and connected to an edge switch through 1 Gbps links. These edge switches are further interconnected through aggregation switches using 10 Gbps links in a tree topology. In the 3-Tier topology, there is one more level, in which the aggregation switches are connected in a fat-tree topology using the core switches either at 10 Gbps or 100 Gbps links (a bundle of 10 Gbps links). This architecture is scalable and fault-tolerant, but oversubscription is necessary to lower the total cost of the network infrastructure (typical edge switch uplinks oversubscription ratios range from 1:5 to 1:20, whereas the oversubscription ratio of aggregation switches uplinks reaches 1:240).

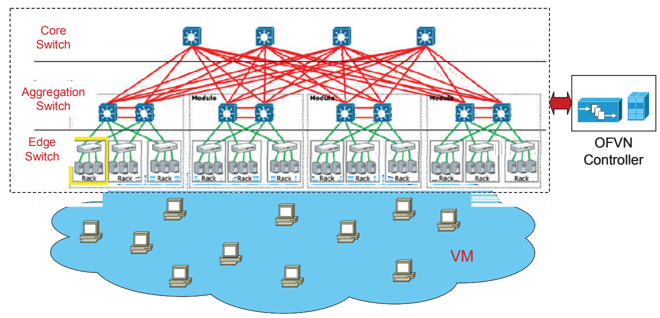

Figure 1 OFVN in a fat-tree 2-tier DC network topology

To optimize the performance of the services provided by a Cloud DC VMs placement into the DC servers should take into account both application traffic profile (e.g., data exchange among VMs) and traffic load of intra-DC network links. Moreover, traffic flows routing across DC network links should be optimized according to link bandwidth utilization in order to guarantee efficient VM-to-VM communications.

Such problems are really challenging because of the great number of DC nodes and, hence, relevant number of possible data exchanges between those nodes that can cause capacity bottlenecks with a huge latency increment. Moreover, the overall performance are also affected by the coexistence of service delivery models (IaaS, SaaS, PaaS) with specific traffic requirements and by the significant number and heterogeneity of applications in terms of computation and communication models, workloads, etc. Currently, the lack of efficient resources allocation algorithms is a major weakness of commercial and open-source tools for DC management. The provision of IT resources (i.e., VM start-up) is usually "location-unaware" because VM placement into physical servers does not take into account interactions dictated by the applications they run. Moreover, the deployment of VMs is done without considering the availability of network resources, (e.g., traffic load over DC links) hence is completely independent of network resources availability (e.g., bandwidth bottlenecks). Therefore, unifying control and management capabilities of IT and network resources is recommended.

The OFVN architecture enhances the traditional architecture of Cloud DCs with a novel functionality enabling the management and control of network resources within the DC based on the OF paradigm (see Figure 1). More specifically, in Cloud DC, OFVN improves the efficiency in resource utilization: higher dynamics of ultra-broadband connectivity will result in an optimization and increased efficiency of the allocation of virtual resources on physical ones. Following the OF paradigm, the OFVN architecture relies on a strict separation between control and forwarding functions. More specifically, an OFVN-compliant switch is controlled and managed by an OFVN controller through the OF protocol over the OF interface. Figure 2 shows the OFVN Controller architecture.

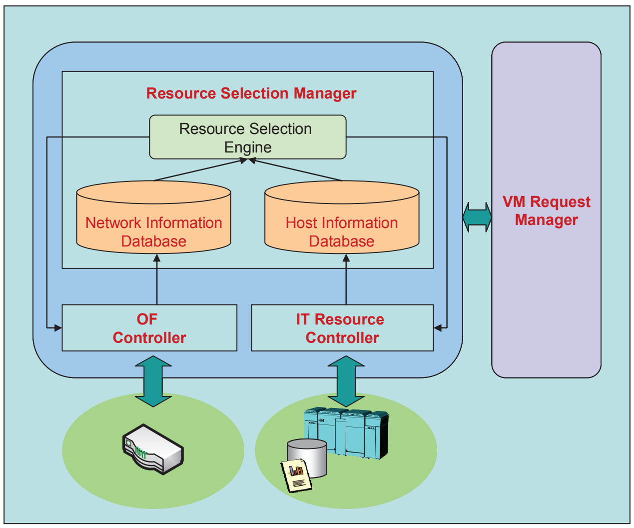

Figure 2 OFVN Controller Architecture

Four main elements compose the controller:

- **VM Request Manager:** is responsible for handling new VM allocation requests.
- **Resource Selection Manager:** includes the following components:

  -Resource Selection Engine: selects the OF switches and the physical server for VM placement by executing a joint IT and network resources selection algorithm leveraging both switch and host status information.

  -Network Information Database: contains information about OF switches status. Traffic flows statistics are retrieved by the OF controller.

  -Host Information Database: contains information about servers status. CPU and memory utilization statistics, number of allocated VMs, etc. are retrieved by the IT resource controller.

- **OF Controller:** is the well-known component of the OF architecture. It adds and removes flow entries from the Flow Table of the OF switches as results from the execution of the resources selection algorithm. Moreover, it retrieves a set of traffic statistics from OF switches while updating the Network Information Database. Specifically: per-flow received/transmitted packets, per flow received/transmitted bytes, per-flow duration (with nanosecond granularity), per-port received/transmitted packets, per-port received/transmitted bytes, per-port received/transmitted errors
- **IT Resource Controller:** manages the VM placement in the available physical servers. Moreover, it retrieves information about servers status and inserts data in the Host Information Database.

Upon the arrival of a new VM placement request, the OFVN controller has to select the server for placing the VM. Therefore, the VM Request Manager forwards the request to the Resource Selection Manager. Such manager, through the Resource Selection Engine (taking into account the status both of the switches - Network Information Database - and of the servers - Host Information Database -) selects the server for the VM and the path to route the traffic exchanged by the VM. Finally, the IT Resource Controller places the VM on the selected server, whereas the OF Controller updates the Flow Table of the selected OF switches with the information necessary to forward the traffic flows to/from the VM.

IV. JOINT IT AND NETWORK RESOURCES SELECTION ALGORITHM

This section discusses two algorithms, called Server-Driven and Network-Driven, respectively, that can be implemented by the Resource Selection Engine. The main difference between them consists in the order in which the physical server for VM placement and OF switches for forwarding the traffic to/from the VM are chosen. Within each set of resources (i.e., switches and servers) basic selection policies (e.g., first fit, best fit or worst fit) are used, i.e., VM placement policy for selecting the server, switch selection policy for selecting the data path across the network. In the following, it is assumed that a DC may be represented as:

- a pool of N physical servers $\mathbf{H}=\{H_n \mid n=1,2,\ldots N\}$,
- an intra-DC network, whose topology is depicted in Figure 1, composed of:
  - a set of K edge switches $\mathbf{ES}=\{ES_k \mid k=1,2,\ldots K\}$
  - a set of L aggregation switches $\mathbf{AS}=\{AS_l \mid l=1,2,\ldots L\}$
  - a set of M core switches $\mathbf{CS}=\{CS_m \mid m=1,2,\ldots M\}$,

a) Server-Driven algorithm

In the Server-Driven algorithm, a physical server is firstly selected for possibly placing the VM, then a sequence of switches is selected for forwarding the traffic flows to/from that VM across the DC network. More specifically, through a VM placement policy (e.g., first fit, best fit or worst fit) a candidate server that satisfies the VM allocation request is designated. At this stage, since oversubscription is usually adopted in the DC, the server is defined as "candidate", because placing a new VM in a physical server requires to check the status of the network. For this reason, the Resource Selection Manager checks the status of the edge switch directly connected to the candidate physical server by analyzing per-flow traffic statistics (i.e., byte sent, duration) collected by the OF Controller. Such statistics allow to estimate the traffic load of the edge switch: in case of overload, the candidate physical server is discarded, and the IT Resource Manager begins to search for another candidate server excluding the ones directly connected to the overloaded edge switch. Otherwise, the edge switch is added to the candidate node list. Next, the switch selection policy (e.g., first fit, best fit or worst fit) and the previously described control about traffic load are repeated to find aggregation and core switches. The VM placement process ends when the OF Controller inserts the proper forwarding rule in the selected edge, aggregation and core switches. At this point the VM is accessible to the applicant. Further details about the algorithm are reported in Figure 3.

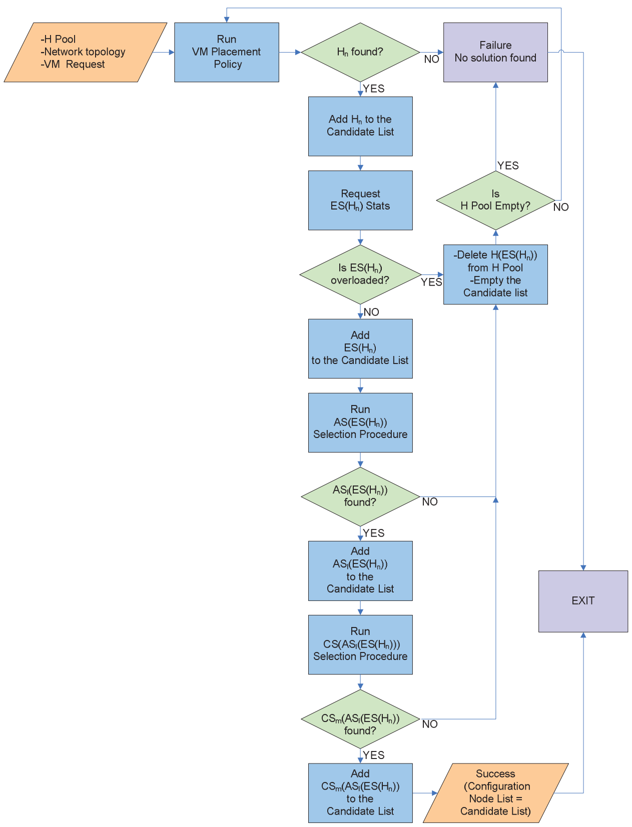

Figure 3 Flowchart of the Server-Driven algorithm

b) Network-Driven algorithm

In case of the Network-Driven algorithm the sequence of OF switches are firstly selected, then the server is selected among those that are reachable through those OF switches. Candidate core, aggregation and edge switches are selected by applying a specific selection policy (e.g., first fit, best fit or worst fit) and checking if the OF switch is able to process all the received traffic (i.e., the switch is not overloaded). As in the case of the Server-Driven algorithm, the load of each switch is estimated by means of per-flow traffic statistics (i.e., byte sent, duration) collected by the OF Controller. Next, the physical server is selected by a VM placement policy (e.g., first fit, best fit or worst fit) applied to the group of servers directly connected to the candidate edge switch. The VM placement process ends when the OF Controller inserts the proper forwarding rule in the edge, aggregation and core switches. At this point the VM is accessible to the applicant. Further details about the algorithm are reported in Figure 4.

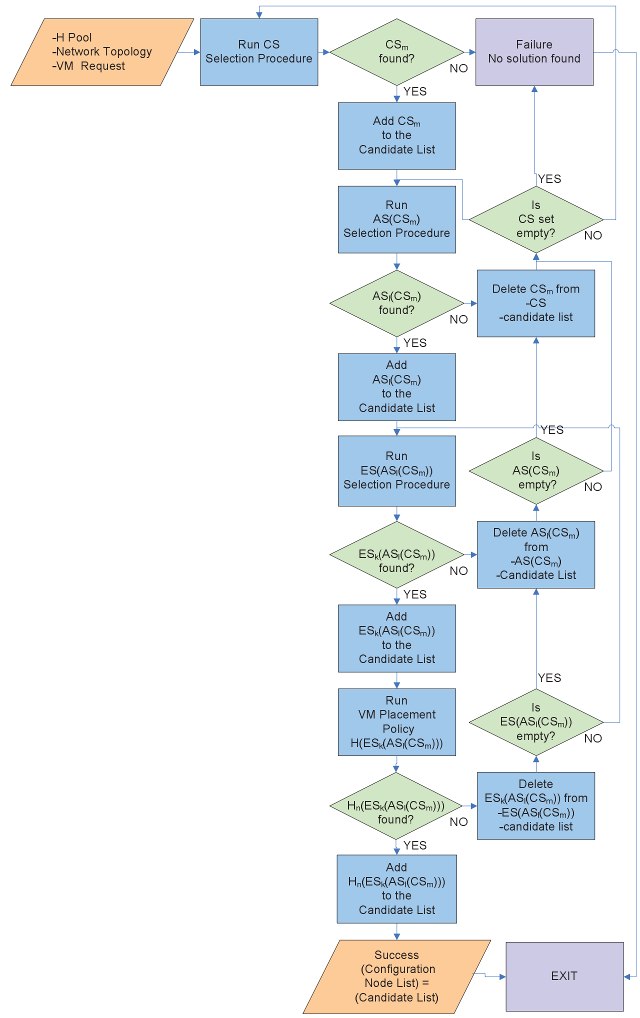

Figure 4 Flowchart of the Network-Driven algorithm

c) OF usage implications in the selection algorithms

The controller is recognized as one of the main bottlenecks of the OF architecture. Every time an OF switch does not know how to treat a packet, it has to ask directly to the controller. Such kind of architecture, in heavily loaded links (as the ones in DCs) may lead to high dropping rates due to the impossibility of the controller to satisfy all the OF switches requests. However, such a case should not happen in our architecture, since a user starts using a VM only after the VM has been placed in a physical server and the proper rules are installed into the switches. While the Server-Driven algorithm focuses firstly on the selection of a candidate server (the network path will depend on the selected server), the Network-Driven algorithm privileges the choice of the best network path (once it has been found, it will choose one of the physical server directly connected to the selected edge switch). The first algorithm may be preferable when there are a lot of VM request placements with low packet-rate while the second one is preferable when high packet-rate per VM placement occurs. Choosing firstly the best network path (Network-Driven Algorithm) forces the OFVN controller to ask for statistics to all the switches in the DC and could lead to a huge data transfer from OF switches to the controller, overloading both the network links and controller itself. However, considering that the VM placement problem is not a real-time process, such huge data transfer could be spread over the time offloading in a way both network links and controller.

V. Simulation Results

a) *Simulation Settings*

The performance of the Server-Driven and Network-Driven algorithms are compared in terms of blocking probability of VM allocation requests, by using a custom-built, event-driven Java simulator. Requests for VM allocation are uniformly distributed in the range [100, 1000] MIPS as regards CPU, and are generated according to a Poisson process, hence inter-arrival and holding times are exponentially distributed with an average of $1/\lambda$ and $1/\mu$, respectively. The average holding time $1/\mu$ is fixed to 85 minutes, while $1/\lambda$ ranges from 1 second to 120 seconds with a step of 20 seconds. For values under 20 seconds (high load), a step of 5 seconds is fixed. All results are plotted with the confidence interval at 95% confidence level. With respect to the DC network topology described in Sec. III a scaled down version is considered in the simulations without lack of generality. More precisely, 20 servers with a maximum computational power of 1000 MIPS each, are connected to all edge switch through 100 Mbps links. Then, 16 edge switches are considered that are further interconnected through 8 aggregation switches using 1 Gbps links in a tree topology. Finally, the 8 aggregation switches are connected in a fat-tree topology using 2 core switches at 1 Gbps links.

In the simulations, the two algorithms are evaluated using two basic policies for selecting each type of resource, i.e., the First Fit (FF) policy and the Worst Fit (WF) policy. The FF policy allocates the VM (traffic load) in the first available server (switch) in the Server-Driven algorithm (Network-Driven algorithm) where the resources are ordered according to an identifier, independently of their availability. On the contrary, the WF policy searches for the most unloaded server (switch) to allocate the requested VM (traffic load) where the largest free space is available.

b) *Simulation Results*

As highlighted in Sec. V.a, the comparison of the algorithms is performed in terms of blocking probability of VM allocation requests. Contributions to the blocking probability with respect to the unavailability of network resources (i.e., link bandwidth, indicated in the following as Bw) or IT resources (i.e., computational power of servers in terms of MIPS, indicated in the following as CPU) are separated. Such contributions are inter-dependent since the blocking probability due to the unavailability of the resource set that is firstly explored (i.e., primary resource) limits the number of requests for the second kind of resource (i.e., secondary resource) thus increasing the chance for successful allocation. Therefore, the order used for selecting the resources is relevant in the two algorithms as well as the amount of resource set (20 possible servers given a selected network path versus 1 possible edge switch and 2 possible aggregation and core switches given a selected server). In fact, according to the topology depicted in Figure 1, for each server the choice of the edge switch is forced while a choice between both 2 aggregation switches and 2 core switches is possible. Figure 5 and Figure 6 plot the blocking probability vs. the average inter-arrival time (IAT) presented by the two algorithms coupled with the FF basic policy. Results show that the contribution of both CPU and Bw to the total blocking probability is relevant which is mainly because FF tends to compact the allocated resources both at the CPU and at the link network levels thus affecting their blocking probability in the same way.

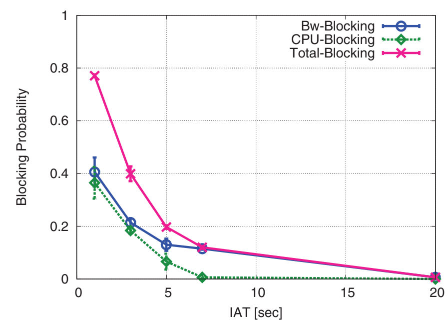

Figure 5 **Server-Driven algorithm coupled with FF policy**

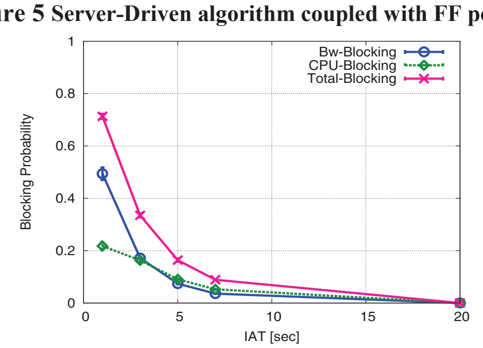

Figure 6 **Network-Driven algorithm coupled with FF policy**

IAT[sec]	ES-Bw	AS+CS-Bw
1	0.0025	0.403
3	0.0016	0.2124
5	0.0002	0.1298
7	0	0.115
20	0	0.0064

Table 1 **Components of the Bw-blocking probability in the Server-Driven algorithm coupled with FF policy**

However, a difference between the two algorithms exists at high loads (i.e., low inter-arrival times) which is due to the order followed for allocating the resources. In fact, FF results more blocking for the primary resources, i.e., CPU in the Server-Driven algorithm and Bw in the Network-Driven algorithm, due to the compacting of the allocated resources, thus limiting the blocking of the secondary resource. However, the trend is not dual due to the amount of secondary resources that is lower for the Server-Driven algorithm. This is confirmed by the results presented in Table 1 showing that almost all the Bw blocking for the Server-Driven algorithms is happening at the (AS+CS) levels since the blocking at the ES is more limited by the blocking at the server (i.e., primary resource) level.

Figure 7 and Figure 8 depict the blocking probability of the algorithms coupled with the WF basic policy. The same trend is observed for the two algorithms where almost all the contribution of the blocking probability is given by the blocking happening during the allocation of the secondary resources (i.e., CPU in the Network-Driven algorithm and Bw in the Server-Driven algorithm). This can be explained by the fact that, since WF tends to spread allocations, it is almost unblocking for the primary resources and thus primarily blocking for the secondary one. Then, once WF is applied to the primary resources, the WF policy is again applied to a limited sub-set of the other type of resource. Due to the fact that WF tends to spread the allocated resources, on the DC global scale, this is practically the same as choosing the other resource in a random way and not as the most free. This explains the relevant difference between the two blocking contributions that does not occur in case of FF, where the global compacting effect of FF still remains on the DC global scale at both resource levels.

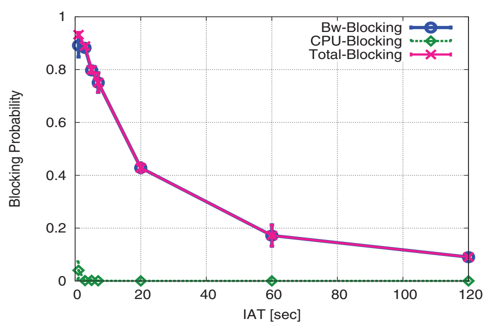

Figure 7 Server-Driven algorithm coupled with WF policy

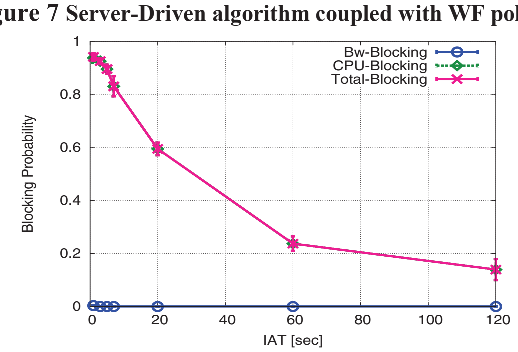

Figure 8 Network-Driven algorithm coupled with WF policy

Moreover, as reported in Table 2, in the Server-Driven case, the total blocking probability is given by the ES-Bw blocking contribution. This is due to the manner in which the WF basic policy is executed: the spreading effect enables the minimization of blocking for CPU unavailability while shifting all the allocation load at network level and, specifically, at the ES level that is automatically selected starting from the server, independently on the amount of resources available at that switch.

IAT[sec]	ES-Bw	AS+CS-Bw
1	0.891	0.001
3	0.888	0
5	0.797	0
7	0.751	0
20	0.428	0

Table 2 Components of the Bw-blocking in the Server-Driven algorithm coupled with WF policy

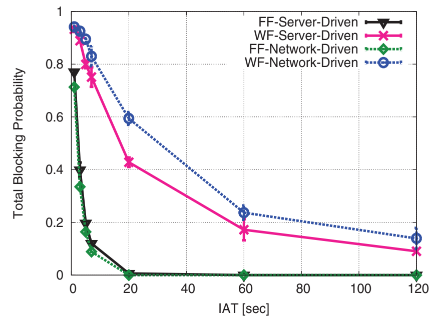

Figure 9 Server-Driven Vs. Network-Driven algorithm

Finally, Figure 9 makes a comparison between the Server-Driven and the Network-Driven algorithms. It indicates that in both cases the FF policy almost behaves in the same way and experiences much less blocking probability with respect to the WF policy. WF on the contrary behaves better when it is coupled to the Server-Driven algorithm. This can be explained by the nature of the policy that tends to scatter the requests on the primary resources unlike FF that tends to compact them. Since the number of servers is large, the scattering has as effect to absorb more "traffic" and then show less blocking. Finally, we can affirm that the performance of the two algorithms mainly depends on the basic policy used for selecting the resources within each set.

VI. Conclusion

The paper presented the design guidelines of a unified control and management functionality, specifically designed for Cloud DCs and provided by the OFVN platform. The OFVN platform allows the optimal placement of VMs into physical servers based on traffic load over DC links leveraging flow-based OF statistics. Two algorithms have been conceived and compared for evaluating the effectiveness of different combined selection strategies of resource from different sets, i.e., network and computation resources. Due to constraints given by DC interconnections, the order of resource selection affects the overall performances, being the first resource affecting the sub-set of the second resource from which perform the selection. In addition, due to different order of resource size, also the basic policy used for selecting resource within each set is of relevant importance.

Acknowledgment

This work was written within the framework of the project EMOTICON (Extending OpenFlow for Unified Management and Control of Cloud Data Center Resources) approved within the 2nd Call for additional beneficiaries of the OFELIA project.

REFERENCES

[1] C. N. Hofer, G. Karagiannis, *Cloud computing services: taxonomy and comparison*, Journal of Internet Services and Applications, Springer, Vol. 2, No. 2, pp. 69-79, September 2011.

[2] Yueping Zhang; Ao-Jan Su; Guofei Jiang; , "Evaluating the impact of data center network architectures on application performance in virtualized environments," *Quality of Service (IWQoS), 2010 18th International Workshop on* , vol., no., pp.1-5, 16-18 June 2010

[3] OFELIA EU Project Home Page, http://www.fp7-ofelia.eu

[4] Openflow, http://www.openflow.org

[5] VMware Capacity Planner, http://www.vmware.com/products/capacity-planner/

[6] S. Metha and A. NEogi, "Recon: a tool to recommend dynamic server consolidation in multi-cluster data centers,", NOMS 2008

[7] IBM Workoload Deployerr Homne Page http://www-01.ibm.com/software/webservers/cloudburst/

[8] J. Shahabuddin, A. Chrungoo, V. Gupta, S. Juneja, S. Kapoor, and A. Kumar, "Stream-packing: Resource allocation in web server farms with a QoS guarantee", HiPC, 2001

[9] Y. Ajiro and A. Tanaka, "Improving packing algorithms for server consolidation", in *Proceedings of the International Conference for the Computer Measurement Group* (CMG), 2007

[10] Sanghwan Lee; Sahu, S.; , "Efficient Server Consolidation Considering Intra-Cluster Traffic," *Global Telecommunications Conference (GLOBECOM 2011), 2011 IEEE* , vol., no., pp.1-6, 5-9 Dec. 201

[11] Xiaoqiao Meng; Pappas, V.; Li Zhang; , "Improving the Scalability of Data Center Networks with Traffic-aware Virtual Machine Placement," *INFOCOM, 2010 Proceedings IEEE* , vol., no., pp.1-9, 14-19 March 2010

[12] Jayasinghe, D.; Pu, C.; Eilam, T.; Steinder, M.; Whally, I.; Snible, E.; , "Improving Performance and Availability of Services Hosted on IaaS Clouds with Structural Constraint-Aware Virtual Machine Placement,"*Services Computing (SCC), 2011 IEEE International Conference on* , vol., no., pp.72-79, 4-9 July 2011

[13] Mihailescu, M.; Rodriguez, A.; Amza, C.; , "Enhancing application robustness in Infrastructure-as-a-Service clouds," *Dependable Systems and Networks Workshops (DSN-W), 2011 IEEE/IFIP 41st International Conference on*, vol., no., pp.146-151, 27-30 June 2011

[14] Theophilus Benson, Ashok Anand, Aditya Akella and Ming Zhang "MicroTE: Fine grained Traffic Engineering for Data Centers", CoNEXT 2011, Tokyo, Japan.

[15] Rastin Pries, Michael Jarschel, Sebastian Goll, "On the Usability of OpenFlow in Data Center Environments", Workshop on Clouds, Networks and Data Centers collocated with the IEEE International Conference on Communications (ICC 2012), Ottawa, Canada, June 2012

[16] Cisco Data Center Interconnect Design and Deployment Guide. Cisco Press, 2010

[17] A. Greenberg, P. Lahiri, D. A. Maltz, P. Patel, S. Sengupta "Towards a next generation DC architecture: scalability and commoditization", in Proc. of PRESTO '08, ACM SIGCOMM Workshop, August 22, 2008 Seattle, Washington, USA.