A Lagrange decomposition based branch and bound algorithm for the optimal mapping of cloud virtual machines

Guanglei Wang, Walid Ben-Ameur, Adam Ouorou

A Lagrange decomposition based Branch and Bound algorithm for the Optimal Mapping of Cloud Virtual Machines

Guanglei Wang Walid Ben-Ameur Adam Ouorou

07 June 2018

Abstract

One of the challenges of cloud computing is to optimally and efficiently assign virtual machines to physical machines. The aim of telecommunication operators is to minimize the mapping cost while respecting constraints regarding location, assignment and capacity. In this paper we first propose an exact formulation leading to a 0-1 bilinear constrained problem. Then we introduce a variety of linear cuts by exploiting the problem structure and present a Lagrange decomposition based B&B algorithm to obtain optimal solutions efficiently. Numerically, we show that our valid inequalities close over 80% of the optimality gap incurred by the well-known McCormick relaxation, and demonstrate the computational advantage of the proposed B&B algorithm with extensive numerical experiments.

1 Introduction

Virtualization technology enables the emergence of cloud computing as a flexible and on-demand service. In a virtualization-based network the placement of Virtual Machines (VM) exerts significant influence on the computation and communication performance of cloud services [23]. A considerable amount of investigations have been devoted to the optimal assignment of VMs to servers accounting for certain objectives and constraints.

Google in 2012 proposed a challenge organized by the French Operational Research and Decision Aid Society (ROADEF) and the European Operational Research society (EURO), where a set of VMs needs to be assigned to a set of servers to minimize the assignment cost while balancing the usage of servers under several resource constraints. As reported in [21], the proposal takes into account capacity constraints of servers regarding CPU, memory, storage. However, it *does not* include bandwidth constraints respecting certain throughput requirements among VMs. Exact formulations of this problem are Mixed Integer Linear Programs (MILP). To deal with large scale problems, different heuristics are proposed. On the other hand, the authors in [19] introduce a traffic-aware virtual machine placement model taking into account bandwidth constraints, which leads to a Quadratic Assignment Problem (QAP) for the solution of which a two-tier heuristic algorithm is proposed.

This research topic is also discussed in the context of *Virtual Network Embedding* (VNE), where virtual networks are required to be mapped to a physical network [7] while respecting different constraints and objectives. For instance, Houidi et al. [12] propose a MILP model to solve the VM assignment problem and later this work is extended in [13] to jointly take into account energy-saving, load balancing and survivability objectives. The authors in [23] present a MILP model

and consider a two-phase heuristic: a node mapping phase and link mapping phase. In the node mapping phase, random rounding techniques [24] are used to correlate flow variables and binary variables. In the link mapping phase, decisions on the mapping of virtual links are made by solving a Multi-Commodity Network Flow (MCNF) problem. Later the authors in [6] propose a chance constrained MILP formulation to handle the uncertain demand of different virtual networks and they propose a couple of heuristics based on MILPs for its solution. For more details about the VNE technology and related investigations, we redirect interested readers to [7, 20] for comprehensive surveys.

A recent thesis [18] studies the virtual network infrastructure provision in a distributed cloud environment, where a 0-1 bilinear constrained model taking into account bandwidth constraints is proposed. Heuristic methods exploiting graph partition and bipartite graph matching techniques are proposed for the solution procedure.

More recently, Fukunaga et al. [9] consider the assignment of VMs under capacity constraints aiming at minimizing certain connection cost. A centralized model and a distributed model are proposed for modeling the connection cost. In the former case, a root node is introduced and the connection cost is defined as the length of network links connecting all host servers and the root node. A couple of approximation algorithms are presented for cases of uniform and nonuniform requests respectively. However all VMs are assumed to be the same and bandwidth constraints are not considered.

In spite of these efforts, few focus on mathematical programming methods in the presence of bandwidth constraints. In contrast to heuristic approaches (e.g., Genetic Programming, Tabu Search) whose performance is usually evaluated by simulations, a mathematical programming approach offers performance guarantees with proved lower and upper bounds. Furthermore, it benefits from off-the-shelf solvers that are being continually improved. Thus mathematical programming based methods deserve in-depth investigations.

In our previous paper [27], we formulate the bandwidth constrained mapping problem as a 0-1 bilinear constrained problem and our numerical results demonstrate that the problem is computationally challenging even for a small number of VMs. This article aims at improving the computational performance by orders of magnitude. It extends the previous work by introducing some effective valid inequalities and proposes a Lagrange decomposition based Branch and Bound (B&B) algorithm to accelerate the solution procedure. Contributions are summarized as follows.

1. We propose a compact model with a number of novel valid inequalities for the mapping problem.

2. We propose a Lagrange decomposition procedure for generating strong valid inequalities thus improving the continuous relaxation lower bound.

3. We develop a B&B algorithm to solve the mapping problem to global optimality. Various valid inequalities are used to strengthen the relaxation at each node dynamically.

4. We conduct extensive numerical experiments showing the effectiveness of the proposed algorithm by orders of computational improvement.

The rest of this paper is organized as follows. In Section 2, we state the background of the mapping problem. In Section 3, the mathematical formulation of the mapping problem is presented and a couple of reformulations involving strong valid inequalities are proposed. Section 4 is dedicated to a

B&B algorithm where lower and upper bounding procedures are elaborated in detail. In Section 5, we evaluate the effectiveness of valid inequalities and the proposed B&B algorithm. Finally, concluding remarks follow in Section 6.

2 Problem background

VMs play an important role in a cloud computing environment. A customer's request consists of a number of VMs, which are allocated on servers to execute a specific program.Without special restrictions, a server can usually run multiple VMs simultaneously.

In order to improve the utility of data center resources, VMs should be dynamically started or stopped and sometimes live migration should be conducted, i.e., move a VM from one server to another. Thus virtual communications should also be mapped to the physical network.

The focus of this paper is on the assignment of virtual resources to a given physical network. The solution to this problem is how to map the cloud resources and which servers and links should be used subject to certain hard constraints, e.g. resource capacity constraints, traffic routing constraints.

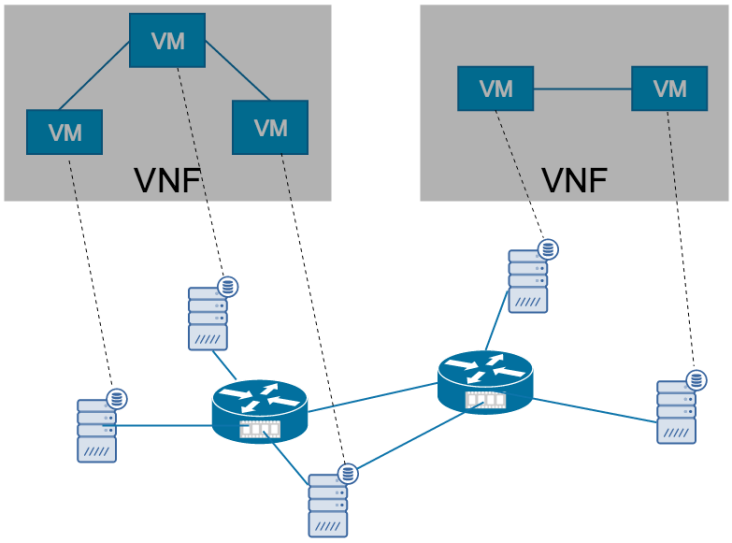

Figure 1: An illustration of the mapping procedure

Figure 1 illustrates the mapping problem involving two virtual requests. One has three VMs and the other has two. In addition, VMs within the same request communicate with each other. Dotted lines show a feasible mapping solution that VMs of each request are mapped to different severs and the communication throughput between each pair of VMs is routed between the corresponding servers that VMs are mapped to.

Furthermore, one may need to be aware that the communication throughput between two servers (which host VMs) should route on a single path, as multi-path routing may cause discrepancies among the arrivals of data at the destination. So we assume that for each origin-destination (O-D) pair the corresponding traffic is routed on a shortest path. Since VMs within a request often

communicate, we also assume that the request graph induced by VMs and virtual links is connected.

3 Formulations

Recall that a virtual request consists of a set of virtual machines and their mutual virtual communications. Therefore, we may represent each virtual request as a directed graph. Our goal is to map such graphs to a physical network. Henceforth, we will use the following notation to construct mathematical expressions.

Sets

R	set of virtual request
$H = (S, E)$	the connected graph of a physical network
S	set of servers in the physical network
E	set of undirected edges in the physical network
$G^r = (V^r, L^r)$	a graph of virtual network for request $r \in R$
V^r	set of VMs of request r
L^r	set of undirected virtual links of request r

Parameters

c^{ri}	required CPU of VM $i \in V^r$
m^{ri}	required memory of VM $i \in V^r$
C_k	CPU cores of server k
M_k	memory capacity of server k
F_k	fixed cost of server $k \in S$
A_k	additional cost of server k imposed from CPU loads
f^{rij}	required throughput associated with logical link $(i, j) \in L^r$
B_e	bandwidth of edge $e \in E$
W_e	fixed cost of edge $e \in E$
P_{kp}	shortest k-p path, $(k, p) \in S \times S : k \neq p$

Variables

$x_k^{ri} \in \{0, 1\}$	1 if VM i of request r is mapped to server k
$\theta_k \in \{0, 1\}$	1 if server k is used (switched on)
$\phi_e \in \{0, 1\}$	1 if edge e is used (switched on)

Notice that for a parameter or a variable, its subscripts (if it has) are associated with physical resources while its superscripts (if it has) are associated with virtual resources. For sake of convenience sometimes variables and parameters are presented in vector form where they are marked in bold. For instance $\mathbf{x}^r$ represents $\left\{x_k^{ri} : i \in S, i \in V^r\right\}$.

We construct the exact mathematical model of the mapping problem as follows

$$\min \quad \sum_{k\in S} F_k\theta_k + \sum_{k\in S} A_k \sum_{r\in R}\sum_{i\in V^r} c^{ri}x_k^{ri} + \sum_{e\in E} W_e\phi_e \tag{$\mathbb{P}$}$$

$$\text{s.t.} \quad \sum_{k\in S} x_k^{ri} = 1 \qquad \forall r\in R, i\in V^r \tag{AC}$$

$$\sum_{i\in V^r} x_k^{ri} \le \theta_k \qquad \forall r\in R, \forall k\in S \tag{LC}$$

$$\sum_{r\in R}\sum_{i\in V^r} c^{ri}x_k^{ri} \le C_k\theta_k, \qquad \forall k\in S \tag{KP}$$

$$\sum_{r\in R}\sum_{i\in V^r} m^{ri}x_k^{ri} \le M_k\theta_k \qquad \forall k\in S \tag{KP'}$$

$$\sum_{r\in R}\sum_{\substack{k,p\in S:\\ k\neq p, e\in P_{kp}}}\sum_{\{i,j\}\in L^r} f^{rij}x_k^{ri}x_p^{rj} \le B_e\phi_e \qquad \forall e\in E \tag{QC}$$

$$\theta_k, \phi_e, x_k^{ri} \in \{0,1\} \qquad \forall r\in R, i\in V^r, k\in S, e\in E. \tag{BC}$$

Henceforth we will use ($\mathbb{P}$) to represent this model and we interpret it as follows

- The objective is to minimize the total cost, which is additively composed of three terms: the fixed cost incurred by switching on servers, the additional cost coming from the CPU load, and the fixed cost from the usage of links. We model the additional cost induced by CPU load as a linear function to represent the fact that CPU is usually categorized as load dependent resource while memory is load independent [14].
- Constraints (AC) mean that each virtual machine must be mapped to a single server. Constraints (LC) model the fact that virtual machines are usually mapped separately in a cloud environment due to some practical issues, e.g., security, reliability.
- Constraints (KP, KP') are knapsack constraints. They ensure that for each server, the aggregated required CPU, memory resource cannot exceed its limits. Constraints (QC) emphasis the fact that for each edge, the aggregated throughputs on the edge cannot exceed the bandwidth.

Before the solution procedure, we briefly analyze problem structure. First, the combination of integrality constraints and bilinear constraints (nonconvex) makes the problem rather difficult. Second, the bilinear products appearing in the formulation are dense. Third, the problem aggregates features of the knapsack problem with multiple constraints and the QAP problem. In fact the authors [1] show that the mapping problem of the form ($\mathbb{P}$) is strongly NP-hard even if $|R| = 1$ as there is a polynomial time reduction from the maximum stable set problem. In what follows, we will focus on the solution procedure of ($\mathbb{P}$).

3.1 Reformulations

Model ($\mathbb{P}$) is a 0-1 bilinear constrained problem. A fundamental idea to deal with such problems is *lifting* it to a higher dimensional space [4, 5]. Introducing new variables y_{kp}^{rij} and enforcing $y_{kp}^{rij} = x_k^{ri}x_p^{rj}$ for each $(r,i,j,k,p) : i\neq j, k\neq p$, we lift the problem to a higher dimensional space

and lead to a MIP, at the expense of introducing non-convex equations. Simple convex relaxations can be achieved by linearization techniques. In this paper, we adopt the well-known McCormick inequalities [17]. Specifically, for each $(r, i, j, k, p) : i \neq j, k \neq p$, we approximate the equation $y_{kp}^{rij} = x_{ik}^r x_{jp}^r$ using the following four inequalities

$$x_k^{ri} + x_p^{rj} - 1 \leq y_{kp}^{rij} \tag{1a}$$

$$y_{kp}^{rij} \leq x_k^{ri} \tag{1b}$$

$$y_{kp}^{rij} \leq x_p^{rj} \tag{1c}$$

$$y_{kp}^{rij} \geq 0. \tag{1d}$$

And (QC) is linearized as

$$\sum_{r \in R} \sum_{\substack{k,p \in S: \\ k \neq p, e \in P_{kp}}} \sum_{\{i,j\} \in L^r} f^{rij} y_{kp}^{rij} \leq B_e \phi_e, \ \forall e \in E \tag{QCL}$$

With this relaxation, we can convert the 0-1 bilinear model ($\mathbb{P}$) to a MIP

$$\mathbb{P}_{\text{MC}} : \{(\text{AC}), (\text{LC}), (\text{KP}), (\text{KP'}), (\text{QCL}), (\text{BC})\} \cap \{(1a - 1d)\}. \tag{$\mathbb{P}_{\text{MC}}$}$$

The numerical results in [27] show that the bound provided by the continuous relaxation of ($\mathbb{P}_{\text{MC}}$) is weak. To strengthen the formulation, we need stronger valid inequalities. An important subset of those can be derived by the Reformulation-Linearization-Technique (RLT) [25], which is a general framework for generating valid inequalities in higher dimensional space for non-convex discrete and continuous formulations. We employ the RLT for the assignment constraints (AC) producing $\sum_{r \in R} |V^r|(|V^r| - 1)|S|$ linear equations:

$$\sum_{k \in S: k \neq p} y_{pk}^{rji} = x_p^{rj} \quad \forall (r, i, j, p) : i \neq j. \tag{AC_{RLT}}$$

Proposition 1. *Constraints (1a-1c) are implied by* (AC_{RLT}).

Proof. It is obvious to see that (AC_{RLT}) imply (1b),(1c). Regarding (1a), for each $(r, i, j, k, p) : k < p, i \neq j$, we have

$$\begin{aligned}
x_k^{ri} + x_p^{rj} - 1 &= y_{kp}^{rij} + \sum_{p' > k : p' \neq p} y_{kp'}^{rij} + \sum_{p' < k} y_{p'k}^{rji} + x_p^{rj} - 1 \\
&= y_{kp}^{rij} + \sum_{p' > k : p' \neq p} y_{kp'}^{rij} + \sum_{p' < k} y_{p'k}^{rji} - \sum_{s : s \neq p} x_s^{rj} \\
&= y_{kp}^{rij} - x_k^{rj} + \sum_{p' > k : p' \neq p} (y_{kp'}^{rij} - x_{p'}^{rj}) + \sum_{p' < k} (y_{p'k}^{rji} - x_{p'}^{rj}) \\
&\leq y_{kp}^{rij} - x_k^{rj} \\
&\leq y_{kp}^{rij}.
\end{aligned}$$

The first equality and the first inequality follow from (AC_{RLT}). The second equality comes from (AC). □

We also employ the RLT for (LC) by multiplying it by itself, leading to

$$\sum_{(i,j)\in V^r\times V^r: i\neq j} y_{kp}^{rij} \leq \theta_k, \quad \forall r \in R, \forall k \neq p \in S^2. \tag{2}$$

Remark 1. *In [26], we generate RLT inequalities for the location constraints (LC) by multiplying both sides x_p^{rj} leading to a new bilinear term $x_p^{rj}\theta_k$. Then we introduce new variables z_{pk}^{rj} and inequalities to relax $z_{pk}^{rj} = x_p^{rj}\theta_k$. This leads to a significant number of constraints and new variables but numerically limited improvement. In contrast, constraints (2) are compact and effective.*

The RLT based formulation can be presented as follows:

$$\begin{aligned} \min \ & \sum_{k\in\mathcal{S}} F_k\theta_k + \sum_{k\in S} A_k \sum_{r\in R}\sum_{i\in V^r} c^{ri}x_k^{ri} + \sum_{e\in E} W_e\phi_e \\ & (\text{AC}), (\text{AC}_{\text{RLT}}), (\text{KP}), (\text{KP'}), (\text{QCL}), \\ & (\text{LC}), (2), (1\text{d}), \\ & \theta_k, \phi_e, x_k^{ri} \in \{0,1\},\ r \in R, i \in V^r, k \in S, e \in E. \end{aligned} \tag{$\mathbb{P}_{\text{RLT}}$}$$

We now present three families of strong linear inequalities exploiting the problem structure. For each $r \in R, (i,j) \in L^r$, if there is a required throughput between i and j (i.e., $f^{rij} > 0$), then for each link $e \in E$, we have

$$\sum_{(k,p)\in S^2: k\neq p, e\in P_{kp}} y_{kp}^{rij} \leq \phi_e. \tag{3}$$

Similarly, for each $r \in R, (k,p) \in S^2 : e \in P_{kp}$, if there exists some throughput mapped along path P_{kp}, then for each link $e \in P_{kp}$ we have

$$\sum_{\{i,j\}\in L^r} y_{kp}^{rij} \leq \phi_e. \tag{4}$$

Another set of valid inequalities can be generated by exploiting some topological property of the substrate graph. Recall that a pair of nodes in a graph is connected if there is path between them. For each virtual request r, we have $|V|^r$ connected VMs mapped to servers. Thus at least $|V|^r$ servers are connected and the number of links connecting these servers should be at least $|V|^r - 1$. This can be represented as

$$\sum_{e\in E} \phi_e \geq \max_{r\in R}\{|V^r|\} - 1. \tag{5}$$

Numerical results in Section 5 show that (3)-(5) strengthen the formulation evidently. We now present the compact MIP model for the exact solution procedure of problem ($\mathbb{P}$) as the first contribution of this paper:

$$\begin{aligned} \min \ & \sum_{k\in\mathcal{S}} F_k\theta_k + \sum_{k\in S} A_k \sum_{r\in R}\sum_{i\in V^r} c^{ri}x_k^{ri} + \sum_{e\in E} W_e\phi_e \\ & (\text{AC}), (\text{AC}_{\text{RLT}}), (\text{KP}), (\text{KP'}), (\text{QCL}), \\ & (\text{LC}), (2), (3)-(5), (1\text{d}), \\ & \theta_k, \phi_e, x_k^{ri} \in \{0,1\},\ r \in R, i \in V^r, k \in S, e \in E. \end{aligned} \tag{$\mathbb{P}_1$}$$

4 The B&B algorithm

As indicated in Section 5, the computational performance of ($\mathbb{P}_1$) is over 10 times more efficient than that of ($\mathbb{P}_{\text{MC}}$) showing the effectiveness of valid inequalities. However it can be still challenging when the number of VMs is over 30.

To further improve the scalability we describe a B&B algorithm to solve the mapping problem to global optimality. We first introduce the Lagrange decomposition based lower bounding procedure, then propose an upper bounding heuristic algorithm. Finally we describe details of the overall algorithm.

4.1 The lower bounds

The convergence of a B&B algorithm largely depends on the strength of lower bounds. This section aims at generating novel valid inequalities leading to lower bounds that are strong than those provided by the continuous relaxation of ($\mathbb{P}_1$). This is achieved by a Lagrange decomposition scheme which involves evaluating a number of subproblems. And for this reason we call these inequalities *Lagrange cuts*. We then generalize the single request decomposition to a decomposition hierarchy.

4.1.1 Request based decomposition

We first present a decomposition leading to a number of subproblems associated with each request, each sever and each link. To this end, we disaggregate constraints (KP), (KP'), (QCL) by reformulating them with a handful of auxiliary variables with corresponding interpretations below

- $w_k^r \in [0, C_k]$ reserved CPU for request r on server k,
- $z_k^r \in [0, M_k]$ reserved Memory for request r on server k,
- $\kappa_e^r \in [0, B_e]$ reserved bandwidth for request r on link e.

The equivalent counterparts of (KP), (KP') and (QCL) are then

$$\sum_{i \in V_r} c^{ri} x_k^{ri} \leq w_k^r, \quad r \in R, k \in S, \tag{6}$$

$$\sum_{i \in V_r} m^{ri} x_k^{ri} \leq z_k^r, \quad r \in R, k \in S, \tag{7}$$

$$\sum_{\substack{i,j \in V_r: \\ i \neq j}} \sum_{\substack{k,p \in S: \\ k \neq p, e \in P_{kp}}} f^{rij} y_{kp}^{rij} \leq \kappa_e^r, \quad r \in R, e \in E, \tag{8}$$

$$\sum_{r \in R} w_k^r \leq C_k \theta_k, \quad k \in S, \quad \boldsymbol{\lambda} \in \mathbb{R}_+^{|S|} \tag{9}$$

$$\sum_{r \in R} z_k^r \leq M_k \theta_k, \quad k \in S, \quad \boldsymbol{\mu} \in \mathbb{R}_+^{|S|} \tag{10}$$

$$\sum_{r \in R} \kappa_e^r \leq B_e \phi_e, \quad e \in E. \quad \boldsymbol{\sigma} \in \mathbb{R}_+^{|E|} \tag{11}$$

To make the problem separable by request while ensuring strong lower bounds, we copy variables $\boldsymbol{\theta}$ and $\boldsymbol{\phi}$ by introducing the following constraints:

$$\theta_k^r \leq \theta_k, \qquad r \in R, k \in S, \quad \boldsymbol{\eta} \in \mathbb{R}_+^{|R|\times|S|}, \tag{12}$$

$$\phi_e^r \leq \phi_e, \qquad e \in E, \quad \boldsymbol{\zeta} \in \mathbb{R}_+^{|R|\times|E|}. \tag{13}$$

(12)-(13) imply the fact that if a server/link is used by one request then it must be on and conversely if a sever/link is on then it is not necessarily used by all the requests. As a result, the upper bounds of w_k^r, z_k^r, κ_e^r can be strengthened to $C_k\theta^r, M_k\theta_k^r$ and $B_e\phi_k^r$ respectively. In addition we relax the connectivity constraint (5) with $\rho \in \mathbb{R}_+$.

We replace θ_k with θ_k^r in constraints (LC) and ϕ_e with ϕ_e^r in constraints (QCL). Let us denote the resulting formulation as a *lifted* version of ($\mathbb{P}_1$):

$$\mathbb{P}_2 : \{(\mathbb{P}_1)\} \cap \{(9), (10), (11), (12), (13)\}. \tag{14}$$

Proposition 2. *The projection of the feasible region of* (14) *in variables* $(\mathbf{x}, \mathbf{y}, \boldsymbol{\theta}, \boldsymbol{\phi})$ *is exactly the feasible region of* ($\mathbb{P}_1$).

Proof. For any feasible point $\mathbf{v} = (\mathbf{w}, \mathbf{z}, \boldsymbol{\kappa}, \boldsymbol{\theta}^r, \boldsymbol{\phi}^r, \mathbf{x}, \mathbf{y}, \boldsymbol{\theta}, \boldsymbol{\phi})$ in (14), one can get a feasible point in ($\mathbb{P}_1$) by truncating the components of variables $(\mathbf{x}, \mathbf{y}, \boldsymbol{\theta}, \boldsymbol{\phi})$ from $\mathbf{v}$. □

Relaxing the five sets of constraints with associated Lagrange multipliers $\boldsymbol{\lambda}, \boldsymbol{\mu}, \boldsymbol{\sigma}, \boldsymbol{\eta}, \boldsymbol{\zeta}, \rho$ leads to the Lagrange function over variables $(\boldsymbol{\lambda}, \boldsymbol{\mu}, \boldsymbol{\sigma}, \boldsymbol{\eta}, \boldsymbol{\zeta}, \rho; \mathbf{w}, \mathbf{z}, \boldsymbol{\kappa}, \mathbf{x}, \boldsymbol{\theta}, \boldsymbol{\phi})$. For ease of notation, let $\mathbf{v} = (\mathbf{w}, \mathbf{z}, \boldsymbol{\kappa}, \mathbf{x}, \boldsymbol{\theta}, \boldsymbol{\phi}) \in \mathbb{R}_+^p, \mathbf{v}^* = (\boldsymbol{\lambda}, \boldsymbol{\mu}, \boldsymbol{\sigma}, \boldsymbol{\eta}, \boldsymbol{\zeta}, \rho) \in \mathbb{R}_+^d$, where p and d represent the respective number of variables in primal and dual space. The Lagrange is then

$$\begin{aligned}
\mathcal{L}(\mathbf{v}^*, \mathbf{v}) = &\sum_{r\in R}\left(\sum_{k\in S}\left(A_k \sum_{i\in V_r} c^{ri}x_k^{ri} + w_k^r\lambda_k + \mu_k z_k^r + \theta_k^r\eta_k^r\right) + \sum_{e\in E}(\sigma_e\kappa_e^r + \zeta_e^r\phi_e^r)\right) \\
&+ \sum_k \left(F_k - C_k\lambda_k - M_k\mu_k - \sum_{r\in R}\eta_k^r\right)\theta_k \\
&+ \sum_e \left(W_e - B_e\sigma_e - \sum_{r\in R}\zeta_e^r - \rho\right)\phi_e \\
&+ (\max_{r\in R}|V^r| - 1)\rho.
\end{aligned} \tag{15}$$

For the sake of clarity let $\tau = (\max_{r\in R}|V^r| - 1)\rho$. For any $\mathbf{v}^* \in \mathbb{R}_+^d$, it holds that the infimum

$$\Psi(\mathbf{v}^*) = \inf_{\mathbf{v}\in\mathcal{X}} \mathcal{L}(\mathbf{v}^*, \mathbf{v}) \tag{16}$$

provides a lower bound of the optimal value of the mapping problem, where $\mathcal{X}$ represents the remaining constraint set. Since $\mathcal{X}$ is compact, the supreme is attainable. Given $\mathbf{v}^*$, the Lagrange function is separable w.r.t. each request, server and edge. Thus evaluating $\min_{\mathbf{v}} \mathcal{L}(\mathbf{v}^*, \mathbf{v})$ reduces to evaluating $|R| + |S| + |E|$ subproblems.

For each $r \in R$, we evaluate $\Psi^r(\mathbf{v}^*)$ by solving

$$\min \quad \sum_{k\in S}\left(A_k \sum_{i\in V_r} x_k^{ri} c^{ri} + \lambda_k w_k^r + \mu_k z_k^r + \theta_k^r \eta_k^r\right) + \sum_{e\in E}(\sigma_e \kappa_e^r + \zeta_e^r \phi_e^r) \tag{Sub$_r$}$$

$$\text{s.t. } (\text{AC}), (\text{AC}_{\text{RLT}}), (\text{QCL}), (\text{LC}), (2), (3)-(5), (1\text{d}), (6), (7), (8),$$

$$w_k^r \le C_k \theta_k^r, \ \forall k \in S \tag{17}$$

$$z_k^r \le M_k \theta_k^r, \ \forall k \in S \tag{18}$$

$$\kappa_e^r \le B_e \phi_e^r, \ \forall e \in E, \tag{19}$$

$$x_k^{ri}, \theta_k^r, \phi_e^r, \in \{0,1\} \ \forall i,k,e.$$

As server k is used by request r if and only if there is a VM mapped to server k, We can strengthen (LC) in (Sub$_r$) as follows

$$\sum_{i\in V^r} x_k^{ri} = \theta_k^r. \tag{20}$$

Moreover, given that each request graph G^r is connected, constraints (5) in (Sub$_r$) can be replaced with

$$\sum_{e\in E} \phi_e^r \ge \sum_{k\in S} \theta_k^r - 1, \tag{21a}$$

$$\theta_k^r \le \sum_{e\in E: k\in e} \phi_e^r, \quad k \in S, \tag{21b}$$

as the graph induced by servers that are used by request r is also connected.

In what follows, let X^r be the feasible region of (Sub$_r$) in primal variables $(\mathbf{w}^r, \mathbf{z}^r, \boldsymbol{\kappa}^r, \mathbf{x}^r, \boldsymbol{\theta}^r, \boldsymbol{\phi}^r)$, i.e.,

$$X^r = \{(\mathbf{w}^r, \mathbf{z}^r, \boldsymbol{\kappa}^r, \mathbf{x}^r, \boldsymbol{\theta}^r, \boldsymbol{\phi}^r) : (\text{AC}), (\text{AC}_{\text{RLT}}), (\text{QCL}), (2), (3)-(4), (1\text{d}),$$
$$(6), (7), (8), (20), (21), (17)-(19), x_{ik}^r, \theta_k^r, \phi_e^r \in \{0,1\}\}.$$

Similarly, for each $k \in S$, and $e \in E$, we evaluate $\Psi^k(\mathbf{v}^*)$ and $\Psi^e(\mathbf{v}^*)$ via

$$\Psi^k(\boldsymbol{\lambda}, \boldsymbol{\mu}, \boldsymbol{\eta}) = \min_{\theta_k \in \{0,1\}} \left(F_k - C_k \lambda_k - M_k \mu_k - \sum_{r\in R} \eta_k^r\right) \theta_k, \tag{22}$$

$$\Psi^e(\boldsymbol{\sigma}, \boldsymbol{\zeta}) = \min_{\phi_e \in \{0,1\}} \left(W_e - B_e \sigma_e - \sum_{r\in R} \zeta_e^r - \rho\right) \phi_e. \tag{23}$$

It is straightforward to see that the optimal solution of the above problem is 1 if the coefficient is negative and 0 if nonnegative. Consequently, the dual objective denoted by $\Psi(\mathbf{v}^*)$ is additively composed of functions $\{\Psi^r\}_{r\in R}$, $\{\Psi^k\}_{k\in S}$ and $\{\Psi^e\}_{e\in E}$ as follows

$$\Psi(\mathbf{v}^*) = \sum_{r\in R} \Psi^r(\mathbf{v}^*) + \sum_{k\in S} \Psi^k(\mathbf{v}^*) + \sum_{e\in E} \Psi^e(\mathbf{v}^*) + \tau \tag{24}$$

Let us assume that primal problem (14) is strictly feasible. Then the Slater's condition holds and the dual problem has a nonempty compact set of maximum points [16]. Thus the dual problem can be defined below

$$\max_{\mathbf{v}^* \in \mathbb{R}^d_+} \Psi(\mathbf{v}^*). \tag{25}$$

By weak duality, it holds that

$$\max_{\mathbf{v}^* \in \mathbb{R}^d_+} \Psi(\mathbf{v}^*) \leq z^*$$

where z^* is the optimum of $(\mathbb{P}_1)$.

4.1.2 On the strength of lower bounds

It is known that the strong duality generally does not hold between $(\mathbb{P}_1)$ and (25). In other words, the duality gap exists. Naturally we may ask:

> *Could we get a priori knowledge or intuition on the quality of the lower bound without actual numerical experiments?*

Much effort has been devoted to relevant investigations in different contexts (see e.g. [10, 11, 16, 15]). Among them, we recall the following theorem.

Theorem 1. *[10] Consider a mixed integer linear problem expressed as*

$$\min_x \{cx : \ Ax \leq b, x \in X\}.$$

where X is bounded and compact. Relaxing constraints $Ax \leq b$ with Lagrange multipliers $\lambda \geq 0$ leads to

$$g(\lambda) = \min_{x \in X} \ cx + \lambda^T (Ax - b).$$

Then it holds that

$$\max_{\lambda \geq 0} \ g(\lambda) = \min \{cx : \ Ax \leq b, x \in \text{conv}(X)\}$$

where $\text{conv}(X)$ *represents the convex hull of X.*

Later Lemaréchal generalized the above result for general Mixed Integer Nonlinear Problems (MINLPs). We refer interested readers to [16, 15] for details.

Corollary 1. *The optimum of (25) dominates (greater than or equal to) that of continuous relaxation of (14) and it amounts to outer-approximating the convex hull of the primal feasible region with the following set*

$$\mathcal{S} = \text{conv}\{\bigtimes_{r \in R} X^r\} \bigcap \{\mathbf{v} : (5), (9), (10), (11), (12), (13)\}, \tag{26}$$

where X^r denote the feasible region of subproblem (Sub_r) and $\times$ denotes the Cartesian product operation: $\bigtimes_{r \in R} X^r = \{(x_1, \ldots, x_{|R|}) : x_i \in X^r, \ r \in R\}$.

Proof. It follows directly from Theorem 1 that the Lagrange decomposition amounts to constructing $\mathcal{S}$. Since $\text{conv}\, X^r$ is the tightest convex relaxation of X^r, the optimum of (25) is greater than or equal to the linear relaxation objective value. □

4.1.3 The choice of Lagrange multipliers

Even though each subproblem (Sub$_r$) is easier than problem ($\mathbb{P}_1$), they are still computationally costly; subgradient algorithms [3] often take hundreds of iterations to converge. To reduce the number of iterations evidently, we propose to first solve the continuous relaxation of the extended formulation (14) to optimality and take the dual multipliers of constraints (9)- (13) as the initial values the Lagrange multipliers. With these initial values, we evaluate (24) and obtain l^1 as the optimal value. Let l^{cts} be optimal value of the continuous relaxation of (14).

Proposition 3. *It holds that* $l^1 \geq l^{cts}$.

Proof. Let $\bar{\mathbf{v}}^*$ be the corresponding dual values w.r.t to constraints (9)- (13) and $\overline{\Psi}^r, \overline{\Psi}^k, \overline{\Psi}^e$ be the respective optimal value of the continuous relaxation of subproblem (Sub$_r$), (22), (23) associated with request $r \in R$ with the Lagrange multipliers $\bar{\mathbf{v}}^*$.

1. Due to integrality constraints in (Sub$_r$), it holds that $\overline{\Psi}^r \leq \Psi^r$ for each $r \in R$; thus $l^1 \geq \sum_{r\in R} \overline{\Psi}^r(\bar{\mathbf{v}}^*) + \sum_{k\in S} \overline{\Psi}^k(\bar{\mathbf{v}}^*) + \sum_{e\in E} \overline{\Psi}^e(\bar{\mathbf{v}}^*) + \tau$.

2. Let $\bar{\mathbf{v}}$ be the optimal solution of the continuous relaxation of (14), then $\sum_{r\in R} \overline{\Psi}^r(\bar{\mathbf{v}}^*) + \sum_{k\in S} \overline{\Psi}^k(\bar{\mathbf{v}}^*) + \sum_{e\in E} \overline{\Psi}^e(\bar{\mathbf{v}}^*)$ is the dual optimal value of the continuous relaxation of (14). By strong duality we have $l^{cst} = \sum_{r\in R} \overline{\Psi}^r(\bar{\mathbf{v}}^*) + \sum_{k\in S} \overline{\Psi}^k(\bar{\mathbf{v}}^*) + \sum_{e\in E} \overline{\Psi}^e(\bar{\mathbf{v}}^*) + \tau$.

Combining the above arguments leads to $l^1 \geq l^{cst}$. □

4.1.4 Lagrange cuts

In this section we show that we can generate some valid inequalities to capture the strength of the Lagrange decomposition characterized by (26). Thus we call these inequalities *Lagrange cuts*. As a result we can strengthen the linear relaxation of (14) upon each resolution of (Sub$_r$).

Proposition 4. *For each* $r \in R$, (Sub$_r$) *amounts to finding a supporting hyperplane of* $\operatorname{conv}(X^r)$ *with outer normal vector* $(-\boldsymbol{\lambda}, -\boldsymbol{\mu}, -\boldsymbol{\sigma}, -\{A_k c^{ri}\}_{i,k}, -\boldsymbol{\eta}, -\boldsymbol{\zeta})$ *defined by equation*

$$H^r(\mathbf{v}) = \Psi^r(\mathbf{v}^*) - \sum_{k\in S}(A_k \sum_{i\in V_r} x_k^{ri} c^{ri} + \lambda_k w_k^r + \mu_k z_k^r + \theta_k^r \eta_k^r) - \sum_{e\in E}(\sigma_e \kappa_e^r + \zeta_e^r \phi_e^r) = 0$$

where $\mathbf{v} = (\mathbf{w}^r, \mathbf{z}^r, \boldsymbol{\kappa}^r, \mathbf{x}^r, \boldsymbol{\theta}^r, \boldsymbol{\phi}^r)$ *and* $\Psi^r(\mathbf{v}^*)$ *is the optimal value of* (Sub$_r$).

Proof. By the definition of (Sub$_r$), it holds that

$$H^r(\mathbf{v}) \leq 0 \tag{27}$$

for any point $\mathbf{v} \in X^r$ and it exists at least one point $\mathbf{v}' \in X^r$ such that $H^r(\mathbf{v}') = 0$, ending the proof. □

The proposition above provides a non-trivial valid inequality $H^r(\mathbf{v}) \leq 0$ for each $\mathbf{v} \in \operatorname{conv} X^r$ and therefore it is also valid for the convex set defined in (26). Thus we can append these inequalities to strengthen the linear relaxation of (14). Moreover we show that the resulting linear relaxation value of (14) will be greater than or equal to the current Lagrange lower bound.

Proposition 5. *The optimal value of the continuous relaxation of (14) augmented by $H^r(\mathbf{v}) \leq 0$ $(r \in R)$ is greater than or equal to l^1.*

Proof. Let $\pi^r = \sum_{k\in S}(A_k \sum_{i\in V_r} x_k^{ri} c^{ri} + \lambda_k w_k^r + \mu_k z_k^r + \theta_k^r \eta_k^r) + \sum_{e\in E}(\sigma_e \kappa_e^r + \zeta_e^r \phi_e^r)$. Then aggregating Lagrange cuts (27) over $r \in R$ leads to

$$\sum_{r\in R} \pi^r \geq \sum_{r\in R} \Psi^r(\mathbf{v}^*)$$

which is equivalent to

$$\sum_{r\in R} \pi^r + b \geq \sum_{r\in R} \Psi^r(\mathbf{v}^*) + b, \tag{28}$$

where $b = \sum_{k\in S}(F_k - C_k\lambda_k - M_k\mu_k - \sum_{r\in R}\eta_k^r)\theta_k + \sum_{e\in E}(W_e - B_e\sigma_e - \sum_{r\in R}\zeta_e^r - \rho)\phi_e + \tau$. On the one hand, (22) and (23) imply that

$$\sum_{r\in R} \Psi^r(\mathbf{v}^*) + b \geq \sum_{r\in R} \Psi^r(\mathbf{v}^*) + \sum_{k\in S} \Psi^k(\mathbf{v}^*) + \sum_{e\in E} \Psi^e(\mathbf{v}^*) + \tau = l^1. \tag{29}$$

On the other hand, we note that $\sum_{r\in R} \pi^r + b \leq f(\mathbf{v})$ is exactly the Lagrange function (15). Thus by weak duality it holds that $\sum_{r\in R} \pi^r + b \leq f(\mathbf{v})$ for any solution satisfying (KP)–(QCL). The proof is then complete. □

Remark 2. *The above discussion reveals the fact that Lagrange decomposition procedure can be regarded as a (Lagrange) cut generation procedure which can be called as needed in our B&B algorithm.*

4.1.5 A generalized decomposition hierarchy

Corollary 1 also indicates a hierarchy of request based Lagrange relaxations. Specifically, we may consider any possible partition of the request set R and apply the aforementioned decomposition strategy to the partition.

For ease of presentation, let us assume that we partition set R into $m \in \{1, \ldots, |R|\}$ disjoint subsets and denote this partition $\mathcal{P}_m = \{I_1, I_2, \ldots, I_m\}$, where $I_i \subset R$, $\forall i = 1, \ldots, m$. If $m = |R|$, then we get the single request based decomposition and if $m = 1$, we end up with the lifted formulation (14). Accordingly each subproblem (Sub_r) is defined on subset I_i. For instance constraint (6) becomes

$$\sum_{r\in I_j} \sum_{i\in V_r} c^{ri} x_k^{ri} \leq w_k^j,\ j \in \{1, \ldots, m\}, k \in S.$$

This leads to a hierarchy of decompositions and we denote by l_m the resulting lower bound associated with parameter m.

Corollary 2. *Let m_1, m_2 be any integer in $\{1, \ldots, |R|\}$. If $m_1 \leq m_2$ and for any subset $I \in \mathcal{P}_{m_2}$ there exists a subset $I' \in \mathcal{P}_{m_1}$ such that $I \subset I'$, then $l_{m_1} \geq l_{m_2}$.*

Proof. By Corollary 1, the Lagrange decomposition amounts to the following outer-approximation

$$S^m = \text{conv}\{\bigtimes_{i\in\{1,\ldots,m\}} X^i\} \bigcap \{\mathbf{v} : (5), (9), (10), (11), (12), (13)\},$$

where X^i represents the feasible region of constraints associated with requests in set I_i. The condition that for any subset $I \in \mathcal{P}_{m_2}$ there exists a subset $I' \in \mathcal{P}_{m_1}$ such that $I \subset I'$ implies that $S^{m_1} \subseteq S^{m_2}$ ending the proof. □

As an illustration, we can partition 6 virtual requests (30 VMs) with $R = \{1, \ldots, 6\}$ to 6 subsets, each has a single request; 3 subsets with $I_1 = \{1, 2\}, I_2 = \{3, 4\}, I_3 = \{5, 6\}$. Let l_6, l_3 be their respective Lagrange lower bounds associated with the partition-based Lagrange relaxation. Then it follows from Corollary 2 that $l_6 \leq l_3$. To accelerate the solution procedure of subproblems involving multiple requests, one can apply the aforementioned request-based decomposition approach recursively.

4.2 The upper bounds

It is known that solving the dual problem (25) (or evaluating (24)) does not produce any primal feasible solution. Recovering a high quality one often calls for appropriate heuristics. In this paper we exploit the information of subproblems (Sub$_r$) and construct a feasible solution and an upper bound. If the resulting upper bound is weak we improve the solution using the *local branching* technique introduced in [8].

The overall algorithm is presented in Alg. 1. Given an optimal solution to the Lagrange problem (15), we partition the index set S into two disjoint subsets by inspecting values of $\sum_{r\in R} \overline{\theta}^r_k$. On the one hand, if $\sum_{r\in R} \theta^r_k = 0$, we guess that server k is unlikely used in a good feasible solution and thus set $\theta_k = 0$. This eliminates binary x^{ri}_k $(r \in R\ i \in V_r)$ by location constraint (LC). Moreover we restrict this unused server k isolated from used ones by imposing $\phi_e = 0 \quad \forall e \in E : k \in e$. On the other hand, if $\sum_{r\in R} \theta^r_k \geq n$ $(n \in \{1, \ldots, |R|\})$ we set $\theta_k = 1$; to further reduce the number of binary variables, we fix binary variables $\{x^{ri}_k : r \in R, i \in V^r\}$ if they do not violate coupling constraints (9)–(10).

As a result we end up with a small MIP which may return a feasible solution. If the MIP is infeasible we switch on the cheapest server among the unused servers and repeat the procedure.

Let LB be the lower bound obtained by solving (15) and $UB, \tilde{\mathbf{v}} = (\tilde{\mathbf{x}}, \tilde{\boldsymbol{\theta}}, \tilde{\boldsymbol{\phi}})$ the resulting upper bound and feasible solution using Alg. 1. If gap $= \frac{UB-LB}{UB}$ is large we mploy the local branching technique [8] to improve it by appending the distance constraint (30) to ($\mathbb{P}_1$)

$$d(\mathbf{v}, \tilde{\mathbf{v}}) \leq \pi. \tag{30}$$

where $d(\mathbf{v}, \tilde{\mathbf{v}})$ represents the distance between the optimal solution $\mathbf{v}$ and the current feasible solution $\tilde{\mathbf{v}}$ and π is an integer parameter. Following [8], we set $d(\mathbf{v}, \tilde{\mathbf{v}}) = \sum_{k\in S:\tilde{\theta}_k=1} (1 - \theta_k) + \sum_{e\in E:\tilde{\phi}_e=1} (1 - \phi_e) + \sum_{r\in R} \sum_{i\in V^r} \sum_{k\in S:\tilde{x}^{ri}_k=1} (1 - x^{ri}_k)$ and choose $\pi \in \{10, \ldots, 20\}$. We then solve the resulting augmented problem and try to get a better solution.

Algorithm 1: Repairing heuristic

Input : Solution $\bar{\mathbf{v}}$ to problem (15); an integer $n \in \{1, |R|\}$
Output: An feasible solution to problem ($\mathbb{P}_1$) and upper bound UB.
Let $\Theta_1 = \{k \in S : \sum_{r\in R} \overline{\theta}_k^r \geq n\}$ and $\Theta_0 = \{k \in S : \sum_{r\in R} \overline{\theta}_k^r = 0\}$.
while $|\Theta_0| \geq 0$ **do**
  Let $\theta_k = 0\ \forall k \in \Theta_0$.
  For each $k \in \Theta_0$, impose $\phi_e = 0,\ \forall e \in E : k \in e$.
  foreach $k \in \Theta_1$ **do**
    if $C_k - \sum_{r\in R} w_k^r \geq 0$ *and* $M_k - \sum_{r\in R} z_k^r \geq 0$ **then**
      Let $\theta_k = 1,\ x_k^{ri} = \overline{x}_k^{ri}\ (\forall r \in R\ i \in V^r)$.
    end
  end
  Solve model ($\mathbb{P}_1$) with partially fixed $\boldsymbol{\theta}, \mathbf{x}$ and constraints (33a).
  if *infeasible* **then**
    Find $k = \text{argmin}\{F_k : k \in \Theta_0\}$.
    Let $\Theta_0 = \Theta_0 \setminus \{k\}$.
  else
    break the loop.
  end
end
if *UB-LB* $\geq \epsilon\, UB$ **then**
  Append (30) to ($\mathbb{P}_1$), which is solved to update the feasible solution.
end

Remark 3. *To accelerate the procedure of Alg. 1 we restrict the MIP solution procedure within* $3 \times |R|$ *seconds. Of-course we do not pretend that it guarantees a high quality feasible solution. For this reason, Alg. 1 is called several times in our B&B algorithm.*

4.3 The algorithm

In this section, we elaborate details of the overall branch-and-bound algorithm. As highlighted before, the algorithm uses Lagrange decomposition stated in Sec. 4.1 and Alg. 1 to generate strong lower and upper bounds. In addition it distinguishes from the standard B&B algorithm in terms of branching rules and dynamic cut generation.

4.3.1 Branching

The B&B bound algorithm focus on branching over $\boldsymbol{\theta}$ and $\boldsymbol{\phi}$ variables. If a node is not fathomed when all $\boldsymbol{\theta}, \boldsymbol{\phi}$ are integral we export the arising subproblem to standard MIP solvers. This is due to the following reasons. First, cost coefficients of $\boldsymbol{\theta}$ and $\boldsymbol{\phi}$ are usually much larger than those of $\mathbf{x}$. And as will be explained in Section 4.4 a number of strong valid inequalities can be generated on the fly by fixing values of $\boldsymbol{\theta}$ or $\boldsymbol{\phi}$. Thus one can usually fathom a node by just branching over $\boldsymbol{\theta}$ and $\boldsymbol{\phi}$. Second, there might be many symmetries among variables $\mathbf{x}$, where symmetry breaking techniques should be investigated. This however is out of the scope of this paper. Instead we exploit relevant features of existing MIP solvers.

For $\boldsymbol{\theta}, \boldsymbol{\phi}$, we select variables that are most inconsistent with respect to linking constraints (12) and (13) as it is more likely to fathom a node. Let us illustrate this point for variables $\boldsymbol{\theta}$ and denote by $\bar{\theta}^r_k, \bar{\theta}_k$ the respective optimal values of (Sub$_r$) and (22) for each $k \in S$. If $\sum_{r\in R} \bar{\theta}^r_k = 0$, we will not branch it as the constraint $\theta^r_k \leq \theta_k$ always holds. Otherwise if $\sum_{r\in R} \bar{\theta}^r_k \geq 1$ and $\bar{\theta}_k = 0$, we branch on variable θ^k and create two nodes with $\theta_k = 0$ or $\theta_k = 1$. Then one can try to improve lower bounds of both nodes by solving a continuous relaxation or a Lagrange decomposition procedure. Note that the Lagrange bounds of these two child nodes can be improved even *without* updating the Lagrange multipliers. For the child node with $\theta_k = 0$, we can improve the Lagrange lower bound by simply evaluating (22) indexed by k and (Sub$_r$) where θ^r_k was determined as 1 at its parent node; while for the child node with $\theta_k = 1$, we can improve the lower bound by evaluating (22) indexed by k. When all $\boldsymbol{\theta}$ are consistent with $\boldsymbol{\theta}^r$, we branch over the most fractional component.

Remark 4. *Our numerical experiments show that without updating the Lagrange multipliers, the improvement of lower bound is significant for the child node with $\theta_k = 0$ but quite small for the node with $\theta_k = 1$. Thus for the latter case, we may still update Lagrange multipliers (by solving the continuous relaxation problem (14)).*

4.3.2 Bounding

As highlighted, variables $\boldsymbol{\theta}$ and $\boldsymbol{\phi}$ have dominating coefficients in the objective function. Thus bounding these variables might be beneficial in the procedure of B&B algorithm.

Let $\mathcal{F}_{(14)}$ be the continuous relaxation of the feasible region of (14). We evaluate bounds of $\sum_{k\in S} \theta_k$, $\sum_{e\in E} \phi_e$ by solving linear programs over $\mathcal{F}_{(14)}$ augmented by Lagrange cuts and the upper bounding constraint $f(\mathbf{v}) \leq UB$. For instance the upper bound of $\sum_{k\in S} \theta_k$ is obtained by solving the following linear problem

$$p = \max \left\{ \sum_{k\in S} \theta_k : \text{s.t. } \mathbf{v} \in \mathcal{F}_{(14)}, f(\mathbf{v}) \leq UB, (27) \right\} \tag{31}$$

and then taking $u_\theta = \lfloor p \rfloor$. Similarly we can get the upper bound of $\sum_{e\in E} \phi_e$ which we denote by u_ϕ. Moreover one can check the connectivity of used servers using the following proposition.

Proposition 6. *Given that all request graphs G^r $(r \in R)$ are connected, all used servers are connected if either of the following inequality holds*

$$u_\theta \leq \min_{r^1, r^2 \in R} \{|V|^{r_1} + |V|^{r_2}\} - 1, \tag{32a}$$

$$u_\phi \leq \min_{r^1, r^2 \in R} \{|V|^{r_1} + |V|^{r_2}\} - 3. \tag{32b}$$

Proof. The first inequality implies that any pair of two virtual requests share at least one server. The second one indicates that that at least one link is shared by any pair of two requests. Indeed suppose that there exist a pair of two requests $r_1, r_2 \in R$ using two disjoint sets of links. Then we have that

$$u_\phi \geq \sum_{e\in E} \phi^{r_1}_e + \phi^{r_2}_e$$

On the other hand (21) implies that

$$\sum_{e\in E}(\phi_e^{r_1}+\phi_e^{r_2}) \geq \sum_{k\in S}(\theta_k^{r_1}+\theta_k^{r_2})-2$$

And constraints (20) and (AC) imply that

$$\sum_{k\in S}(\theta_k^{r_1}+\theta_k^{r_2}) = \sum_{k\in S}\left(\sum_{i\in V^{r_1}} x_k^{r_1 i}+\sum_{i\in V^{r_2}} x_k^{r_2 i}\right) = |V|^{r_1}+|V|^{r_2}$$

These lead to that

$$u_\phi \geq \{|V|^{r_1}+|V|^{r_2}\}-2$$

which contradicts the inequality $u_\phi \leq \min_{r^1,r^2\in R}\{|V|^{r_1}+|V|^{r_2}\}-3$. □

Note that the fact that the graph induced by used servers is connected will be exploited in the next section to derive some strong cuts.

4.4 Cuts

In our B&B algorithm, two sets of cuts are added to the continuous relaxation of (14) at each child node dynamically, namely, the Lagrange cuts (27) and *connectivity cuts*. The former has been elaborated in Subsection 4.1.4 and we now introduce the latter.

For ease of presentation, we denote by H' the subgraph of H induced by all used servers and used links. If H' is connected, then the following inequalities are valid

$$\sum_{e\in E}\phi_e \geq \sum_{k\in S}\theta_k - 1, \tag{33a}$$

$$\theta_k \leq \sum_{e\in E:k\in e}\phi_e,\ k\in S, \tag{33b}$$

where the first one stipulates that at least $\sum_{k\in S}\theta_k - 1$ links are used and the second one comes from the definition of connectivity. It is straightforward to see that (33a) dominates (5). Let us call (33a) and (33b) connectivity cuts.

The overall B&B procedure is presented in Algorithm 2.

5 Numerical experiments

In this section, we assess the computational performance of the proposed formulation ($\mathbb{P}_1$), Lagrange decomposition procedure and Algorithm 2. Results in this section illustrate the following key points.

1. The proposed reformulation ($\mathbb{P}_1$) is effective. Numerically it outperforms the McCormick formulation ($\mathbb{P}_{\text{MC}}$) by orders of magnitudes.

2. The proposed Lagrange decomposition provides stronger lower bounds than the continuous relaxation bound of ($\mathbb{P}_1$).

3. Algorithm 2 outperforms the standard B&B algorithm of CPLEX 12.7 solver with formulation ($\mathbb{P}_1$) by orders of magnitudes.

Algorithm 2: The B&B algorithm

input : an instance data, the optimality tolerance ϵ, upper bound tolerance δ
output: An optimal solution and the optimal objective value
Step 1: Initialization
At root node solve the continuous relaxation of the reformulation (14).
Extract the dual multipliers with respect to constraints (9)-(13).
For each $r \in R$ evaluate (Sub_r); evaluate(22), (23).
Initialize lower bound LB with (24) and store $|R|$ Lagrange cuts (27).
Initialize UB using Alg. 1 and go to step 4.
Step 2: Evaluation
if *the branched variable (e.g.* θ_k*) is fixed to be* 0 *and* $\sum_{r \in R} \bar{\theta}_k^r > 0$ **then**
  For each $r \in R$ such that $\theta_k^r = 1$ evaluate (Sub_r); evaluate (22) and (23).
else
  Solve the continuous relaxation of the reformulationcenhanced by (27)
  Extract the dual multipliers with respect to constraints (9)-(13).
  For each $r \in R$ evaluate (Sub_r); evaluate(22), (23).
  Update lower bound LB with (24) and store $|R|$ Lagrange cuts (27).
end
If $UB - LB \geq \delta UB$, then update UB using Alg. 1. Go to Step 3.
Step 3: Termination
A node is fathomed if one of the following conditions is met:

1. $UB - LB \leq \epsilon UB$.
2. A feasible solution is found.
3. Any subproblem (Sub_r) is infeasible.

Otherwise, go to Step 4.
Step 4: Bounding
Update the upper bounds of $\sum_{k \in K} \theta_k$, $\sum_{e \in E} \phi_e$ with (31). **if** (32) *holds* **then**
  Replace (5) with (33a)-(33b)
end
Go to step 5.
Step 5: Branching
if *there exists a fractional component in* $(\boldsymbol{\theta}, \boldsymbol{\phi})$ **then**
  Select one variable from $\boldsymbol{\theta}, \boldsymbol{\phi}$ to according to rules in Sec. 4.3.1
  Create two child nodes by fixed the selected variable to 1 and 0 respectively. Go to step 2.
else
  Export the problem to a MIP solver.
  If the objective value is less than UB, update UB and go to Step 3.
end

5.1 Test instances

To the best of our knowledge, there is no public data sets for the virtual machine mapping problem. Following [18], we randomly generate virtual request instances following rules below.

1. As presented in Section 2, each virtual request typically involves a small number of VMs, so the size of each virtual request is fixed to 5. The number of virtual requests ranges from 1 to 10. In other words, we deal with problem instances up to 50 VMs.

2. For each virtual request, the communication traffic between each two VMs is uniformly generated as an integer in $\{0, 100\}$. For each virtual machine, the required number of CPU cores is randomly generated from 1 to 10 and the size of memory is generated from 2GB to 8GB.

Physical graph instances are from SND library [22]. Each node of a graph represents a server, whose number of CPU cores and memory size are randomly chosen as an ordered pair from set $\{(8, 128), (16, 256), (32, 512), (64, 1024)\}$. Parameters of link capacity and fixed cost are consistent with the data in SND library. The additional cost regarding CPU resource A_k is set as 10 for all $k \in S$. For each O-D pair of servers, the shortest path is computed using Dijkstra's algorithm before the solution procedure.

To measure the computational difficulty of each problem instance, we report the topology of the physical graph, the number of VMs and the number of binary variables.

5.2 Implementation and experiments setup

All computations are implemented with C++ and all problem instances are solved by the state-of-the-art solver CPLEX 12.7 with default settings on a Dell Latitude E7470 laptop with Intel Core(TM) i5-6300U CPU clocked at 2.40 GHz and with 8 GB of RAM. The B&B tree is implemented using a priority queue. To have a fair and unbiased comparison, CPU time is used as computational measurement. For all the tests, the computational time is limited to 10 hours and the feasibility tolerance is set as 10^{-4}. The optimality tolerance for Algorithm 2 is set to 0.5%. To improve the stability of computation, Lagrange multipliers are preserved to 4 decimal places. If no solution is available at solver termination or the solution process is killed by the solver, $(-)$ is reported.

5.3 The evaluation of formulation $(\mathbb{P}_1)$

We evaluate the strength of formulations $(\mathbb{P}_{\text{RLT}})$, $(\mathbb{P}_1)$ with respect to the McCormick based formulation $(\mathbb{P}_{\text{MC}})$. For each problem instance let v_{MC} denote the optimal value of the continuous relaxation of $(\mathbb{P}_{\text{MC}})$ and v^* the global optimal value. Thus the optimality gap induced by McCormick formulation is $\frac{v^* - v_{MC}}{v^*}$. Formulations $(\mathbb{P}_{\text{RLT}})$, $(\mathbb{P}_1)$ are expected to close this optimality as more valid inequalities are added. We measure the closed gap via

$$\text{Closed gap} = \frac{v - v_{MC}}{v^* - v_{MC}} \times 100 \tag{34}$$

where v represents the continuous relaxation value of $(\mathbb{P}_{\text{RLT}})$ or $(\mathbb{P}_1)$. Results are summarized in Table 1 which imply the following.

(i) The RLT inequalities close $2\% - 15\%$ of the optimality gap while the combination of the RLT and (3)-(5) closes $48\% - 100\%$ gap.

(ii) As might be expected the continuous relaxations of RLT formulation and the compact formulation ($\mathbb{P}_1$) are computationally expensive due to the addition of valid inequalities. In particular the solution time for the continuous relaxation of ($\mathbb{P}_1$) is becoming evidently more expensive than that of ($\mathbb{P}_{\text{RLT}}$) as the size of the physical network increases.

Table 1: Numerical evaluation for ($\mathbb{P}_{\text{RLT}}$), ($\mathbb{P}_1$)

		Continuous relaxation statistics				
$(\|S\|, \|E\|)$	#VMs.	($\mathbb{P}_{\text{MC}}$)	($\mathbb{P}_{\text{RLT}}$)		($\mathbb{P}_1$)	
		#Cpu(Sec.)	#CPU(Sec.)	closed gap(%)	#CPU(Sec.)	closed gap(%)
(8, 10)	5	0.00	0.01	15.2	0.12	**84.80**
(8, 10)	10	0.02	0.01	11.8	0.18	**89.21**
(12, 15)	5	0.02	0.01	7.29	0.22	**92.28**
(12, 15)	10	0.07	0.02	2.37	0.43	**82.51**
(12, 15)	15	0.13	0.22	3.34	1.02	**95.12**
(12, 15)	20	0.27	0.72	3.33	1.49	**100.00**
(12, 15)	25	0.33	0.92	6.53	2.69	**88.11**
(12, 15)	30	0.34	1.2	4.89	3.18	**80.30**
(12, 15)	35	0.49	1.33	8.32	3.64	**94.30**
(12, 15)	40	0.67	1.42	7.31	3.75	**86.83**
(12, 15)	50	0.92	1.43	9.98	4.23	**89.83**
(15, 22)	10	0.07	0.27	2.33	0.22	**89.78**
(15, 22)	15	0.13	0.48	7.03	2.34	**72.56**
(15, 22)	20	0.26	3.77	4.09	4.22	**86.04**
(15, 22)	25	0.85	2.75	4.89	5.09	**87.77**
(15, 22)	30	1.04	1.98	5.98	4.96	**90.14**
(15, 22)	35	0.63	0.72	14.53	4.57	**76.51**
(15, 22)	40	1.25	3.44	12.42	5.83	**79.43**
(22, 36)	10	0.92	1.46	5.92	4.59	**100.00**
(22, 36)	15	3.01	3.04	2.51	16.20	**69.80**
(22, 36)	20	3.39	3.55	3.58	26.87	**67.75**
(22, 36)	25	6.22	4.12	2.90	28.10	**63.64**
(22, 36)	30	8.83	3.23	3.91	29.91	**69.12**
(22, 36)	35	8.91	3.34	3.34	33.81	**53.73**

5.4 The Lagrange lower bounds

Section 4.1.2 shows that the proposed Lagrange decomposition scheme (25) and its generalized hierarchy can close further the McCormick relaxation gap. We illustrate this point numerically with following settings.

1. For a given problem instance, we first partition the virtual request set R to a number of subsets. Each subset has at most δ requests. For instance if $|R| = 7$ and $\delta = 2$, set R is partition to 4 subsets $\{\{1,2\},\{3,4\},\{5,6\},\{7\}\}$.

2. We perform numerical experiments with $\delta \in \{1,2,3\}$ and solve each subproblem with CPLEX 12.7.

3. For instances with network $(22, 36)$ it is time consuming for solving problems associated with $\delta = 3$. For this reason, we skip them.

The corresponding results are summarized in Table 2 and they may indicate the following.

1. The Lagrange lower bound is generally stronger than the continuous relaxation bound of ($\mathbb{P}_1$); in particular, it can close the optimality gap completely for certain instances in a reasonable time. For most problem instances, the proposed Lagrange lower bounding procedure closes 80% McCormick optimality gap.

2. For a given instance, its Lagrange lower bound generally increases as δ increases. Correspondingly the computational time increases.

3. There exist a couple of instances where the Lagrange lower bound with $\delta = 1$ is equal to the continuous relaxations bound (e.g. network (15, 22) with 30VMs). For such instances, the computational cost of Lagrange lower bounding procedure is usually quit small. This is probably due to the fact that the formulation of each subproblem (Sub_r) is strong.

5.5 The evaluation of Algorithm 2

In this section, we evaluate the performance of Algorithm 2 in comparison with model ($\mathbb{P}_{\text{MC}}$) and model ($\mathbb{P}_1$) in terms of solution time and B&B nodes. For this purpose we setup numerical experiments as follows.

1. For each problem instance we try to solve it to global optimality within a time limit of 10 hours with formulation ($\mathbb{P}_{\text{MC}}$) and ($\mathbb{P}_1$). Both CPU time and the number of B&B nodes of CPLEX 12.7 are recorded. We also implement branching priority strategy over $\boldsymbol{\theta}, \boldsymbol{\phi}$ using a `BranchCallback` for ($\mathbb{P}_1$).

2. For each problem instance we solve it to global optimality with Algorithm 2. Lagrange cuts are generated using the single request based decomposition (25). The upper bound tolerance parameter δ is set as 5% and the optimality tolerance is 0.5%.

3. The heuristic algorithm 1 is used in Algorithm 2 for the generation of upper bounds. Its input parameter n is set to $\lfloor R/2 \rfloor$.

Table 2: Numerical evaluation of the Lagrange lower bounds

$(\|S\|, \|E\|)$	#VMs.	$\delta = 1$		$\delta = 2$		$\delta = 3$	
		#CPU(s)	Closed gap(%)	#CPU(s)	Closed gap(%)	#CPU(s)	Closed gap(%)
(8, 10)	10	0.91	**100.00**	-	-	-	-
(12,15)	10	8.34	**100.00**	-	-	-	-
(12,15)	15	5.23	**100.00**	-	-	-	-
(12,15)	20	6.87	**100.00**	-	-	-	-
(12,15)	25	15.78	**89.68**	16.31	**91.98**	50.23	**92.12**
(12,15)	30	12.31	**88.78**	25.20	**88.87**	68.38	**89.34**
(12,15)	35	12.56	**94.67**	28.93	**95.03**	35.45	**95.03**
(12,15)	40	27.56	**82.34**	38.03	**83.56**	63.34	**83.98**
(12,15)	50	37.22	**90.12**	43.92	**90.12**	93.40	**90.24**
(15, 22)	10	7.64	**93.55**	30.10	**100.00**	-	-
(15, 22)	15	6.21	**76.37**	13.34	**76.67**	-	-
(15, 22)	20	13.22	**89.56**	28.90	**90.43**	90.31	**92.77**
(15, 22)	25	17.88	**91.40**	33.04	**91.81**	50.23	**96.36**
(15, 22)	30	4.11	**90.14**	4.18	**90.14**	33.21	**93.13**
(15, 22)	35	6.09	**79.01**	23.94	**80.32**	24.82	**80.33**
(15, 22)	40	23.34	**81.34**	24.43	**81.34**	50.32	**82.10**
(22, 36)	10	4.31	**100.00**	-	-	-	-
(22, 36)	15	62.31	**75.12**	72.12	**81.60**	-	-
(22, 36)	20	90.31	**74.40**	139.76	**75.45**	-	-
(22, 36)	25	95	**81.36**	139.12	**83.74**	-	-
(22, 36)	30	150	**70.12**	223	**73.20**	-	-
(22, 36)	35	158.18	**60.56**	413.93	**64.32**	-	-

4. As highlighted before our B&B algorithm uses a heuristic for generating upper bounds whose quality might influence the overall computational time. In order to facilitate a fair comparison we solve each problem instance 10 times and take the respective averages of CPU time and B&B nodes as measurement.

Numerical results are summarized in Table 3 and we make some comments below.

1. For problem instances with 10 VMs, all three approaches can solve the problem to global optimality. The compact formulation ($\mathbb{P}_1$) performs the best in terms of computational efficiency.

2. For problem instances with more than 30 requests, the McCormick formulation ($\mathbb{P}_{\text{MC}}$) is the most time-consuming approach while Algorithm 2 is computationally most efficient. For some small instances (e.g. (12, 15) 20 VMs), formulation ($\mathbb{P}_1$) performs better than Algorithm 2. This is probably due to the fact that CPLEX sometimes finds better upper bound than Algorithm 1.

3. For problem instances having more than 40 VMs, CPLEX cannot solve the problem to optimality within the 10-hour time limit even using the compact model ($\mathbb{P}_1$). In contrast Algorithm 2 provides optimal solutions much faster. For most problem instances, Algorithm 2 is at least 10 times and sometimes 30 times more efficient than the CPLEX default branch and bound algorithm with Model ($\mathbb{P}_1$).

4. Given a set of VMs, formulation ($\mathbb{P}_1$) and Algorithm 2 become less advantageous in terms of computational efficiency as the size of the physical network increases. This is largely due to two reasons. First our subproblem (Sub_r) is not separable in terms of physical network components thus making the lower bounding procedure computationally more expensive. Second Algorithm 1 becomes less effective in finding good upper bounds leading to more branch nodes.

5. The current implementation of Algorithm 2 fails to solve problems instances with more than 700 variables due to memory issues. Numerically we observed that the implementation incurs memory issues when the number of branch nodes is over 1000.

6 Conclusion

This work proposes a couple of mathematical programming based algorithms for the optimal mapping of virtual machines while taking into account the bilinear bandwidth constraints and other knapsack constraints regarding CPU and memory. The first one is a compact model involving RLT inequalities and some strong valid inequalities exploiting the problem structure. The second one is a Lagrange decomposition based B&B algorithm for solving larger problem instances. We show both theoretically and numerically that the proposed valid inequalities and bounding procedures can improve the continuous relaxation bounds significantly. We also demonstrate that the proposed B&B algorithm is numerically encouraging.

Based on the results presented in this paper, several research directions can be considered. First, decomposition strategies exploiting the structure of the physical network should be investigated to accelerate the solution procedure of problem instances associated with a large physical network. Second, specialized branch-and-cut algorithm incorporating the Lagrange lower bounding procedure and some recent findings of the convex and concave estimators in [2] can be devised. Third, some approximation algorithm might be devised to achieve guaranteed feasible solutions of high quality.

References

[1] Amaldi, E., Coniglio, S., Koster, A.M., Tieves, M.: On the computational complexity of the virtual network embedding problem. Electronic Notes in Discrete Mathematics **52**, 213 – 220 (2016). DOI http://dx.doi.org/10.1016/j.endm.2016.03.028. URL http://www.sciencedirect.com/science/article/pii/S1571065316300336. INOC 2015 7th International Network Optimization Conference

[2] Ben-Ameur, W., Ouorou, A., Wang, G.: Convex and concave envelopes: Revisited and new perspectives. Operations Research Letters **45**(5), 421–426 (2017)

Table 3: The evaluation of Algorithm 2

$(\|S\|, \|E\|)$	#VMs.	#Bin.	($\mathbb{P}_{MC}$)		($\mathbb{P}_1$)		Alg. 2	
			CPU	Nodes	CPU	Nodes	CPU	Nodes
(8, 10)	5	68	0.86	0	**0.29**	0	0.58	0
(8, 10)	10	108	29.1	229	1.46	3	**0.91**	0
(12, 15)	5	102	1.63	16	**0.95**	0	1.21	0
(12, 15)	10	162	123.31	8897	**4.31**	0	8.87	12
(12, 15)	15	222	321.32	9873	56.27	56	**22.53**	22
(12, 15)	20	282	1035.32	10766	**1.85**	0	3.21	2
(12, 15)	25	342	32492.00	10645	270.32	240	**20.54**	6
(12, 15)	30	402	-	-	358.89	197	**59.50**	32
(12, 15)	35	447	-	-	7684.81	6534	**169.00**	29
(12, 15)	40	507	-	-	-	-	**534.22**	138
(12, 15)	50	627	-	-	-	-	**1340.18**	82
(15, 22)	10	187	3781	2383	**21.39**	20	24.01	3
(15, 22)	15	262	-	-	2366.72	6414	**123.01**	12
(15, 22)	20	337	-	-	1702.9	2264	**239.22**	49
(15, 22)	25	412	-	-	11400.8	12011	**304.44**	46
(15, 22)	30	487	-	-	12103.3	2750	**406.92**	24
(15, 22)	35	562	-	-	-	-	**731.12**	134
(15, 22)	40	637	-	-	-	-	**2649.20**	316
(15, 22)	50	787	-	-	-	-	-	-
(22, 36)	10	228	10982	9948	**4.43**	0	5.21	1
(22, 36)	15	388	-	-	208.23	42	**98.21**	19
(22, 36)	20	498	-	-	-	-	**763.33**	394
(22, 36)	25	608	-	-	-	-	**3381.22**	498
(22, 36)	30	718	-	-	-	-	**8382.12**	913
(22, 36)	40	938	-	-	-	-	-	-

[3] Bonnans, J.F., Gilbert, J.C., Lemaréchal, C., Sagastizábal, C.A.: Numerical optimization: theoretical and practical aspects. Springer Science & Business Media (2006)

[4] Burer, S., Letchford, A.N.: Non-convex mixed-integer nonlinear programming: A survey. Surveys in Operations Research and Management Science **17**(2), 97 – 106 (2012). DOI http://dx.doi.org/10.1016/j.sorms.2012.08.001

[5] Burer, S., Saxena, A.: The milp road to miqcp. In: Mixed Integer Nonlinear Programming,

pp. 373–405. Springer (2012)

[6] Coniglio, S., Koster, A., Tieves, M.: Data uncertainty in virtual network embedding: robust optimization and protection levels. Journal of Network and Systems Management **24**(3), 681–710 (2016)

[7] Fischer, A., Botero, J.F., Beck, M.T., De Meer, H., Hesselbach, X.: Virtual network embedding: A survey. IEEE Communications Surveys & Tutorials **15**(4), 1888–1906 (2013)

[8] Fischetti, M., Lodi, A.: Local branching. Mathematical programming **98**(1-3), 23–47 (2003)

[9] Fukunaga, T., Hirahara, S., Yoshikawa, H.: Virtual machine placement for minimizing connection cost in data center networks. Discrete Optimization **26**, 183–198 (2017)

[10] Geoffrion, A.M.: Lagrangean relaxation for integer programming. Mathematical programming Study pp. 82–114 (1974)

[11] Guignard, M., Kim, S.: Lagrangean decomposition: A model yielding stronger lagrangean bounds. Mathematical programming **39**(2), 215–228 (1987)

[12] Houidi, I., Louati, W., Ben Ameur, W., Zeghlache, D.: Virtual network provisioning across multiple substrate networks. Comput. Netw. **55**(4), 1011–1023 (2011). DOI 10.1016/j.comnet.2010.12.011. URL http://dx.doi.org/10.1016/j.comnet.2010.12.011

[13] Houidi, I., Louati, W., Zeghlache, D.: Exact multi-objective virtual network embedding in cloud environments. The Computer Journal **58**(3), 403–415 (2015)

[14] Karve, A., Kimbrel, T., Pacifici, G., Spreitzer, M., Steinder, M., Sviridenko, M., Tantawi, A.: Dynamic placement for clustered web applications. In: Proceedings of the 15th International Conference on World Wide Web, WWW '06, pp. 595–604. ACM, New York, NY, USA (2006). DOI 10.1145/1135777.1135865

[15] Lemaréchal, C., Renaud, A.: A geometric study of duality gaps, with applications. Mathematical Programming **90**(3), 399–427 (2001)

[16] Lemarchal, C.: Lagrangian relaxation. In: M. Jnger, D. Naddef (eds.) Computational Combinatorial Optimization, *Lecture Notes in Computer Science*, vol. 2241, pp. 112–156. Springer Berlin Heidelberg (2001). DOI 10.1007/3-540-45586-8_4

[17] McCormick, G.: Computability of global solutions to factorable nonconvex programs: Part i-convex underestimating problems. Mathematical Programming **10**(1), 147–175 (1976). DOI 10.1007/BF01580665. URL http://dx.doi.org/10.1007/BF01580665

[18] Mechtri, M.: Virtual networked infrastructure provisioning in distributed cloud environments. Ph.D. thesis, Institut National des Télécommunications (2014)

[19] Meng, X., Pappas, V., Zhang, L.: Improving the scalability of data center networks with traffic-aware virtual machine placement. In: INFOCOM, 2010 Proceedings IEEE, pp. 1–9. IEEE (2010)

[20] Mijumbi, R., Serrat, J., Gorricho, J.L., Bouten, N., De Turck, F., Boutaba, R.: Network function virtualization: State-of-the-art and research challenges. IEEE Communications Surveys & Tutorials **18**(1), 236–262 (2016)

[21] Murat Afsar, H., Artigues, C., Bourreau, E., Kedad-Sidhoum, S.: Machine reassignment problem: the roadef/euro challenge 2012. Annals of Operations Research **242**(1), 1–17 (2016). DOI 10.1007/s10479-016-2203-7. URL http://dx.doi.org/10.1007/s10479-016-2203-7

[22] Orlowski, S., Pióro, M., Tomaszewski, A., Wessäly, R.: SNDlib 1.0–Survivable Network Design Library. In: Proceedings of the 3rd International Network Optimization Conference (INOC 2007), Spa, Belgium (2007). URL http://www.zib.de/orlowski/Paper/OrlowskiPioroTomaszewskiWessaely2007-SNDlib-INOC.pdf.gz. Http://sndlib.zib.de, extended version accepted in Networks, 2009.

[23] Papagianni, C., Leivadeas, A., Papavassiliou, S., Maglaris, V., Cervello-Pastor, C., Monje, A.: On the optimal allocation of virtual resources in cloud computing networks. Computers, IEEE Transactions on **62**(6), 1060–1071 (2013). DOI 10.1109/TC.2013.31

[24] Raghavan, P., Tompson, C.: Randomized rounding: A technique for provably good algorithms and algorithmic proofs. Combinatorica **7**(4), 365–374 (1987). DOI 10.1007/BF02579324

[25] Sherali, H., Adams, W.: A hierarchy of relaxations between the continuous and convex hull representations for zero-one programming problems. SIAM Journal on Discrete Mathematics **3**(3), 411–430 (1990). DOI 10.1137/0403036. URL http://epubs.siam.org/doi/abs/10.1137/0403036

[26] Wang, G.: Relaxations in mixed-integer quadratically constrained programming and robust programming. Ph.D. thesis, Evry, Institut national des télécommunications (2016)

[27] Wang, G., Ben-Ameur, W., Neto, J., Ouorou, A.: Optimal mapping of cloud virtual machines. Electronic Notes in Discrete Mathematics **52**, 93 – 100 (2016). INOC 2015 7th International Network Optimization Conference