Locating Virtual Infrastructures: Users and InP Perspectives

Guilherme Koslovski
INRIA - University of Lyon
guilherme.koslovski@ens-lyon.fr

Sebastien Soudan
Lyatiss
ssn@lyatiss.com

Paulo Gonçalves
INRIA - University of Lyon
paulo.goncalves@ens-lyon.fr

Pascale Vicat-Blanc
Lyatiss
pvb@lyatiss.com

***Abstract*—The Cloud Computing wave consolidates the on-demand provisioning of configurable virtual machines. Recent projects have proposed the extension of the original IaaS paradigm to provide dynamic virtual networks to interconnect virtual IT resources, composing Virtual Infrastructures (VIs). In this new scenario, users with different objectives and expectations can rent dynamically provisioned virtual infrastructures to execute their applications during a given time slot. VIs can be allocated anywhere on top of a distributed and virtualized substrate. This decoupling from the geographical location introduces concerns such as a latency increase in network communications (user's perspective), and the fragmentation of physical resources (Infrastructure Provider's - InP - perspective). This context motivates efforts to investigate and deploy new models and tools which consider the geographical location of virtual infrastructures. Our work concentrates on the allocation of VIs guided by both the user's and the InP's constraints. We propose a formulation of the allocation problem considering the user's expectations as well as the physical-substrate provider's goals. Our initial experiments demonstrate that it is possible to improve the quality of the virtual-infrastructure allocation (user perspective) while simultaneously decreasing the physical substrate's fragmentation and the substrate's cost.**

I. Introduction

Internet is becoming a world-wide reservoir of interconnected resources that can be reserved and shared among many users and applications with specific requirements and independent contexts. The Cloud Computing wave consolidates the on-demand provisioning of configurable virtual machines (VMs) to perform computation [1], [2] and provide large data storage [3]. Single users, companies and institutions have deployed dynamic VMs to execute their applications, exploring the advantages proposed by the *Infrastructure as a Service (IaaS)* paradigm, such as costs reduction [4] and reliability support [5].

New propositions have extended the original IaaS paradigm to provide dynamic virtual networks as well as computing resources [6], [7], [8], [9], composing the Virtual Infrastructures (VIs). With this extension, the Internet's computational and storage resources, as well as its communication and interconnection capacities, can be exposed as services, accessible for reservation and dynamic provisioning.

The complete abstraction of physical network and IT resources, combined with the geographical independency of VIs, brings together new challenges in the allocation and execution of high-end applications. Factors that directly impact application performance such as the amount of data to be processed, the geographical location and distribution of data, the confidentiality of the information, and the indeterminism of the required computational power, must be translated into virtual-infrastructure requirements in terms of computation, storage, and communication capacities [10].

Moreover, a virtual infrastructure can be allocated anywhere on top of a distributed substrate. The allocation of geographically fragmented virtual infrastructures introduces concerns for users and infrastructure providers (InPs): i) the network-communication latency of spread virtual resources is increased, which can augment the application's runtime [10]; ii) the physical substrate's fragmentation decreases the potential for accepting new requests due to an increased congestion factor on communication channels and IT resources [11]. The more the physical resources which host a virtual infrastructure are spread, the more bandwidth capacity is required (reserved) to interconnect the virtual IT resources. In the long term, the physical substrate's fragmentation can increase costs such as energy consumption, cooling and administration, due to the simultaneous activation of many distributed physical racks and network equipments.

This scenario justifies making efforts to allocate virtual infrastructures taking their geographical location into consideration. In this context, the main contributions of this work are:

- an allocation-problem formulation and an allocation heuristic, both guided by the geographical location of virtual and physical components;
- experimental results demonstrating that it is possible to minimize the physical substrate's fragmentation and costs (the InP perspective): the fragmentation can be decreased by almost 28% on a medium-size physical substrate, while the cost can be decreased by approximately 21%;
- results highlighting an improvement in allocation quality (the user perspective) of about 39% for different sizes of virtual infrastructures allocated on a medium-size physical substrate.

The description of our contributions is organized as follows: Section II presents the motivations, the goals, and the definitions that guide our proposal. Section III formulates the allocation problem using a graph notation, and describes the constraints and objective functions. Section IV presents the allocation method proposed in this work, and Section V describes the initial experiments. Related works are reviewed in Section VI. Section VII concludes this work.

II. Motivations, goals and definitions

In [12] we defined a *Virtual Infrastructure (VI)* as a time-limited interconnection of virtual computing resources, which are interconnected by a virtual network. The proposed network model integrates virtual-link provisioning (including bandwidth-sharing control) with traffic-control components [13]. By combining IT resources virtualization and network virtualization, the user of a VI has the illusion that he is using a private distributed system, while in reality he is using multiple systems that are part of a virtualized physical substrate.

VIs are dynamically-provisioned entities which can be defined and modeled to represent the application's requirements in terms of computing and communication [14]. The required configuration and the temporal aspects of VIs are specified by the user during the establishment of the Service Level Agreement (SLA). Specific attributes enable the description and parameterization of: a) individual resources and groups of resources; b) an extensible list of parameters to represent the necessary configuration—e.g., RAM, CPU speed, and storage capability; c) the required software (e.g., operating systems) and tools (e.g., communication and programming tools) that must be provisioned with the VI; d) each resource functionality, for example, a set of *computing* nodes or *storage* nodes; e) the virtual network topology, detailing each virtual link. On the other hand, parameters enable an abstract description, for example, informing a virtual link description that must be used to interconnect a group of resources; f) the internal execution timeline of the VI; g) security attributes for each resource (e.g., access control and confidentiality level); h) reliability level required for each VI component; i) commercial attributes (maximum cost); and j) temporal attributes for each resource (the time window for provisioning). Also, a user can inform a specific location where the virtual resources must be provisioned, respecting an application reason (data location, for example) or a user reason (e.g., confidentiality and security).

Figure 1 exemplifies a VI composition using a graph notation: the vertices represent the virtual resources (r_v1, r_v2, and r_v3), which are interconnected by virtual links (the edges). In this example, a particular user has access to all VI components during the reservation time.

A VI is deployed and provisioned on top of a distributed and virtualized physical substrate. Usually, this physical substrate is hierarchically organized and interconnected. Figure 2 presents an example of this hierarchical organization using a tree structure, where the leaves are the IT resources, and the parent nodes represent the geographical organization and interconnection. Starting from the bottow, the physical resources (for example, r_p1, r_p2, and r_p3) are grouped on racks. These racks are positioned at different locations, such as *lyon.fr.eu* or *paris.fr.eu*. More specifically, in this example the exact location of the physical node r_p1 is noted as *rp1.rack1.lyon.fr.eu*. In this hierarchical organization, the distance between physical resources is given by the number of intermediate hops. For example, the distance between r_p2 and r_p3 is one hop.

VIs must be efficiently allocated on top of geographically distributed physical substrates. Considering the users and InP perspectives, we defined two metrics to be investigated during the allocation process: the *physical substrate's fragmentation* and the *allocation quality*, as defined below.

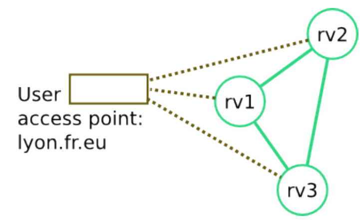


Figure 1. A VI composed by resources r_v1, r_v2 and r_v3 is requested by a user with access point located at *lyon.fr.eu*. In this example, the user has access to all VI components during the reservation time.

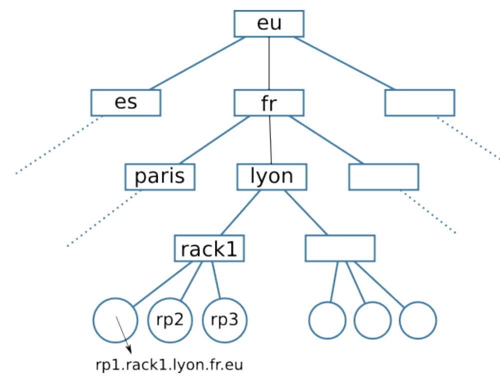


Figure 2. The physical substrate that must allocate the VI describe on Figue 1 is made up by components hierarchically distributed and interconnected.

A. Physical substrate fragmentation

The allocation of spread VI components induces long-term issues on physical substrates, such as an increase in administration costs, energy consumption and cooling due to the simultaneous activation of many racks and network equipments. It also decreases the acceptance ratio of new requests due to an increased congestion factor on communication and computational resources [11].

We define the *physical substrate's fragmentation* as a metric to qualify the number of physical resources (IT and network) reserved and activated on a distributed and virtualized physical substrate. This metric is given by the ratio between the number of activated resources and the total number of physical resources available.

To exemplify the fragmentation, lets consider the VI request presented on Figure 1 and the physical substrate described on Figure 2. Considering this metric, an efficient allocation map will place all the virtual nodes and virtual links using the minimum number of physical components. For example, allocating the virtual nodes into physical machines r_p1, r_p2, and r_p3, respectively, only requires activating one rack and using a single physical communication path to provide network access to those components. Furthermore, an optimal allocation will place all virtual nodes (r_v1, r_v2, and r_v3) into physical machine r_p1 only requiring the activation of a single computing resource. In the long term, it is expected that minimizing the physical substrate's fragmentation can increase the acceptance ratio of a InP.

B. Allocation quality

From the user's perspective, the allocation and provisioning of spread VI components result in a latency increase on network communications [10]:

- node-to-node: usually, the more distant the physical hosts, the higher the latency in communication. This issue is more perceptible in communication intensive applications, but can also affect regular applications;
- user interaction: interactive applications (such as remote terminals or visualization tools) are explored by users to control both the virtual infrastructure and their applications. The physical distance between the user and the virtual infrastructure components increases the response time of those applications proportionally.

Advanced users can specify the exact configuration required to efficiently execute their applications [14]. For example, there are reasons why a certain application should run in a certain location, such as data-location dependency, security, and even limitations on data mobility because of governmental law. This location-based provisioning can be explicitly required. However, regular users are not aware of the efficient configuration in terms of networking and computation power. Moreover, some users are not familiar with the meaning of latency in communications, and only wish a set of VMs to execute their applications.

In this context, defining the allocation quality from the user's perspective is a difficult task. The allocation quality is optimal when the virtual request is precisely defined and all resources correctly provisioned. When the request is non-absolutely defined, quality is subjective: the user do not care on the optimal configuration to execute its application, however, he wants the application runs well, usually with an efficient interaction between its distributed components. Figure 1 describes this scenario: a VI is requested by a user with access point located at *lyon.fr.eu*. In this case, the user does not specify any requirement in terms of virtual-resources location or network configuration. Thus, the InP can allocate this request anywhere on top of the distributed substrate represented by Figure 2.

For both types of users, independently of the VI description level, a common factor can be optimized: the resources' proximity. We call *allocation quality* of a VI the average of all distances (calculated in hops) between the virtual resources and one specific geographical landmark (the reference point). More specifically, the reference point can be i) the location of the user, or ii) the location of a certain virtual resource, as specified by the user.

To exemplify the definition of allocation quality, let's consider the user's location as the geographical landmark for hops calculation (*lyon.fr.eu*). Allocating resources distributed among others locations (such as *es*) consequently results in a greater number of hops from the user's location than allocating resources near *lyon*, or even *fr*. More specifically, allocating all the components of this VI on physical node *rp1.rack1.lyon.fr.eu* results in a minimum distance (in this case, only one hop, from *lyon*'s access point), and consequently, an optimal allocation quality.

In the following sections we formulate the VI allocation problem and propose an embedding heuristic under the constraints of fragmentation and allocation quality optimization, discussed in this section.

III. Problem formulation

The problem of mapping VIs to a physical substrate corresponds to a classical graph embedding problem. The graph describing the VI, $G^v(R^v, L^v)$, must be mapped on the physical substrate graph, $G^p(R^p, L^p)$, where R^v and R^p are the set of virtual and physical nodes, respectively, and L^v and L^p are the set of virtual and physical links, respectively.

Let's denote by $Q_R(r, t)$ the vector of capacities (memory and CPU) of node r ($\in R^v$ or $\in R^p$) at time $t \in [0, T]$. Further, let P^p be the set of all the simple physical paths between any two physical nodes, $Q_P(p, t)$ be the vector of capacities of physical path $p \in P^p$, and $Q_L(l, t)$ be the vector of capacities of link $l \in L^v$, both at time t. For a path $p = (l_1^p, l_2^p, \dots)$, the bandwidth capacity is the minimum of all bandwidth values of l_i^p in p. By adopting capacity vectors indexed by time, the capacities of IT resources and links can vary during the reservation's time.

Finally, let $A_R(r)$ be the set of geographical locations of resource r ($\in R^v$ or $\in R^p$). For virtual resources, the value of $A_R(r)$ can be specified by users, and for physical resources it represents the exact geographical location.

A map of a VI on a physical substrate represents the reservation of all the capacity requirements specified by the user, noted as:

$$\text{Resources mapping}: \mathcal{M}_R : R^v \to R^p$$

$$\text{Links mapping}: \mathcal{M}_L : L^v \to P^p$$

Given a set of VI requests S^v, the embedding problem is to obtain a map that maps virtual nodes R^v to physical nodes R^p, denoted by $\mathcal{M}_R$, and virtual links L^v to physical paths P^p, denoted by $\mathcal{M}_L$, such that the following conditions are satisfied:

$$Q_R(\mathcal{M}_R(r_i), t) \geq Q_R(r_i, t), \forall r_i \in R_j^v, \forall G_j^v \in S^v, \forall t \in [0, T]; \quad (1)$$

$$Q_P(\mathcal{M}_L(l_i), t) \geq Q_L(l_i, t), \forall l_i \in L_j^v, \forall G_j^v \in S^v, \forall t \in [0, T]; \quad (2)$$

$$A_R(r_i) \subset A_R(\mathcal{M}_R(r_i)), \forall r_i \in R_j^v, \forall G_j^v \in S^v; \quad (3)$$

A. Physical substrate fragmentation and cost

Let's initially define the functions $C_r(r,t)$ and $C_l(l,t)$, which set the physical substrate cost for an amount of resource $r \in R^v$ and the cost for an amount of link $l \in L^v$, respectively, both at time t. The total cost of resources R^v, over a reservation time $[0,T]$ is given by:

$$C_R(R^v,T) = \int_0^T (\sum_{r_i \in R^v} C_r(r_i,t))\ dt \tag{4}$$

Similarly, the total cost of links L^v, over a reservation time $[0,T]$ is defined as:

$$C_L(L^v,T) = \int_0^T (\sum_{l_i \in L^v} C_l(l_i,t) \times len(\mathcal{M}_L(l_i)))\ dt \tag{5}$$

where $len(\mathcal{M}_L(l_i))$ gives the length of the path provisioned to allocate virtual link l_i. Consequently, the total cost of a VI G^v is noted as:

$$C_{VI}(G^v,T) = \alpha C_R(R^v,T) + \beta C_L(L^v,T) \tag{6}$$

Constants α and β are tunable weights which allow the balance and the normalization between resources and links costs. Given a set of VI requests S^v, an immediate metric is the minimization of the total cost for the infrastructure provider, as defined by:

$$\text{minimize:} \sum_{G_i^v \in S^v} C_{VI}(G_i^v,T) \tag{7}$$

subject to the constraints defined by conditions (1)-(3).

As discussed in Section II-A, InPs aim the minimization of the physical resources fragmentation. Let's denote by F_R the subset of physical resources $r \in R^p$ which support at least one virtual resource running, and similarly, by F_L the subset of physical links $l \in L^p$ which host at least one virtual link activated. The objective is to minimize the number of physical resources (IT and networking) involved to allocate S^v:

$$\text{minimize:} \ \frac{\#F_R + \#F_L}{\#R^p + \#L^p} \tag{8}$$

subject to the constraints defined by conditions (1)-(3).

B. VI allocation quality

From the user's perspective, the objective is to maximize the allocation quality. The quality of an allocation is directly related to the location of the virtual resources (as discussed on section II-B). Let's define the distance function $D_R(a_i,a_j)$ that gives the distance between locations a_i and a_j in number of hops, where $a \in A_R$, and each hop is equivalent as one unit. Further, define a^u as the location specified by the user (the reference point). The optimal allocation quality is given by the minimization of the average resources distance:

$$\text{minimize:} \sum_{r_i \in R^v} \frac{D_R(A_R(\mathcal{M}_R(r_i)), a^u)}{\#R^v} \tag{9}$$

subject to the constraints defined by the conditions (1)-(3).

Conceptually, the quality improvement is related to the minimization of physical substrate fragmentation. This relationship is observed in results presented in Section V.

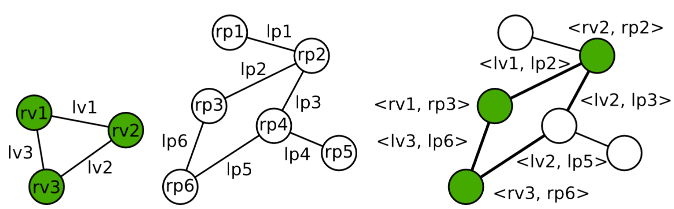
IV. VI ALLOCATION

This section discusses the models, algorithms and techniques used to allocate VIs considering the proposed problem formulation. The VI graph-embedding problem is well-know to be NP-hard [15]. There are numerous works on solving the problem with heuristics based on path-splitting methods [16], multi-commodity flow modeling [17], and substrate characteristics [11]. We choose a subgraph-isomorphism detection [18] (which is solvable in polynomial time) to incorporate the allocation constraints and to examine the metrics proposed.

A. Subgraph isomorphism detection

The process of embedding a virtual graph on a physical one, in order to find a possible map allocation, can be solved as a subgraph isomorphism detection [19], [5]. An isomorphism, with edges extension, from G^v to G^p is a function f that maps R^v to R^p and L^v to P^p such as each edge $l_i \in L^v$ with endpoints $r_m \in R^v$ and $r_n \in R^v$ is mapped to a path of edges $l_j \in P^p$ with endpoints $f(r_m) \in R^p$ and $f(r_n) \in R^p$, subject to the constraints defined by the conditions (1)-(3).

Figure 3 exemplifies a subgraph isomorphism map between graphs G^v and G^p. Applying function f for all $r_i \in R^v$ and for all $l_i \in L^v$ results in a set of maps called $< x, f(x) >$, where x represents both vertices and edges.



(a) graph $G^v(R^v, L^v)$. (b) graph $G^p(R^p, L^p)$. (c) Subgraph isomorphism between graphs G^v and G^p.

Figure 3. The figures show a subgraph-isomorphism map between G^v (Figure 3a) and G^p (Figure 3b) given by the application of a function f, where $f(r_v1) = r_p3, f(r_v2) = r_p2, f(r_v3) = r_p6, f(l_v1) = l_p2, f(l_v2) = l_p3 \cup l_p5, f(l_v3) = l_p6$, as exemplified on Figure 3c.

A subgraph-isomorphism map of non-simple graphs (i.e., VIs) requires the extension of edges to interconnect non-adjacent vertices. Figure 3c exemplifies this requirement: r_v2 and r_v3 were mapped on r_p2 and r_p6, respectively. Consequently, l_v2 must be extended over l_p3 and l_p5 to interconnect these resources.

To find a map solution between both physical and virtual graphs, a capacity comparison must be performed between vertices, edges and paths. The virtual and physical components being compared are called candidates. Usually, in a subgraph-isomorphism detection, a virtual candidate is tested with all physical candidates [19]. This approach requires a large number of comparisons between nodes and links (for example, to identify the shortest physical path that can host a virtual link), and consequently results in a elevated computational cost to find an allocation solution.

B. Location-aware algorithm

To accelerate the processing we have produced a patented allocation heuristic to allocate virtual infrastructures [20]. In the context of this paper we propose an extension of this method able to exploit the location of virtual resources. Initially, the location-aware algorithm identifies the set of physical landmarks specified by the user, as well as the set of virtual resources without location constraint.

An iteration is performed on those sets. Each time, one physical location specified by the user is defined as the required location constraint for components that do not have this information. At this moment, a subgraph-isomorphism detection is performed to find a map solution. If no allocation solution is found for this configuration, the location constraint is relaxed for those resources, i.e., the location's precision is decreased.

Let's use Figure 2 as example. In the first iteration, the required location of virtual resources r_v1, r_v2, and r_v3 are defined as *lyon.fr.eu* (the user's location). Considering that no solution was found with this configuration, the geographical location of these virtual resources is relaxed from *lyon.fr.eu* to *fr.eu*. Observe that, this way, the algorithm always tries to allocate virtual components as close as possible to the geographical landmark.

The allocator has been implemented as a module of the Lyatiss Weaver (more information about Lyatiss Weaver is available on http://www.lyatiss.com/). The Lyatiss Weaver combines system- and networking-virtualization technologies with bandwidth-sharing and advance-reservation mechanisms to offer dynamic networking- and computing-infrastructures as services [12], [21]. At run-time, the Lyatiss Weaver communicates with physical resources to deploy virtual nodes (configured respecting the users' requirements), monitor their status and configure control tools to supervise the resources usage. In this fully virtualized scenario, Lyatiss Weaver interacts with multiple resource providers to plan, monitor and control them. Functions such as fault management, load balancing, bandwidth management and performance control are handled taking both network- and resource-virtualization techniques into account.

The Lyatiss Weaver, and consequently the allocator, receives requests for VIs provisioning using the VXDL language [14]. VXDL is an XML-based language that allows an efficient description of virtual infrastructures; more specifically, the identification and parameterization of virtual resources and groups of resources (according to their functionalities), as well as the network topology (based on the link-organization concept), using the same grammar. VXDL also introduces the internal virtual infrastructure timeline, which explores the elasticity of VIs, enabling application providers to specify the exact intervals where virtual resources must be activated. An important feature of VXDL is that it proposes cross-layer parameters (i.e. application level and physical level attributes) for all components. For example, with the specification of *location* and *exclusivity*, users can directly transmit application-specific information and constrains the management framework.

We adopted a multi-thread implementation of the proposed solution. The execution is finished when all threads return an empty answer, or when a map solution is found. We omitted implementation details, but basically, our implementation relies on future objects and synchronization mechanisms proposed by the Java language.

V. Experiments

This section describes our initial experiments and performs an analysis considering the metrics proposed on Section III.

A. Scenario composition

We ran the experiments on machines belonging to Grid'5000 [22] with the following configuration: 2 CPUs Intel Xeon L5420, 4 cores, 2.5 GHz, 6 MB cache, and 32 GB RAM. We used Java Runtime Environment version 1.6.0_20 to run the allocator.

The physical-substrate graphs and the virtual-requests graphs were generated by the topology generation tool GT-ITM [23]. Physical substrates use the *transit-stub model*, which results in graphs composed by domains interconnected by a backbone. Virtual requests were generated considering the *normal model* without backbone routers, following setups similar to previous works [16], [17].

Two physical substrates were simulated:

- a *small-size substrate* composed by 100 resources (domains and backbone routers) and approximately 200 physical links, organized in 4 geographical domains, and interconnected by a backbone composed by 8 resources;
- a *medium-size substrate* composed by 500 resources, approximately 4000 links, divided in 8 geographical domains, and interconnected by a backbone composed by 20 resources.

The values of CPU cores (2, 4 or 8) and of memory capacity (2 GB, 4 GB, 8 GB or 16 GB) follow a uniform distribution. The network's bandwidth capacity was defined as 1 Gbps within a domain and 10 Gbps between domains (in the backbone). To represent a more realistic scenario, the allocation of computing nodes on backbone resources was disabled.

Virtual requests vary the number of nodes among 2, 4, 6, 8 and 10. Each pair of virtual resources is randomly connected with a probability of 0.5. All VI requests require a reservation period of one hour. The values of CPU cores (1, 2 or 4) and memory capacity (256 MB, 512 MB, 1 GB, or 2 GB) also follow a uniform distribution. The same approach was used to generate network-bandwidth requirements (10 Mbps, 20 Mbps, 40 Mbps, 100 Mbps, 200 Mbps, 400 Mbps, or 600 Mbps).

The values of α and β used to calculate the VI cost (C_{VI}) were defined as $\alpha = \beta = 1$, indicating that nodes and links have the same weight to the InP, as explored on previous scenarios [16]. The cost functions ($C_r(r,t)$ and $C_l(l,t)$) require an equivalent metric to calculate the costs. As an example of this equivalence, we arbitrarily set that 1 GB, 1 core, and 100 Mbps are the basic units for memory, CPU and bandwidth, respectively, being equivalents in terms of cost calculation. The definition of those values requires a specific study based on InP policies and current substrate load, as proposed by [11]. We leave this implementation and analysis for future work.

The number of VIs requests submitted were: 100 to the small-size substrate, and 300 to the medium-size substrate. The number of requests' sources varies (1 or 3) to represent different request-submission scenarios. Results identified by *allocation with basic algorithm* were obtained by executing the regular subgraph-isomorphism detection, without the geographical-location optimization. The optimized execution is identified by the *allocation with our optimized algorithm* label.

All averages presented on the following experiments were calculated considering 10 executions and have a confidence interval of 95%.

B. Physical substrate's fragmentation and cost

This experiment investigates the variation of the InP allocation costs and the physical-substrate fragmentation. For both metrics, the results were obtained considering two configurations: *small-size* and *medium-size* physical substrate.

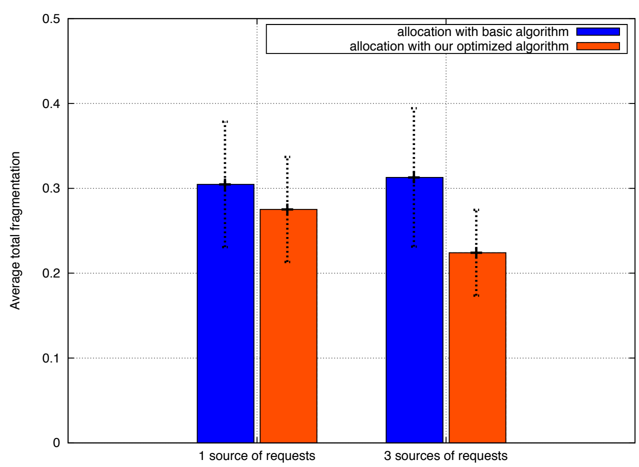


Figure 4. Fragmentation of a small-size physical substrate. The number of sources requesting VIs is 1 and 3.

Figure 4 shows the average total fragmentation of a small-size physical substrate. The comparison performed on Figure 4 highlights that our algorithm improves the physical substrate usage by decreasing the fragmentation approximately 10% in the case where all requests come from the same location, and that its performance increase with the number of requests source: 28% when 3 sources are submitting requests. Similar results are obtained with requests submitted to the medium-size substrate (see Figure 5).

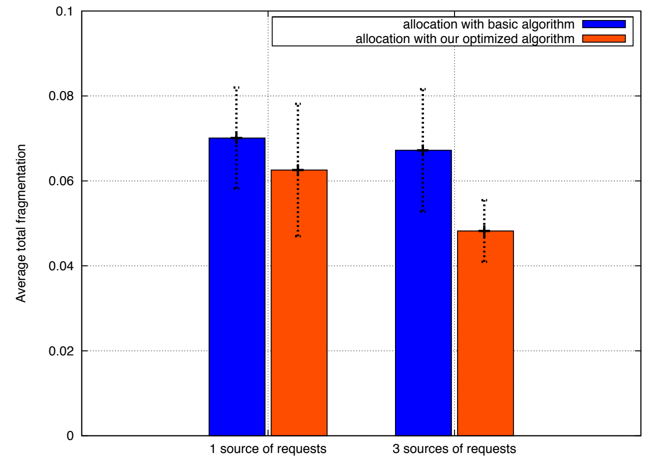


Figure 5. Fragmentation of a medium-size physical substrate. The number of sources requesting VIs is 1 and 3.

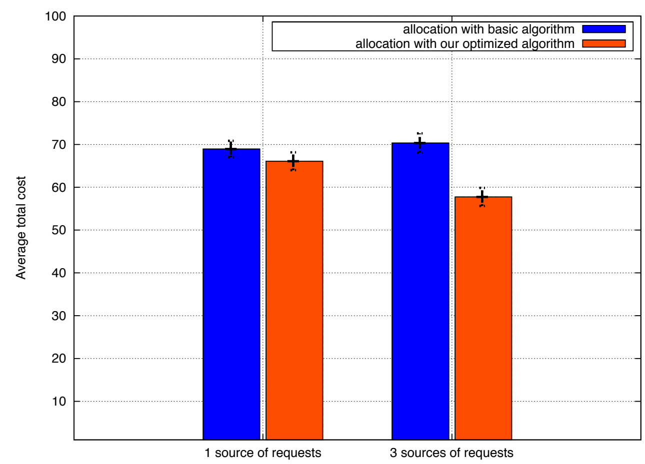


Figure 6. InP cost of allocations on a small-size physical substrate. The number of sources requesting VIs is 1 and 3.

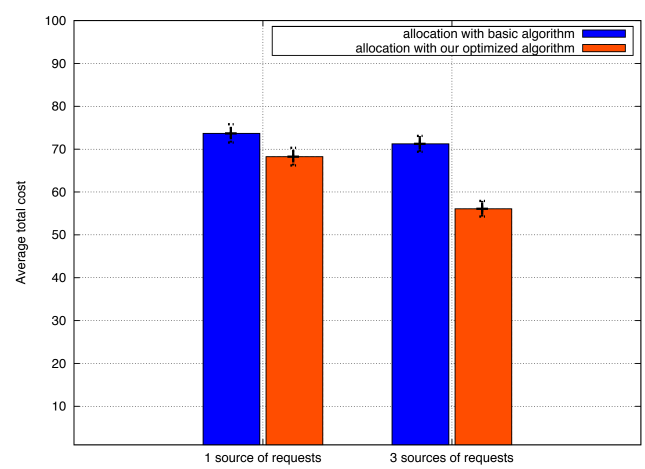


Figure 7. InP cost of allocations on a medium-size physical substrate. The number of sources requesting VIs is 1 and 3.

The average of the total cost required to allocate these requests were analyzed and presented on Figure 6 and Figure 7. For these scenarios, the average cost is also decreased by our optimized algorithm. Considering requests submitted by only 1 source, the cost is decreased in both scenarios: approximately 4% on a small-size physical substrate, and 8% on a medium-size one. When analyzing requests submitted by 3 sources, the cost also decreases close by 17% and 21% for the small-size substrate and the medium-size substrate, respectively.

C. Allocation quality

We also measured the metric defined to quantify the allocation quality (discussed in Section III). The user location (source of requests) was defined as the reference point for distances calculation.

Figure 8 and Figure 9 present the average for the small-size and medium-size substrate, respectively. The results show that the average total distance is smaller with our algorithm in both scenarios, for requests submitted by 1 and 3 source(s). When only 1 source was submitting requests, the average total distance was decreased by almost 13% for small-size and medium-size substrates. However, when 3 sources were submitting requests the average total distance is highly decreased: by approximately 36% and 39% for small-size and medium-size substrates, respectively. Consequently, it is expected that the users of VIs allocated by our optimized algorithm will have a lower perception of the physical-resources distribution in terms of network communication.

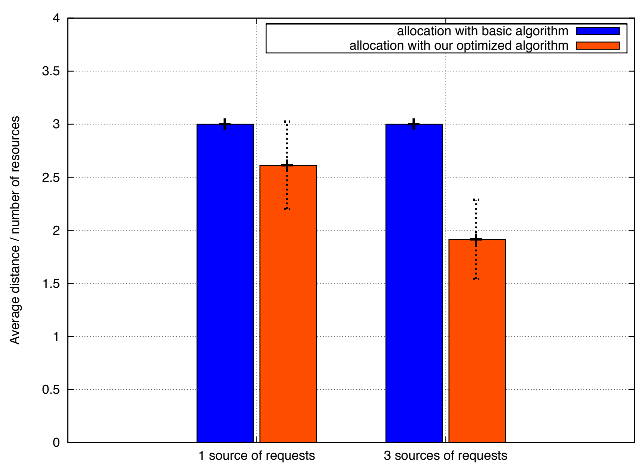


Figure 8. Aggregated distance of VI components allocated on a small-size physical substrate. The number of sources requesting VIs is 1 and 3.

Analyzing the results of the optimized algorithm highlights the relationship between performance and the number of geographical landmarks used as reference points (in this case, the user's location). For all metrics, the results obtained with three sources of requests showed a better performance than those obtained with one source of requests, independently of the physical substrate size. Consequently, increasing performance of these metrics (physical fragmentation, allocation cost, and allocation quality) is expected when the number of sources submitting VI requests is augmented.

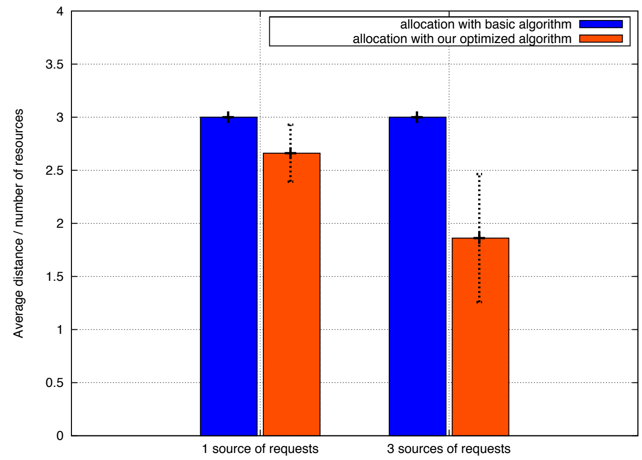


Figure 9. Aggregated distance of VI components allocated on a medium-size physical substrate. The number of sources requesting VIs is 1 and 3.

VI. Related works

The allocation of virtual networks has been investigated in previous works. Some algorithms inherited the problem formulation from the Virtual Private Networks (VPNs) perspective. These formulations only consider bandwidth requirements [24], [25]. In this scenario, the allocation consists in finding paths between source-destination pairs. The allocation of virtual IT resources was not addressed since they were placed in advance.

Some works have focused on problem formulation considering the nodes requirements together with network configuration. This problem is well-know to be NP-hard [15]. There are numerous works on solving the graph-embedding problem: isomorphism-based detection [19], [5], path-splitting methods [16], multi-commodity flow modeling [17], and heuristics based on substrate characteristics [11]. The main metrics of these proposals are maximizing the resource usage and minimizing the maximum link load. Our work discusses the issues involved with the allocation of spread virtual resources. The results highlighted that the minimization of resources fragmentation leads to a decrease in average substrate costs. Consequently, the fragmentation metric can be combined with those problem formulations proposed by previous works aiming an efficient usage of InP resources. In addition, we extend the discussion adding the user's perspective in terms of allocation quality.

Specifically regarding the search-space restriction and the acceleration of execution time, Ricci et al. [26] developed the *assign* program, which explores the resources' homogeneity of Emulab by the definition of equivalence classes which aggregate these resources, limiting the search space of an allocation. [27] and [5] investigate the virtual-infrastructure allocation considering mechanisms to pool back-up nodes in order to achieve the desired level of reliability together with resources allocation.

Recently, some authors have proposed decentralized solutions to allocate virtual infrastructures: [28] focused on a distributed fault-tolerant embedding and [29] investigated virtual-network embedding across multiple domains. Our work extends these discussions by exploring the geographical location of virtual infrastructures to optimize the allocation, in both the user's and the InP's perspectives. The user's perspective was developed considering the results of [10] and [4].

VII. Conclusions

Our work concentrates on the allocation of virtual infrastructures guided by the geographical location of virtual resources. We have presented an allocation-problem formulation considering both the user's and the InP's perspectives. The metrics proposed enable an analysis of the virtual-infrastructure allocation considering the user's objectives (allocation quality) as well as the InP's (cost and fragmentation).

An allocation heuristic that optimizes both perspectives has been implemented as a patented module of the Lyatiss Weaver. Our initial experiments show that is possible to improve the virtual-infrastructure allocation's quality (approximately 39% for VIs allocated on a medium-size substrate), and simultaneously decrease the physical substrate's fragmentation and the substrate's cost (almost 28% and 21% on a medium-size substrate, respectively).

Acknowledgement

This work has been funded by the FP-7 SAIL project (http://www.sail-project.eu/), the FP-7 GEYSERS project (http://www.geysers.eu/), the French ministry of Education and Research, INRIA, and CNRS, via ACI GRID's Grid'5000 project and Aladdin ADT. The authors would like to thank Augustin Ragon for his contribution.

References

[1] "Amazon's Elastic Compute Clouds (EC2)." [Online]. Available: http://aws.amazon.com/ec2

[2] "Enomaly's Elastic Computing Platform (ECP)." [Online]. Available: http://www.enomaly.com

[3] "3tera's AppLogic." [Online]. Available: http://www.3tera.com

[4] T. Truong Huu, G. Koslovski, F. Anhalt, P. Vicat-Blanc Primet, and J. Montagnat, "Joint elastic cloud and virtual network framework for application performance optimization and cost reduction," *Journal of Grid Computing (JoGC)*, 2010.

[5] G. Koslovski, W.-L. Yeow, C. Westphal, T. Truong Huu, J. Montagnat, and P. Vicat-Blanc Primet, "Reliability support in virtual infrastructures," *2nd IEEE International Conference on Cloud Computing Technology and Science (CloudCom 2010)*, 2010.

[6] B. Rochwerger, D. Breitgand, E. Levy, A. Galis, K. Nagin, I. M. Llorente, R. Montero, Y. Wolfsthal, E. Elmroth, J. Caceres, M. Ben-Yehuda, W. Emmerich, and F. Galan, "The reservoir model and architecture for open federated cloud computing," *IBM Journal of Research and Development*, vol. 53, no. 4, 2009.

[7] "Scalable and Adaptive Internet Solutions - SAIL project," 2010. [Online]. Available: http://www.sail-project.eu/

[8] "Exploring networks of the future - GENI project." [Online]. Available: http://www.geni.net/

[9] "Generalised Architecture for Dynamic Infrastructure Services - GEYSERS project," 2010. [Online]. Available: http://www.geysers.eu/

[10] G. Koslovski, T. Truong Huu, J. Montagnat, and P. Vicat-Blanc Primet, "Executing distributed applications on virtualized infrastructures specified with the VXDL language and managed by the HIPerNET framework," in *First International Conference on Cloud Computing (CLOUDCOMP'09)*, Munich, Germany, Oct. 2009.

[11] Y. Zhu and M. Ammar, "Algorithms for assigning substrate network resources to virtual network components," *INFOCOM 2006. 25th IEEE International Conference on Computer Communications. Proceedings*, pp. 1–12, April 2006.

[12] F. Anhalt, G. Koslovski, and P. Vicat-Blanc Primet, "Specifying and provisioning Virtual Infrastructures with HIPerNET," *ACM Int. Journal of Network Management (IJNM) - Special issue on Network Virtualization and its Management*, 2010.

[13] F. Anhalt, D. M. Divakaran, and P. Vicat-Blanc Primet, "A virtual switch architecture for hosting virtual networks on the internet," in *11th International Conference on High Performance Switching and Routing*, Dallas, Texas, USA, 6 2010.

[14] G. Koslovski, P. Vicat-Blanc Primet, and A. S. Charão, "VXDL: Virtual Resources and Interconnection Networks Description Language," in *The Second International Conference on Networks for Grid Applications (GridNets 2008)*, Oct. 2008.

[15] N. M. M. K. Chowdhury and R. Boutaba, "Network virtualization: State of the art and research challenges," *IEEE Commun. Mag.*, vol. 47, no. 7, pp. 20–26, Jul. 2009.

[16] M. Yu, Y. Yi, J. Rexford, and M. Chiang, "Rethinking virtual network embedding: substrate support for path splitting and migration," *SIGCOMM Comput. Commun. Rev.*, vol. 38, no. 2, pp. 17–29, 2008.

[17] N. M. M. K. Chowdhury, M. R. Rahman, and R. Boutaba, "Virtual network embedding with coordinated node and link mapping," *INFOCOM 2009. 28th IEEE International Conference on Computer Communications. Proceedings*, April 2009.

[18] L. P. Cordella, P. Foggia, C. Sansone, and M. Vento, "A (sub)graph isomorphism algorithm for matching large graphs," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 26, no. 10, pp. 1367–1372, Oct. 2004.

[19] J. Lischka and H. Karl, "A virtual network mapping algorithm based on subgraph isomorphism detection," in *ACM SIGCOMM workshop VISA '09*, 2009.

[20] G. Koslovski and P. Vicat-Blanc Primet, "VxAlloc. APPcode: IDDN.FR.001.260009.000.S.P.2009.000.10800," Lyon, France, 2010. [Online]. Available: http://www.lyatiss.com/

[21] P. Vicat-Blanc Primet, J.-P. Gelas, O. Mornard, G. Koslovski, V. Roca, L. Giraud, J. Montagnat, and T. T. Huu, "A scalable security model for enabling dynamic virtual private execution infrastructures on the internet," in *IEEE/ACM International Conference on Cluster Computing and the Grid (CCGrid2009)*, May 2009.

[22] F. Cappello, P. Primet et al., "Grid'5000: A Large Scale and Highly Reconfigurable Grid Experimental Testbed," in *GRID '05: Proceedings of the 6th IEEE/ACM International Workshop on Grid Computing*. IEEE Computer Society, 2005, pp. 99–106.

[23] K. Calvert, M. B. Doar, A. Nexion, E. W. Zegura, G. Tech, and G. Tech, "Modeling internet topology," *IEEE Communications Magazine*, 1997.

[24] A. Gupta, J. Kleinberg, A. Kumar, R. Rastogi, and B. Yener, "Provisioning a virtual private network: a network design problem for multicommodity flow," in *STOC '01: Proc. of the 33 annual ACM symposium on Theory of computing*. New York, USA: ACM, 2001.

[25] J. Lu and J. Turner, "Efficient mapping of virtual networks onto a shared substrate," Washington University in St. Louis, Tech. Rep., 2006.

[26] R. Ricci, C. Alfeld, and J. Lepreau, "A solver for the network testbed mapping problem," *SIGCOMM Computer Comm. Rev.*, 2003.

[27] W.-L. Yeow, C. Westphal, and U. C. Kozat, "Designing and embedding reliable virtual infrastructures," in *Proc. of 2nd ACM VISA workshop*, Sep. 2010.

[28] I. Houidi, W. Louati, D. Zeghlache, P. Papadimitriou, and L. Mathy, "Adaptive virtual network provisioning," in *Proc. of 2nd ACM VISA workshop*, ser. VISA '10. ACM, 2010.

[29] M. Chowdhury, F. Samuel, and R. Boutaba, "Polyvine: policy-based virtual network embedding across multiple domains," in *Proc. of 2nd ACM VISA workshop*, ser. VISA '10. ACM, 2010.