A New Linear Model for Placement of Virtual Machines across Geo-Separated Data Centers

Fernando Stefanello, Luciana S. Buriol
Instituto de Informática – Universidade Federal do Rio Grande do Sul
Porto Alegre, RS, Brazil – 91.501-970
[fstefanello,buriol]@inf.ufrgs.br

Vaneet Aggarwal
School of Industrial Engineering
Purdue University
West Lafayette, IN, USA – 47907
vaneet@purdue.edu

Mauricio G. C. Resende
Mathematical Optimization and Planning
Amazon.com
Seattle, WA,USA – 98109
resendem@amazon.com

RESUMO

A computação em nuvem surgiu recentemente como uma nova tecnologia para hospedagem e fornecimento de serviços através da Internet. Com uma demanda crescente por computação em nuvem, se torna cada vez mais importante fornecer garantias de desempenho para as aplicações que são executadas sobre a nuvem. As aplicações podem ser abstraídas em um conjunto de máquinas virtuais com certas garantias que retratam a qualidade de serviço. Neste artigo, consideramos a alocação de máquinas virtuais em vários data centers, atendendo à qualidade do serviço e minimizando o custo de largura de banda dos centros de dados. Esse problema é uma generalização do problema de atribuição generalizada quadrática (GQAP), o qual é NP-Difícil. Nós apresentamos um novo modelo matemático linear, estendendo uma formulação para GQAP da literatura e incluímos um conjunto adicional de cortes. Resultados experimentais mostram que os cortes melhoraram significativamente a qualidade de limitantes inferiores e ajudam o CPLEX a reduzir o tempo de execução para encontrar soluções de boa qualidade.

PALAVRAS CHAVE: Otimização combinatória; Computação em nuvem; Programação matemática.
Área Principal: PM - Programação Matemática; OC - Otimização Combinatória.

ABSTRACT

Cloud computing has recently emerged as a new technology for hosting and supplying services over the Internet. With an increasing demand for cloud computing, providing performance guarantees for applications that run over cloud become important. Applications can be abstracted into a set of virtual machines with certain guarantees depicting the quality of service of the application. In this paper, we consider the placement of these virtual machines across multiple data centers, meeting the quality of service requirements while minimizing the bandwidth cost of the data centers. This problem is a generalization of the NP-hard Generalized Quadratic Assignment Problem (GQAP). We present a new linear mathematical model, extending a formulation for GQAP from the literature and include a set of additional cuts. Experimental results show that the cuts improve significantly the quality of lower bounds and help CPLEX to reduce the running time to find good quality solutions.

KEYWORDS: Combinatorial optimization; Cloud computing; Mathematical programming.
Main Area: PM - Mathematical Programming; OC - Combinatorial Optimization.

1. Introduction

Virtualization of physical servers have gained prominence in enterprise data centers. This is because virtualization offers virtually unlimited resources without any upfront capital investment and a simple pay-as-you-go charging model. Long term viability of virtualization depends, among other factors, on cost and performance. In order to attain performance guarantees, application providers can offer requirements for a number of virtual machines, bandwidth/latency requirements between virtual machines, and latency requirements between users of the service and virtual machines. Having all these performance guarantees for the application can help give an optimized service to the users. However, the service provider has to match the requirements of different applications to the placement of virtual machines with the limited bandwidth links between geographically separated data centers while minimizing its cost.

Unfortunately, today's public cloud platforms such as Amazon EC2 (Amazon Elastic Compute Cloud, 2015) do not provide any performance guarantee, which in turn affects tenant cost. Specifically, the resource reservation model in today's clouds only provisions CPU and memory resources but ignores networking completely. Because of the largely oversubscribed nature of today's data center networks (e.g., Greenberg et al. (2009)), network bandwidth is a scarce resource shared across many tenants. In order to meet the reliability and the demand requirements, the data centers have to be placed all across the world. For instance, a teleconference call connects people from all over the world, and a data center within a reasonable distance to the end users is needed. For distributed data centers, networking cost is the major cost, which has not been accounted in the prior works on virtual machine placement to the best of our knowledge. With the limited bandwidth links between the data centers, networking intensive phases of applications collide and compete for the scarce network resources, which leads to their running times become unpredictable. The uncertainty in execution time further translates into unpredictable cost as tenants need to pay for the reserved virtual machines (VMs) for the entire duration of their jobs.

Placement of virtual machines within a data center have been widely explored (Guo et al., 2010; Ballani et al., 2011; Xie and Hu, 2012). These papers account for the networking needs in addition to the CPU and memory needs within a data center. For example, Guo et al. (2010) proposes bandwidth reservation between every pair of VMs. Ballani et al. (2011) proposes a simpler virtual cluster (VC) model where all virtual machines are connected to a virtual switch with links of bandwidth B. Xie and Hu (2012) extends these approaches to consider time-varying network requirement. However, all these works account for a single data center where the bandwidths are much larger as compared to the bandwidths across data centers. Instead, this paper deals with the placement of virtual machines across geo-separated data centers

In this paper, we consider multiple data centers that are connected with limited bandwidth links. The latency between every pair of data centers is known. In order to meet the application's quality of service guarantees, there is a required minimum bandwidth and maximum latency between each pair of virtual machines. We assume that there are multiple users who would use these services, and users are connected to some data center. In order to meet the overall application performance, there is an additional requirement of maximum latency between users and the virtual machines. Intuitively, if there is a set of VMs needed by a user and the set does not have any requirement with any other user or VM, it can be placed in a single data center. However, a VM interacts with multiple VMs which may be needed by other users, thus increasing the set of options for placement. There is a cost of transferring data between data-centers and the placement minimizes this cost thus preferring placement of all VMs in a single data center which may not be feasible due to the quality of service requirements for the application.

In this paper we address a Virtual Machine Placement Problem (VMPlacement), which is the problem of minimizing the cost of virtual machines placement across geo-separated data

centers, first introduced by Stefanello et al. (2015). This problem is a generalization of the NP-hard GQAP given in Lee and Ma (2004).

We present a new linear mathematical model for the VMPlacement problem. We extend a formulation for GQAP from the literature and include an additional set of cuts to improve the relaxation quality for the new mathematical model. Experimental results show that the cuts improve significantly the quality of lower bounds, and helps to CPLEX to reduce the running time to find good quality solutions.

The rest of the paper is organized as follows. In Section 2, we present tree mathematical formulations for the Virtual Machine Placement Problem in multiple data centers as well as the additional set of cuts. Computational results are presented in Section 3. Finally, conclusions are drawn in Section 4.

2. Virtual Machine Placement Problem

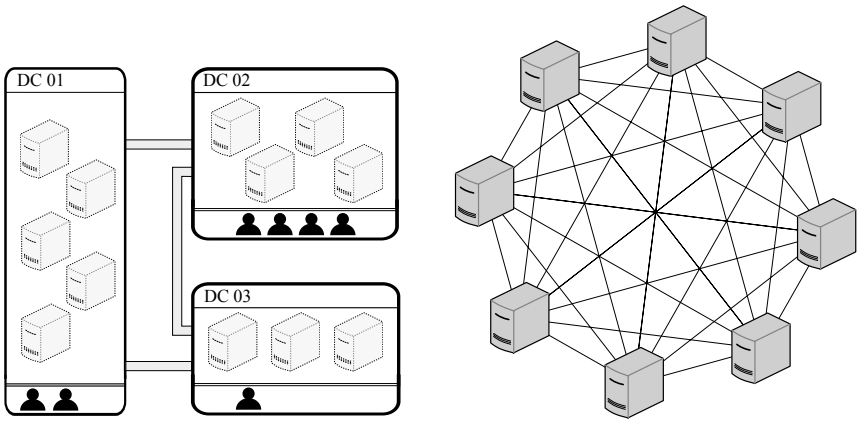
In the Virtual Machine Placement Problem, the objective is to place a set K of virtual machines (VM) in a set N of data centers (DC) in order to minimize the communication cost among virtual machines.

In this problem, each data center has a capacity a_i, which represents the number of virtual machines that can be placed in DC i. Also, between two data centers i and j, there are a bandwidth capacity (B_{ij}), a latency (L_{ij}), and a cost C_{ij} to transfer a data unit between the pair of data centers.

In order to meet the reliability and demand requirements of the applications, certain bandwidth and latency requirements can be imposed on the different VMs that are placed on the data centers. Each pair of virtual machines v and w has a required bandwidth (b_{vw}) whose sum overall VMs placed between DCs i and j cannot exceed B_{ij}. Furthermore, there is a required latency (l_{vw}), such that VMs v and w cannot be placed in data centers i and j if the required latency is greater than the respective data center latency.

Finally, there is a set U of users who access the system. Each user u is located at a data center $d(u)$ and has a required latency t_{uv} for each VM v.

Figure 1 shows a representation of the input data components: the data centers (Figure 1(a)), and the virtual machines (Figure 1(b)). The first component is composed by three data centers (rounded rectangles). Each data center has a number of users and a capacity (represented as a number of spots where VMs can be placed). The connection between each pair of DCs represents the bandwidth capacity, latency, and cost. The second component is composed by eight virtual machines, where each link represents the bandwidth and required latency.



(a) Data centers. (b) Virtual machines.

Figure 1: Input data representation.

In the next subsections, we present three formulations for the VMPlacement problem. Since VMPlacement is a generalization of the NP-hard Generalized Quadratic Assignment Problem (GQAP), we extend the linear mathematical models proposed for the GQAP in Lee and Ma (2004) to VMPlacement. The models in Lee and Ma (2004) were also extended from the mixed integer linear programming formulation from Kaufman and Broeckx (1978) and Frieze and Yadegar (1983) to the Quadratic Assignment Problem, all of them based on the formulation for the QAP described in Koopmans and Beckmann (1957).

2.1. Quadratic mathematical model

A natural formulation for VMPlacement is based in a quadratic formulation as a generalization of GQAP. In what follows we summarize the parameters and present a quadratic mathematical model for VMPlacement (QMVMP) introduced in Stefanello et al. (2015).

Parameters:

N : set of data centers;
K : set of virtual machines;
U : set of users;
a_i : capacity in number of VMs DC i can host;
B_{ij} : bandwidth between DCs i and j;
L_{ij} : latency between DCs i and j;
C_{ij} : cost of transferring a data unit between DCs i and j;
b_{vw} : required bandwidth between VMs v and w;
l_{vw} : required latency between VMs v and w;
$d(u)$: DC which hosts user u;
t_{vu} : required latency between user u and VM v;
c_{iv} : cost of place a VM v in a DC i;
z : scaling cost factor.

Equations (1a)-(1g) present the quadratic mathematical model for the VMPlacement (QMVMP), where the binary decision variable x_{iv} is set to one when VM v is located into DC i, and zero otherwise.

$$\min \sum_{i\in N}\sum_{v\in K} c_{iv}x_{iv} + z\sum_{i\in N}\sum_{j\in N}\sum_{v\in K}\sum_{w\in K} x_{iv}x_{jw}C_{ij}b_{vw} \tag{1a}$$

subject to:

$$\sum_{v\in K} x_{iv} \leq a_i \qquad \forall\, i \in N, \tag{1b}$$

$$\sum_{i\in N} x_{iv} = 1 \qquad \forall\, v \in K, \tag{1c}$$

$$\sum_{v\in K}\sum_{w\in K} x_{iv}x_{jw}b_{vw} \leq B_{ij} \qquad \forall\, i,j \in N, \tag{1d}$$

$$\sum_{i\in N}\sum_{j\in N} x_{iv}x_{jw}L_{ij} \leq l_{vw} \qquad \forall\, v,w \in K, \tag{1e}$$

$$\sum_{i\in N} x_{iv}L_{i,d(u)} \leq t_{vu} \qquad \forall\, u \in U,\ \forall\, v \in K, \tag{1f}$$

$$x_{iv} \in \{0,1\} \qquad \forall\, i \in N,\ \forall\, v \in K. \tag{1g}$$

Objective function (1a) minimizes the cost of placing each pair of virtual machines v and w to DCs i and j. Constraints (1b) require that the number of VMs in each DC must not

exceed the DC capacity. Constraints (1c) require that each VM must be assigned to exactly one DC. Constraints (1d) require that the given bandwidth between each pair i and j of DCs should not be surpassed by the total sum of bandwidth required among the virtual machines placed in these DCs. Constraints (1e) assure that the latency required between each pair of VMs should be respected, i.e, if VMs v and w are placed respectively to DCs i and j, then the latency between DCs i and j should not exceed the required latency between VMs v and w. Constraints (1f) require that the latency between a VM v and the DC where the user u is located be respected, i.e, a VM v can be only placed on a DC i if the latency between i and $d(u)$ is less than or equal to a given latency between the VM v and the user u. Finally, constraints (1g) define the variables domain.

2.2. Linear mathematical model I - LMVMP

The performance of mixed integer linear programming solvers has improved considerably over the last few years. IBM ILOG CPLEX Optimizer (2015) is a general-purpose black-box solver based on simplex and branch-and-bound algorithm with the state-of-the-art exact algorithm for integer programming and has been successfully applied in many combinatorial optimization problems. In order to analyze the CPLEX performance and provide baseline results for comparison of heuristic methods, we present a linear mathematical model for VMPlacement problem.

Based on model $L3$ from Lee and Ma (2004) for GQAP, and from Frieze and Yadegar (1983) for QAP, we present a mixed-integer linear model for the VMPlacement. Let $y_{ivjw} = x_{iv}x_{jw}$, $\forall\, i, j = \{1, \ldots, N\}$ and $v, w = \{1, \ldots, K\}$, the mixed-integer linear mathematical model for VMPlacement named as LMVMP can be formulated as the following:

$$\min \sum_{i \in N} \sum_{v \in K} c_{iv} x_{iv} + z \sum_{i \in N} \sum_{j \in N} \sum_{v \in K} \sum_{w \in K} y_{ivjw} C_{ij} b_{vw} \tag{2a}$$

subject to:

$$\sum_{v \in K} x_{iv} \leq a_i \qquad \forall\, i \in N, \tag{2b}$$

$$\sum_{i \in N} x_{iv} = 1 \qquad \forall\, v \in K, \tag{2c}$$

$$\sum_{i \in N} y_{ivjw} = x_{jw} \qquad \forall\, v, w \in K,\ \forall\, j \in N, \tag{2d}$$

$$y_{ivjw} = y_{jwiv} \qquad \forall\, v, w \in K,\ \forall\, i, j \in N, \tag{2e}$$

$$\sum_{v \in K} \sum_{w \in K} y_{ivjw} b_{vw} \leq B_{ij} \qquad \forall\, i, j \in N, \tag{2f}$$

$$\sum_{i \in N} \sum_{j \in N} y_{ivjw} L_{ij} \leq l_{vw} \qquad \forall\, v, w \in K, \tag{2g}$$

$$\sum_{i \in N} x_{iv} L_{i,d(u)} \leq t_{vu} \qquad \forall\, u \in U,\ \forall\, v \in K, \tag{2h}$$

$$x_{iv} \in \{0, 1\} \qquad \forall\, i \in N,\ \forall\, v \in K, \tag{2i}$$

$$0 \leq y_{ivjw} \leq 1 \qquad \forall\, i, j \in N,\ \forall\, v, w \in K. \tag{2j}$$

The LMVMP is obtained by replacing the product $x_{iv}x_{jw}$ by y_{ivjw} from QMVMP. In addition four sets of constraints are considered. Constraints (2d) and (2e) define the relation between variables x and y. Constraints (2e) also impose the symmetry relation to variables y. Finally, constraints (2j) define the domain of variables y.

We note that the model QMVMP has quadratic constraints, while LMVMP not. The objective function also changes from a quadratic function in QMVMP to linear in LMVMP. However, the mixed-integer linear problem LMVMP has a considerable higher number of variables, having variables y_{ivjw} in addition to the previous variables x_{iv}. Thus, the number of variables change from $O(NK)$ in QMVMP to $O(N^2K^2)$ in LMVMP. We note that if the optimal solution of LMVMP is (x^*_{iv}, y^*_{ivjw}), then (x^*_{iv}) is the optimal solution for QMMVP. The proof that both models are equivalent can be easily obtained by extending the proof for QAP provided in Lee and Ma (2004).

2.3. Linear mathematical model II - LMVMP-II

Next we present a second linearization for VMPlacement problem (LMVMPII). This linear model is derived from Kaufman and Broeckx (1978) for QAP, which is probably the linearization for QAP with the lower number of variables and constraints. In Lee and Ma (2004) the authors extend the formulation for GQAP.

In this model, each binary decision variable x_{iv} is set to one when VM v is located into DC i, and zero otherwise. The auxiliary variables y_{iv} aggregate the cost for each placed VM v in a DC i, and n_{ijb} aggregate the bandwidth from VM v between data centers i and j.

$$\min \sum_{i\in N}\sum_{v\in K} c_{iv}x_{iv} + z\sum_{i\in N}\sum_{v\in K} y_{iv} \tag{3a}$$

subject to:

$$\sum_{v\in K} x_{iv} \leq a_i \qquad \forall\, i \in N, \tag{3b}$$

$$\sum_{i\in N} x_{iv} = 1 \qquad \forall\, v \in K, \tag{3c}$$

$$\sum_{j\in N}\sum_{w\in K} C_{ij}b_{vw}x_{jw} - y_{iv} \leq m_{iv}(1 - x_{iv}) \qquad \forall\, v \in K,\ \forall\, i \in N, \tag{3d}$$

$$\sum_{w\in K} b_{vw}x_{jw} - n_{ijv} \leq M(1 - x_{iv}) \qquad \forall\, v \in K,\ \forall\, i, j \in N, \tag{3e}$$

$$\sum_{v\in K} n_{ijv} \leq B_{ij} \qquad \forall\, i, j \in N, \tag{3f}$$

$$L_{ij}x_{iv} \leq l_{vw} + L_{ij}(2 - x_{iv} - x_{jw}) \qquad \forall\, v, w \in K, \forall\, i, j \in N, \tag{3g}$$

$$\sum_{i\in N} x_{iv}L_{i,d(u)} \leq t_{vu} \qquad \forall\, u \in U,\ \forall\, v \in K, \tag{3h}$$

$$x_{iv} \in \{0, 1\} \qquad \forall\, i \in N,\ \forall\, v \in K, \tag{3i}$$

$$y_{iv} \geq 0 \qquad \forall\, i \in N,\ \forall\, v \in K, \tag{3j}$$

$$n_{ijv} \geq 0 \qquad \forall\, v, w \in K,\ \forall\, i, j \in N. \tag{3k}$$

where

$$m_{iv} \geq \sum_{j\in N}\sum_{w\in K} C_{ij}b_{vw}, \quad \forall\, i \in N,\ \forall\, v \in K.$$

Constraints (3d) impose the cost between the data centers i and j to the variables y_{ij}. Constraints (3e) and (3f) imposed the bandwidth constraints while the constraints (3g) imposed the latency constraints. The constraints (3g) can be replaced by the constraints

$$x_{iv} + x_{jw} \leq 1 \quad \forall\, i, j \in N,\ \forall\, v, w \in K \text{ if } L_{ij} > l_{vw}. \tag{4}$$

We observe that CPLEX convert (3g) into (4) in the pre-processing phase. Finally, constraints (3i), (3j), and (3k) define the domain of variables.

This model is not used in practice since it uses big-M constraints (3d), and the root-node bound is always zero. However, a much stronger formulation can be obtained by adding the following cuts

$$y_{iv} \geq Y_{iv} x_{iv} \quad \forall\, i \in N,\ \forall\, v \in K, \tag{5}$$

where Y_{iv} is defined as the optimal value of the following assignment problem:

$$Y_{iv} = \min \sum_{j \in N} \sum_{w \in K} C_{ij} b_{vw} x_{jw} \tag{6a}$$

subject to:

$$\sum_{w \in K} x_{jw} \leq a_j \qquad \forall\, j \in N, \tag{6b}$$

$$\sum_{j \in N} x_{jw} = 1 \qquad \forall\, w \in K, \tag{6c}$$

$$\sum_{j \in N} \sum_{w \in K} b_{vw} x_{jw} \leq B_{ij} \qquad \forall\, w \in K, \tag{6d}$$

(3g), (3h) (6e)

$$x_{iv} = 1 \tag{6f}$$

$$x_{jw} \in \{0, 1\} \qquad \forall\, j \in N,\ \forall\, w \in K. \tag{6g}$$

These additional cuts were applied for the 3-dimensional assignment problem in Mittelmann and Salvagnin (2015). Note that this cuts can also be applied for GQAP suppressing the constrains (3e)–(3h) and (3k) from LMVMPII, and (6d)–(6e) from the cuts, which are the specific constraints for VMPlacement.

3. Computational experiments

In this section we present computational experiments for comparing the performance of CPLEX with the mathematical models presented in the previous sections. The experiments were conducted on a cluster with quad-core Intel Xeon E5530 2.4 GHz CPUs, with at least 48 GB of RAM running GNU/Linux. The commercial solver IBM ILOG CPLEX Optimizer version 12.6.0.0 (C++ API) was used to evaluate the mathematical models. All experiments used a single thread but multiple experiments were run in parallel.

Experiments were conducted with the main objective of investigating the CPLEX performance considering the different mathematical models. We seek to carry out an empirical study in order to analyze which size of instances the CPLEX can handle and prove optimality, and which model provides better integer solutions and lower bounds when CPLEX ends by time limit.

Two sets of instances are used in our experiments. The first, that contains small size instances that contain only 5 data centers, was generated as described in Stefanello et al. (2015). The second is the set used in Stefanello et al. (2015). All instances encoded in the name the number of data centers, number of virtual machines, number of users and percentage of overall data center occupation. The first set of instances are listed in Table 1, while the second set is listed in Table 2. The instances and their best known solutions are available at `www.inf.ufsm.br/~stefanello/instances/`.

Part of the objective function from Equation (1a) is in fact not used for these instances since we are not considering the fix cost $c_{iv} \in \mathbb{R}$ of placement or installation a VM to a DC, as well the scaling factor $z \in \mathbb{R}$. Thus, without loss of generality, for these instances of the VMPlacement problem we may assume $c_{iv} = 0$, $\forall\, i \in N$ and $v \in K$, and $z = 1$. However, these values can be different for GQAP instances, or a more general case of VMPlacement instances that consider installation cost.

3.1. Results for small size instances

In the first experiment we evaluated the performance of CPLEX with the mathematical models described in Section 2. We used the standard CPLEX solvers for models LMVMP and LMVMPII. The running time limit was set to thee hours ($10,800$ seconds) and the number of threads was set to one. The remaining parameters were maintained on the default values.

In Fischetti and Monaci (2014) the authors exploited erraticism in search and how to take advantage of this behavior. Also, the authors show that CPLEX can have a wide contrast in its behavior due randomized initial conditions. Thus, to better evaluate the CPLEX performance we made ten runs for each instance, each one with different random seed defined by the CPLEX parameter `RandomSeed`.

Table 1 shows CPLEX results. The first column shows the name of instances. The second column (BKS) shows the objective function of the best known solution value for each instance (optimal solutions are showed in boldface). The next two columns show the average running times of CPLEX to solve each instance for each mathematical model. Numbers in parenthesis denote the number of runs that the solver did not prove optimality within the time limit. The next two columns show the average running times that CPLEX spent to find the BKS value. The average is over ten runs or the number of times that CPLEX found BKS (indicated by the number in parenthesis). A signal '$-$' indicates that no run found a solution as good as BKS within the time limit. Finally, the last two columns show the best running times over all runs that CPLEX spent to find the BKS.

Table 1: CPLEX detailed results for small instances.

		AVG Time(s)		AVG BKS Time(s)		MIN BKS Time(s)	
Instance	**BKS**	**LMVMP**	**LMVMPII**	**LMVMP**	**LMVMPII**	**LMVMP**	**LMVMPII**
05_015_007_70	**25,844.02**	2.9	6.2	2.3	2.2	1.1	0.4
05_015_007_90	**23,557.30**	26.0	61.9	17.4	15.4	10.2	1.4
05_015_015_70	**10,904.78**	0.4	1.0	0.3	0.4	0.2	0.3
05_015_015_90	**24,354.96**	4.2	6.9	3.1	1.8	2.3	0.8
05_015_022_70	**14,163.60**	0.2	0.7	0.1	0.1	0.1	0.0
05_015_022_90	**32,318.02**	22.4	21.3	21.2	5.2	12.7	0.9
05_020_010_70	**38,572.62**	292.8	701.7	232.4	138.1	6.4	1.2
05_020_010_90	**64,710.80**	68.3	107.3	58.5	24.7	31.2	1.9
05_020_020_70	**55,288.76**	134.5	437.6	100.9	212.4	10.5	37.8
05_020_020_90	**57,574.90**	1.7	2.1	1.3	0.4	0.8	0.4
05_020_030_70	**28,433.34**	11.1	17.4	7.1	3.7	2.8	0.4
05_020_030_90	**66,088.70**	1.9	2.6	1.8	0.7	1.2	0.3
05_025_012_70	**43,300.76**	1271.5	4,025.5	1,063.8	862.6	195.0	2.2
05_025_012_90	100,865.02	10,800 (10)	10,800 (10)	9,902.6 (4)	6,165.6 (8)	9,746.0	28.7
05_025_025_70	**42,890.40**	58.4	126.7	52.5	39.0	26.9	1.3
05_025_025_90	103,791.96	10,800 (10)	10,800 (10)	9,764.3 (1)	–	9,764.3	–
05_025_037_70	**97,335.12**	161.4	592.8	117.0	161.2	19.6	2.8
05_025_037_90	**73,363.56**	10,262 (7)	10,800 (10)	8,993.2	3,782.8 (9)	5,916.3	33.8

We can draw three main observations from this experiment. First, for both models, the solver tends to increase significantly the time spent to prove optimality for instances with 5 data centers or more, and 25 virtual machines or more. This indicates a baseline for the size of instances that this version of CPLEX can handle and prove the optimality using these models in this computer. The second observation is that the performance of CPLEX for both models was relatively similar. CPLEX was able to prove or not the optimality in the same instances, except for the instance `05_025_037_90` which CPLEX proved optimality in tree runs for LMVMP, while for LMVMPII CPLEX has an average gap of 8%. However, considering the time that each model spent to find a solution with objective function equal to BKS, in most cases CPLEX with LMVMPII spent less time than considering the model LMVMP. Finally, the last observation is about the variance of time to find the BKS. With the randomized initial conditions, CPLEX presented a large variance in its behavior. For example, for the instance `05_025_012_70` and LMVMPII, in the best case

CPLEX found BKS in 2.2 seconds, while in the worst case, it takes $3,458.01$ seconds to find the same solution. For the same instance, the time to prove the optimality also have large variance, alternating between a minimum of $3,685.67$ and a maximum of $4,828.05$.

We also evaluate the model LMVMPII without including the additional set of cuts (5). In comparison with the model with the cuts, we observe a higher decrease in the performance of CPLEX. For six instances CPLEX was not able to prove the optimality in any of the ten runs. For these cases, the gap of CPLEX is still higher. For example, for the instance `05_025_012_90`, the average gap was 53%, confirming that the relaxation quality of this model is low. Furthermore, the time to find BKS increased around 8 times, and the number of nodes explored within the time limit or to prove the optimality increased around 7 times in comparison with the model that includes the set of cuts (5).

3.2. Results for median and large size instances

In this subsection we present results and an analyzes of CPLEX performance for a set of median and large instances. In this experiment, we evaluate CPLEX for each model and each instance with one run for a time limit of one day ($86,400$ seconds).

Table 2: CPLEX detailed results for median and large instances.

		FO		CPLEX GAP		BKS GAP	
Instance	**BKS**	**LMVMP**	**LMVMPII**	**LMVMP**	**LMVMPII**	**LMVMP**	**LMVMPII**
10_025_012_70	114,582.50	116,264.44	115,734.58	42.37	24.29	1.47	1.01
10_025_012_90	84,461.30	88,087.12	85,814.72	53.92	28.31	4.29	1.60
10_025_025_70	90,997.90	93,729.48	92,407.94	29.59	19.30	3.00	1.55
10_025_025_90	124,763.66	125,365.26	125,335.92	31.18	16.56	0.48	0.46
10_025_037_70	100,801.80	104,350.38	100,801.80	24.16	15.04	3.52	0.00
10_025_037_90	106,617.94	107,558.00	107,471.04	24.06	20.06	0.88	0.80
10_050_025_70	414,689.30	442,548.40	435,929.26	83.22	39.98	6.72	5.12
10_050_025_90	460,414.96	480,146.00	477,439.06	75.60	30.79	4.29	3.70
10_050_050_70	360,102.12	374,071.02	366,972.40	68.85	43.23	3.88	1.91
10_050_050_90	403,272.24	420,173.88	416,615.52	72.88	36.32	4.19	3.31
10_050_075_70	349,135.78	362,853.92	353,979.20	54.03	35.20	3.93	1.39
10_050_075_90	500,668.88	513,161.02	510,062.50	54.45	25.23	2.50	1.88
10_100_050_70	1,677,015.90	1,884,262.62	1,732,985.94	91.55	42.59	12.36	3.34
10_100_050_90	1,804,385.34	1,916,126.76	1,826,484.56	89.56	34.79	6.19	1.22
10_100_100_70	1,465,034.60	1,546,897.78	1,500,763.50	86.22	47.12	5.59	2.44
10_100_100_90	2,145,917.62	2,256,408.46	2,169,460.06	82.73	31.03	5.15	1.10
10_100_150_70	1,572,976.60	1,702,573.74	1,586,082.86	78.44	45.02	8.24	0.83
10_100_150_90	1,858,242.74	1,968,341.96	1,883,289.84	70.74	33.59	5.92	1.35
Average				61.86	31.58	4.59	1.83
25_100_050_70	1,910,010.90	-	1,976,649.48	-	44.15	-	3.49
25_100_050_90	2,138,992.76	-	2,184,190.20	-	34.89	-	2.11
25_100_100_70	1,986,347.70	-	2,056,428.04	-	44.55	-	3.53
25_100_100_90	2,051,178.60	-	2,091,723.56	-	35.72	-	1.98
25_100_150_70	2,003,146.04	-	2,061,181.36	-	43.91	-	2.90
25_100_150_90	2,199,389.38	-	2,235,936.02	-	35.35	-	1.66
25_150_075_70	4,651,277.60	-	4,768,514.02	-	42.99	-	2.52
25_150_075_90	4,656,263.22	-	4,786,179.50	-	36.54	-	2.79
25_150_150_70	3,927,147.34	-	4,085,463.10	-	46.34	-	4.03
25_150_150_90	4,759,888.80	-	4,899,667.66	-	36.82	-	2.94
25_150_225_70	4,391,141.90	-	4,530,866.46	-	46.00	-	3.18
25_150_225_90	4,560,237.08	-	4,708,994.42	-	35.51	-	3.26
25_200_100_70	7,008,419.48	-	7,539,073.36	-	50.79	-	7.57
25_200_100_90	9,096,638.02	-	9,391,760.06	-	34.93	-	3.24
25_200_200_70	7,221,163.64	-	7,579,453.48	-	49.13	-	4.96
25_200_200_90	8,638,447.06	-	8,909,349.84	-	36.53	-	3.14
25_200_300_70	7,707,491.66	-	8,033,640.92	-	42.32	-	4.23
25_200_300_90	8,262,794.54	-	8,579,652.20	-	38.94	-	3.83
Average					40.86		3.41

Table 2 shows for each instance, the BKS obtained in Stefanello et al. (2015). The column FO shows, for each model, the objective function value of the best integer solution found by CPLEX. Column CPLEX GAP shows the gap returned by CPLEX. The last column shows the gap from BKS to the best integer solution returned by CPLEX.

The first observation from this experiment is that for instances with 10 data centers, CPLEX was able to start solve the models, but still with a higher gap even after 24 hours of computation. However, we observed that the gap returned by CPLEX as well as the gap to BKS are lower for LMVMPII in comparison with the values of LMVMP. The average gap returned by CPLEX with LMVMPII was around half of the average gap for LMVMP. The average gap to BKS was 1.83 for LMVMPII, while for LMVMP the gap was 4.59.

The second observation is that for instances with 25 data centers and LMVMP model, CPLEX spent the whole time in the presolve phase without solving the root relaxation node. Instead for LMVMPII, CPLEX was able to start a node exploration and find feasible solutions. For instances with 25 data centers, CPLEX explored an average of approximately 34800 nodes for instances with 100 VMs, 5700 nodes for instances with 150 VMs, and 1000 nodes for instances with 200 VMs. Even CPLEX maintaining a higher gap, the gap from BKS was relatively small, considering the size of the instance and the difficult to find feasible solutions (Stefanello et al., 2015).

4. Conclusions and future works

In this paper we address the problem of minimizing the cost of virtual machines placement across geo-separated data centers, first introduced by Stefanello et al. (2015). We present a new linear mathematical model (LMVMPII), extending a formulation for GQAP from the literature and include an additional set of cuts. Experimental results using CPLEX show that by adding the set of cuts the solver improves significantly the quality of the lower bounds for the new model. We also observed that for this model, CPLEX can handle larger instances than considering LMVMP model, obtaining better lower bounds and feasible solutions in less computational time.

In the future works, we plan use the information provided by the relaxation of both models or the described set of cuts to improve heuristic and metaheuristic techniques.

Acknowledgments

This work has been partially supported by CAPES, CNPq project 462425/2014-2, PRH PB-217 Petrobras S.A. from Brazil.

References

Amazon Elastic Compute Cloud (2015). http://aws.amazon.com/ec2/. Last accessed: February, 2015.

Ballani, H., Costa, P., Karagiannis, T., Rowstron, A. (2011). Towards predictable datacenter networks. In: ACM SIGCOMM Computer Communication Review. Vol. 41 of SIGCOMM '11. ACM Press, New York, NY, USA, pp. 242–253.

Fischetti, M., Monaci, M. (2014). Exploiting Erraticism in Search. Operations Research 62 (1), 114–122.

Frieze, A., Yadegar, J. (1983). On the quadratic assignment problem. Discrete Applied Mathematics 5 (1), 89–98.

Greenberg, A., Hamilton, J. R., Jain, N., Kandula, S., Kim, C., Lahiri, P., Maltz, D. a., Patel, P., Sengupta, S. (2009). VL2: A Scalable and Flexible Data Center Network. ACM SIGCOMM Computer Communication Review 39 (4), 51.

Guo, C., Lu, G., Wang, H. J., Yang, S., Kong, C., Sun, P., Wu, W., Zhang, Y. (2010). SecondNet: A Data Center Network Virtualization Architecture with Bandwidth Guarantees. In: Proceedings of the 6th International Conference on - Co-NEXT '10. Co-NEXT '10. ACM, New York, NY, USA, p. 1.

IBM ILOG CPLEX Optimizer (2015). www.cplex.com. Last accessed: February, 2015.

Kaufman, L., Broeckx, F. (1978). An algorithm for the quadratic assignment problem using Bender's decomposition. European Journal of Operational Research 2 (3), 207–211.

Koopmans, T. C., Beckmann, M. (1957). Assignment problems and the location of economic activities. Econometrica 25 (1), 53–76.

Lee, C. G., Ma, Z. (2004). The generalized quadratic assignment problem. Tech. rep., Department of Mechanical and Industrial Engineering at the University of Toronto, Toronto, Ontario, M5S 3G8, Canada.

Mittelmann, H., Salvagnin, D. (2015). On solving a hard quadratic 3-dimensional assignment problem. Mathematical Programming Computation, 1–16.

Stefanello, F., Aggarwal, V., Buriol, L. S., Gonçalves, J. F., Resende, M. G. C. (2015). A Biased Random-key Genetic Algorithm for Placement of Virtual Machines across Geo-Separated Data Centers. In: Accepted to GECCO'15 – Proceedings of the 2015 conference on Genetic and evolutionary computation. ACM, Madrid, pp. 1–8.

Xie, D., Hu, Y. C. (2012). The Only Constant is Change: Incorporating Time-Varying Network Reservations in Data Centers. In: Sigcomm. SIGCOMM '12. ACM, New York, NY, USA, pp. 199–210.