Alocação de Infraestruturas Virtuais em *data centers* implementados com Redes Definidas por *Software*

Felipe Rodrigo de Souza, Charles C. Miers, Adriano Fiorese, Guilherme Koslovski

[1]Programa de Pós-Graduação em Computação Aplicada
Universidade Estadual de Santa Catarina (UDESC) – Joinville/SC – Brasil
dcc6frs@joinville.udesc.br, {charles.miers, adriano.fiorese, guilherme.koslovski}@udesc.br

Resumo. *A virtualização de data centers combinada com o paradigma de Software-Defined Networking (SDN) impulsionaram a criação de serviços nas nuvens computacionais. Dentre as atividades gerenciais de um provedor de nuvem baseado em SDN, o presente trabalho aborda a alocação de infraestruturas virtuais com requisitos de Quality-of-Service (QoS). Inicialmente, o problema é modelado como programação inteira mista. Depois, restrições em variáveis são relaxadas e uma heurística para aproximação dos resultados é discutida. Além de alocar a qualidade de serviço solicitada, a análise experimental indica a redução da latência de comunicação observada por clientes.*

Abstract. *The virtualization of data centers combined with the SDN paradigm lead to the provisioning of new cloud services. Among the management challenges faced by a SDN-based cloud provider, this work improves the virtual infrastructure allocation problem with QoS requirements. First, the problem is modeled as a mixed integer programing. Thus, integer constraints are relaxed and rounding heuristics are discussed. Besides guaranteeing the QoS requirements, the results indicate a reduction on virtual network communication latency.*

1. Introdução

Com a computação em nuvem, os clientes tem acesso a recursos computacionais fornecidos pelos provedores, de maneira dinâmica e de acordo com a demanda de suas aplicações [Mell and Grance 2011]. Provedores de *Infrastructure-as-a-Service (IaaS)* tem utilizado virtualização para provisionar Infraestruturas Virtuais (IVs), que são compostas por Máquinas Virtuais (MVs) em conjunto com serviços de *Network-as-a-Service (NaaS)* [Fischer et al. 2013, Manvi and Shyam 2014]. Dentre os desafios enfrentados por provedores IaaS, a alocação de IVs consiste em identificar dentro do *data center* um conjunto de recursos para hospedar os componentes da IV. A alocação de IVs pode ser vista como um mapeamento de grafos. Os vértices representam os equipamentos (processamento e encaminhamento) e as arestas representam os enlaces de comunicação. Assim, o problema consiste em mapear o grafo que representa a IV no grafo que representa o *data center*, caracterizado como um problema NP-difícil [Fischer et al. 2013].

A complexidade do processo de alocação tende a aumentar, visto que são criados cada vez mais critérios para garantir a qualidade da IV provisionada. Por exemplo, aplicações em IVs geram grandes volumes de tráfego e boa parte do tempo de execução é devido a transferência de dados. Um *cluster* do Facebook pode consumir até 33% de seu tempo de execução somente realizando transferências [Rost et al. 2015]. É crescente

o número de aplicações na nuvem que solicitam requisitos de rede, sendo um mecanismo de alocação focado somente em MVs uma visão simplista.

Recentemente, *Software-Defined Networking (SDN)* surgiu como uma alternativa para simplificar o provisionamento de IVs. SDN separa a parte de controle, normalmente embutida nos equipamentos de rede, para um controlador externo e logicamente centralizado. Além disso, fornece mecanismos para isolamento de IVs e garantia de *Quality-of-Service (QoS)* [Sherwood et al. 2009]. Apesar de SDN facilitar tanto o processo de provisionamento de IVs quanto a representação dos recursos de encaminhamento, algumas características exclusivas do ambiente introduzem outros níveis de complexidade. Em SDN, a garantia de largura de banda é realizada diretamente nos *switches*. Portanto, todos os equipamentos para alocação de um enlace virtual devem conter regras para respeitar os limites preestabelecidos. Ainda, as entradas nas tabelas de encaminhamento dos *switches* são limitadas [Kreutz et al. 2015], o que leva ao armazenamento temporário de entradas no controlador. Assim, a eventual latência até o controlador deve ser considerada no cálculo para verificação da conformidade com os requisitos de QoS do cliente.

A contribuição do presente trabalho é a proposta de um mecanismo intitulado QVIA-SDN (*Quality-Aware Virtual Infrastructure Allocation SDN*), que é composto por um modelo para alocação de IVs em um *data center* de nuvem baseado em SDN, com *switches* OpenFlow. O problema é modelado como programação inteira mista, depois as restrições em variáveis são relaxadas e uma heurística para aproximação dos resultados é discutida. Os resultados experimentais indicam que além de respeitar o QoS, há uma redução na latência média percebida por clientes de IVs.

Este trabalho está organizado da seguinte forma. A Seção 2 contextualiza a proposta frente aos trabalhos relacionados. A Seção 3 apresenta a formulação do problema, enquanto a Seção 4 descreve um *Mixed Integer Program (MIP)* ótimo para alocação de IVs. O relaxamento das variáveis e as heurísticas são abordados na Seção 5. A análise experimental é discutida na Seção 6 e as considerações finais são apresentadas na Seção 7.

2. Trabalhos relacionados

A literatura define que o provisionamento de IVs em *data centers* de *IaaS* deve ser guiado por políticas, que podem ser aplicadas com foco local (por enlace físico ou caminho congestionado) ou global (considerando a topologia do *data center*) [Ballani et al. 2011]. No trabalho de [Popa et al. 2011] foi proposto o compartilhamento proporcional dos recursos de rede, e isso só pode ser alcançado em um cenário controlado, como SDN, visto que requer coordenação entre o controlador e os recursos virtuais. As políticas são baseadas em parâmetros que configuram o compartilhamento dos caminhos físicos que estão alocando enlaces virtuais, alinhado com a nossa proposta: parâmetros indicados pelos clientes (largura de banda e latência) definem os requisitos para configuração dos enlaces.

Quanto a seleção de recursos físicos para alocar IVs, é possível encontrar propostas de formulação ótima e heurísticas de aproximação [de Oliveira and Koslovski 2017, Cavalcanti et al. 2014, Fischer et al. 2013]. Em geral, cada proposta é guiada por um objetivo diferente, com destaque para [Chowdhury et al. 2012], que inovou ao propor a alocação conjunta de recursos de processamento e comunicação (discutido na Seção 4).

A alocação de IVs em *data center* SDN segue um desafio para provedores de nuvem IaaS. [Sherwood et al. 2009] identificou os principais desafios na alocação de re-

cursos SDN, destacando as tabelas de fluxo e o compartilhamento de controladores entre as IVs alocadas. Além disso, propôs um mecanismo para provisionar redes virtuais semelhante aos *hypervisores* de MVs. De maneira complementar, [Al-Shabibi et al. 2014] propôs um *framework* para virtualização de redes utilizando SDN. O presente trabalho propõe um algoritmo de alocação baseado em QoS (Seção 5) que considera os desafios e particularidades do processo de alocação [Sherwood et al. 2009, Al-Shabibi et al. 2014].

[Demirci and Ammar 2014] identificou que o posicionamento do controlador tem impacto no desempenho da aplicação, devido a latência até o controlador. Propostas de garantia de largura de banda em ambientes SDN estão presentes em [Tao et al. 2015, Mijumbi et al. 2014]. No presente trabalho, largura de banda e latência são impostas através da formulação MIP relativa ao ambiente SDN. Ainda, o MIP proposto na Seção 4 considera que os *switches* podem ser reservados e gerenciados pelos clientes.

É importante ressaltar que o presente trabalho inova na alocação de IVs com requisitos de QoS em um *data center* de nuvem baseado em SDN, considerando os desafios indicados na literatura. Além disso, o problema de alocação é formulado como uma alocação conjunta de MVs, *switches* e enlaces, alegando que o desempenho das aplicações hospedadas na nuvem pode ser impactado por políticas de alocação de rede.

3. Formulação do problema

3.1. *Data center* IaaS e requisições de IVs

Tanto a infraestrutura de *data center* quanto as requisições de IVs são representadas por grafos não direcionados com pesos. O *data center* é representado pelo grafo $G^s(N_h^s, N_n^s, C^s, E^s)$, sendo que N_h^s representa o conjunto de servidores e N_n^s o conjunto de *switches* OpenFlow que compõem a infraestrutura física. A notação utilizada ao longo do trabalho é resumida na Tabela 1.

Notação	Descrição	Notação	Descrição
$G^s(N_h^s, N_n^s, C^s, E^s)$	grafo do *data center*	$G^v(N_h^v, N_n^v, E^v, D^v)$	Requisição de IV
N_h^s	servidores físicos	N_h^v	MVs
N_n^s	*swiches* SDN	N_n^v	*switches* virtuais
C^s	controlador SDN	E^v	enlaces virtuais
E^s	enlaces físicos	D^v	matriz de latência máxima aceita
$R(.)$	capacidade física residual	d_{ij}	latência máxima entre i e j
$\Omega(i)$	candidatos físicos para hospedar i	c_i	capacidade solicitada para i
$P^s(i,j)$	caminhos entre $\Omega(i)$ e $\Omega(j)$		

Tabela 1. Notação utilizada ao longo do artigo. i e j representam recursos virtuais, enquanto u e v são utilizados para recursos físicos.

Os controladores SDN são representados por C^s. Os enlaces que interconectam os servidores e equipamentos de rede são representados pelo conjunto E^s. Cada recurso (servidores, equipamentos de rede ou enlaces) possui uma capacidade residual, denotada pela função $R(.)$. De maneira similar, uma requisição de IV é um grafo $G^v(N_h^v, N_n^v, E^v, D^v)$, sendo que N_h^v representa o conjunto de MVs, N_n^v o conjunto de equipamentos de rede e

E^v o conjunto de enlaces. D^v é uma matriz que contêm a latência máxima aceita fim-a-fim, na qual d_{ij} representa a maior latência aceita entre os recursos i e j. A capacidade solicitada para cada recurso virtual é representada por c. Como a configuração da rede virtual é um aspecto importante para o desempenho de aplicações hospedadas em IVs, provedores estão buscando a alocação de IVs com requisitos de QoS de rede. Neste sentido uma requisição de IV pode ser baseada em IaaS e NaaS, exemplificado na Figura 1.

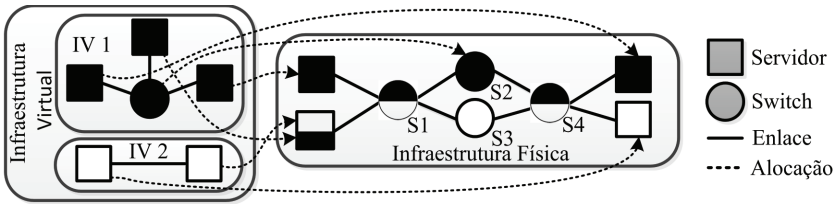


Figura 1. Alocação de IVs em provedores de nuvens baseados em SDN.

Requisições somente IaaS representam MVs solicitadas sem a descrição da topologia virtual. Entretanto, além das configurações de MVs, regiões ou zonas, alguns requisitos de rede, como por exemplo a latência máxima fim-a-fim ou uma largura de banda mínima podem ser especificados. Este tipo de requisição é representado pela IV 2 (Figura 1), na qual duas MVs devem ser provisionadas com requisitos de QoS de rede. É importante destacar que os equipamentos de rede utilizados para garantir a comunicação das MVs são abstraídos do cliente sendo de conhecimento somente do provedor.

Requisições combinando IaaS e NaaS permitem a completa descrição da IV. O cliente pode realizar uma descrição detalhada dos equipamentos de rede que compõem a rede virtual, bem como os requisitos de QoS. A IV 1 (Figura 1) exemplifica que o provedor pode utilizar mais de um recurso físico para para alocar um virtual: enquanto a IV solicitou um único *switch* virtual, o provedor utilizou 3 *switches* SDN físicos para garantir a comunicação dos equipamentos virtuais. Neste caso, o cliente só tem acesso para configuração do *switch* que efetivamente está alocando o equipamento virtual solicitado.

3.2. Alocando recursos físicos para hospedar IVs

As requisições de IVs são processadas individualmente pelo *framework* de alocação da nuvem, o qual é responsável por tomar a decisão de aceitar ou não uma IV. Os requisitos da IV podem alterar durante seu ciclo de vida, entretanto este trabalho não aborda a reconfiguração, pois acreditamos que alocação e reconfiguração devem ser analisadas individualmente. Ao fim da execução da IV, os recursos destinados a mesma são liberados.

A alocação de IVs em um *data center* de nuvem pode ser dividida em duas etapas: alocação de nós e alocação de enlaces [Chowdhury et al. 2012]. O mapeamento de MVs em servidores físicos é dada por $M_h : N_h^v \mapsto N_h^s$, enquanto para *switches* virtuais mapeados em *switches* SDN por $M_n : N_n^v \mapsto N_n^s$, sendo $M_h(i) \in N_h^s$ e $M_n(i) \in N_n^s$. De maneira similar, o mapeamento de um enlace virtual ij é realizado em um caminho $p \in P^s$, entre os nós que alocam os nós virtuais finais (*switch* ou MV) do enlace ij. Para hospedar um recurso virtual, o nó físico deve ter uma capacidade residual (não reservada) maior ou igual a capacidade virtual solicitada. Para as MVs e os *switches*, $c_i \leq R(M_h(i))$ e $c_j \leq R(M_n(j))$, respectivamente, enquanto para os enlaces virtuais, $c_{ij} \leq R(M_e(ij))$.

3.3. Objetivos do provedor de nuvem IaaS

O principal objetivo neste trabalho é a alocação de IVs com requisito de QoS de rede. Além disso, a formulação busca a redução do custo de alocação, aumentando o *revenue* do provedor. De maneira similar a outros trabalhos, o custo para alocar uma IV é proporcional a capacidade física reservada para garantir o *Service Level Agreement (SLA)*, como mostrado na Eq. (1), onde $|ij|$ representa o tamanho do caminho para alocar ij, em número de saltos [Fischer et al. 2013]. O *revenue* ao alocar uma IV é dado pela Eq. (2), visto como o somatório das capacidades requisitadas.

$$\mathcal{C}(G^v) = \sum_{i \in N_h^v} c_i + \sum_{j \in N_n^v} c_j + \sum_{ij \in E^v} c_{ij}|ij| \tag{1}$$

$$\mathcal{R}(G^v) = \sum_{i \in N_h^v} c_i + \sum_{j \in N_n^v} c_{ij} + \sum_{ij \in E^v} c_{ij} \tag{2}$$

A *Quality-of-Service (QoS)* fornecida ao cliente é medida pela latência média dos recursos virtuais comunicantes, como descrito na Eq. (3), na qual $\mathcal{L}(i,j)$ representa a latência média do caminho físico alocando ij, e $|E^v|$ indica o número de enlaces virtuais.

$$\mathcal{Q}(G^v) = \frac{\sum_{ij \in E^v} \mathcal{L}(i,j)}{|E^v|} \tag{3}$$

4. MIP ótimo para alocação de IVs com requisitos de QoS

4.1. Seleção de candidatos para alocar recursos virtuais

Em provedores públicos de IaaS, clientes podem selecionar regiões e zonas para alocação das MVs. Assim, cada recurso virtual requisita uma localização (zona ou região) representado por $loc(i)$. Qualquer recurso físico na localização $loc(i)$ é um candidato para alocar i. Candidatos para alocar o recurso virtual i são representados pelo conjunto $\Omega(i)$.

Além da localização, requisitos de QoS de rede são essenciais para inquilinos [Stallings 2015]. Sobretudo, a latência e o desempenho da rede virtual não são fatores totalmente dependentes da localização física [Persico et al. 2015]. Sendo assim, além da seleção usual baseada na localização [Mijumbi et al. 2015] [Chowdhury et al. 2012], é proposta a seleção de candidatos baseada em latência.

Candidatos para alocar *switches* virtuais com requisitos de latência. Para requisições de IVs com enlaces contendo requisitos de latência e *switches* virtuais, o conjunto de candidatos físicos para alocar o *switch* i é composto por equipamentos que possuem enlaces capazes de hospedar o pior requisito de latência de i. Em suma, $\Omega(i) = \{u \in N_n^s | max(lat(u,v)) < max(d_{ij})\}; \forall v \in adj(u); \forall j \in adj(i)$, onde $adj(.)$ retorna o conjunto de adjacentes, e $lat(u,v)$ indica a latência do caminho físico entre u e v.

Candidatos para alocar *switches* virtuais sem requisitos de latência. Neste caso, todos os *switches* físicos com capacidade residual suficiente são candidatos para alocar o *switch* virtual i. Assim, $\Omega(i) = \{u \in N_n^s | R(u) \geq c_i\}$.

Candidatos para alocar MVs sem requisitos de latência. Os servidores físicos com suficiente capacidade residual são selecionados para alocar a MV i. Em outras palavras, $\Omega(i) = \{u \in N_h^s | R(u) \geq c_i\}$.

Candidatos para alocar MVs com requisitos de latência.
(a) MVs conectadas com *switches* virtuais: neste caso, os candidatos são selecionados com base na latência de comunicação do *switch* conectado com MV i. Assim, $\Omega(i)$ é composto por $\{u \in N_h^s | lat(u,v) \leq d_{ij}\}; \forall v \in \Omega(j); j \in adj(i) \setminus N_h^v$.
(b) MVs conectadas com MVs: os candidatos, são selecionados com base nos requisitos de latência fim-a-fim. Portanto, $\Omega(i) = \{u \in N_h^s | lat(u,v) \leq d_{ij}\} \forall v \in N_h^s$.

Seguindo a abordagem proposta por [Chowdhury et al. 2012], cada recurso virtual i é inserido no grafo físico, através de um enlace temporário com seus candidatos, com capacidade infinita e sem latência. O grafo aumentado resultante é representado por $G^{s'}(N^{s'}, C^s, E^{s'})$. Assim, $N^{s'} = N_h^s \cup N_h^v \cup N_n^s \cup N_n^v$; e $E^{s'} = E^s \cup \{iu \,|\, i \in N_h^v, u \in \Omega(i)\} \cup \{ju \,|\, j \in N_n^v, u \in \Omega(j)\}$. Os controladores físicos (C^s) não são afetados pela criação do grafo aumentado. A Figura 2 exemplifica um grafo aumentado, conectando a IV 1 da Figura 1 a um *data center* simplificado.

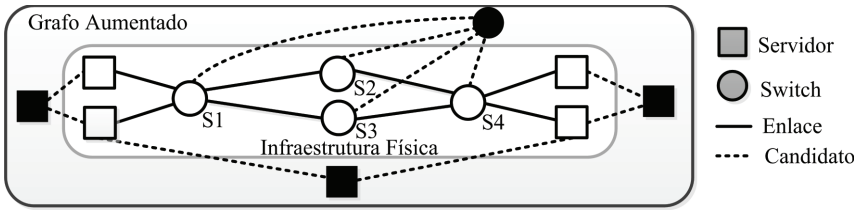


Figura 2. Grafo aumentado combinando recursos físicos e virtuais.

4.2. Variáveis e objetivo

Para representar uma solução de mapeamento de IVs com QoS de rede foram utilizadas três variáveis. A variável f_{ijuv} representa o quanto do fluxo (largura de banda solicitada para o enlace virtual) de ij está alocado no enlace físico entre u e v. Com domínio binário, a variável x_{uv} representa a presença ou não de um fluxo (requisição de enlace virtual) entre os recursos u e v. Se a condição $\sum_{ij \in E^v}(f_{ijuv} + f_{ijvu}) > 0$ for atendida o valor em x_{uv} é definido como 1, caso contrário 0. Por fim, a variável binária e_{ijpu} representa as entradas das tabelas de encaminhamento de fluxos que estão alocadas no controlador (1 em caso de ocorrência). Quando 0, o fluxo ij está alocado sobre o caminho p no *switch* u.

De acordo com os objetivos do provedor (Seção 3.3), uma versão modificada das Equações (1), (2) e (3) compõem a função objetivo (Eq. 4). A minimização da função objetivo reduz o custo para alocação de uma IV, através da redução do número de servidores e enlaces físicos utilizados, bem como pela alocação de fluxos no controlador. Como proposto na literatura, ao ponderar o custo pela capacidade residual dos recursos físicos (servidores, *switches* e enlaces), são selecionados os recursos com maior capacidade residual, balanceando a carga do *data center* [Chowdhury et al. 2012]. Os parâmetros α_{uv}, β_u, e γ_e controlam a importância de cada recurso de acordo com a visão do provedor, com valores entre 1 e $R(.)$, enquanto δ é um número positivo para evitar divisão por zero.

$$min: \sum_{u \in N_h^s \cup N_n^s} \frac{\beta_u}{R(u)+\delta} \sum_{i \in N^v} x_{iu} c_i + \sum_{u \in N_n^s} \frac{\gamma_e}{R(u)+\delta} \sum_{ij \in E^v} \sum_{p \in P^s(i,j)} (1 - e_{ijpu}) + \sum_{uv \in E^s} \frac{\alpha_{uv}}{R(uv)+\delta} \sum_{ij \in E^v} f_{ijuv} \quad (4)$$

4.3. Restrições

Para garantia dos requisitos de QoS solicitados no SLA, um conjunto de restrições de capacidade, fluxo, binárias e meta restrições deve ser atendido pelo mecanismo de alocação.

Restrições de capacidade dos servidores e enlaces. A Eq. (5) define a restrição de capacidade para os enlaces, enquanto a Eq. (6) se aplica aos servidores. Em suma, os recursos físicos para devem ter capacidade residual suficiente para alocar as requisições.

$$\sum_{ij \in E^v} (f_{ijuv} + f_{ijvu}) \leq R(uv) x_{uv} \qquad \forall u, v \in N^{s'} \tag{5}$$

$$R(u) \geq \sum_{i \in N_h^v} x_{iu} c_i \qquad \forall u \in N_h^s \tag{6}$$

Restrições de SDN. Os equipamentos com SDN possuem uma particularidade quanto as suas tabelas de encaminhamento: os fluxos que passam por um determinado *switch* (representado por x) podem estar alocados somente nas tabelas de encaminhamento do controlador (identificado por e). Neste sentido, a Eq. (7) busca garantir que a capacidade residual de um *switch* u deve ser suficiente para alocar todos os fluxos que passam por aquele *switch*, entretanto, as entradas alocadas no controlador não devem ser consideradas. s_{ij} e t_{ij} representam a origem e o destino, do enlace virtual ij.

$$\begin{aligned} R(u) \geq & \sum_{k \in N_n^v} \sum_{ij \in E^v : s_{ij} = k \vee t_{ij} = k} \sum_{p \in P^s(i,j)} (x_{ku} - e_{ijpu}) c_k \\ & + \sum_{ij \in E^v} \sum_{p \in P^s(i,j)} (x_{iu} - e_{ijpu}) \qquad \forall u \in N_n^s \end{aligned} \tag{7}$$

Restrições de fluxo de dados. As requisições de enlaces virtuais, representam um fluxo que deve ser transferido no *data center*. Assim, a Eq. (8) garante que para cada enlace virtual ij, o fluxo iniciado em s_{ij} devem ser igual a capacidade solicitada para o enlace virtual, enquanto a Eq. (9) tem significado equivalente para o fluxo chegando em t_{ij}. Complementando, a Eq. (10) é responsável pelos fluxos encaminhados no *data center* SDN: cada *switch* intermediário deve encaminhar os fluxos que recebeu em sua totalidade.

$$\sum_{u \in N^{s'}} f_{ijs_{ij}u} - \sum_{u \in N^{s'}} f_{ijus_{ij}} = c_{ij} \qquad \forall ij \in E^v \tag{8}$$

$$\sum_{u \in N^{s'}} f_{ijt_{ij}u} - \sum_{u \in N^{s'}} f_{ijut_{ij}} = -c_{ij} \qquad \forall ij \in E^v \tag{9}$$

$$\sum_{v \in N^{s'}} f_{ijuv} - \sum_{v \in N^{s'}} f_{ijvu} = 0 \qquad \forall ij \in E^v, \forall u \in N^{s'} \setminus \{s_{ij}, t_{ij}\} \tag{10}$$

Restrições de latência. As Equações (11) e (12) garantem que a latência do caminho físico alocando o enlace virtual é menor ou igual ao solicitado, mesmo quando algumas regras de encaminhamento estão no controlador.

$$d_{ij} \leq \sum_{u,v \in p} (lat(u,v) x_{uv} + lat(u,c) e_{ijpu}) \forall ij \in E^v; \forall p \in P^s(i,j) \tag{11}$$

$$d_{ij} \geq \sum_{u,v \in p} lat(u,v) x_{uv} \forall ij \in E^v; \forall p \in P^s(i,j) \tag{12}$$

Meta restrições e restrições binárias. Pela Eq. (13), um recurso virtual será alocado em somente um recurso físico, enquanto a Eq. (14) garante que x irá receber valores quando existir algum fluxo passante. Eqs. de (15) a (17) definem os domínios das variáveis.

$$\sum_{u \in \Omega(i)} x_{iu} = 1 \qquad \forall i \in N^v \tag{13}$$

$$x_{uv} = x_{vu} \qquad \forall u, v \in N^{s'} \tag{14}$$

$$f_{ijuv} \geq 0 \qquad \forall u, v \in N^{s'}; \forall ij \in E^v \tag{15}$$

$$x_{uv} \in \{0, 1\} \qquad \forall u, v \in N^{s'} \tag{16}$$

$$e_{ijpu} \in \{0, 1\} \qquad \forall u \in N_n^s; \forall ij \in E^v; \forall p \in P^s(i, j) \tag{17}$$

5. QVIA-SDN

É fato que a solução de um MIP é um problema NP-difícil [Chowdhury et al. 2012]. Diante da limitação da aplicabilidade da formulação ótima, as restrições binárias são relaxadas, compondo um *Linear Program (LP)* Em um segundo momento, o número de candidatos e caminhos físicos são reduzidos. Após a solução do LP, os resultados são tratados por uma heurística para obtenção de uma solução. QVIA-SDN (*QoS-Aware VI Allocation on SDN-based data-centers*) é composto pelas técnicas listadas.

5.1. Relaxando as variáveis

Para obter o LP, as Eqs. (16) e (17) tiveram seus domínios relaxados. Assim, respectivamente, $1 \geq x_{uv} \geq 0; \forall u, v \in N^{s'}$ e $1 \geq e_{ijpu} \geq 0; \forall u \in N_n^s; \forall ij \in E^v; \forall p \in P^s(i, j)$. A função objetivo bem como as demais restrições permanecem inalteradas no LP.

5.2. Redução do número de candidatos físicos

Os *data centes* normalmente são compostos por recursos homogêneos, interconectados por uma infraestrutura de rede estruturada [Stallings 2015]. Assim, recursos podem ser agrupados de acordo com aspectos comuns (*e.g.*, largura de banda, latência, CPU, RAM). Esta informação pode ser utilizada para composição de grupos de candidatos [Rost et al. 2015]. Nesta linha, QVIA-SDN reduz o número de candidatos físicos utilizando somente uma porcentagem de cada região do *data center*. Esta informação pode ser ajustada pelo provedor de acordo com a quantidade de recursos físicos disponíveis por região ou zona. É esperado que ao reduzir o número de candidatos, a eficiência do algoritmo seja impactada. Essa parametrização é discutida na Seção 6.

É comum que topologias de *data center* tenham caminhos redundantes [Stallings 2015, Al-Fares et al. 2008], gerando um grande volume de caminhos, que são impraticáveis de calcular a cada alocação. A contabilização de todos os caminhos entre os recursos que compõem um *data center*, a cada alocação, é impraticável. Como o LP requer um conjunto P^s para encontrar uma solução aproximada, QVIA-SDN emprega uma redução de caminhos físicos candidatos. Essa redução somente oculta a existência de caminhos, permitindo futuramente a aplicação de algoritmo de resiliência ou engenharia de tráfego. Assim, P^s é composto por um (i) caminho mais curto e (ii) um caminho com a menor latência, entre todos os candidatos físicos (servidores e *switches*), sendo (i) diferente de (ii).

5.3. Heurísticas

Ao resolver o LP relaxado, a correlação entre as variáveis f, x e e é perdida. Portanto, QVIA-SDN aplica uma heurística para avaliar os valores obtidos e aproximar a solução. A heurística é composta por duas etapas, como descrito pelos Algoritmos 1 e 2.

Input: G^v, G^s
Output: M_n, M_e
1 Cria o grafo aumentado $G^{s'}$
2 Resolve QVIA-SDN com variáveis relaxadas
3 **for** $k \in N^v$ **do**
4 **for** $z \in \Omega(k)$ **do**
5 $p_z = \alpha(\sum_{ij \in E^v} f_{ijkz} + f_{ijzk}) + (1-\alpha)x_{kz}$
6 **end**
7 $z_{max} = \text{argmax}\{p_z | z \in \Omega(k)\}$
8 **if** $z_{max} = \emptyset$ **then**
9 Rejeita G^v
10 **end**
11 $M_n(n) \leftarrow z_{max}$
12 **end**
13 **if** *BDC*$(G^v, G^{s'}, M_n, M_e)$ **then**
14 Atualizada capacidades residuáis dos recursos físicos
15 Return M_n, M_e
16 **else**
17 Rejeita G^v
18 **end**

Algorithm 1: QVIA-SDN baseado em [Chowdhury et al. 2012].

Input: $G^v, G^{s'}, M_n, M_e$
Output: Verdairo ou falso e os caminhos para M_e
1 **for** $ij \in E^v$ **do**
2 **if** $M_n(i) == M_n(j)$ **then**
3 **continue**
4 **end**
5 **for** $p \in P^s(i,j)$ **do**
6 $path \leftarrow \{\}$
7 $lat\_path \leftarrow 0$
8 **for** $u \in p$ **do**
9 **if** $e_{ijpu} > 0$ **then**
10 $lat\_path \leftarrow lat\_path + lat(u,c)$
11 $path \leftarrow path + u + controller(u)$
12 **else**
13 $path \leftarrow path + u$
14 **end**
15 **end**
16 **if** $R(path) \geq c_{ij} \wedge lat\_path \leq d_{ij}$ **then**
17 Set $M_e(ij) \leftarrow path$
18 **break**
19 **else**
20 $path = resolveMochila(path, d_{ij})$
21 **if** $path$ **then**
22 $M_e(ij) \leftarrow path$
23 **break**
24 **else**
25 Rejeita G^v
26 **end**
27 **end**
28 **end**
29 **end**

Algorithm 2: Busca Determinística de Caminhos.

No Alg. 1, baseado em [Chowdhury et al. 2012], é criado o grafo aumentado conectando os recursos virtuais a seus candidatos, de acordo com as definições da Seção 4.1. Então o LP é resolvido (linhas 1 e 2). Posteriormente, apenas um candidato adequado para alocar o recurso virtual é identificado. Para tal, p_z é calculado para os candidatos de k (linhas 3 a 11), dado pela soma ponderada de x_{kz} e o total de fluxo passando por kz em ambas as direções. Esta abordagem reconstrói a correlação entre f e x, identificando o mapeamento de vértices a arestas. O peso α indica a preferência (rede ou nós) durante esta etapa. O candidato com o maior p_z é selecionado para alocar o recurso virtual. Quando nenhum candidato é identificado, a alocação da IV é rejeitada (linhas 8 e 9).

Depois de identificar um mapeamento para MVs e *switches*, os caminhos físicos para alocar os enlaces virtuais são selecionados, considerando as particularidades do ambiente SDN (controladores, *switches* e tabelas de encaminhamento). A Busca Determinística de Caminhos (BDC) é guiada pela variável e, bem como por requisitos de QoS (latência e largura de banda) para identificar se um caminho físico é capaz de alocar um enlace virtual ij (Alg. 2). Além disso, busca reduzir a utilização dos *switches* através da

alocação de entradas de encaminhamento no controlador, quando possível. Quando dois recursos virtuais comunicantes são alocados no mesmo recurso físico, é assumido que o último possui capacidade suficiente para respeitar as restrições de largura de banda e latência (linhas 2 a 4). Todos os caminhos candidatos pré-selecionados para alocar um enlace virtual ij (linha 5) são analisados considerando a presença ou não de entradas na tabela de encaminhamento do controlador (linhas 8 a 14). Se existe algum valor em e, QVIA-SDN contabiliza a latência até o controlador ao invés de considerar a alocação na tabela de fluxo do *switch* correspondente.

Caminhos físicos com latência superior a requisitada são descartados. Ideia similar é aplicada em relação a largura de banda, ignorando aqueles caminhos que não atendem os requisitos mínimos. Entretanto, em alguns casos, o caminho pode ser reconfigurado para atender os requisitos de latência. Neste caso, resolvendo o problema da mochila, QVIA-SDN verifica se existe uma combinação de *switches* que pode alocar o enlace virtual respeitando os requisitos de QoS. Dentre os caminhos, são selecionados aqueles com o menor número de saltos e maior número de entradas de fluxo alocadas no controlador (reduzindo a carga do *switches*). Por fim, na ausência de caminhos físicos para alocar um enlace virtual, a IV é rejeitada.

6. Análise experimental

Os experimentos quantificam a utilização do *data center* na perspectiva do provedor, bem como a QoS percebida pelo cliente (latência) além da configuração da IV requisitada.

6.1. Métricas

Para representar os objetivos do provedor da nuvem (Seção 3.3), cinco métricas foram coletadas. (i) *Razão custo-revenue* fornece uma visão dos lucros do provedor ao alocar uma IV. (ii) *Fragmentação* do *data center* indica a porcentagem de recursos ativos, calculada através da divisão do número de recursos ativos pelo número total de recursos disponíveis. (iii) O *tempo médio de alocação* de uma requisição. (iv) A *taxa de aceitação* de requisições. (v) A *latência média* de uma IV, calculada pela soma das latências dos caminhos físicos que hospedam os enlaces virtuais, dividia pelo número de enlaces da IV.

6.2. Cenários de simulação

QVIA-SDN e um simulador de eventos discretos foram implementados em Java v1.8, utilizando CPLEX (v12.6.1.0) para resolver o LP. Os experimentos foram executados em um Intel Xeon E5-2620 2.0GHz - 24 núcleos, 256GB (DDR3) de RAM e 2TB. Para avaliar a aplicabilidade de QVIA-SDN em cenários de nuvem, a topologia *fat-tree* foi selecionada para o *data center*, enquanto para as IVs duas topologias comumente utilizadas são discutidas: multi-camadas e *Virtual Private Clouds (VPC)*.

Topologia *fat-tree* [Al-Fares et al. 2008]. Uma *fat-tree* é baseada em *switches* com k-portas. Assim, k *pods*, cada um contendo duas camadas de $k/2$ *switches*, são compostos, comportando até $k^3/4$ servidores. Neste trabalho, são consideradas duas configurações, $k = 4$ e $k = 8$. A capacidade dos servidores foi calculada atribuindo pesos a cada um dos recursos (CPU, RAM e armazenamento). Os pesos foram definidos empiricamente e representam a importância de cada recurso do processo de alocação de IVs. Assim, a CPU tem um peso de 50%, a RAM e o armazenamento 25% cada.

Tomando como base o servidor no qual a análise experimental foi executada, a capacidade ponderada dos servidores do *data center* é igual a 576. Quanto aos *switches* físicos (núcleo, agregação e aresta), a capacidade foi definida como 100 entradas nas tabelas de encaminhamento. A largura de banda entre os *switches* de núcleo e os *pods* é definida como 10Gbps, e 1Gbps para enlaces dentro dos *pods*. A latência entre qualquer par de recursos físicos é definida como 1ms, enquanto a latência entre os *switches* de núcleo e o controlador SDN é 2ms. Os *switches* de núcleo são diretamente conectados com o controlador SDN, enquanto os outros *switches* estão conectados por caminhos lógicos. A organização hierárquica das regiões e zonas na *fat-tree* é definida da seguinte maneira: cada *pod* representa uma zona, e cada par de *pods* representa uma região.

Requisições de IVs multi-camadas (NC). Diversos inquilinos organizam suas IVs em multi-camadas [Jennings and Stadler 2014], incluindo um balanceador de carga, encarregado de distribuir as requisições para um conjunto de servidores, que eventualmente consultam um banco de dados. Para simulação, cada camada possui um *switch* virtual que solicita QoS de rede, além das MVs. A distribuição de MVs ocorre da seguinte forma: um balanceador de carga, quatro servidores e quatro bancos de dados.

Requisições de IVs VPC. Amazon EC2 introduziu o provisionamento de VPCs[1], que é composta por um conjunto de regras de acesso e um conjunto de MVs, compondo uma rede privada, gerenciada pelo cliente. Para compor as requisições VPC, um conjunto de 9 MVs é conectado a um *switch* SDN, representando as regras de acesso.

As capacidades dos recursos virtuais segue as instâncias M3 da Amazon EC2[2], nas configurações *medium*, *large*, *xlarge* e *2xlarge*. A capacidade das MVs é calculada utilizando os mesmo pesos aplicados para o *data center*, obtendo os valores 2, 10, 25 e 51 para cada configuração, respectivamente. Já a capacidade dos enlaces virtuais segue uma distribuição normalizada entre 10, 20, 40 e 80% das larguras de bandas média observadas por cada instância [Persico et al. 2015]: 492, 739, 958 e 1188 Mbps, respectivamente. Quanto a localização geográfica, todas as IVs recebem uma região, porém existe só 50% de chances de uma IV especificar uma zona. Um conjunto com 50 requisições (VPC ou multicamadas) igualmente distribuído entre os quatro tipo de instâncias M3 é submetido para cada cenário físico. O tempo de chegada de cada IV é uniformemente distribuído entre 100 intervalos discretos, com tempo ativo máximo de 30 intervalos. A latência atual, bem como a capacidade residual dos recursos são coletadas a cada requisição submetida.

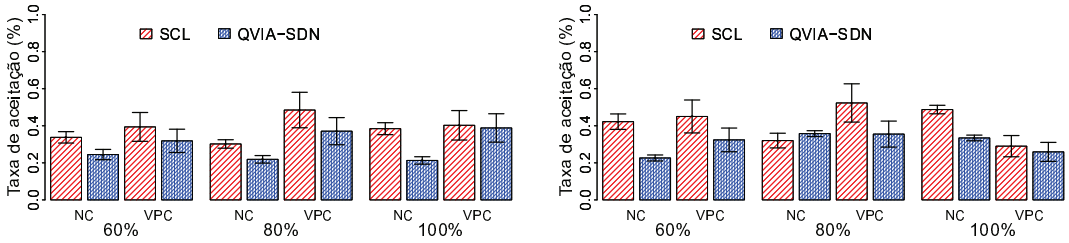
6.3. Resultados da simulação

Os resultados mostram médias com 95% de intervalo de confiança. QVIA-SDN é comparado com um algoritmo base sem controle de latência (SCL) baseado na proposta de [Chowdhury et al. 2012]. Baseado em observações empíricas, $\alpha = 0.9$ (Seção 5.3) e $\beta_u = \gamma_e = \alpha_{uv} = 1$ (Seção 4.2). Dois cenários são definidos variando o tipo de IV.

Taxa de aceitação. Os resultados para $k = 4$ e $k = 8$ são apresentados na Figura 3. A Figura 3(a) indica que diante de poucos recursos é possível alocar um número maior de VPCs. Para o cenário com $k = 8$ houve um equivalência entre os dois tipos de IVs. Sobretudo, QVIA-SDN e SCL possuem taxas de aceitação equivalentes, porém QVIA-SDN fornece uma melhora de QoS na perspectiva do inquilino.

[1]Virtual Private Cloud (VPC): `https://aws.amazon.com/vpc/`.

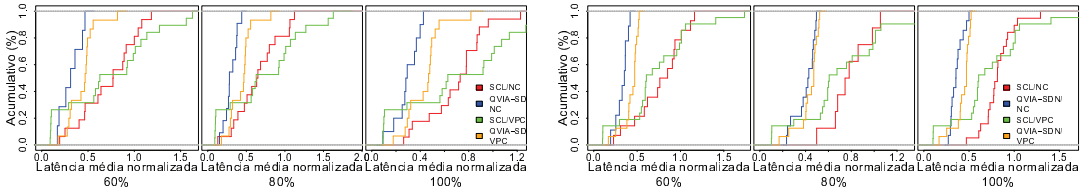
[2]Instâncias M3 - Amazon EC2: `https://aws.amazon.com/pt/ec2/instance-types/`



(a) $k = 4$. (b) $k = 8$.

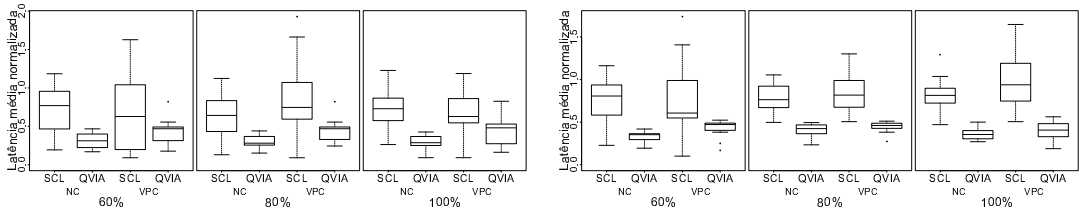
Figura 3. Taxa de aceitação.

Latência média. A Figura 4 mostra a distribuição acumulada da latência média normalizada. A latência experimentada nas IVs de NC foi inferior aquela presente nas IVs de VPC, ou seja, requisições NCs ocuparam mais entradas nos *switches*, justificando uma menor taxa de aceitação. Em todos os cenários QVIA-SDN ofereceu menor latência e variabilidade quando comparado ao algoritmo SCL (Figura 5).



(a) $k = 4$. (b) $k = 8$.

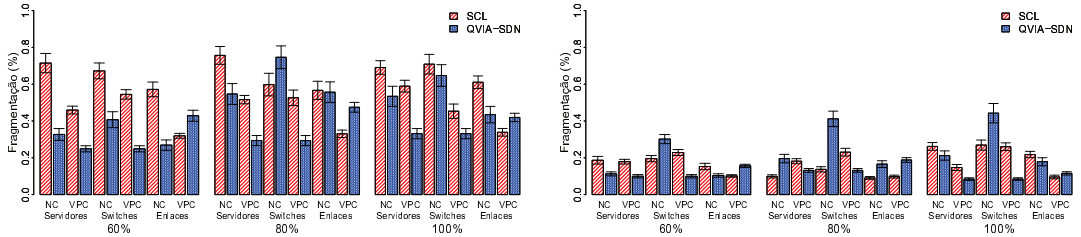
Figura 4. Distribuição acumulada da latência média normalizada.



(a) $k = 4$. (b) $k = 8$.

Figura 5. Variabilidade da latência média.

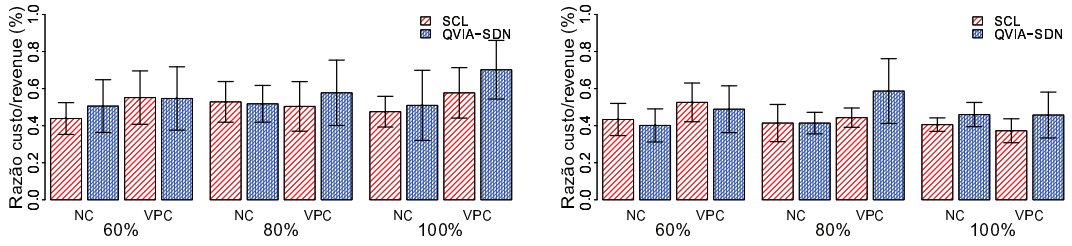
Fragmentação. As Figuras 6(a) e 6(b) mostram a fragmentação para $k = 4$ e $k = 8$, respectivamente. Como esperado, requisições NC utilizam mais recursos de comunicação e requisições VPC permitem uma maior consolidação de servidores, o que também impactou nas taxas de aceitação. Ainda, QVIA-SDN além de melhorar a latência experimentada pelo inquilino reduziu a fragmentação do *data center* (exceto *switches* em NC).



(a) $k = 4$. (b) $k = 8$.

Figura 6. Fragmentação do *data center*.

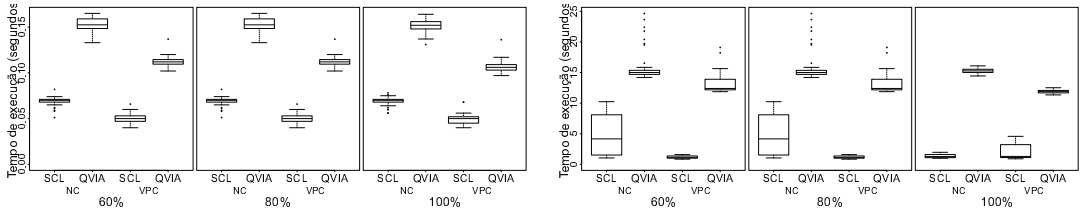
Relação entre custo e *revenue*. As Figuras 7(a) e 7(b) indicam que existe uma equivalência entre QVIA-SDN e SCL e entre NC e VPC. Ainda, é importante realizar uma relação com a taxa de aceitação. SCL obteve maior aceitação, porém uma menor razão custo-*revenue*. Ou seja, os cenários com QVIA-SDN conseguiram alocar IVs com cargas maiores do que aquelas alocadas por SCL. O mesmo raciocínio pode ser aplicado a VPC em relação a NC. Por fim, usando QVIA-SDN é possível melhorar a latência na perspectiva do inquilino sem prejudicar as métricas que representam a perspectiva do provedor.



(a) $k = 4$. (b) $k = 8$.

Figura 7. Relação entre custo e *revenue*.

Tempo médio de alocação de IVs. As Figuras 8(a) e 8(b) mostram que o tempo médio para alocação de IVs com QVIA-SDN é superior na maioria dos casos, entretanto, inferior a 30 segundos no pior caso (para os cenários com $k = 8$).



(a) $k = 4$. (b) $k = 8$.

Figura 8. Tempo médio de alocação de IVs.

7. Conclusões

SDN introduziu benefícios e desafios no provisionamento de IVs em ambientes de nuvens IaaS. O presente trabalho fornece uma formulação baseada em MIP para alocação de IVs em ambiente SDN considerando as restrições impostas. Em seguida, variáveis do MIP foram relaxadas obtendo um LP. O mecanismo proposto, QVIA-SDN, combina o LP com heurísticas de arredondamento. QVIA-SDN, foi comparado com um mecanismo de base, considerando dois tipos de requisições de IVs (NC e VPC). Os resultados obtidos indicam que em *data centers* baseados em SDN é possível realizar a alocação de IVs com garantias de QoS sem afetar as métricas do provedor. Ainda, as técnicas de redução de candidatos, indicam que nem todos os candidatos precisam ser considerados no processo de alocação de IVs. Os resultados promissores, abrem caminhos para trabalhos futuros. Uma primeira linha de pesquisa pode explorar o conhecimento centralizado do controlador para realizar o compartilhamento de recursos de largura de banda residual, enquanto outra linha indica a implementação em um *framework* de gerenciamento.

Agradecimentos: ao laboratório LabP2D, programa PROMOP/UDESC e FAPESC.

Referências

Al-Fares, M., Loukissas, A., and Vahdat, A. (2008). A scalable, commodity data center network architecture. *SIGCOMM Comput. Commun. Rev.*, 38(4):63–74.

Al-Shabibi, A., De Leenheer, M., Gerola, M., Koshibe, A., Parulkar, G., Salvadori, E., and Snow, B. (2014). Openvirtex: Make your virtual sdns programmable. In *Proc. of the HotSDN '14*, pages 25–30. ACM.

Ballani, H., Costa, P., Karagiannis, T., and Rowstron, A. (2011). Towards predictable datacenter networks. *SIGCOMM Comput. Commun. Rev.*, 41(4):242–253.

Cavalcanti, G., Obelheiro, R., and Koslovski, G. (2014). Optimal resource allocation for survivable virtual infrastructures. In *IEEE Conference on Design of Reliable Communication Networks*, pages 1–8.

Chowdhury, M., Rahman, M., and Boutaba, R. (2012). Vineyard: Virtual network embedding algorithms with coordinated node and link mapping. *Networking, IEEE/ACM Transactions on*, 20(1):206–219.

de Oliveira, R. and Koslovski, G. P. (2017). A tree-based algorithm for virtual infrastructure allocation with joint virtual machine and network requirements. *International Journal of Network Management*, 27(1):e1958–n/a. e1958 nem.1958.

Demirci, M. and Ammar, M. (2014). Design and analysis of techniques for mapping virtual networks to software-defined network substrates. *Computer Communications*, 45:1 – 10.

Fischer, A., Botero, J., Till Beck, M., de Meer, H., and Hesselbach, X. (2013). Virtual network embedding: A survey. *Communications Surveys Tutorials, IEEE*, 15(4):1888–1906.

Jennings, B. and Stadler, R. (2014). Resource management in clouds: Survey and research challenges. *Journal of Network and Systems Management*, pages 1–53.

Kreutz, D., Ramos, F. M. V., Verissimo, P. E., Rothenberg, C., Azodolmolky, S., and Uhlig, S. (2015). Software-Defined Networking: A Comprehensive Survey. *Proceedings of the IEEE*, 103(1):14–76.

Manvi, S. S. and Shyam, G. K. (2014). Resource management for infrastructure as a service (iaas) in cloud computing: A survey. *Journal of Network and Computer Applications*, 41:424 – 440.

Mell, P. M. and Grance, T. (2011). SP 800-145. The NIST Definition of Cloud Computing. Technical report, NIST, Gaithersburg, MD, United States.

Mijumbi, R., Serrat, J., Gorricho, J. L., and Boutaba, R. (2015). A path generation approach to embedding of virtual networks. *IEEE Transactions on Network and Service Management*, 12(3):334–348.

Mijumbi, R., Serrat, J., Rubio-Loyola, J., Bouten, N., De Turck, F., and Latre, S. (2014). Dynamic resource management in sdn-based virtualized networks. In *Int. Conf. on Network and Service Management*.

Persico, V., Marchetta, P., Botta, A., and Pescape, A. (2015). Measuring network throughput in the cloud: The case of amazon ec2. *Computer Networks*, 93:408 – 422. Cloud Networking and Comm. {II}.

Popa, L., Krishnamurthy, A., Ratnasamy, S., and Stoica, I. (2011). Faircloud: Sharing the network in cloud computing. In *Proc. of the Workshop on Hot Topics in Networks*, pages 22:1–22:6. ACM.

Rost, M., Fuerst, C., and Schmid, S. (2015). Beyond the stars: Revisiting virtual cluster embeddings. In *In Proc. ACM SIGCOMM Computer Communication Review*.

Sherwood, R., Gibb, G., kiong Yap, K., Casado, M., Mckeown, N., and Parulkar, G. (2009). Flowvisor: A network virtualization layer. Technical report, Deutsche Telekom, Stanford University, Nicira Networks.

Stallings, W. (2015). *Foundations of Modern Networking: SDN, NFV, QoE, IoT, and Cloud*. Addison-Wesley Professional, 1st edition.

Tao, F., Jun, B., and Ke, W. (2015). Allocation and scheduling of network resource for multiple control applications in sdn. *Communications, China*, 12(6):85–95.