Cerroni *Journal of Internet Services and Applications* (2015) 6:6
DOI 10.1186/s13174-015-0020-x

Journal of Internet Services and Applications
a SpringerOpen Journal

RESEARCH Open Access

Network performance of multiple virtual machine live migration in cloud federations

Walter Cerroni

Abstract

The idea of pay-per-use computing incarnated by the cloud paradigm is gaining a lot of success, both for entertainment and business applications. As a consequence, the demand for computing, storage and communication resources to be deployed in data center infrastructures is increasing dramatically. This trend is fostering new forms of infrastructure sharing such as cloud federations, where the excess workload is smartly distributed across multiple data centers, following some kind of mutual agreement among the participating cloud providers. Federated clouds can obtain great advantages from virtualization technologies and, in particular, from multiple virtual machine live migration techniques, which allow to flexibly move bulk workload across heterogeneous computing environments with minimal service disruption. However, a quantitative characterization of the performance of the inter-data center network infrastructure underlying the cloud federation is essential to guarantee user's quality of service and optimize provider's resource utilization. The main contribution of this paper is the definition and application of an analytical model for dimensioning inter-data center network capacity in order to achieve some given performance levels, assuming some simple multiple virtual machine live migration strategies. An extensive set of results are provided that allow to understand the impact of the many parameters involved in the design of a cloud federation network.

Keywords: Cloud computing; Cross-cloud communication; Inter-data center communication; Virtualization; Virtual machine live migration

1 Introduction

Software applications based on the cloud computing paradigm have become very popular in the last few years, both for entertainment and business purposes, and an increasing number of new services—including entire virtual IT infrastructures—are today considered as part of "the Cloud" [1]. Such an idea of the cloud as a ubiquitous computing utility has become a reality owing to recent advances in data center (DC) technologies. However, in order to cope with the exponentially increasing number of cloud service subscribers—especially mobile cloud users—more advanced networking infrastructures and technologies are expected to be deployed for both intra-DC and inter-DC communications [2].

Over-provisioning DC processing power may not always be the right answer, as increasing the size of a DC can result in very expensive and energy-demanding operations. For this reason, the emerging *federated cloud computing* model adopts the idea of smartly sharing the workload across the DC resources of multiple cloud providers, following some kind of mutual agreement [3,4]. However, in order for cloud federations to become current practice, several issues still remain to be solved, among which the correct design of the inter-DC interconnection network by means of suitable communication infrastructure planning to achieve the required level of quality of service (QoS) [5].

The use of virtual machines (VMs) to implement end-user services is one of the key enablers of cloud federations. In fact, decoupling service instances from the underlying processing and storage hardware allows to flexibly deploy any application on any server within any DC, independently of the specific operating system used. One of the main advantages is that a VM can be instantiated, cloned, migrated, rolled-back to a previous state without expensive hardware interventions. This is particularly useful in a cloud federation, where VMs can be easily moved from one DC to another as long as hypervisor compatibility is guaranteed. Live VM migration is an additional feature that allows to move services from

Correspondence: walter.cerroni@unibo.it
Department of Electrical, Electronic and Information Engineering, University of Bologna, via Venezia 52, 47521 Cesena (FC), Italy

one server/DC to another with minimal disruption to the end-user service availability [6].

Migrating a running VM to a different DC requires to maintain the guest's network state consistency. Emerging technologies, such as software defined networking (SDN) [7], offer new opportunities to seamlessly migrate virtualized environments and their current network states [8]. This is particularly useful when considering groups of correlated VMs that must be live-migrated together while maintaining reciprocal connectivity. In fact, many multi-tier applications are often executed across multiple VMs [9] (e.g., front-end, business logic and back-end tiers of e-commerce services, or clustered MapReduce computing environments) and the relative services are available to the end-user only when all VMs in the group are active and connected to each other.

The main contribution of this paper is the definition and application of an analytical model to assess the network performance of a federated cloud, specifically assuming some simple multi-VM live migration strategies. The proposed model should be intended as a useful design tool to dimension inter-DC network capacity in order to achieve some given performance levels in a cloud federation, taking into account both the cloud provider's and end-user's points of view. This paper extends and generalizes a previously published, simpler version of the model [10], and provides an extensive set of results that allow to characterize the impact of the many parameters involved in the design of a cloud federation network.

The paper is organized as follows. After a brief look into related work in Section 2, the problem statement and the federated cloud network scenario considered in the model are introduced in Section 3. Then, the main parameters of interest for multiple VM live migration are discussed in Section 4, whereas the proposed Markovian model of the inter-DC network is presented in Section 5. After reporting extensive numerical results in Section 6, the role of the different cloud federation design parameters is discussed in Section 7. Finally, conclusions are drawn in Section 8.

2 Related work

One of the most relevant advantages of cloud federations is that they act as a single overlay entity across the participating networks and DCs, thus boosting the capacity of what can be seen by the end-user as a "virtual data center". However, the distributed nature of these cross-cloud infrastructures makes the network a very critical element for an effective deployment and management of cloud federations, confirming the importance of being able to assess and control the performance of inter-cloud communication resources [11].

The joint effect of computing and network resource availability in distributed cloud systems has been investigated mainly from an optimization perspective. The importance of revisiting the algorithms for dynamically mapping user-driven virtual resources into physical resources within distributed clouds is highlighted in [12], where an optimal unified resource allocation framework for improving cost efficiency of networked clouds is formulated, and efficient heuristics to solve the problem are proposed.

Other works deal with the virtual DC optimization problem, where a set of interconnected VMs must be provisioned to the end-user according to a given service level agreement that specifies not only computing and storage requirements, but also bandwidth requirements. Virtual DCs must then be mapped to physical resources following efficient and bandwidth-aware heuristics, such as the one implemented by SecondNet [13]. More recent algorithms and frameworks, such as AppAware [14] and VDC Planner [15], take into account also the effects of VM migration while performing dynamic virtual DC network embedding and consolidation.

A holistic solution to the issue of large-scale, distributed cloud system design is proposed in [16]: the authors formalize an optimal VM placement strategy aimed at minimizing both intra-DC resource demand and inter-DC energy consumption. Other related cross-cloud network optimization works include a resilient optical inter-DC infrastructure planning scheme based on dynamic electricity pricing [17] and a framework for joint computing and communication resource allocation targeted at satisfying green service level agreements [18]. Differently from the aforementioned approaches, the analytic evaluation of cloud federation network performance presented here is not based on linear programming optimization models or related heuristics; instead, it relies on relatively simple closed-form formulas that are more straightforward to compute.

A completely different approach is followed in [19], where an analytical model based on stochastic reward nets is proposed to evaluate the performance of infrastructure-as-a-service cloud systems. The model is scalable and can cope with thousands of resources. It is also flexible to represent different resource management policies and cloud-specific strategies. However, although several performance metrics are defined and evaluated to analyze the behavior of a cloud DC, this approach does not provide an effective tool to dimension the inter-cloud network considering both communication and computing resource availability.

The issue of VM live migration has been extensively studied in literature and successfully solved and optimized in commercially available hypervisors. Of course, any live migration procedure must ensure consistency in the VM memory, storage, and network states before and after the transfer to the new hosting server. The most typical solution for live memory migration is the so-called *pre-copy*

strategy [6], which is currently adopted by many virtualization systems such as Xen and KVM [20,21]. With pre-copy, the virtual memory is repeatedly copied to the destination while the VM is still running at the source, until the residual number of modified memory pages is small enough to ensure very quick VM pause, transfer and resume. The opposite approach is adopted by the *post-copy* migration technique [22]: the VM is immediately paused, a minimal processor state is copied, and the VM is quickly resumed at the destination; then, any memory page needed by the running applications is pulled from the source. Post-copy is able to reduce the migration time with respect to pre-copy, as each memory page is transferred only once. However, in case of failure at the destination, the VM state may become unrecoverable.

The file system state consistency during live migration within a local cloud environment is typically ensured by adopting well-established shared storage solutions, such as Network Attached Storage (NAS) and Storage Area Network (SAN). In this case the migrating VMs are attached to the same file system, available at both source and destination hosts, so that there is no need to copy disk images. Different is the case of live migration to a remote DC: the storage located at the destination must be synchronized with the one at the source, and this may require to perform large data transfers—in the worst case to copy an entire VM disk image. An efficient solution consists in: (i) executing the bulk storage data transfer before launching the actual VM live migration, (ii) recording all write operations happening during the transfer, and (iii) applying the changes at the destination when the VM is being migrated [23,24]. Template disk images and write throttling mechanisms allow to reduce the amount of bulk storage data to be transferred and the number of changes to be applied, respectively.

The network state consistency issue is easily solved in a local cloud environment: in fact, since each VM is connected via a virtual bridge to the same physical LAN at both source and destination hosts, it will keep the same IP address and, when the execution is resumed at the destination, a gratuitous ARP packet is sufficient to make all switches and neighbors aware of the new VM location. More complex is the case of migrating a VM to a remote DC, because ongoing connections need to be rerouted. Indeed, when the relocated VM is hosted in a DC connected to a different IP network, or even within a different domain, some kind of IP mobility solution must be adopted, such as those based on the so-called identifier/locator split principle [25,26]. Another possibility is to take advantage of the highly flexible, dynamic network reconfiguration capabilities of SDN, which allows to migrate an entire virtual network from one DC to another [27], and to smartly reroute external traffic after a VM has been migrated [28].

While most of the existing literature focuses on the single VM migration, the case of migrating multiple correlated VMs is still to be investigated in detail. Some studies have been carried out to understand the implications of live-migrating a group of VMs together with the virtual network interconnecting them [27], whereas other works focused on different optimization aspects [29-31]. A starting point for a quantitative analysis of the performance of multiple VM live migration assuming some simple scheduling strategies can be found in [32,33].

The analytic inter-DC network performance evaluation, with particular emphasis on communication resources consumed by multi-VM live migration, deserves specific attention in order to understand if and how a cloud federation can be designed and dimensioned in an efficient way. To the best of the author's knowledge, this topic has not been previously investigated in detail, apart from the early work in [10] which is generalized and extended here.

3 Federated cloud network scenario

The general problem addressed in this paper is how to quantify the effects of the main design parameters in a cloud federation on the performance of VM live migration procedures. The design parameters include the size of the federation, the amount of communication and computing resources, and how they are allocated within the federation. The migration performance is assessed from both the end-user's perspective, in terms of impact of the migration on the availability of cloud services, and the cloud provider's point of view, considering resource utilization and availability of the migration service itself. If an analytical model that can reasonably capture all these aspects is developed, then it can be used to properly choose the aforementioned design parameters and dimension the cloud federation network.

The federated cloud network scenario considered here consists of $n + 1$ DCs interconnected by a full mesh of guaranteed-bandwidth network pipes. Figure 1 shows the assumed connectivity model from the the point of view of a given DC (local DC), which can reach the n remote DCs via as many established network pipes $L_1, L_2, \ldots, L_n$. Such network pipes could be implemented as MPLS label-switched paths (LSPs) or as lightpaths established between the edge nodes of an inter-DC optical network. Those LSPs or lightpaths are assumed to be established according to a long-term network resource planning strategy (e.g., by means of well-known routing and wavelength assignment techniques [34]). It is reasonable to assume that some QoS requirements (e.g., minimum bandwidth) must be guaranteed within a cloud federation network, to be able to control the performance of the inter-cloud communications.

Let us consider the case of a set of correlated VMs, currently running in the local DC, that must be migrated,

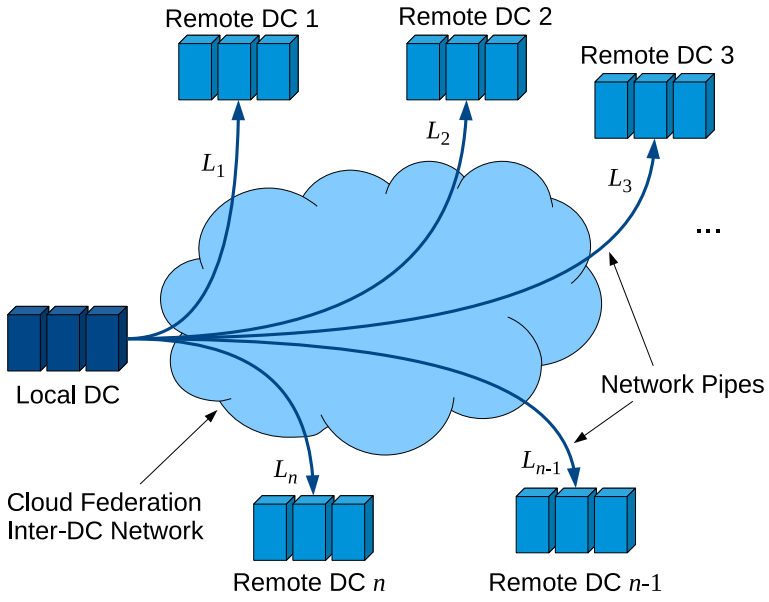

Figure 1 Cloud federation network connectivity model, as seen by a given traffic source (local DC).

e.g. for load balancing, energy saving, server consolidation or DC maintenance reasons. A *migration request* is then sent to the *federated cloud management system*, which is in charge of finding a suitable DC where the VMs can be hosted. In principle, any DC within the cloud federation could host the moving VMs. However, it would be more realistic to assume that only a subset of the DCs are actually able to receive the workload of the set of VMs to be migrated. This can be true for a number of reasons: for instance, not all DCs may provide the specific computing or storage resources required by the given VM set; or maybe the services implemented by the VMs have some latency requirements that cannot be satisfied if the VMs are migrated to a DC located too far from their final users; also, not all DCs in a cloud federation are equivalent in terms of energy savings or maintenance schedules; last but not least, load balancing reasons may force to choose some DCs instead of others. These limitations are caused by situations that can be either permanent (e.g., the nature, size, and location of a DC) or contingent (e.g., maintenance or load balancing schedules within the federation), but in any case known to the federated cloud management system, which is thus able to identify, for each migration request, the subset of DCs in the federation that are suitable to satisfy it.

More formally, in this work it is assumed that the generic request z of migrating a group of M_z VMs can be satisfied by the set of resources C_z available in a subset of the n remote DCs, as illustrated in Figure 2. In general, different requests may need different resource sets. It is assumed that the m_z resource set instances $C_z^{(1)}, C_z^{(2)}, \ldots, C_z^{(m_z)}$ available in the cloud federation are randomly distributed over the n remote DCs. Any of the m_z resource set instances is equivalent for hosting the VMs, according to a general *anycast* service model.

The federated cloud management system associates each migration request z to the resource set C_z by means of a sort of anycast address, defined internally in the federation and used to identify the relevant subset of suitable DCs, that is then translated into:

- the set of network pipes between the local DC and any possible location of C_z instances, i.e. $\{L_2, L_3, L_n\}$ in the example of Figure 2;
- the minimum amount of network pipe capacity b_z that must be guaranteed to migrate the whole group of M_z VMs, according to the live migration performance model presented in the next section.

Then the management system is in charge of finding the location of the most suitable instance of C_z, namely $C_z^{(x)}$. The choice is made based on the availability of the required capacity b_z towards the location of $C_z^{(x)}$. More specifically, let B_i and $B_{\mathrm{a},i}$ respectively denote the total and currently available capacity of network pipe L_i, $i = 1, 2, \ldots, n$. The capacity currently used by other data transfers between the local DC and the i-th remote DC is $B_i - B_{\mathrm{a},i}$. Any network pipe towards a remote DC hosting at least one of the m_z resource set instances and such that $B_{\mathrm{a},i} \geq b_z$ is considered equivalent in the model and can be used to migrate the M_z VMs. A migration request z is blocked when there is not enough capacity available

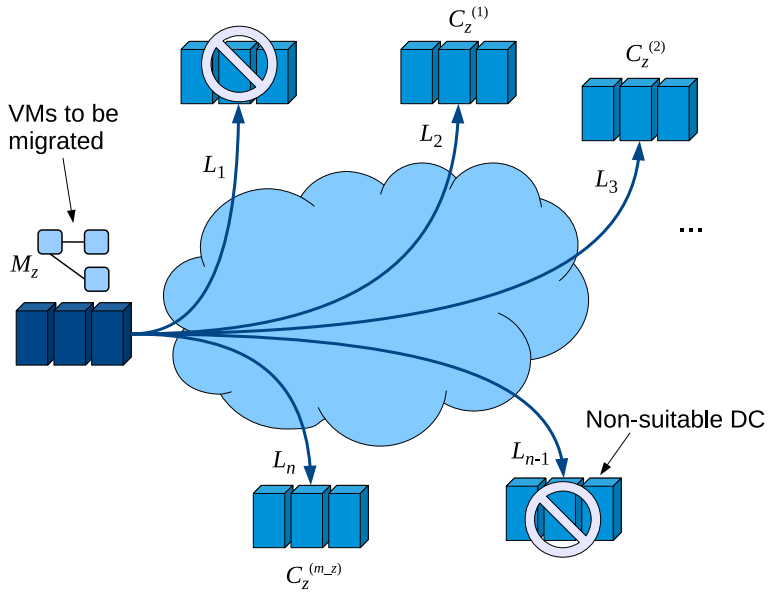

Figure 2 Example of resource availability in a cloud federation when M_z VMs are scheduled for migration.

between the local DC and any remote DC where the instances of resource set C_z are located.

As an example, consider the case when $n = 5$ and $B_i = 4$ Gbps, $\forall i = 1, \ldots, 5$. Assume that request $z = j$ asks to migrate $M_j = 2$ VMs with a guaranteed bit rate $b_j = 1$ Gbps. Suppose that, when the request arrives, it finds the available network pipe capacity as expressed by vector $\mathbf{B}_a = [B_{a,1}, B_{a,2}, B_{a,3}, B_{a,4}, B_{a,5}] = [2, 1, 0, 4, 2]$ Gbps, and only $m_j = 3$ instances of resource set C_j are available, e.g. located in remote DCs 1, 3, and 5. There is enough available capacity in network pipes L_1 and L_5, so the group of VMs can be migrated to either remote DC 1 or remote DC 5. If the latter one is chosen and none of the ongoing transfers is completed when the next request arrives, then the available capacity vector becomes $\mathbf{B}_a = [2, 1, 0, 4, 1]$ Gbps. If request $z = j + 1$ needs to migrate a larger number of VMs, e.g. $M_{j+1} = 5$, it can happen that the number of suitable DCs is smaller, e.g. only $m_{j+1} = 2$ instances of resource set C_{j+1} are available, located in remote DCs 2 and 5. Then, if the bit rate requirement is also higher, e.g. $b_{j+1} = 2$ Gbps, none of the network pipes L_2 and L_5 towards the suitable remote DCs have enough capacity left, and the migration request is blocked.

The previous example shows that a request may be blocked because of lack of communication resources even when the total network capacity is not fully consumed. The occurrence of this kind of blocking situations depends on the availability of proper computing and storage resources in the remote DCs, or, in other words, it depends on which remote DCs are part of the anycast group associated to a given request. Of course it could be argued that, if request $z = j + 1$ reduces its bit rate requirement to $b_{j+1} = 1$ Gbps, then the migration can be performed over either L_2 or L_5: however, this choice could negatively affect the service offered to the VM users, since the reduced transfer rate can have a significant impact on the live migration performance, as discussed in the next section. It could also be argued that, by choosing network pipe L_1 instead of L_5 to serve request $z = j$, the capacity left on L_5 would be enough to serve request $z = j + 1$: however, this implies the capability of the federated cloud management system to know future requests in advance, which could be possible only if all the VM migrations are already planned and pre-scheduled. This would be a case of static load patterns, for which an optimization model such as the one in [12] would be a better choice, but it is out of the scope of this paper.

The purpose of the model described in the next sections is to evaluate the performance, in terms of migration request blocking probability, of the aforementioned federated cloud network scenario and to provide a useful design tool for network pipe dimensioning, considering dynamic network load patterns generated by the specific nature of VM live migration traffic. In order to keep the model simple enough to be tractable and understand the role of the different design parameters, the following homogeneity assumptions are made:

A.1 each multi-VM migration request z needs the same amount of guaranteed transfer bit rate, i.e. $b_z = b \ \forall z$;

A.2 each network pipe provides the same total capacity, i.e. $B_i = B$, $\forall i = 1, \ldots, n$;

A.3 each remote DC has the capability of hosting up to k resource set instances;

A.4 each migration request finds the same number of instances of resource set C_z, i.e. $m_z = m$, which are uniformly distributed over the n remote DCs (considering the general case when multiple instances of the same resource set could be available in the same DC).

The migration request blocking model is obtained in two steps. First, an exact formulation of the multiple VM migration time is derived and used to compute for how long each migration request consumes the guaranteed amount of network capacity (Section 4). Then, the previous result is applied to characterize the average network pipe capacity occupancy time in a Markov chain that approximates the state evolution of the inter-DC network and allows to compute the migration request blocking probability (Section 5).

4 Modeling multiple VM migration

The main advantage of *live migration*, i.e. of moving the VM from one hosting server to another while it is still running, is that the current state of VM kernel and running processes is maintained and the migration procedure has minimum impact on the end-user service availability. It also reduces the risk of inconsistencies due to duplicate VM instances running simultaneously in both the source and destination DCs. This work is focused mainly on the memory migration issue, assuming the pre-copy strategy [6].

As illustrated in Figure 3, the migration starts with an *iterative push phase*: a first snapshot of the VM memory is transferred, while the VM is still running; during the transfer, some memory pages can still be modified by the running processes. Therefore, "dirty" memory pages (i.e., pages modified during a given transfer) are transmitted again during the next round, until the total size of dirty pages is below a given threshold, or a maximum number of iterations is reached. After that, the *stop-and-copy phase* takes place: the VM is suspended at the source host and the remaining dirty pages are copied to the destination. Finally, during the *resume phase* the VM is brought back on-line at the destination host with consistent memory and network states. Network state can be either migrated during the resume phase or cloned at the beginning of the push phase, as suggested in [27]. The former solution is more straightforward, whereas the latter seems to be more efficient in terms of latency and network load.

The two key performance parameters that are typically considered in the single VM migration process are the so-called *downtime* (T_{down}) and *total migration time* (T_{mig}). The former is defined as the amount of time the VM is paused during the transfer and it measures the impact of the migration on the end-user's perceived quality of the service offered by the VM. Keeping the downtime as small as possible helps to make the migration process look transparent to the end-user, even for time-critical services such as audio/video streaming and on-line gaming. On the other hand, the total migration time is also very important because it measures the impact of the migration process on the cloud federation resources: in fact, the network pipe capacity consumed for transferring the VM as well as the computing resources dedicated to the VM in both source and destination DCs are occupied during the whole migration phase and cannot be used to perform other tasks.

A simple model used to evaluate T_{down} and T_{mig} in case of a single VM migrated with the pre-copy strategy was proposed in [35]. Based on this model, two generalized extensions to the case of multiple VM migration were presented in [32,33]. In this case, the definition of

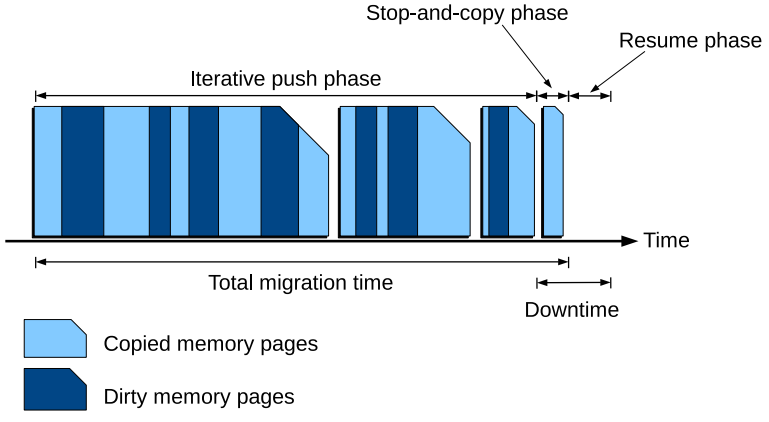

Figure 3 Phases of the pre-copy live migration strategy and related timings.

the performance parameters depends on how the multiple VMs are scheduled for the live transfer and on their mutual interactions when providing services to the end-user. The formulation of the multiple VM migration model is briefly recalled here, under the following assumptions:

A.5 the M_z VMs to be migrated as per request z are allotted the same amount of memory V_z;
A.6 the applications running on the VMs show the same constant memory page dirtying rate D;
A.7 all VMs have the same memory page size P;
A.8 the bit rate $R_j^{(z)}$ used to transfer the j-th VM in the requested set is constant during the whole migration process; this is the amount of network pipe capacity dedicated to transfer VM j, i.e. $R_j^{(z)} \leq b,\ \forall j = 1, \ldots, M_z$.

Assumptions **A.5** and **A.6** may not be completely true in a real-word scenario, since the memory profile of each VM strongly depends on the specific applications that are being executed. However, these assumptions allow to simplify the equations and to capture the macroscopic performance aspects of multiple VM live migration. Anyways, the case of VMs with different memory sizes in the same migration group can be studied by extending the migration model as in [33].

Let $T_{i,j}^{(z)}$ be the time needed to complete the i-th iteration in the push phase of VM j migration as per request z. From the general equations derived in [32] that describe the iterative migration process of each VM $j = 1, \ldots, M_z$, it is possible to compute the time needed to migrate the j-th VM and the number of iterations required as:

$$T_{\mathrm{mig},j}^{(z)} = \sum_{i=0}^{n_j^{(z)}} T_{i,j}^{(z)} = \sum_{i=0}^{n_j^{(z)}} V_z \frac{(PD)^i}{\left(R_j^{(z)}\right)^{i+1}} = \frac{V_z}{R_j^{(z)}} \frac{1 - \left(\gamma_j^{(z)}\right)^{n_j^{(z)}+1}}{1 - \gamma_j^{(z)}} \tag{1}$$

$$n_j^{(z)} = \min\left\{ \left\lceil \log_{\gamma_j^{(z)}} \frac{V_{\mathrm{th}}}{V_z} \right\rceil, n_{\max} \right\} \tag{2}$$

where V_{th} is the dirty memory size threshold and $n_{\max}$ is the maximum number of iterations that trigger the stop-and-copy phase, whereas $\gamma_j^{(z)} = (PD)/R_j^{(z)}$ must always be lesser than 1, because the pre-copy migration algorithm is sustainable as long as the average memory dirtying rate is smaller than the transfer rate.

The computation of the total migration time and downtime of a set of VMs strictly depends on how many of them are simultaneously transferred. In the following, two simple cases are considered: (i) when the M_z VMs are transferred one at a time (*sequential migration*); (ii) when all the M_z VMs are simultaneously transferred (*parallel migration*). A useful system parameter is the ratio of the dirtying rate to the maximum transfer rate $\gamma = (PD)/b$.

When the M_z VMs are migrated one at a time, each transfer is performed at full rate, i.e. $R_j^{(z)} = b,\ \forall j$. In this case, $\gamma_j^{(z)} = \gamma,\ \forall j$ and the sequential migration time of the whole VM set z is given by

$$T_{\mathrm{s-mig}}^{(z)} = \sum_{j=1}^{M_z} T_{\mathrm{mig},j}^{(z)} = M_z \frac{V_z}{b} \frac{1 - \gamma^{n_{\mathrm{s}}^{(z)}+1}}{1 - \gamma} \tag{3}$$

where $n_{\mathrm{s}}^{(z)} = \min\left\{ \left\lceil \log_{\gamma} V_{\mathrm{th}}/V_z \right\rceil, n_{\max} \right\}$.

As illustrated in Figure 4, the downtime of the whole VM set starts when the first VM is stopped at the source host (i.e., when the last iteration of the first VM begins) and ends when the last VM is resumed at the destination host. If T_{res} is the fixed time required to resume a VM at the destination host, the sequential migration downtime can be computed as

$$T_{\mathrm{s-down}}^{(z)} = \frac{V_z}{b} \gamma^{n_{\mathrm{s}}^{(z)}} + (M_z - 1) \frac{V_z}{b} \frac{1 - \gamma^{n_{\mathrm{s}}^{(z)}+1}}{1 - \gamma} + T_{\mathrm{res}} \tag{4}$$

When the M_z VMs are migrated simultaneously, each one of them is transferred at a bit rate that depends on how the requested capacity b is shared among the on-going connections. Assuming an equal share of the channel capacity and considering that all the VMs in set z have the same memory profile, all VMs start and end their iterations at the same instants. Therefore, there are always M_z simultaneous transfers and the transfer rate used for each VM is $R_j^{(z)} = b/M_z,\ \forall j$. In this case, $\gamma_j^{(z)} = M_z\gamma,\ \forall j$ and, as shown in Figure 4, the parallel migration time of the whole VM set is equivalent to the migration time of any single VM, given by

$$T_{\mathrm{p-mig}}^{(z)} = T_{\mathrm{mig},j}^{(z)} = M_z \frac{V_z}{b} \frac{1 - (M_z\gamma)^{n_{\mathrm{p}}^{(z)}+1}}{1 - M_z\gamma} \tag{5}$$

where $n_{\mathrm{p}}^{(z)} = \min\left\{ \left\lceil \log_{M_z\gamma} V_{\mathrm{th}}/V_z \right\rceil, n_{\max} \right\}$.

The parallel migration downtime of the whole VM set corresponds to the last iteration (stop-and-copy phase) of any single VM and is given by

$$T_{\mathrm{p-down}}^{(z)} = M_z \frac{V_z}{b} (M_z\gamma)^{n_{\mathrm{p}}^{(z)}} + T_{\mathrm{res}} \tag{6}$$

As proved in [32], $T_{\mathrm{p-mig}}^{(z)} \geq T_{\mathrm{s-mig}}^{(z)}$ and $T_{\mathrm{p-down}}^{(z)} \leq T_{\mathrm{s-down}}^{(z)}$, meaning that, while parallel migration is better than sequential migration in terms of end-user service provisioning because the downtime is smaller, sequential migration shows a smaller total transfer time and thus is better than parallel migration in terms of communication and computing resource usage and transmission overhead.

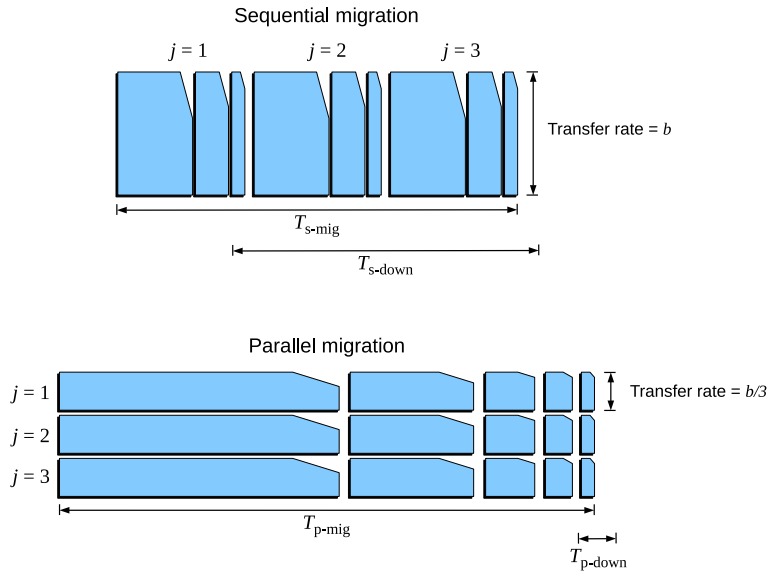

Figure 4 Different definitions of total migration time and downtime for sequential and parallel migration of $M_z = 3$ VMs.

Besides migration time and downtime, other parameters have been defined to quantify the overall *cost of VM migration* [36]. One is the VM performance loss, taking into account the overhead of monitoring memory write-access operations during the push phase: this can significantly slow down the execution of processes running inside the VM and reduce their throughput. Another important cost factor is the energy overhead, i.e. the additional amount of energy consumed by the computing and communication resources involved in the VM live migration. A different approach is followed here, taking advantage of the quantities obtained from the multi-VM migration model. The idea is to quantify the migration cost of a given set of VMs by measuring two factors: (i) the volume of data transferred on the network pipe during the migration, representing the communication resource overhead; (ii) the volume of data that the applications running inside the VMs are not able to exchange with the external end-users during the downtime, representing the VM processing performance degradation.

The former cost component can be evaluated as $T^{(z)}_{\text{s}-\text{mig}}\, b$ and $T^{(z)}_{\text{p}-\text{mig}}\, b$ for the sequential and parallel migration, respectively, whereas the latter one can be obtained assuming the application throughput to be equal to the transfer rate, i.e. $T^{(z)}_{\text{s}-\text{down}}\, b$ and $T^{(z)}_{\text{p}-\text{down}}\, b$. Then, in order to come up with two cost contributions with comparable values, each product is normalized to its maximum value, which is obtained when γ approaches its upper bound, i.e.

$$\lim_{\gamma \to 1} T^{(z)}_{\text{s}-\text{mig}}\, b = M_z\, V_z \lim_{\gamma \to 1} \frac{1-\gamma^{n_{\text{s}}^{(z)}+1}}{1-\gamma} = M_z\, V_z\, (n_{\max}+1)$$

$$\lim_{\gamma \to \frac{1}{M_z}} T^{(z)}_{\text{p}-\text{mig}}\, b = M_z\, V_z \lim_{\gamma \to \frac{1}{M_z}} \frac{1-(M_z\gamma)^{n_{\text{p}}^{(z)}+1}}{1-M_z\gamma} = M_z\, V_z\, (n_{\max}+1)$$

$$\lim_{\gamma \to 1} T^{(z)}_{\text{s}-\text{down}}\, b = V_z\, (1+(M_z-1)\,(n_{\max}+1)) + T_{\text{res}}\, b$$

$$\lim_{\gamma \to \frac{1}{M_z}} T^{(z)}_{\text{p}-\text{down}}\, b = M_z\, V_z + T_{\text{res}}\, b$$

Therefore, the multi-VM migration cost factors related to network overhead and application throughput degradation as per request z for sequential and parallel migration can be written as

$$c^{(z)}_{\text{s}-\text{net}} = \frac{1-\gamma^{n_{\text{s}}^{(z)}+1}}{(n_{\max}+1)\,(1-\gamma)} \tag{7}$$

$$c^{(z)}_{\text{p}-\text{net}} = \frac{1-(M_z\gamma)^{n_{\text{p}}^{(z)}+1}}{(n_{\max}+1)\,(1-M_z\gamma)} \tag{8}$$

$$c^{(z)}_{\text{s}-\text{app}} = \frac{V_z\left(\gamma^{n_{\text{s}}^{(z)}} + (M_z-1)\,\frac{1-\gamma^{n_{\text{s}}^{(z)}+1}}{1-\gamma}\right) + T_{\text{res}}\, b}{V_z\,(1+(M_z-1)\,(n_{\max}+1)) + T_{\text{res}}\, b} \tag{9}$$

$$c^{(z)}_{\text{p}-\text{app}} = \frac{M_z\, V_z\, (M_z\gamma)^{n_{\text{p}}^{(z)}} + T_{\text{res}}\, b}{M_z\, V_z + T_{\text{res}}\, b} \tag{10}$$

and a weighted average of the two factors can be computed as

$$c_{\mathrm{s}}^{(z)} = \alpha\, c_{\mathrm{s-net}}^{(z)} + (1-\alpha)\, c_{\mathrm{s-app}}^{(z)} \tag{11}$$

$$c_{\mathrm{p}}^{(z)} = \alpha\, c_{\mathrm{p-net}}^{(z)} + (1-\alpha)\, c_{\mathrm{p-app}}^{(z)} \tag{12}$$

where $0 \leq \alpha \leq 1$.

5 Markovian model of inter-DC network

According to the federated cloud network scenario described in section 3, migration request z is refused when other ongoing transfers consume all the capacity on network pipes reaching the remote DCs where the m instances of resource set C_z are located. In order to compute the migration request blocking probability, it is necessary to determine which states of the inter-DC network are such that all the resource set instances $C_z^{(1)}, C_z^{(2)}, \ldots, C_z^{(m)}$ are located inside unreachable DCs. To this purpose, the state of the network is defined as the number r of ongoing VM group migrations originated at the local DC. From assumptions **A.1** and **A.2**, B is the total capacity of each of the n network pipes connecting the local DC to the remote ones, and b is the capacity consumed by each group migration. Then, each network pipe is able to carry at most $h = \lfloor B/b \rfloor$ simultaneous group migrations, and the maximum total number of simultaneous transfers originating from the local DC is nh. Therefore, the state space of the network, as seen by the local DC, is $r \in \{0, 1, \ldots, nh\}$.

Assuming that the multi-VM migration requests follow a Poisson arrival process, a Markov chain describing the network state evolution can be defined using the following parameters:

- λ: migration request arrival rate;
- μ: migration service rate;
- $A_0 = \lambda/\mu$: load offered to the inter-DC network as seen by the local DC.

Since this Markov chain represents a system without a waiting line, it can be classified as a loss system and its solution is valid for any service time distribution with a finite mean [37].

To obtain the service rate, the reciprocal of the average total migration time must be computed. Therefore, we can define two possible values of μ, depending on the multi-VM migration strategy adopted (either sequential or parallel):

$$\mu_{\mathrm{s}} = \frac{1}{E\left[T_{\mathrm{s-mig}}^{(z)}\right]} \quad \text{or} \quad \mu_{\mathrm{p}} = \frac{1}{E\left[T_{\mathrm{p-mig}}^{(z)}\right]}$$

where the migration time must be averaged over all migration requests z according to the statistical distribution of V_z and M_z.

The inter-DC network, as seen by the local DC, is therefore modeled as a loss system where the number of servers is equal to nh and migration request blocking events may occur in a generic state r, depending on the requested resource set instance locations. In fact, the transition to state $r+1$ occurs only when at least one of the m requested resource set instances is placed in a DC reachable via a network pipe i with available capacity $B_{\mathrm{a},i} \geq b$, otherwise the request is blocked. Let $P_{\mathrm{bl}|r}$ be the blocking probability in state r. Transitions from state r to state $r+1$ occur with rate $\lambda_r = (1 - P_{\mathrm{bl}|r})\,\lambda$, whereas transitions from state r to state $r-1$ occur with rate $r\mu$.

To obtain $P_{\mathrm{bl}|r}$, the combinatorial behavior of the anycast approach must be analyzed first. Considering assumptions **A.3** and **A.4**, when the number of fully occupied network pipes is ℓ, the probability that the m resource set instances are located in the ℓ unreachable remote DCs, each capable of hosting up to k instances, is given by the probability of choosing m objects from a subset of ℓk elements out of nk possible choices, i.e.

$$p(m|\ell) = \prod_{i=0}^{m-1} \frac{\ell k - i}{nk - i} \quad \ell = 1, \ldots, n \tag{13}$$

Then, to correctly use formula (13), all the possible sub-states of state r must be considered, i.e. all the possible ways the r ongoing migrations can be distributed over any subset of the n remote DCs. This means finding all the possible partitions of number r into up to $\min\{n, r\}$ positive terms not greater than h. In principle, a more complex Markov chain including all sub-states of each state r should be solved in order to obtain the exact sub-state probabilities. However, this approach may become impractical, as the number of sub-states quickly becomes very large. Therefore, it was decided to approximate the sub-state probabilities considering only the "forward" evolution of the state, i.e. by recursively computing the sub-state probabilities of state r from those of state $r-1$.

As an example, consider the case when $n = 3$, $h = 3$ and $r \in \{0, 1, \ldots, 9\}$. Figure 5 shows the possible sub-states of all the states, and the probabilities of moving from a sub-state to another when a new request arrives, based on the location of the chosen instance. When the system is empty, i.e. $r = 0$, any incoming request is accepted and the state moves to $r = 1$, where only sub-state (1) is possible. Then, when a second request arrives, the chosen resource set instance can be located either in the same DC as the previous request, or in any of the other two DCs. The request is accepted and the system moves to either sub-state (2)—representing two migrations on the same network pipe—with probability 1/3, or to sub-state (1, 1)—representing two migrations on two different pipes—with probability 2/3. If $s(\varphi|r)$ denotes the probability of sub-state φ, given state r, the sub-state probabilities in the first three states are: $s(0|0) = 1$;

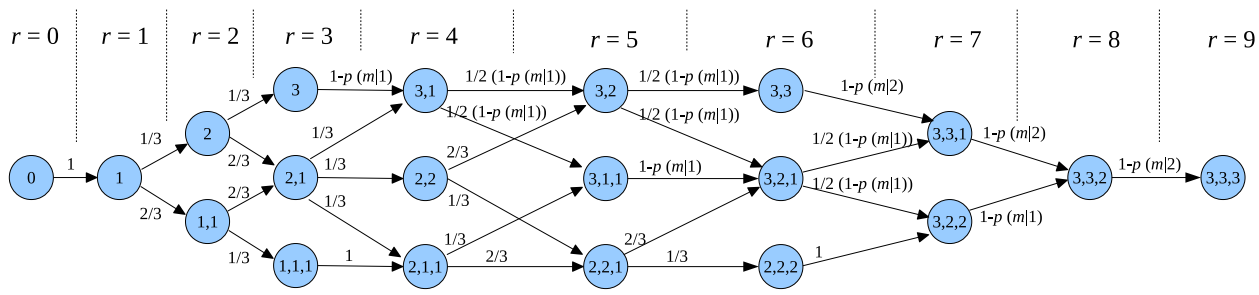

Figure 5 Possible sub-states of states $r = 0, 1, \ldots, 9$ and related "forward" transition probabilities, for the case $n = 3$ and $h = 3$.

$s(1|1) = 1$; $s(2|2) = 1/3$ and $s(1,1|2) = 2/3$. When moving further to state $r = 3$, the third migration request can bring the system to one of sub-states (3), $(2,1)$ or $(1,1,1)$, depending on the previous sub-state and the location of the chosen resource set instance. Therefore, the sub-state probabilities are $s(3|3) = 1/3\,s(2|2) = 1/9$, $s(2,1|3) = 2/3\,s(2|2) + 2/3\,s(1,1|2) = 2/3$, and $s(1,1,1|3) = 1/3\,s(1,1|2) = 2/9$. If the system is in sub-state (3), one of the remote DCs has a full network pipe and a new request can be blocked with probability $p(m|1)$, otherwise it moves forward to sub-state $(3,1)$. The chance of a missing transition due to request blocking must be taken into account when computing the sub-state probabilities for state $r = 4$, by normalizing to the sum of all the possible sub-state transitions, e.g.

$$s(3,1|4) = \frac{(1-p(m|1))\,s(3|3) + 1/3\,s(2,1|3)}{(1-p(m|1))\,s(3|3) + s(2,1|3) + s(1,1,1|3)}$$

$$s(2,2|4) = \frac{1/3\,s(2,1|3)}{(1-p(m|1))\,s(3|3) + s(2,1|3) + s(1,1,1|3)}$$

$$s(2,1,1|4) = \frac{1/3\,s(2,1|3) + s(1,1,1|3)}{(1-p(m|1))\,s(3|3) + s(2,1|3) + s(1,1,1|3)}$$

It is worth to mention that only partitions $(3,1)$, $(2,2)$ and $(2,1,1)$ of number 4 correspond to feasible sub-states of state $r = 4$. In fact, sub-state (4) can never occur because only up to $h = 3$ simultaneous transfers are possible on a given network pipe, whereas sub-state $(1,1,1,1)$ is impossible because there are only $n = 3$ remote DCs. With the help of the forward sub-state transition probabilities shown in Figure 5, the remaining sub-state probabilities can be computed in a similar way. This procedure can be generalized to a recursive sub-state probability computation algorithm that is relatively simple to execute.

Once all the sub-states probabilities are known, the blocking probability in a given state r can be obtained by averaging $p(m|\ell)$ over all sub-states such that at least one DC has a full network pipe, e.g. in the example of Figure 5:

$$\begin{aligned}
P_{\mathrm{bl}|3} &= p(m|1)\,s(3|3)\\
P_{\mathrm{bl}|4} &= p(m|1)\,s(3,1|4)\\
P_{\mathrm{bl}|5} &= p(m|1)\,s(3,2|5) + p(m|1)\,s(3,1,1|5)\\
P_{\mathrm{bl}|6} &= p(m|2)\,s(3,3|6) + p(m|1)\,s(3,2,1|6)\\
P_{\mathrm{bl}|7} &= p(m|2)\,s(3,3,1|7) + p(m|1)\,s(3,2,2|7)\\
P_{\mathrm{bl}|8} &= p(m|2)\,s(3,3,2|8)\\
P_{\mathrm{bl}|9} &= p(m|3)\,s(3,3,3|9) = 1
\end{aligned}$$

Then, solving the Markov chain of states r gives the following general steady-state probabilities:

$$\begin{aligned}
P_0 &= \left(1 + A_0 + \sum_{r=2}^{nh}\prod_{\ell=1}^{r-1}\left(1 - P_{\mathrm{bl}|\ell}\right)\frac{A_0^r}{r!}\right)^{-1}\\
P_1 &= P_0 A_0\\
P_r &= P_0 \prod_{\ell=1}^{r-1}\left(1 - P_{\mathrm{bl}|\ell}\right)\frac{A_0^r}{r!} \qquad 2 \le r \le nh
\end{aligned} \tag{14}$$

Finally, the total request blocking probability can be obtained by adding the contributions from each state, resulting in

$$P_{\mathrm{bl}} = \sum_{r=1}^{nh} P_{\mathrm{bl}|r}\,P_r \tag{15}$$

6 Numerical results

This section presents an extensive set of numerical results, obtained with the federated cloud network model introduced above, that help to understand the role of the different parameters involved in the design of a cloud federation network. In the following it is assumed that the VM memory size V_z follows a bimodal distribution, i.e. $V_z = V_0$, with probability q, and $V_z = u\,V_0$, $u > 1$, with probability $1 - q$. This approximation is intended to capture the fact that VMs can have either small or large memory requirements, depending on the specific nature of the applications they run. Also, the number of VMs in each migration group M_z is assumed to be uniformly distributed between 1 and M. Therefore, the

Table 1 Model parameters and their reference values

Parameter	Symbol and reference value
Maximum number of VMs to be migrated in a group	$M = 12$
Small VM memory size	$V_0 = 1$ GB
Large VM memory size multiplying factor	$u = 4$
Fraction of small VMs	$q = 0.75$
Memory page dirtying rate	$D = 2500$ pps
Memory page size	$P = 4$ KB
Residual dirty memory size threshold	$V_{\text{th}} = 100$ MB
Maximum number of iterations	$n_{\max} = 8$
VM resume time	$T_{\text{res}} = 100$ ms
Network pipe capacity reserved for any multi-VM migration	$b = 1$ Gbps
Total network pipe capacity	$B = 4$ Gbps
Number of remote DCs in the cloud federation	$n = 5$
Number of resource set instances per request	$m = 3$
Number of resource set instances supported by each remote DC	$k = 8$

average multi-VM migration time (e.g., in the sequential case) can be computed as

$$E\left[T_{\text{s-mig}}^{(z)}\right] = \frac{M+1}{2}\,\frac{V_0}{b}\,\frac{q\left(1-\gamma^{n_{\text{s}}(V_0)+1}\right)+u\,(1-q)\left(1-\gamma^{n_{\text{s}}(uV_0)+1}\right)}{1-\gamma} \tag{16}$$

where $n_{\text{s}}(x) = \min\left\{\left\lceil \log_\gamma V_{\text{th}}/x \right\rceil, n_{\max}\right\}$. Unless explicitly mentioned, the charts included in this section show the performance trends as a function of the migration request arrival rate λ, when one of the model parameters varies and the others are assigned the reference values reported in Table 1.

Figures 6 and 7 show the migration request blocking probability as a function of the arrival rate for different values of the total network pipe capacity B, assuming a sequential and a parallel migration strategy respectively. The curves have been obtained by applying the proposed model, whereas the points correspond to measurements obtained with a discrete-event simulator, specifically developed to evaluate the accuracy of the approximation introduced by the model. The analytic results show a quite good match with simulations, with a slight overestimation of the blocking probability, noticeable for small values of P_{bl}. The same data validation was performed for all the following graphs. However, for the sake of readability, only the curves obtained with the model will be shown.

As expected, the blocking probability increases when migration requests are more frequent, but a proper dimensioning of the network pipe capacity allows to keep the negative performance below a target level. Indeed, the proposed model can be used as an effective dimensioning tool to quantify the amount of communication resources needed for the design of a federated cloud network. Comparing the two figures clearly shows that parallel migration has a more detrimental effect on the inter-DC network performance than sequential migration. This is mainly caused by the higher migration time

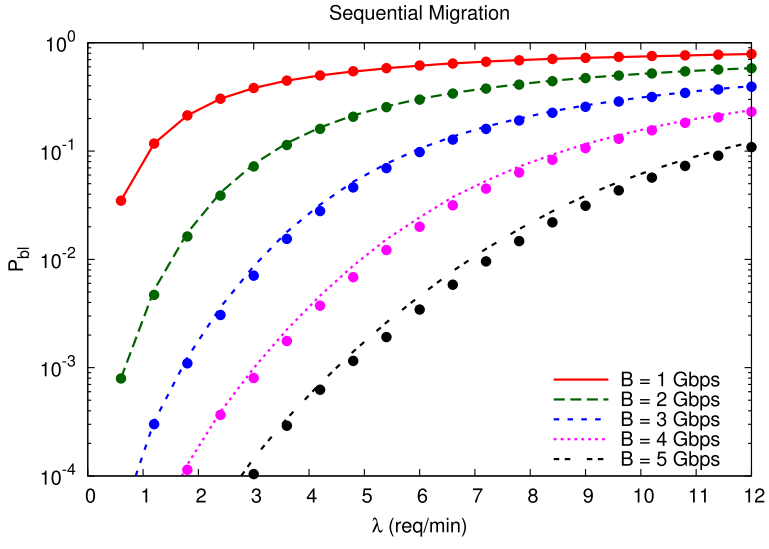

Figure 6 Sequential migration request blocking probability as a function of the arrival rate, for different values of the total network pipe capacity. Lines: analysis. Points: simulations.

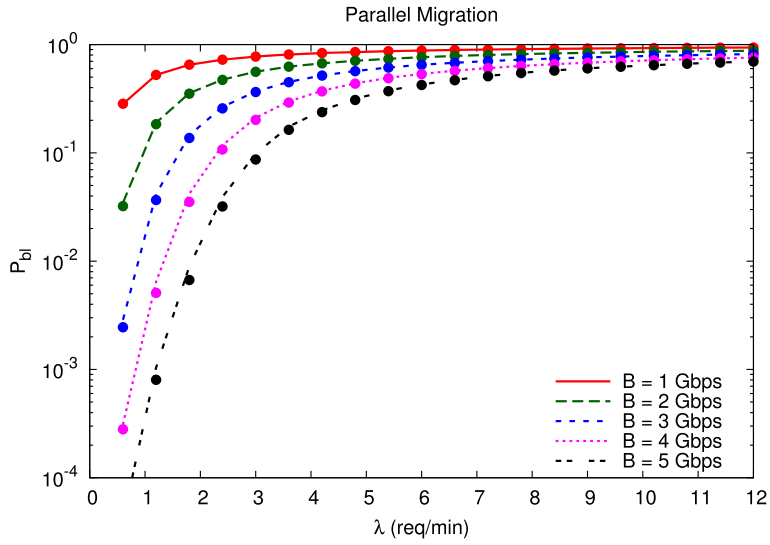

Figure 7 Parallel migration request blocking probability as a function of the arrival rate, for different values of the total network pipe capacity. Lines: analysis. Points: simulations.

experienced by VMs transferred simultaneously: in fact, in parallel migration the transfer rate of each VM is reduced due to network pipe capacity sharing, while the memory dirtying rate remains the same, thus increasing the number of iterations needed to complete the migration [32].

The worse network performance of parallel migration from the cloud federation management perspective is the price to pay to keep the total downtime as small as possible. In fact, as shown in Figure 8, while the average parallel migration downtime can be significantly smaller than the sequential one, the migration time can be much larger in the parallel case. This behavior affects also how the blocking probability depends on the ratio b/B, as reported in Figure 9: the step-like trend of the curves is caused by the abrupt change in the number of simultaneous group migrations supported by each network pipe, which decreases from 4 to 1. However, choosing a proper value of the network pipe capacity b to be reserved to each multi-VM transfer can help to control the migration timings and performance. In this sense, the proposed model allows to: (i) quantify the existing trade-off between sequential

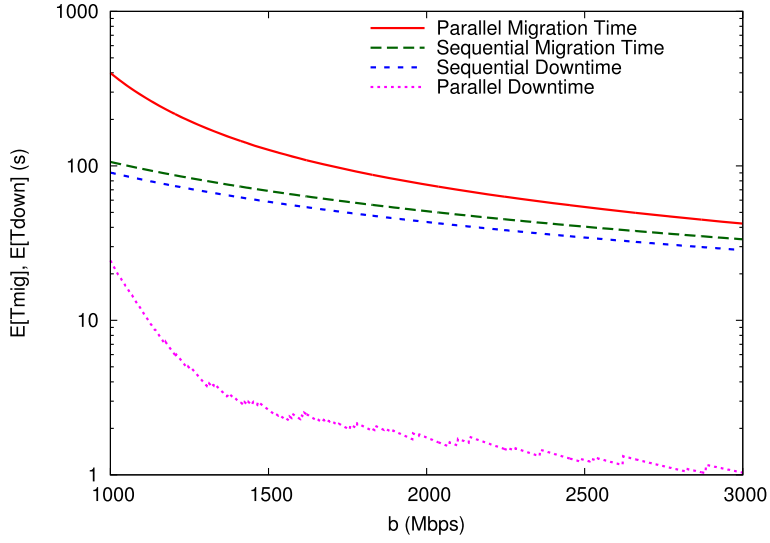

Figure 8 Total migration time and downtime of both sequential and parallel multi-VM migration strategies as a function of the network pipe slice capacity reserved to each transfer.

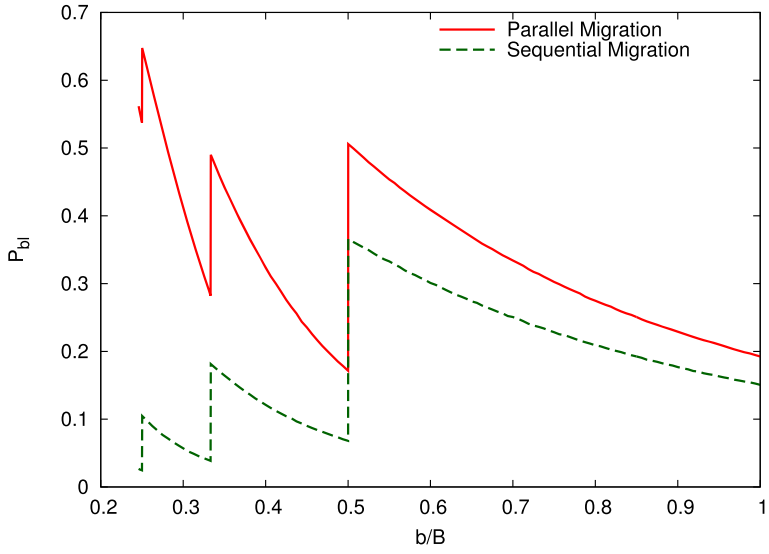

Figure 9 Sequential and parallel migration request blocking probability as a function of the fraction of network pipe capacity reserved to each multi-VM migration, when $\lambda = 6$ req/min.

migration, which tends to reduce the network resource occupancy, and parallel migration, which is better for the end-user's perceived quality; (ii) properly dimension the network pipe capacity b to be dedicated to a multi-VM transfer.

Another important aspect that the proposed model allows to quantify is the impact of the size of the cloud federation. To this purpose, Figures 10 and 11 show how the request blocking rate can be significantly reduced by increasing the number $n + 1$ of DCs participating in the cloud federation. Obviously, the performance improvement is a direct consequence of the higher number of communication and computing resources available when n increases, as there are more DCs capable of hosting the VMs to be migrated. However, increasing the size of the federated cloud network may have a significant infrastructure cost: the proposed model can help in finding a good cost/performance trade-off, given that a complementary infrastructure cost model is devised.

To understand the role of the availability of the computing resource set instances, i.e. the impact of the size of the anycast group associated to each migration request,

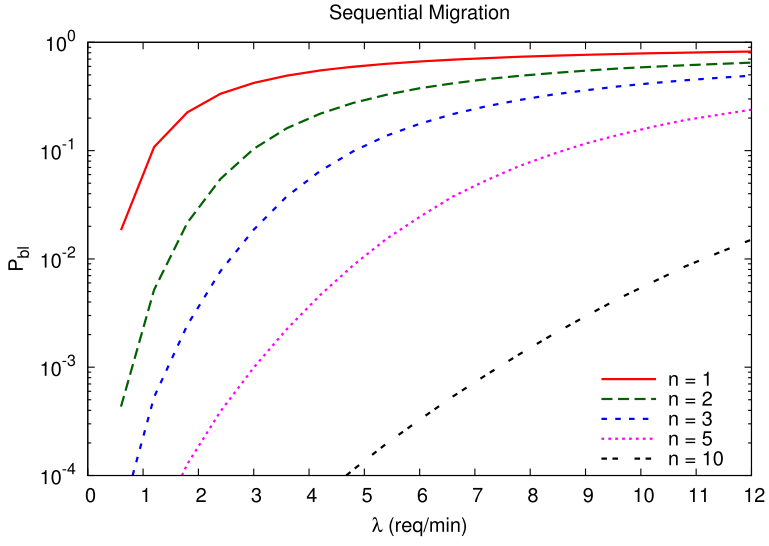

Figure 10 Sequential migration request blocking probability as a function of the arrival rate, for different values of the number of remote DCs.

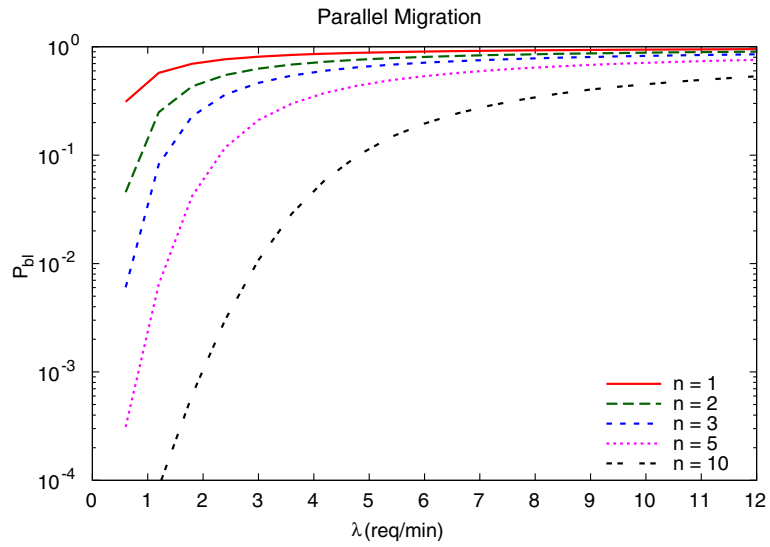

Figure 11 **Parallel migration request blocking probability as a function of the arrival rate, for different values of the number of remote DCs.**

Figures 12 and 13 show how P_{bl} decreases when m increases. As expected, the performance improves when it is possible to choose among a larger number of instances, since the computing resources available in the cloud federation can be better utilized. In other words, the model allows to quantify the improvement achieved by migrating groups of VMs that are compatible with a higher number of DCs participating in the federation, e.g. choosing VMs with less stringent requirements. However, it is worth to note that such an improvement saturates at a given point, e.g., there is little difference between $m = 10$ and $m = 40$. So, increasing m further does not bring any significant advantage, as long as space diversity and the related amount of communication resources, quantified by n, is the limiting factor. This effect is more evident in the parallel migration case, due to the worse network performance level.

The same dominant effect of the communication resource availability can be verified by varying the number k of resource set instances supported by each remote

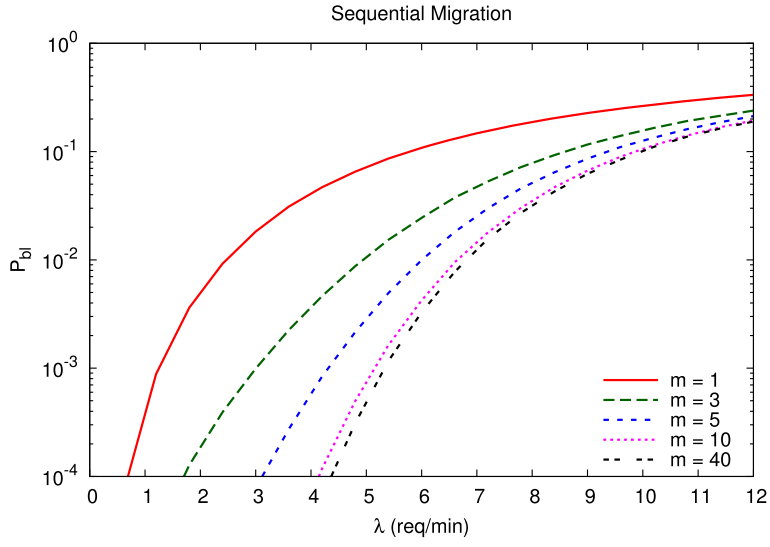

Figure 12 **Sequential migration request blocking probability as a function of the arrival rate, for different values of the number of resource set instances available per request.**

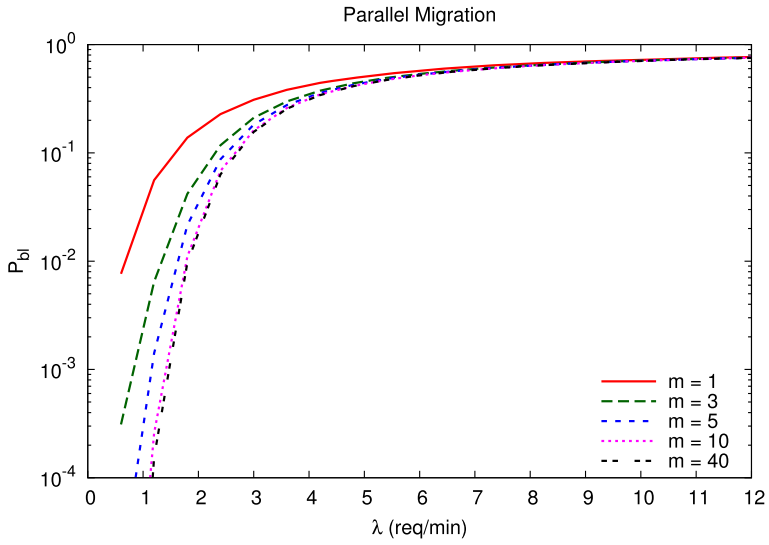

Figure 13 **Parallel migration request blocking probability as a function of the arrival rate, for different values of the number of resource set instances available per request.**

DC, as shown in Figures 14 and 15. In this case, P_{bl} quickly reaches saturation when k increases, especially for parallel migration. Therefore, increasing the DC capacity in terms of computing resources is not helpful if communication resources are not adequately improved as well. Indeed, the curves show that larger values of k give even worse request blocking probabilities when n and m are fixed. The reason of such counterintuitive behavior in the system model is the general assumption **A.4** of uniform distribution of the m compatible computing resource set instances to be chosen among the nk instances available in all remote DCs, which includes the possibility of choosing multiple instances located in the same DC. Increasing k means a higher chance of finding most or all of the m instances in the same remote DC without provisioning additional communication resources, thus increasing the chance of finding a full network pipe.

The number of VMs to be migrated in a single request and the size of their allotted memory are two key parameters with significant impact on the federated cloud network performance. In fact, increasing any of them results in a higher total migration time, and then in a higher resource occupancy period. This is exemplified by Figures 16 and 17 with relation to the maximum

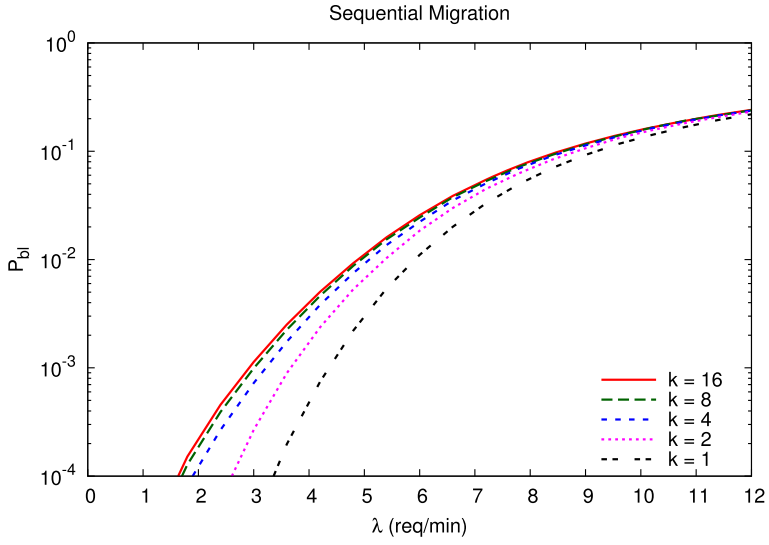

Figure 14 **Sequential migration request blocking probability as a function of the arrival rate, for different values of the number of resource set instances supported by each remote DC.**

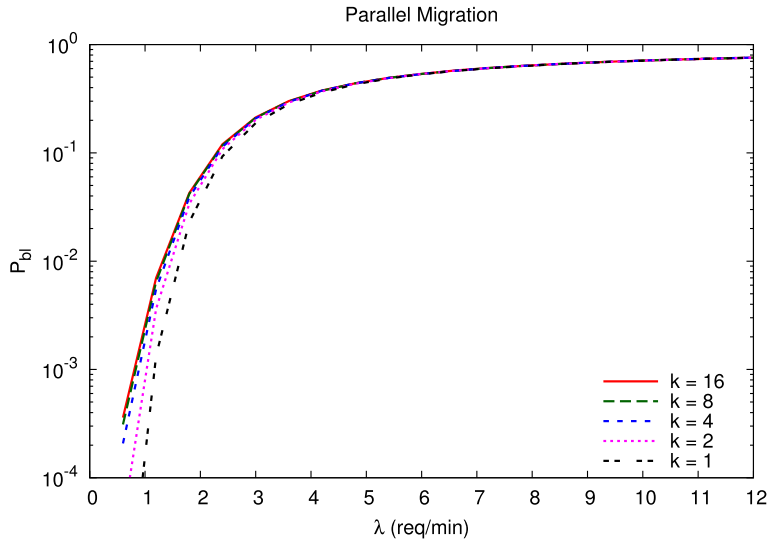

Figure 15 **Parallel migration request blocking probability as a function of the arrival rate, for different values of the number of resource set instances supported by each remote DC.**

number M of VMs to be migrated for each request, and by Figures 18 and 19 in terms of the large VM memory size multiplying factor u, according to the statistical assumptions made at the beginning of this section that led to the average migration time expressed in (16) for the sequential case. The proposed model allows to determine the limits of the VM migration group size and the VM memory size that are compatible with a desired average performance level, given the amount of computing and communication resources available in the cloud federation.

As a last example of the possible applications of the proposed model, Figure 20 compares the sequential and parallel migration costs as defined in (11) and (12), assuming an equal weight for network overhead and application throughput degradation, i.e. $\alpha = 0.5$. The sequential migration cost is less sensitive to the variation of the VM transfer capacity than the parallel migration cost, due to the similar behavior of the migration time and downtime as shown in Figure 8. The interesting result here is that there is a value of b above which parallel migration becomes more convenient than sequential migration:

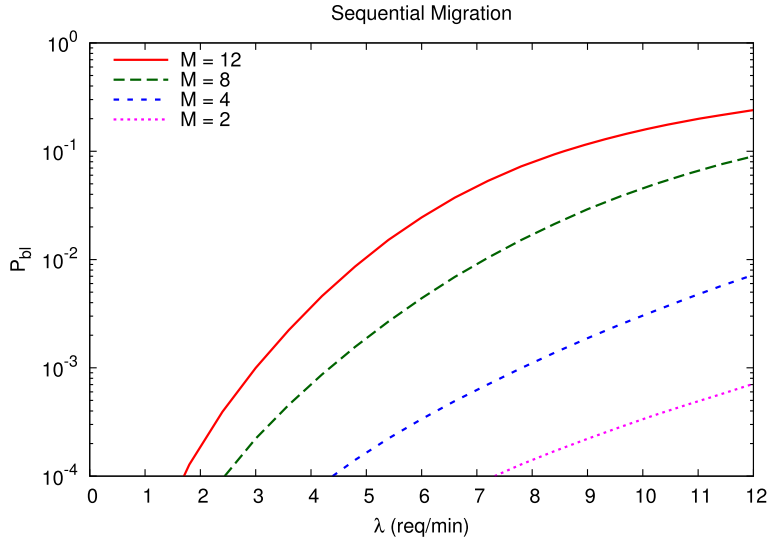

Figure 16 **Sequential migration request blocking probability as a function of the arrival rate, for different values of the maximum number of VMs to be migrated for each request.**

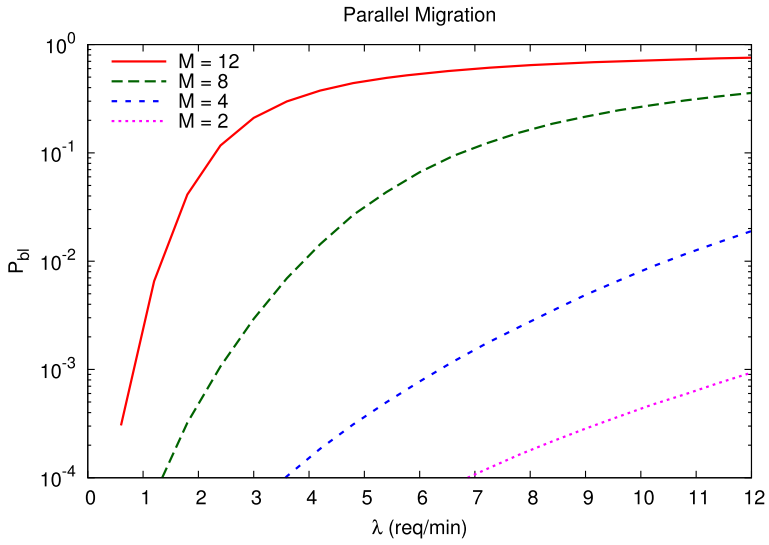

Figure 17 **Parallel migration request blocking probability as a function of the arrival rate, for different values of the maximum number of VMs to be migrated for each request.**

according to the aforementioned definition of migration cost, this happens when the difference in terms of network overhead becomes less significant than the difference in application throughput degradation, which is reduced when adopting a parallel migration. Of course, a different choice of the weight α would give a different result, but this simple cost model can help comparing different design choices for the federated cloud network infrastructure, after the relative importance of network overhead and customer's perceived quality has been determined.

7 Discussion

The results presented in the previous section show how to quantify the effects of the main cloud federation design parameters on the performance of multi-VM live migration procedures. The most relevant findings include:

- parallel migration is more demanding than sequential migration in terms of communication and computing resources, resulting in worse network performance from the cloud provider's viewpoint, although the

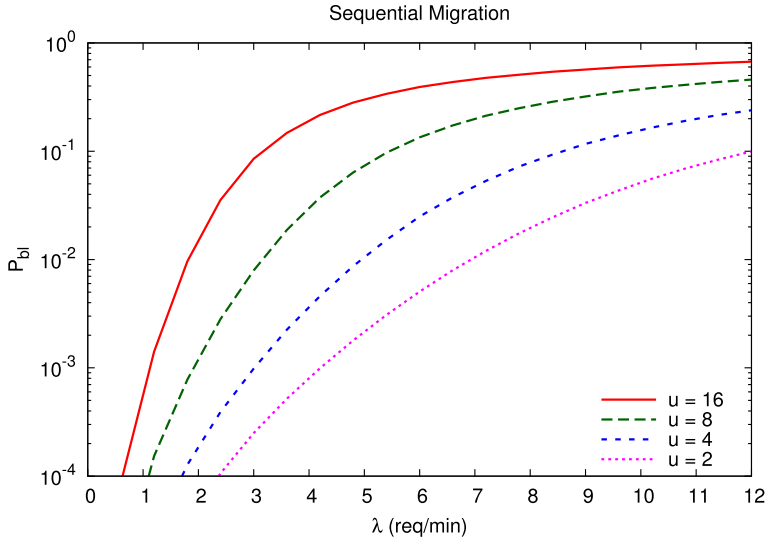

Figure 18 **Sequential migration request blocking probability as a function of the arrival rate, for different values of the large VM memory size multiplying factor.**

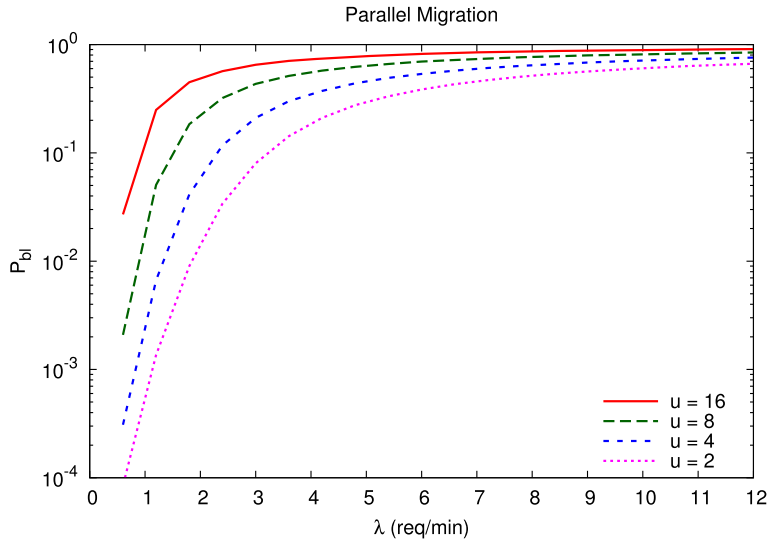

Figure 19 **Parallel migration request blocking probability as a function of the arrival rate, for different values of the large VM memory size multiplying factor.**

downtime experienced by the end-user can be significantly smaller;

- the total inter-DC network pipe capacity (B) and the minimum capacity guaranteed to migration requests (b) are key parameters that allow to control the performance from both end-user's and cloud provider's perspective;
- the significant effect of an increased size of the cloud federation (n) must be traded with the additional communication infrastructure cost that this implies;
- increasing the amount of computing resources available in the DCs (m, k) can have some positive effects on the performance, but only if supported by adequate space diversity (n) and/or network capacity (B);
- the number of VMs to be migrated together (M_z) must be carefully chosen in order to obtain the desired performance levels, also taking into account the VM memory size (V_0, u) and the amount of communication resources (B, b);

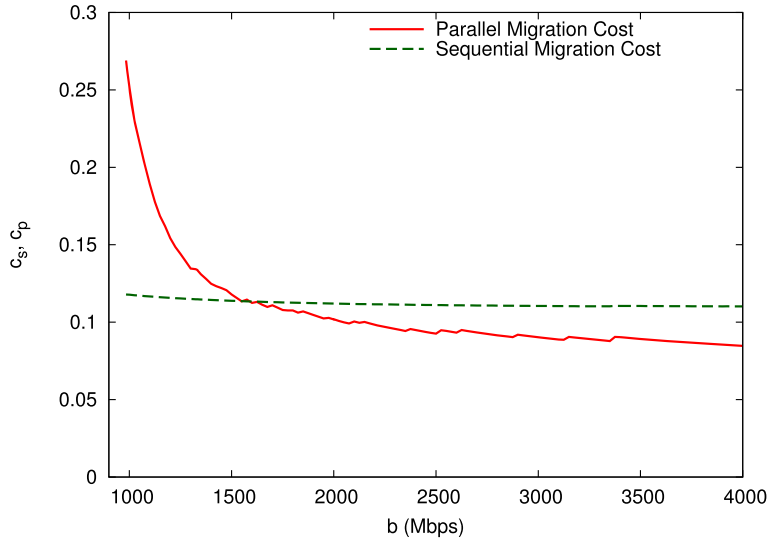

Figure 20 **Sequential and parallel migration cost as a function of the network pipe slice capacity reserved to each transfer, when $\lambda = 6$ req/min and $\alpha = 0.5$.**

- a well-defined balance between network overhead and end-user's perceived quality is essential to decide which multi-VM strategy is better under some given conditions (c_s, c_p).

The proposed model can then be used to properly dimension the cloud federation network. As a matter of fact, from the live migration model presented in Section 4, and by carefully choosing the number of VMs to be migrated, it is possible to compute the minimum amount of network pipe capacity (b) that must be guaranteed to migrate a group of VMs while ensuring a given maximum total downtime (Figure 8). Then, based on the expected load of migration requests (λ), it is possible to determine the minimum amount of total network pipe capacity (B) that must be provided by the inter-DC communication infrastructure to achieve a given performance level (Figures 6 and 7). As discussed above, other parameters can be used to refine the design of the cloud federation.

8 Conclusion

This paper presented a model for assessing inter-DC network performance in cloud federations, assuming that network load is caused by live migration of multiple VMs cooperating to provide the end-user with a given service. After characterizing the multi-VM migration time for the two alternatives of sequential and parallel migration strategies, it was demonstrated how the former one has a less detrimental effect on network performance, although parallel migration results in a much smaller service downtime. A possible trade-off in terms of migration cost has been quantified, showing how the choice of the optimal multi-VM migration strategy depends on the capacity provisioned in the network. Although some assumptions made can be considered abstractions, the proposed model can be used to properly dimension the inter-DC network capacity and to understand the macroscopic impact of the many parameters involved in the design of a federated cloud network.

Possible future extensions include a more accurate and dynamic modeling of the availability of computing resources in remote DCs, and the analysis of the impact of more realistic inter-DC network topologies. In addition, extensions to existing open-source hypervisors are currently being investigated in order to implement customized multi-VM migration strategies and build an experimental test-bed for validating the proposed model in a real inter-DC environment.

Competing interests
The author declares that he has no competing interests.

Authors' contributions
WC is the single author of this manuscript. He developed the analytical model presented here, wrote a software tool to compute the model as well as a discrete-event simulator to validate it, prepared all the graphs and tables with numerical results, and wrote the text of the paper. The author read and approved the final manuscript.

Acknowledgements
This work was partially funded by EIT ICT Labs, Action Line on Future Networking Solutions, under Activity 2013 "Smart Networks at the Edge" and Activity 2015 "SDN at the Edges".

Received: 24 October 2014 Accepted: 2 March 2015
Published online: 20 March 2015

References

1. Höfer CN, Karagiannis G (2011) Cloud computing services: taxonomy and comparison. J Internet Serv Appl 2(2):81–94
2. Chen M, Jin H, Wen Y, Leung VCM (2013) Enabling technologies for future data center networking: a primer. IEEE Netw 27(4):8–15
3. Rochwerger B, Breitgand D, Epstein A, Hadas D, Loy I, Nagin K, Tordsson J, Ragusa C, Villari M, Clayman S, Levy E, Maraschini A, Massonet P, Muñoz H, Toffetti G (2011) Reservoir - When one cloud is not enough. IEEE Comput 44(3):44–51
4. Toosi AN, Calheiros RN, Buyya R (2014) Interconnected cloud computing environments: challenges, taxonomy, and survey. ACM Comput Surv 47(1):7
5. BTI Systems (2014) Data Center Interconnect. http://www.btisystems.com/solutions/data-center-interconnect.aspx. Accessed 10 Sept 2014
6. Clark C, Fraser K, Hand S, Hansen JG, Jul E, Limpach C, Pratt I, Warfield A (2005) Live migration of virtual machines. In: Proc. of the 2nd USENIX Symp on Networked Systems Design & Implementation (NSDI 2005), Boston, MA. USENIX Association, Berkeley
7. The Open Networking Foundation (2012) Software-defined networking: the new norm for networks. ONF White Paper. https://www.opennetworking.org
8. Jain R, Paul S (2013) Network virtualization and software defined networking for cloud computing: a survey. IEEE Commun Mag 51(11):24–31
9. Wang X, Du Z, Chen Y, Li S (2008) Virtualization-based autonomic resource management for multi-tier web applications in shared data center. J Syst Software 81(9):1591–1608
10. Cerroni W (2014) Multiple virtual machine live migration in federated cloud systems. In: Proc. of the IEEE INFOCOM Workshop on Cross-cloud Systems (CrossCloud 2014), Toronto, Canada. IEEE, USA
11. Cuadrado F, Navas A, Dueñas JC, Vaquero LM (2014) Research challenges for cross-cloud applications. In: Proc. of the IEEE INFOCOM Workshop on Cross-cloud Systems (CrossCloud 2014), Toronto, Canada. IEEE, USA
12. Papagianni C, Leivadeas A, Papavassiliou S, Maglaris V, Cervelló-Pastor C, Monje A (2013) On the optimal allocation of virtual resources in cloud computing networks. IEEE T Comput 62(6):1060–1071
13. Guo C, Lu G, Wang HJ, Yang S, Kong C, Sun P, Wu W, Zhang Y (2010) SecondNet: a data center network virtualization architecture with bandwidth guarantees. In: Proc. of the 6th ACM Int Conf on Emerging Networking Experiments and Technologies (CoNEXT 2010), Philadelphia, PA. ACM, New York
14. Shrivastava V, Zerfos P, Lee K-W, Jamjoom H, Liu Y-H, Banerjee S (2011) Application-aware virtual machine migration in data centers. In: Proc. of the 30th IEEE Int Conf on Computer Communications (INFOCOM 2011), Shanghai, China. IEEE, USA
15. Zhani MF, Zhang Q, Simon G, Boutaba R (2013) VDC Planner: dynamic migration-aware virtual data center embedding for clouds. In: Proc. of the IFIP/IEEE Int Symp on Integrated Network Management (IM 2013), Ghent, Belgium. IEEE, USA
16. Kantarci B, Foschini L, Corradi A, Mouftah HT (2013) Design of energy-efficient cloud systems via network and resource virtualization. Int J Network Mgmt. doi:10.1002/nem.1838. John Wiley & Sons, USA
17. Mouftah HT, Kantarci B (2014) Optical inter-data-center network design under resilience requirements and dynamic electricity pricing. In: Proc. of the 16th Int Conf on Transparent Optical Networks (ICTON 2014), Graz, Austria. IEEE, USA
18. Amokrane A, Zhani MF, Zhang Q, Langar R, Boutaba R, Pujolle G (2014) On satisfying green SLAs in distributed clouds. In: Proc. of 10th Int Conf on Network and Service Management (CNSM 2014), Rio de Janeiro, Brazil. IEEE, USA

19. Bruneo D (2014) A stochastic model to investigate data center performance and QoS in IaaS cloud computing systems. IEEE T Parall Distr 25(3):560–569
20. Ibrahim KZ, Hofmeyr S, Iancu C, Roman E (2011) Optimized pre-copy live migration for memory intensive applications. In: Proc. of the Int Conf of High Performance Computing, Networking, Storage and Analysis (SC 2011), Seattle, WA. IEEE, USA
21. Medina V, Garcia JM (2014) A survey of migration mechanisms of virtual machines. ACM Comput Surv 46(3):30
22. Hines MR, Deshpande U, Gopalan K (2009) Post-copy live migration of virtual machines. ACM SIGOPS Operat Syst Rev 43(3):14–26
23. Bradford R, Kotsovinos E, Feldmann A, Schioberg H (2007) Live wide-area migration of virtual machines including local persistent state. In: Proc. of the 3rd ACM Int Conf on Virtual Execution Environments (VEE 2007), San Diego, CA. ACM, New York
24. Wood T, Ramakrishnan KK, Shenoy P, van der Merwe J (2011) CloudNet: dynamic pooling of cloud resources by live WAN migration of virtual machines. In: Proc. of the 7th ACM Int Conf on Virtual Execution Environments (VEE 2011), Newport Beach, CA. ACM, New York
25. Koponen T, Gurtov A, Nikander P (2005) Application mobility with HIP. In: Proc. of the 12th Int Conf on Telecommunications (ICT 2005). Cape Town, South Africa
26. Bhatti SN, Atkinson R (2012) Secure and agile wide-area virtual machine mobility. In: Proc. of the IEEE Military Communications Conf (MILCOM 2012), Orlando, FL. IEEE, USA
27. Keller E, Arora D, Perez Botero D, Rexford J (2012) Live migration of an entire network (and its hosts). Tech Rep TR-926–12 Dept of Computer Science, Princeton University
28. Bifulco R, Brunner M, Canonico R, Hasselmeyer P, Mir F (2012) Scalability of a mobile cloud management system. In: Proc. of the 1st ACM Workshop on Mobile Cloud Computing (MCC 2012), Helsinki, Finland. ACM, New York
29. Ye K, Jiang X, Ma R, Yan F (2012) VC-migration: live migration of virtual clusters in the cloud. In: Proc. of the 13th ACM/IEEE Int Conf on Grid Computing (GRID 2012), Beijing, China. IEEE, USA
30. Deshpande U, Wang X, Gopalan K (2011) Live gang migration of virtual machines. In: Proc. of the 20th Int Symp on High Performance Distributed Computing (HPDC 2011), San Jose, CA. ACM, New York
31. Bari MF, Zhani MF, Zhang Q, Ahmed R, Boutaba R (2014) CQNCR: optimal VM migration planning in cloud data centers. In: Proc. of the IFIP Networking Conf, Trondheim, Norway. IEEE, USA
32. Callegati F, Cerroni W (2013) Live migration of virtualized edge networks: analytical modeling and performance evaluation. In: Proc. of first IEEE Workshop on Software Defined Networks for Future Networks and Services (SDN4FNS 2013), Trento, Italy. IEEE, USA
33. Cerroni W, Callegati F (2014) Live migration of virtual network functions in cloud-based edge networks. In: Proc. of the IEEE Int Conf on Communications (ICC 2014), Sydney, Australia. IEEE, USA
34. Mukherjee B (2006) Optical WDM networks. Springer, New York
35. Liu H, Jin H, Xu C-Z, Liao X (2013) Performance and energy modeling for live migration of virtual machines. Cluster Comput 16(2):249–264
36. Strunk A (2012) Costs of virtual machine live migration: a survey. In: Proc. of the 8th IEEE World Congress on Services (SERVICES 2012), Honolulu, HI. IEEE, USA
37. Kelly FP (1991) Loss networks. Ann Appl Probab 1(3):319–378