A Multi-Objective Approach for Multi-Cloud Infrastructure Brokering in Dynamic Markets

Lino Chamorro

Politechnic School. National University of Asunción.
Campus, San Lorenzo. Paraguay.
lgchamorro@gmail.com

Abstract. Cloud Service Brokers (CSBs) simplify complex resource allocation decisions, efficiently linking up the tenant dynamic requirements in to providers dynamic offers, where several objectives should ideally be considered. Nowadays, both demands and offers should be considered in dynamic environments, representing particular challenges in cloud computing markets. This work proposes for the first time a pure multi-objective formulation of a broker-oriented Virtual Machine Placement (VMP) problem for dynamic environments, simultaneously optimizing following objective functions: (1) Total Infrastructure CPU (TICPU), (2) Total Infrastructure Memory (TIMEM) and (3) Total Infrastructure Price (TIP) subject to load balancing across providers. To solve the formulated multi-objective problem, a Multi-Objective Evolutionary Algorithm (MOEA) is proposed. When a change arises in the demands or in the offers, a set of non-dominated solutions is found (usually more than one solution), selection strategies were considered in order to automatically select a solution at each reconfiguration. The proposed MOEA and selection strategies, were compared in different scenarios composed by real data from providers in actual markets. Experimental results demonstrate the good quality of the obtained solutions for the proposed scenarios.

Keywords: Cloud Computing, Cloud Brokering, Virtual Machine Placement, Multi-Objective Optimization, Evolutive Algorithm

1 Introduction

Cloud computing datacenters delivers infrastructure, platform and software as services available to end users in a pay-as-you-go basis [23]. Particularly, the Infrastructure as a Service (IaaS) model provides processing, storage, network and other fundamental computing resources where a customer is able to deploy and run arbitrary software, which can include operating systems and applications [1]. Cloud computing markets are currently composed by a wide ecosystem of heterogeneous Cloud Service Providers (CSPs) with different pricing schemes, virtual machine (VM) offers and features. Also, there are heterogeneous Cloud Service Tenants (CSTs) with different requirements and budgets to deploy complex cloud infrastructures. In this context, Cloud Service Brokers (CSBs) play a strategic role providing the CSTs with abstractions of complex resource allocation decisions, mapping specific requirements of each cloud infrastructure according to particular CST preferences (demands) onto available resources of CSPs (offers) [2].

Efficiently mapping demands from CSTs to available CSP offers could be defined as broker-oriented Virtual Machine Placement (VMP) [3]. In the VMP literature, this process is also known as Cloud Application Brokerage (CAB) [4]. CSBs allow CSTs to deploy multi-cloud infrastructures, avoiding vendor lock-in problems, while optimizing infrastructure costs and improving infrastructure performance [4]. Multi-cloud deployment architecture help CSTs avoid vendor-locking problems, also help optimizing total economical costs of the virtual infrastructure and even improve performance of cloud applications compared to a single-cloud (i.e. deploying all VMs of a cloud application in only one CSP) [4].

Broker-oriented VMP problems could be studied in both static and dynamic environments [3]. In static environments, CSP offers and CST requirements do not change any time; in dynamic environments CSPs could change VMs features, offers and prices over time, while CSTs could increase or decrease their demands such as resources requirements. The broker-oriented VMP problem could be formulated considering mono-objective or multi-objective approaches [3]. To the best of the authors' knowledge, there is no published work presenting a multi-objective formulation of the considered problem for dynamic environments. This work proposes for the first time a multi-objective formulation of the problem for dynamic environments, simultaneously optimizing the following three objective functions: (1) Total Infrastructure CPU (TICPU), (2) Total Infrastructure Memory (TIMEM) and (3) Total Infrastructure Price (TIP) while considering load balancing across providers. Considered dynamic environment parameters are: (1) dynamic VM type offers from CSPs, (2) dynamic pricing schemes from CSPs and (3) variable number of required VMs from CSTs.

For solve the formulated multi-objective problem, a Multi-Objective Evolutionary Algorithm (MOEA) is also proposed. Given that when a change arises in the offer or in the demand, a set of non-dominated solutions is initially found, to latter select a unique solution. Differents selection strategies were evaluated in order to automatically select one solution of the Pareto set [24] at each discrete time instant. Obtained results were evaluated in different scenarios with real data from cloud providers.

Research objectives of this work are:

- **Objective 1:** Propose for the first time a pure multi-objective formulation of a broker-oriented VMP problem, simultaneously optimizing three objective functions: (1) TICPU, (2) TIMEM and (3) TIP, subject to load balancing across providers and takeing account dynamic parameters such as: (1) dynamic VM resources offers, (2) dynamic pricing schemes from CSPs and (3) variables requirements in terms of resources and budget from CSTs.
- **Objective 2:** Develop a Multi-Objective Evolutionary Algorithm (MOEA) that is able to effectively solve large-scale instances of the proposed formulation.

– **Objective 3:** Perform experimental assessments of the obtained results by MOEA developed, considering the selection strategies of the solution from a Pareto set: (S1) random, (S2) minimum distance to origin, (S3) preferred solution, (S4) maximum TICPU, (S5) maximum TIMEM and (S6) minimum TIP.

The remainder of this work is organized as follows: Section 2 presents related works summary and research motivation. Next, Section 3 details the proposed multi-objective formulation of a broker-oriented VMP problem in dynamic environments. Design concepts of the proposed MOEA are explained in Section 4 while in Section 6.1 experimental results are summarized. Finally, conclusions and future work are left to Section 5.

2 Related Work and Motivation

Broker-oriented VMP problems have been mostly studied in static environments in the literature, considering both mono-objective and multi-objective approaches. Dynamic environments were briefly explored, but with a mono-objective optimization approach.

2.1 Mono-Objective Brokerage in Static Environments

Tordsson et al. proposes in [4], scheduling algorithms for the cloud application brokering (CAB) problem in static environments, taking into account fixed CST requirements, fixed possible number of VM hardware configurations and known hourly prices for running VMs in a CSP. The proposed Integer Linear Programming (ILP) model considers the maximization of the Total Infrastructure Capacity (TIC) while defining a maximum budget constraint [4]. CSTs may also consider the following deployment constraints: (1) VM hardware configurations, where a minimum and maximum instance type indexes are requested, (2) number of VMs of each instance type, where a minimum and maximum number of VMs of each instance type are specified and (3) load balancing, where a minimum and maximum percentage of VMs that can be located at each CSP are defined. It should be mentioned that the TIC objective proposed in [4] represents a particular capacity of CPU rather than other possible resources. To avoid ambiguous terminology, this work redefines the mentioned objective function as Total Infrastructure CPU (TICPU).

Chaisiri et al. proposes in [5], a Stochastic Linear Programming (SLP) model to minimize the costs associated to hosting VMs in a multi-cloud deployment architecture under future demand and price uncertainty, subject to constraints to ensure that the requested demand is met and the allocations of VMs do not exceed the resource capacity offered by CSPs. This work considers the costs minimized in [5] as Total Infrastructure Price (TIP). The proposed model take account two payment plans for CSTs: (1) reservation and (2) on-demand, as offered by Amazon Web Services [6]. Extending the work proposed in [5], Mark et al.

proposed in [7] an Evolutionary Optimal Virtual Machine Placement (EOVMP) algorithm with a demand forecaster to allocate VMs using reservation and on-demand plans for job processing. The proposed EOVMP is a hybridized algorithm of Genetic Algorithm (GA), Particle Swarm Optimization (PSO) and Ant Colony Optimization (ACO). The model proposed in [7] minimizes the total price of the virtual infrastructure, denoted in this work as TIP.

2.2 Multi-Objective Brokerage in Static Environments

Kessaci et al. proposes in [8], a job scheduler using a Multi-Objective Evolutionary Algorithm (MOEA) considering response time and service price optimization in order to maximize the CSTs satisfaction and simultaneously maximize the CSBs profit, providing a Pareto set of non-dominated solutions rather than a single solution.

Amato and Venticinque defines in [9, 10] that cloud brokering problems may need to deal with several and (at the same time) contradictory objectives, finally combining all objectives into just one objective function with a weighted sum method. The considered objectives include [9, 10]: (1) TIP, (2) TICPU, (3) CSPs Reputation and/or (4) Availability. The mentioned objective functions could also be considered as two different types of constraints: mandatory (hard) and optional (soft) constraints. The developed brokering tool is flexible enough to define a custom model to meet particular CST requirements [9, 10].

2.3 Mono-Objective Brokerage in Dynamic Environments

Lucas-Simarro et al. proposes in [2], an optimization model for CSTs virtual cluster placement across available CSPs offers. This scheduler considers average prices or cloud price trends to suggest an optimal multi-cloud deployment. A mono-objective approach is considered for TIP minimization, selecting the best possible combination of CSPs that offer the lowest prices using an ILP formulation. The following constraints are also considered in the proposed model [2]: (1) distance, representing a minimum and maximum number of VMs that can be relocated across CSPs (to guarantee a certain number of VMs working all the time) and (2) load balancing, where a minimum and maximum percentage of all VMs should be located at each CSP (to avoid vendor lock-in problems).

Li et al. proposes in [11], an ILP formulation for cloud service brokering in dynamic environments where instance types, prices and service performance are continuously changing throughout the service life-cycle. This work proposes three different mono-objective optimization models: (1) TIC maximization, (2) TIP minimization and (3) Migration Costs (MC) minimization. The following constraints are applied to each optimization model: (1) budget, where TIP cannot exceed a specified budget limit, (2) unique placement, where each VM has to be of exactly one instance type and placed in exactly one CSP and (3) load balancing, representing a minimum and maximum percentage of all VMs to be located at each CSP. These optimization models were experimentally evaluated in different dynamic scenarios: (1) new instance type offers, (2) changing prices and (3) service performance elasticity (number of VMs).

Since the scalability limitations of the ILP formulation proposed in [11], Chamorro et al. presented in [12] a Genetic Algorithm (GA) for dynamic cloud application brokerage also considering a mono-objective optimization approach. The [12] works studies a broker-oriented VMP problem with large problem instances considering dynamic environments such as: (1) variable resource offers and (2) varying pricing from CSPs, as well as (3) dynamic requirements of CSTs.

2.4 Motivation: Multi-Objective Brokerage in Dynamic Environments

Since that several different objective functions were already studied in the broker-oriented VMP literature, these important objectives should be ideally taken into account simultaneously. Trending dynamic markets of Infrastructure as a Service (IaaS) present competitive advantages that should be exploited by formulating the broker-oriented VMP problem considering dynamic parameters in actual cloud markets. Then, this work proposes for the first time a pure multi-objective formulation of a broker-oriented VMP problem in dynamic environments for the simultaneous optimization of the following three objective functions: (1) Total Infrastructure CPU (TICPU), (2) Total Infrastructure Memory (TIMEM) and (3) Total Infrastructure Price (TIP) while considering a Minimum Distribution Index (LOC_{min}) constraint to meet load balancing of VMs among CSPs, avoiding vendor lock-in problems.

This work additionally considers as an important resource for any type of virtual infrastructure or application the RAM Memory, denoted as TIMEM. Also this work takes account soft constraints associated to each objective function cost to guide the decision space exploitation and reduce the number of non-dominated solutions in a Pareto set approximation [13].

To be able of effectively scale the resolution of the formulated broker-oriented VMP problem to large problem instances, a MOEA is proposed. Considering that the output of the proposed algorithm is a set of non-dominated solutions, a single solution could be manually selected by a decision maker according to particular needs. In order to facilitate it, this work uses different selection strategies to automatically select a convenient solution at each stage, i.e. at any change of the considered parameters that compose a dynamic environment.

In summary, the main contributions of this work are:

- a first pure multi-objective formulation of a broker-oriented VMP problem, simultaneously optimizing three objective functions: (1) TICPU, (2) TIMEM and (3) TIP, subject to load balancing across providers for dynamic environments: (1) dynamic VM type offers from CSPs, (2) dynamic pricing schemes from CSPs and (3) variable number of required VMs from CSTs;
- a Multi-Objective Evolutionary Algorithm (MOEA) that is able of effectively solve large-scale instances of the proposed formulation of the problem; and

– experimental assessments of selection strategies for automatically selecting a convenient solution from a Pareto set approximation for the studied problem: (S1) random, (S2) minimum distance to origin, (S3) preferred solution, (S4) maximum TICPU, (5) maximum TIMEM and (6) minimum TIP.

3 Proposed Multi-Objective Broker-oriented VMP Formulation

Based on the state of the art research made by the author, this work presents for the first time a formulation of a broker-oriented VMP problem in dynamic environments, simultaneously optimizing the following objective functions: (1) TICPU, (2) TIMEM and (3) TIP subject to load balancing of VMs between CSPs. Formally, the proposed pure multi-objective broker-oriented VMP problem formulation can be enunciated as follows. Given:

– a set of m CSPs;
– a set of $l(t)$ instance types IT_j available at each CSP c_k (denoted as $IT_{j,k}$), including its characteristics (as explained in Section 3.1);
– a set of $n(t)$ VMs v_i to be deployed across available CSPs c_k lifetime expected of the requested infrastructure;

it's sought convenient combinations of instance types and CSPs to deploy the requested VMs in dynamic cloud computing markets, satisfying the constraints and simultaneously optimizing all objective functions defined in this formulation in a pure multi-objective context. Based on hourly prices, each instant t typically represents an hour of resource provisioning and placement reconfiguration is triggered for next instant $t+1$ only if a change on any of the dynamic parameters of the cloud computing market is detected and whether it represents a competitive advantage for the customer.

3.1 Problem Inputs

The proposed broker-oriented VMP problem receives information as data input, maintaining an updated information on CSP offers and CST requirements. Thus, this work considers a dynamic environment composed by the following dynamic parameters: (1) new instance types may be introduced in the cloud market, (2) changes in prices of resources and (3) changes in the number of requested VMs or CST available budget.

The set of available CPU resources in the cloud computing market is represented as a matrix $CPU(t) \in \mathbb{Z}^{l(t)\times m}$, where each element $CPU_{j,k}$ represents the available CPU resources associated to each instance type IT_j at each c_k CSP.

$$CPU(t) = \begin{bmatrix} CPU_{1,1} & CPU_{1,2} & \dots & CPU_{1,m} \\ \dots & \dots & \dots & \dots \\ CPU_{l(t),1} & CPU_{l(t),2} & \dots & CPU_{l(t),m} \end{bmatrix} \qquad (1)$$

where:

$CPU_{j,k}$:	CPU resources [in # of cores] associated to each IT_j at each c_k;
$l(t)$:	Number of instance types IT_j at time instant t, where $1 \leq j \leq l(t)$;
m:	Number of CSPs c_k, where $1 \leq k \leq m$.

Similarly, the set of available RAM memory resources is represented as a matrix $MEM(t) \in \mathbb{Z}^{l(t)\times m}$. This set of resources can be formulated as:

$$MEM(t) = \begin{bmatrix} MEM_{1,1} & MEM_{1,2} & \dots & MEM_{1,m} \\ \dots & \dots & \dots & \dots \\ MEM_{l(t),1} & MEM_{l(t),2} & \dots & MEM_{l(t),m} \end{bmatrix} \tag{2}$$

where:

$MEM_{j,k}$:	Memory resources [in GB] associated to each IT_j at each c_k;
$l(t)$:	Number of instance types IT_j at time instant t, where $1 \leq j \leq l(t)$;
m:	Number of CSPs c_k, where $1 \leq k \leq m$.

Note that $CPU(t)$ and $MEM(t)$ matrices are functions of time because new instance type offers may be introduced in the cloud computing market (e.g. Amazon EC2 announced micro instances [14, 15]). Consequently, the number of instance types $l(t)$ could vary, representing potential placement reconfigurations.

The set of prices associated to IT_j at each CSP c_k is represented as a matrix $PRC(t) \in \mathbb{R}^{l(t)\times m}$, where each element $PRC_{j,k}(t)$ represents its hourly price.

$$PRC(t) = \begin{bmatrix} PRC_{1,1}(t) & PRC_{1,2}(t) & \dots & PRC_{1,m}(t) \\ \dots & \dots & \dots & \dots \\ PRC_{l(t),1}(t) & PRC_{l(t),2}(t) & \dots & PRC_{l(t),m}(t) \end{bmatrix} \tag{3}$$

where:

$PRC_{j,k}(t)$:	Price [in $] associated to each IT_j at each c_k;
$l(t)$:	Number of instance types IT_j at time instant t, where $1 \leq j \leq l(t)$;
m:	Number of CSPs c_k, where $1 \leq k \leq m$.

Additionally, this work takes account the overhead for placement reconfigurations in the proposed simulation, information about allocation ($AT_{j,k}$) and release times ($RT_{j,k}$) [in hours] of each $IT_{j,k}$ are represented in the data input of the formulated broker-oriented VMP problem. These information could be experimentally obtained by CSBs, as presented by Iosup et al. in [21]. Similarly to the cited matrices, both sets of allocation ($AT_{j,k}$) and release ($RT_{j,k}$) times associated to each $IT_{j,k}$ are represented as $AT(t)$ and $RT(t) \in \mathbb{R}^{l(t)\times m}$ respectively. These can be formulated as:

$$AT(t) = \begin{bmatrix} AT_{1,1} & AT_{1,2} & \dots & AT_{1,m} \\ \dots & \dots & \dots & \dots \\ AT_{l(t),1} & AT_{l(t),2} & \dots & AT_{l(t),m} \end{bmatrix} \tag{4}$$

and:

$$RT(t) = \begin{bmatrix} RT_{1,1} & RT_{1,2} & \dots & RT_{1,m} \\ \dots & \dots & \dots & \dots \\ RT_{l(t),1} & RT_{l(t),2} & \dots & RT_{l(t),m} \end{bmatrix} \tag{5}$$

where:

$AT_{j,k}$:	Allocation time [in hours] associated to each IT_j at each c_k;
$RT_{j,k}$:	Release time [in hours] associated to each IT_j at each c_k;
$l(t)$:	Number of instance types IT_j at time instant t, where $1 \leq j \leq l(t)$;
m:	Number of CSPs c_k, where $1 \leq k \leq m$.

CSTs have to specify the number of requested VMs to be deployed, denoted as $n(t)$. Note that the number of VMs could be dynamically adjusted according to particular CST requirements. Also, CST defines an estimated lifetime of the infrastructure [in hours] (denoted as $H(t)$) and it indicates the time that the all requested VMs will remain in operation.

3.2 Problem Outputs

A matrix $P(t) \in \mathbb{B}^{n(t)\times l(t)\times m}$, composed by decision variables $x_{i,j,k}(t)$, represents a possible instance type selection and placement of VMs on available CSPs at instant t. Consequently, the output data of the proposed multi-objective broker-oriented VMP problem is a new matrix $P(t+1) \in \mathbb{B}^{n(t+1)\times l(t+1)\times m}$, composed by $x_{i,j,k}(t+1)$, representing a new instance type selection and placement of VMs on available CSPs at the next instant $t+1$, considering changes presented in the cloud market. It is important to note that $P(t+1)$ is selected in this work from a set of non-dominated solutions (Pareto set approximation).

3.3 Objective Functions

If one or more of the dynamic parameters considered in the proposed formulation present a change in the cloud computing market (see Section 3.1), new opportunities could be exploited by reconfiguring the current placement of requested VMs (e.g. migrating VMs among CSPs and/or changing selected instance types). Inspired in [11], this work presents a Reconfiguration Overhead (RO) based on the wasted resources during each placement reconfiguration period, considered for both TICPU and TIMEM objectives to be presented in the following subsections.

The **Total Infrastructure CPU** objective function is defined as the CPU cores sum of the instance types selected for the execution of each VM required by the CST, subtracting a possible overhead reconfiguration of the infrastructure. The Total Infrastructure CPU $TICPU$ is mathematically formulated below:

$$TICPU = H(t) \times \sum_{i=1}^{n(t)} \sum_{j=1}^{l(t)} \sum_{k=1}^{m} CPU_{j,k} \times x_{i,j,k}(t+1) \qquad (6)$$

where:

$TICPU$:	Total Infrastructure CPU;
$H(t)$:	Expected remaining lifetime of the infrastructure [in hours];
$l(t)$:	Number of instance types IT_j at time instant t, where $1 \leq j \leq l(t)$;
m:	Number of CSPs c_k, where $1 \leq k \leq m$;
$CPU_{j,k}$:	CPU capacity [in # of cores] of instance type IT_j at CSP c_k;
$x_{i,j,k}(t)$:	Binary variable equals 1 if v_i is of instance type IT_j and is located at CSP c_k at instant t; otherwise, it is 0.

CPU Overhead, the RO_{CPU} is modeled as the CPU capacity wasted during the reconfiguration period (i.e. allocation and release time [in hours] of modified VMs). This overhead can mathematically be formulated as:

$$RO_{CPU} = \sum_{i=1}^{n(t)} \sum_{\substack{j=1 \\ j\prime=1}}^{l(t)} \sum_{\substack{k=1 \\ k\prime=1}}^{m} [CPU_{j,k} \times RT_{j,k} \times x_{i,j,k}(t) + CPU_{j\prime,k\prime} \times AT_{j\prime,k\prime} \times x_{i,j\prime,k\prime}(t+1)] \qquad \forall j \neq j\prime; k \neq k\prime \tag{7}$$

where:

RO_{CPU}:	Total CPU reconfiguration overhead;
$n(t)$:	Number of requested VMs v_i, where $1 \leq i \leq n(t)$;
$l(t)$:	Number of instance types IT_j at time instant t, where $1 \leq j \leq l(t)$;
m:	Number of CSPs c_k, where $1 \leq k \leq m$;
$CPU_{j,k}$:	CPU resources [in # of cores] associated to each IT_j at each c_k;
$RT_{j,k}$:	Release time [in hours] associated to each IT_j at each c_k;
$AT_{j,k}$:	Allocation time [in hours] associated to each IT_j at each c_k;
$x_{i,j,k}(t)$:	Binary variable equals 1 if v_i is of instance type IT_j and is located at CSP c_k at instant t; otherwise, it is 0.

Finally, the objective function $f_1(x)$ is expressed as the difference between the TICPU and its corresponding RO_{CPU}(there is a release and allocation time conversion from seconds to hours):

$$f_1(x) = TICPU - RO_{CPU} \tag{8}$$

The **Total Infrastructure Memory** objective function is defined as the RAM Memory sum (in GB) of the instance types selected for the execution of each VM required by the CST, subtracting a possible overhead reconfiguration of the infrastructure. The Total Infrastructure Memory $TIMEM$ is mathematically formulated below:

$$TIMEM = H(t) \times \sum_{i=1}^{n(t)} \sum_{j=1}^{l(t)} \sum_{k=1}^{m} MEM_{j,k} \times x_{i,j,k}(t+1) \tag{9}$$

where:

$TIMEM$:	Total Infrastructure RAM memory;
$H(t)$:	Expected remaining lifetime of the infrastructure [in hours];
$n(t)$:	Number of requested VMs v_i, where $1 \leq i \leq n(t)$;
$l(t)$:	Number of instance types IT_j at time instant t, where $1 \leq j \leq l(t)$;
m:	Number of CSPs c_k, where $1 \leq k \leq m$;
$MEM_{j,k}$:	RAM Memory capacity [in GB] of instance type IT_j at CSP c_k;
$x_{i,j,k}(t)$:	Binary variable equals 1 if v_i is of instance type IT_j and is located at CSP c_k at instant t; otherwise, it is 0.

RAM Memory Overhead, the RO_{MEM} is modeled as the RAM Memory capacity wasted during the reconfiguration period (i.e. allocation and release time [in hours] of modified VMs). This overhead can mathematically be formulated as:

$$RO_{MEM} = \sum_{i=1}^{n(t)} \sum_{\substack{j=1\\ j'=1}}^{l(t)} \sum_{\substack{k=1\\ k'=1}}^{m} (MEM_{j,k} \times RT_{j,k} \times x_{i,j,k}(t) + MEM_{j',k'} \times AT_{j',k'} \times x_{i,j',k'}(t+1))$$
$$\forall j \neq j'; k \neq k' \tag{10}$$

where:

RO_{MEM}:	Total RAM memory reconfiguration overhead;
$n(t)$:	Number of requested VMs v_i, where $1 \leq i \leq n(t)$;
$l(t)$:	Number of instance types IT_j at time instant t, where $1 \leq j \leq l(t)$;
m:	Number of CSPs c_k, where $1 \leq k \leq m$;
$MEM_{j,k}$:	RAM memory resources [in GB] associated to each IT_j at each c_k;
$RT_{j,k}$:	Release time [in hours] associated to each IT_j at each c_k;
$AT_{j,k}$:	Allocation time [in hours] associated to each IT_j at each c_k;
$x_{i,j,k}(t)$:	Binary variable equals 1 if v_i is of instance type IT_j and is located at CSP c_k at instant t; otherwise, it is 0.

Finally, the objective function $f_2(x)$ is expressed as the difference between the TIMEM and its corresponding RO_{MEM}(there is a release and allocation time conversion from seconds to hours):

$$f_2(x) = TIMEM - RO_{MEM} \tag{11}$$

The **Total Infrastructure Price** objective function is defined as the execution price sum (in USD) of the instance types selected for each VM required by the CST. The Total Infrastructure Price TIP is mathematically formulated below:

$$f_3(x) = TIP = H(t) \times \sum_{i=1}^{n(t)} \sum_{j=1}^{l(t)} \sum_{k=1}^{m} PRC_{j,k}(t) \times x_{i,j,k}(t+1) \tag{12}$$

where:

TIP:	Total Infrastructure Price;
$H(t)$:	Expected lifetime of the infrastructure [in hours];
$n(t)$:	Number of requested VMs v_i, where $1 \leq i \leq n(t)$;
$l(t)$:	Number of instance types IT_j at time instant t, where $1 \leq j \leq l(t)$;
m:	Number of CSPs c_k, where $1 \leq k \leq m$;
$PRC_{j,k}(t)$:	Price [in \$] of instance type IT_j at CSP c_k;
$x_{i,j,k}(t)$:	Binary variable equals 1 if v_i is of instance type IT_j and is located at CSP c_k at instant t; otherwise, it is 0.

3.4 Constraints

Feasible solutions of the proposed broker-oriented VMP problem are restricted by the following constraints:

– unique placement of VMs;
– load balancing of VMs among CSPs;
– lower and upper bounds (according to optimization context) associated to each objective function $f_1(x)$ to $f_3(x)$.

Each of these constraints are detailed as follows.

Unique placement of VMs. For each VM v_i, $\forall i \in [1, \ldots, n(t)]$, should be provisioned selecting a single instance type IT_j, $\forall j \in [1, \ldots, l(t)]$, and located to run on a single CSP c_k, $\forall k \in [1, \ldots, m]$.

Load balancing of VMs between CSPs. In order to avoid vendor lock-in problems [11], a load balancing constraint (LOC_{min}) is modeled as a minimum percentage of VMs to be located at each CSP c_k, $\forall k \in [1, \ldots, m]$.

Adjustable constraints. Considering the multi-objective formulation of the proposed broker-oriented VMP problem (Section 3.5), the set of non-dominated solutions may comprise a large number of feasible solutions, being increasingly hard of discriminate between solutions using only the dominance relation [22]. For this reason, this work proposes the utilization of lower and upper threshold (depending on the optimization context) associated to each objective function $z \in \{1, \ldots, q\}$ ($L_z \leq f_z(x) \leq U_z$) in order to be able of reduce the number of possible solutions to be delivered to the CST particular requirements. Since for the CSTs may be hard to define appropriate thresholds because these values are unknown a-priori, these lower and upper thresholds are modeled as soft constraints [9], where a percentage of the expected thresholds could be exceeded, always depending on the optimization context (e.g. if a CST expects at least 100 [GB] for TIMEM with an acceptable margin of 10%, the TIMEM of feasible solutions for $f_2(x)$ must be higher or equal to L_2=90 [GB] or if CST have a budget of 120 USD for all the infrastructure with an acceptable margin of 10%, the TIP of feasible solutions for $f_3(x)$ must be less or equal to U_3=132 USD).

3.5 Multi-Objective Problem Formulation

Finally, the problem formulation of the broker-oriented VMP problem studied in this work is based on several objectives functions and constraints detailed in 3.3 and 3.4 respectively. The problem formulation is proposed as follows:
Maximize:

$$y_1 = f(x) = [f_1(x), f_2(x)] \tag{13}$$

y *Minimize:*

$$y_2 = f(x) = [f_3(x)] \tag{14}$$

where:

$$\begin{aligned} f_1(x) &= \text{Total Infrastructure CPU (TICPU);} \\ f_2(x) &= \text{Total Infrastructure Memory (TIMEM);} \\ f_3(x) &= \text{Total Infrastructure Price (TIP).} \end{aligned} \tag{15}$$

subject to:

$$
\begin{aligned}
&e_1(x) : \text{unique placement of VMs;}\\
&e_2(x) : \text{load balancing of VMs between CSPs;}\\
&e_3(x) : f_1(x) \geq L_1;\\
&e_4(x) : f_2(x) \geq L_2;\\
&e_5(x) : f_3(x) \leq U_3;
\end{aligned} \tag{16}
$$

4 Proposed Multi-Objective Evolutionary Algorithm (MOEA)

In order to solve the proposed multi-objective broker-oriented VMP problem presented in the Section 3, a Multi-Objective Evolutionary Algorithm (MOEA) was developed, taking into account that it is a well studied solution technique presenting good results for a large set of combinatorial optimization problems [13]. The proposed MOEA is inspired in the Non-dominated Sorting Genetic Algorithm (NSGA-II) [17] and it mainly works in the following way (see Algorithm 1):

At step 1, a set P_0 of candidates is randomly generated. These candidates are repaired at step 2 to ensure that P_0 contains only feasible solutions. With the obtained non-dominated solutions, the first set P_{known} (Pareto set approximation) is generated at step 3, considering lower and upper bounds associated to each objective function $z \in \{1, \ldots, q\}$ ($L_z \leq f_z(x) \leq U_z$). After initialization at step 4, evolution begins (between steps 5 and 12).

Algorithm 1: Proposed MOEA for multi-objective broker-oriented VMP.

```
   Data: CPU(t), MEM(t), PRC(t), AT(t), RT(t), n(t), H(t), P(t), selection
         strategy.   See Section 3.1 for notation details.
   Result: P(t+1). See Section 3.2 for notation details.
 1 initialize set of solutions $P_0$
 2 $P'_0$ = repair infeasible solutions of $P_0$
 3 update set of solutions $P_{known}$ from $P'_0$ applying lower and upper bounds
 4 $u = 0; P_u = P'_0$
 5 while is not stopping criterion do
 6     $Q_u$ = selection of solutions from $P_u \cup P_{known}$
 7     $Q'_u$ = crossover and mutation of solutions of $Q_u$
 8     $Q''_u$ = repair infeasible solutions of $Q'_u$
 9     update set of solutions $P_{known}$ from $Q''_u$ applying lower and upper bounds
10     increment number of generations $u$
11     $P_u$ = non-dominated sorting from $P_u \cup Q''_u$
12 end
13 $P_{selected}$ = selected solution (selection strategy parameter)
14 return $P_{selected}$
15 increment instant $t$; reset $P_{known}$
```

The evolutionary process basically follows the same behavior: solutions are selected from the union of P_{known} with the evolutionary set of solutions (or population) also known as P_u (step 6), crossover and mutation operators are applied as usual (step 7), and eventually solutions are repaired, as there may be infeasible solutions (step 8). At step 9, the Pareto set approximation P_{known} is updated (if applicable); while at step 10 the generation (or iteration) counter is updated. At step 11 a new evolutionary population P_u is selected. The evolutionary process is repeated until the algorithm meets a stopping criterion (such as a maximum number of generations), returning one solution $P_{selected}$ from the set of solutions P_{known} (step 13), using one of the strategies to be presented in Section 4.3.

4.1 Chromosome representation

Considering the output parameters presented in section 3.2, the proposed MOEA represents a current instance types selection and placement of VMs on available CSPs P(t) at instant t as a chromosome. A chromosome (or solution representation of the proposed broker-oriented VMP problem) is represented as an integer matrix C(t) $\in \mathbb{Z}^{2\times n(t)}$. The first row defines the selected instance types (IT_j) for each requested VM, while the second row indicates the selected CSP (c_k) in which the VM will be deployed. Note that the chromosome column size is variable, since $n(t)$ can change over time according to CST requirements.

Example: Suppose that a CST initially requires to deploy 13 VMs $(n(t) = 7)$ at $t = 1$. There are three available CSPs $(m = 3)$ with four different instance types $(l(t) = 4)$ (see Table 1). Then, due to growing demands, the CST requires to increase number of deployed VMs $(n(t) = 10)$ at $t = 2$ (see Table 2). For simplicity no reconfigurations of instance types or CSPs are presented in this very simple example.

According to Table 1, the obtained infrastructure configuration at $t = 1$ is:

- v_1 is of the type IT_2 and is located at CSP c_1;
- v_2 is of the type IT_1 and is located at CSP c_3;
- v_3 is of the type IT_3 and is located at CSP c_1;
- v_4 is of the type IT_3 and is located at CSP c_2;
- v_5 is of the type IT_1 and is located at CSP c_2;
- v_6 is of the type IT_2 and is located at CSP c_3;
- v_7 is of the type IT_1 and is located at CSP c_3.

When demand is increased, three additional VMs are deployed $(n(t) = 10)$ at $t = 2$, resulting in the infrastructure configuration presented in Table 2, where:

- v_8 is of the type IT_2 and is located at CSP c_3;
- v_9 is of the type IT_3 and is located at CSP c_2;
- v_{10} is of the type IT_3 and is located at CSP c_1.

v_1	v_2	v_3	v_4	v_5	v_6	v_7
2	1	3	3	1	2	1
1	3	1	2	2	3	3

Table 1. Example chromosome at $t = 1$.

v_1	v_2	v_3	v_4	v_5	v_6	v_7	v_8	v_9	v_{10}
2	1	3	3	1	2	1	2	3	3
1	3	1	2	2	3	3	3	2	1

Table 2. Example chromosome at $t = 2$.

4.2 Evolutive Operators

The proposed MOEA considers a *Binary Tournament* method for selecting individuals for crossover and mutation operations [18]. The crossover operator implemented in the presented work is the single point cross-cut [18]. This work uses a mutation method in which each gene is mutated with a probability $\frac{1}{n(t)}$, where $n(t)$ represents the number of requested VMs. The fitness function considered in the proposed algorithm is based on the one presented by Deb. et al. in [17].

The population evolution in the proposed MOEA is based on the population evolution proposed in the Non-dominated Sorting Genetic Algorithm II [17]. A new population P_{u+1} is formed from the union of the best known population P_{known} and offspring population Q_u, applying non-domination rank and crowding distance operators, as defined in [17].

4.3 Solutions Selection Strategies

In Pareto based algorithms, the Pareto set approximation can include a large number of non-dominated solutions; therefore, in a dynamic environment, automatically selecting only one of the non-dominated solutions (step 13 of Algorithm 1) can be considered as a new difficulty for the problem. This work uses and evaluates the following selection strategies:

– **Random (*S1*)** Considering that the Pareto set approximation is composed by non-dominated solutions, randomly selecting one of the solutions could be an acceptable selection strategy.
– **Minimum Distance to Origin (*S2*)** The solution with the minimum Euclidean distance to the origin is selected, since all objective functions in a minimization context. For this purpose, $f_1(x)$ and $f_2(x)$ were redefined as the difference between the maximum possible value at instant t. When several solutions have equal Euclidean distance, only one of these solution is randomly selected.
– **Prefered Solution (*S3*)** A solution is defined as prefered to another when it is better in more objective functions [19]. When several solutions can be

considered as prefered ones (there is a tie), the solution with higher TICPU or TIMEM function value is selected.
- **Maximum TICPU (*S4*)** This strategy select the solution with maximum TICPU function value*.
- **Maximum TIMEM (*S5*)** This strategy select the solution with maximum TIMEM function value*.
- **Minimum TIP (*S6*)** This strategy select the solution with minimum TIP function value*.

* When several solutions have the same minimum or maximun objective function value, only one of these solutions is randomly selected.

5 Conclusion and Future Works

Current cloud computing markets have dynamic environment where the providers offers variability about pricing schemes and computational resources, and the CSTs requirement may change over time (e.g. available budget, VM resources demands). In this context, the broker-oriented VMP problem resolution represents a true challenge.

This work proposes for the first time a formulation for broker-oriented VMP problem resolution in a pure multi-objective context. This formulation simultaneously optimize three objective functions: (1) TICPU, (2) TIMEM and (3) TIP, subject to load balancing across CSPs (Objective 1). In order to solve the proposed multi-objective formulation of broker-oriented VMP problem, a Multi-Objective Evolutionary Algorithm (MOEA) was developed that is able of effectively solve large-scale instances of the problem (Objective 2).

In Pareto based algorithms, the Pareto set can include a large number of non-dominated solutions but only one can be used for the reconfiguration of the new infrastructure.
This work evaluates the following six solution selection strategies: (1) random, (2) minimum distance to origin, (3) preferred solution, (4) Maximum TICPU, (5) Maximum TIMEM and (6) Minimum TIP.

Several experiments were performed to assess the MOEA performance against large instances of the problem and to evaluate the quality of the obtained solutions. The experiments were focused on two scenarios which comprises a dynamic environment through resources offers and pricing by CSPs and requirements of CST, as well as large instances of the problem (up to 500 VMs). They proved that the algorithm obtains good solutions that meet CSTs requirements. Between the selection strategies, preferred solution strategy has showed that gets the best solution in terms of CSTs requirements (Objective 3).

As future works, the author suggests to study heterogeneity that may exist between different providers, generally in terms of computational resources. In addition, a study on the feasibility of implementing a predictive model of the behavior of CSPs offers and CSTs requirements in a purely multi-objective context is proposed.

6 Appendix

6.1 Experimental Results

Proposed MOEA (see section4) was implemented using Java programming language. The source code as well as experimental results are available online[1].

Experimental scenarios include real data input from different CSPs taken from Amazon Web Services [16], these consist of several instances types in terms of quantity of CPU and memory in GB (see Table 3), pricing schemes in USD (see Table 4), and allocation and release times in seconds (see Table 5).

Resources	**m**	**s**	**M**	**L**	**XL**
$CPU(Cores)$	1	1	2	2	4
$Memory(GB)$	1	2	4	8	16

Table 3. VMs hardware configuration per instance type. Micro(m), Small(s), Medium(M), Large(L), Extra Large(XL).

CSP	**m**	**s**	**M**	**L**	**XL**
$EC2-US$	0.013	0.026	0.052	0.104	0.239
$EC2-EU$	0.014	0.028	0.056	0.112	0.264
$EC2-OC$	0.020	0.040	0.080	0.160	0.336

Table 4. Hourly instance type prices per CSP in \$. Micro(m), Small(s), Medium(M), Large(L), Extra Large(XL).

	Allocation					*Release*				
CSP	**m**	**s**	**M**	**L**	**XL**	**m**	**s**	**M**	**L**	**XL**
$EC2-US$	71	82	85	90	64	20	21	20	20	25
$EC2-EU$	71	82	85	90	64	20	21	20	20	25
$EC2-OC$	71	82	85	90	64	20	21	20	20	25

Table 5. Allocation and release times per instance type in seconds. Micro(m), Small(s), Medium(M), Large(L), Extra Large(XL).

Table 6 summarizes parameters related to the proposed MOEA. The parameters are: number of scenarios; t represent the time period in which CSTs maintain the same requirements (for practical reasons, the value of t is constant); the proposed MOEA was was executed several times (Runs per t) in order to average

[1] https://github.com/lgchamorro/broker-VMP-MOEA

all the obtained results and finally the GA parameters (population quantity and generations).

Parameter	**Value**
# of Scenarios	2
t	24 hours
Runs per t	10
Population Size	50
Number of Generations	200

Table 6. MOEA general parameters.

CSTs can define the minimum and maximum (depending of the optimization context) values that they expect for every objective function (also knows as minimum or maximum acceptable required values) for each scenario in 7 and 8. Also, CSTs define a tolerance margin (in percent over specific expected value) within which an output value could be considered as valid. Both, expected values and tolerance are problem inputs defined by CSTs.

		Expected			Restriction	Tolerance
t	VMs	*TICPU*	*TIMEM*	*TIP*	LOC_{min} in %	Margin in %
1	100	300	1300	26	30	10
2	100	300	1300	26	30	10
3	120	380	1400	30	30	10
4	120	380	1400	30	30	10
5	120	380	1400	30	30	10
6	120	380	1400	30	30	10
7	100	300	1300	26	30	10

Table 7. *CST* requirements for Experiment 1.

Output values are assessed against the expected values in a optimization context, this means that according to the objective function, the expected value could designate an acceptable minimum (TICPU, TIMEM) or maximum (TIP). In the same way, tolerance could extend below minimum acceptable values (TICPU, TIMEM) or above the maximum acceptable value (TIP) based on the tolerance percentage against expected value indicated by CSTs.

Experiments for each of the six evaluated selection strategies were repeated 10 times, given the probabilistic nature of Evolutionary Algorithms (EAs).

Experiments were performed considering two different scenarios. The different changes that may occur in the cloud computing market are described below:

Experiment 1:

- $t = 1$. Initial infrastructure placement. Requested VMs $n(t) = 100$.

		Expected			Restriction	Tolerance
t	VMs	$TICPU$	$TIMEM$	TIP	LOC_{min} in %	Margin in %
1	400	1100	4300	100	30	10
2	400	1100	4300	100	30	10
3	500	1400	5100	130	30	10
4	500	1400	5100	130	30	10
5	500	1400	5100	130	30	10
6	500	1400	5100	130	30	10
7	400	1100	4300	100	30	10

Table 8. CST requirements for Experiment 2.

		Average		
No.	**Selection Strategy**	$TICPU$	$TIMEM$	TIP
S1	Random	2,502.69	9,474.74	165.09
S2	Distance	2,543.16	9,696.35	169.75
S3	Prefered	2,725.89	10,556.18	184.84
S4	Maximum TICPU	2,711.18	10,476.18	183.67
S5	Maximum TIMEM	2,712.62	10,501.05	183.87
S6	Minimum TIP	2,315.17	8,567.42	148.20

Table 9. Obtained averaged results for Experiment 1.

- $t = 2$. Changes in CSP offers. New instance type offer (micro instances).
- $t = 3$. Changes in CST requirements. CST needs more VMs $n(t) = 120$.
- $t = 4$. Changes in CSP offers. Nighttime instance type discounts. The $EC2-OC$ provider offers 50% off.
- $t = 5$. Changes in CSP offers. Nighttime instance type discounts ends.
- $t = 6$. Changes in CSP offers. An instance type is removed from the market (xlarge instances).
- $t = 7$. Changes in CST requirements. CST needs less VMs $n(t) = 100$.

Experiment 2:

- $t = 1$. Initial infrastructure placement. Requested VMs $n(t) = 400$.
- $t = 2$. Changes in CSP offers. New instance type offer (micro instances).
- $t = 3$. Changes in CST requirements. CST needs more VMs $n(t) = 500$.
- $t = 4$. Changes in CSP offers. Nighttime instance type discounts. $EC2-OC$ provider offers 50% nighttime off.
- $t = 5$. Changes in CSP offers. Nighttime instance type discounts ends.
- $t = 6$. Changes in CSP offers. An instance type is removed from the market (xlarge instances).
- $t = 7$. Changes in CST requirements. CST needs less VMs $n(t) = 400$.

		Average		
No.	**Selection Strategy**	$TICPU$	$TIMEM$	TIP
S1	Random	8,767.46	31,615.79	547.48
S2	Distance	8,926.14	32,362.23	561.90
S3	Prefered	9,137.18	33,369.80	579.78
S4	Maximum TICPU	9,096.16	33,132.56	576.34
S5	Maximum TIMEM	9,107.74	33,233.36	576.87
S6	Minimum TIP	8,433.01	29,995.62	518.41

Table 10. Obtained averaged results for Experiment 2.

6.2 Selection Strategy Analysis

Since six selection strategies are evaluated, these strategies are compared through two methods [20] in order to check which strategy generates best solution in terms of the values of the objective functions:

– Pareto Dominance: a solution u_1 dominates another u_2 if considering each objective function u_1, is better or equal than u_2 and strictly better in at least one objective.
– Pareto Preference: a solution u_1 is defined as prefered over another u_2 if u_1 has more objective functions better evaluated than u_2 in terms of quantity.

Tables 9 and 10 summarize the results obtained in both experiments. As seen in Tables 11 and 12, **in average none of the strategies is dominated by another strategy** in both experiments; consequently, no strategies dominates the others, i.e. we can no establish that a given strategy is better than the other.

		Pareto Dominance						*Pareto Preference*					
No.	**Selection Strategy**	$S1$	$S2$	$S3$	$S4$	$S5$	$S6$	$S1$	$S2$	$S3$	$S4$	$S5$	$S6$
S1	Random	N/A	-	-	-	-	-	N/A	-	-	-	-	$\succ$
S2	Distance	-	N/A	-	-	-	-	$\succ$	N/A	-	-	-	$\succ$
S3	Prefered	-	-	N/A	-	-	-	$\succ$	$\succ$	N/A	$\succ$	$\succ$	$\succ$
S4	Maximum TICPU	-	-	-	N/A	-	-	$\succ$	$\succ$	-	N/A	-	$\succ$
S5	Maximum TIMEM	-	-	-	-	N/A	-	$\succ$	$\succ$	-	$\succ$	N/A	$\succ$
S6	Minimum TIP	-	-	-	-	-	N/A	-	-	-	-	-	N/A

Table 11. Experiment 1. Strategy Dominance and Preference.

Given that none of the considered strategies can be declared as the best strategy considering exclusively Pareto Dominance, a further comparison of selection strategies using the preference method (i.e. more quantity of better objective functions values) criteria [13] is presented in Table 11. It may seem intuitive that the $S3$ strategy (that uses the preference criterion itself) should be the best; as experimentally validated and presented in Table 11 in both experiments, given that in each algorithm iteration, the Preference strategy selects the solution containing the greatest number of best objective functions.

		Pareto Dominance						*Pareto Preference*					
No.	**Selection Strategy**	$S1$	$S2$	$S3$	$S4$	$S5$	$S6$	$S1$	$S2$	$S3$	$S4$	$S5$	$S6$
S1	Random	N/A	-	-	-	-	-	N/A	-	-	-	-	$\succ$
S2	Distance	-	N/A	-	-	-	-	$\succ$	N/A	-	-	-	$\succ$
S3	Prefered	-	-	N/A	-	-	-	$\succ$	$\succ$	N/A	$\succ$	$\succ$	$\succ$
S4	Maximum TICPU	-	-	-	N/A	-	-	$\succ$	$\succ$	-	N/A	-	$\succ$
S5	Maximum TIMEM	-	-	-	-	N/A	-	$\succ$	$\succ$	-	$\succ$	N/A	$\succ$
S6	Minimum TIP	-	-	-	-	-	N/A	-	-	-	-	-	N/A

Table 12. Experiment 2. Strategy Dominance and Preference.

References

1. Buyya, Rajkumar and Yeo, Chee Shin and Venugopal, Srikumar. *Market-oriented cloud computing: Vision, hype, and reality for delivering it services as computing utilities*. High Performance Computing and Communications, 2008. HPCC'08. 10th IEEE International Conference on, 2008.
2. Lucas Simarro, José Luis and Moreno-Vozmediano, Rafael and Montero, Ruben S and Llorente, Ignacio Martín. *Dynamic placement of virtual machines for cost optimization in multi-cloud environments*. High Performance Computing and Simulation (HPCS), International Conference on, 2011.
3. Lopez-Pires, F. and Baran, B. *A Virtual Machine Placement Taxonomy*. Cluster, Cloud and Grid Computing (CCGrid), 15th IEEE/ACM International Symposium on, 2015.
4. Tordsson, Johan and Montero, Rubén S and Moreno-Vozmediano, Rafael and Llorente, Ignacio M. *Cloud brokering mechanisms for optimized placement of virtual machines across multiple providers*. Future Generation Computer Systems, 2012.
5. Chaisiri, Sivadon and Lee, Bu-Sung and Niyato, Dusit *Optimal virtual machine placement across multiple cloud providers*. Services Computing Conference, 2009. IEEE Asia-Pacific.
6. Amazon Web Services*Amazon EC2 pricing schemes* https://aws.amazon.com/es/ec2/pricing.
7. Mark, Ching Chuen Teck and Niyato, Dusit and Chen-Khong, Tham *Evolutionary optimal virtual machine placement and demand forecaster for cloud computing*. Advanced Information Networking and Applications (AINA), 2011. IEEE International Conference on.
8. Kessaci, Yacine and Melab, Nouredine and Talbi *A pareto-based genetic algorithm for optimized assignment of VM requests on a cloud brokering environment*. Evolutionary Computation (CEC), 2013 IEEE Congress on.
9. Amato, Alba and Di Martino, Beniamino and Venticinque, Salvatore *Cloud brokering as a service*. P2P, Parallel, Grid, Cloud and Internet Computing (3PGCIC). 2013 Eighth International Conference on
10. Amato, Alba and Venticinque, Salvatore *Multi-objective decision support for brokering of cloud SLA*. Advanced Information Networking and Applications Workshops (WAINA), 2013 27th International Conference on.
11. Li, Wubin and Tordsson, Johan and Elmroth, Erik *Modeling for dynamic cloud scheduling via migration of virtual machines*. Cloud Computing Technology and Science (CloudCom), 2011 IEEE Third International Conference on.

12. Chamorro, Lino and Lopez-Pires, Fabio and Baran, Benjamin *A Genetic Algorithm for Dynamic Cloud Application Brokerage*. IEEE International Conference on Cloud Engineering, 2016.
13. A survey on multi-objective evolutionary algorithms for many-objective problems*von Lücken, Christian and Barán, Benjamín and Brizuela, Carlos*. Computational Optimization and Applications, 2014.
14. Announcing Micro Instances for Amazon EC2 *Amazon Web Services*. https://aws.amazon.com/es/about-aws/whats-new/2010/09/09/announcing-micro-instances-for-amazon-ec2
15. New Low Cost EC2 Instances with Burstable Performance *Amazon Web Services*. https://aws.amazon.com/es/blogs/aws/low-cost-burstable-ec2-instances/
16. Cloud Services Pricing *Amazon Web Services*. https://aws.amazon.com/es/pricing/services/
17. A Fast and Elitist Multiobjective Genetic Algorithm: NSGA-II *Kalyanmoy Deb, Amrit Pratap, Sameer Agarwal and T. Meyarivan*. IEEE Transactions on Evolutionary Computation
18. Evolutionary algorithms for solving multi-objective problems *Coello, Carlos Coello and Lamont, Gary B and Van Veldhuizen, David A*. Springer, 2007
19. Policies for Dynamical MultiObjective Environment of Multicast Traffic Engineering *Talavera, Francisco and Crichigno, Jorge and Barán, Benjamín*. IEEE ICT, 2005
20. Many-Objective Virtual Machine Placement for Dynamic Environments *Ihara, Diego and Lopez-Pires, Fabio and Crichigno, Jorge and Barán, Benjamín*. International Conference on Utility and Cloud Computing, 2015
21. Performance analysis of cloud computing services for many-tasks scientific computing *Iosup, Alexandru and Ostermann, Simon and Yigitbasi, M Nezih and Prodan, Radu and Fahringer, Thomas and Epema, Dick*. IEEE Transactions on Parallel and Distributed systems, 2011
22. A Many-Objective Optimization Framework for Virtualized Datacenters *Lopez-Pires Fabio and Barán, Benjamín*. 5th International Conference on Cloud Computing and Service Science, 2015
23. The NIST definition of cloud computing *Mell, Peter and Grance, Tim*. National Institute of Standards and Technology, 2009
24. Evolutionary algorithms for solving multi-objective problems *Carlos Coello Coello and Gary B Lamont and David A Van Veldhuizen*. Springer, 2007