TRELIS: Posicionamento de Funções Virtuais de Rede com Economia de Energia e Resiliência

Gabriel F. C. de Queiroz[1,3], Rodrigo de S. Couto[1], Alexandre Sztajnberg[2] *

[1]Universidade do Estado do Rio de Janeiro - PEL - DETEL/FEN

[2]Universidade do Estado do Rio de Janeiro - PEL - DICC/IME

[3]Universidade Federal do Rio de Janeiro - PEE/COPPE/GTA

gqueiroz@gta.ufrj.br, rodrigo.couto@uerj.br, alexszt@ime.uerj.br

Abstract. *Network Function Virtualization increases flexibility and reduces the costs of telecommunication infrastructures. To do so, network functions are implemented on virtualized servers, rather than embedded in dedicated equipment. One of the challenges of this approach is the placement of virtual network functions over the infrastructure, which must be resilient and energy-efficient. This work formulates a mixed integer programming problem to place such functions, choosing the servers meeting service demands. The problem formulated minimizes energy utilization and provides resilience to the services through function replication. As solution, TRELIS heuristic is proposed, which reduces the size of the problem and saves up to 15% of the energy consumption.*

Resumo. *A Virtualização de Funções de Rede aumenta a flexibilidade e reduz custos de infraestruturas de telecomunicações. Para tal, as funções de rede são implementadas em servidores virtualizados, ao invés de embarcadas em equipamentos dedicados. Um dos desafios dessa abordagem é o posicionamento de funções virtuais de rede, que deve ser resiliente e energeticamente eficiente. Este trabalho formula um problema de programação inteira mista para posicionar tais funções, escolhendo os servidores para atender às demandas de serviço. O problema formulado minimiza o uso da energia e provê resiliência aos serviços através de replicação de funções. Como solução, propõe-se a heurística TRELIS, que reduz o tamanho do problema e economiza até 15% da energia.*

1. Introdução

A Virtualização de Funções de Rede (*Network Functions Virtualization*, NFV) oferece uma alternativa ao uso de equipamentos dedicados (*middleboxes*), através da alocação de Funções Virtuais de Rede (*Virtual Network Functions*, VNF). As VNFs são módulos de *software* que oferecem os serviços dos *middleboxes* em servidores genéricos, amplamente disponíveis nas diversas instalações. Se por um lado abre-se mão de equipamentos construídos especificamente para prover serviços de rede, por outro lado as VNFs podem oferecer os mesmos serviços de forma modular e adaptável, resiliente e com qualidade aceitável, porém com menor custo [Mijumbi et al., 2015, John et al., 2013].

*Este trabalho foi realizado com recursos da FAPERJ, CNPq e CAPES.

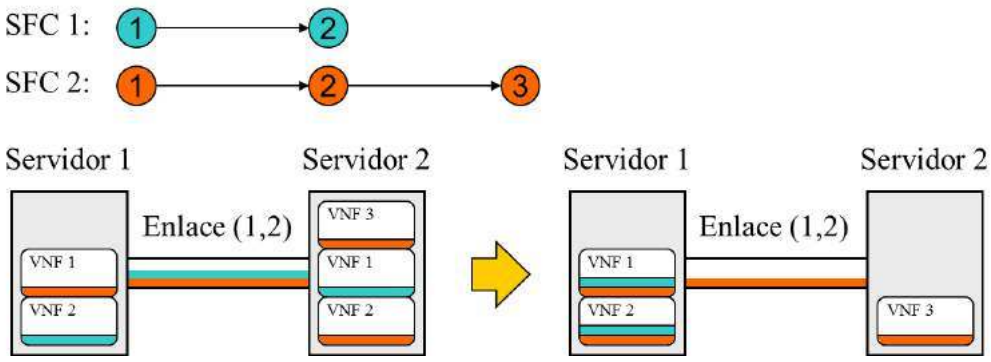


Figura 1. Compartilhamento de VNFs pelas SFCs.

Os serviços de NFV são providos por uma sequência de VNFs, denominada Encadeamento de Funções de Serviço (*Service Function Chaining*, SFC) [Sahhaf et al., 2015, Herrera e Botero, 2016]. Um dos desafios da abordagem NFV é a alocação ótima de SFCs em uma infraestrutura, garantindo a qualidade de serviço [Han et al., 2015]. Em um cenário geral, uma política de alocação deve considerar os recursos e requisitos demandados por várias SFCs, os recursos disponíveis nos servidores físicos e a largura de banda dos enlaces. Um servidor pode ser capaz de hospedar várias VNFs e, da mesma forma, uma VNF pode atender às requisições de várias SFCs [Herrera e Botero, 2016].

No posicionamento de SFCs, dois requisitos representam desafios interessantes: a resiliência e economia de energia. A resiliência pode ser garantida através da replicação de VNFs para manter as SFCs operantes mesmo em caso de falha [Couto et al., 2015]. A economia de energia pode ser alcançada por uma estratégia de compartilhar VNFs entre diversas SFCs, reduzindo os recursos ociosos e, consequentemente, o uso da energia. A Figura 1 ilustra à esquerda uma infraestrutura sem compartilhamento de VNFs pelas SFCs e, à direita, sua posterior reorganização considerando o compartilhamento. Como as duas SFCs solicitam duas VNFs do mesmo tipo, elas podem ser compartilhadas, já que as VNFs possuem capacidade para atender às requisições de ambas. Assim, são alocadas duas instâncias a menos no Servidor 2 e também é poupada a largura de banda no enlace.

Este trabalho formula um problema de Programação Linear Inteira Mista (*Mixed Integer Linear Programming*, MILP) para posicionar SFCs em uma infraestrutura. A formulação matemática do problema é resolvida para três redes, RNP, GEANT e RENATER, através do IBM CPLEX. No entanto, devido à elevada complexidade do problema, o consumo de memória e os tempos de execução inviabilizam a solução ótima. Assim, a heurística Redução em Árvore e Solução em Linha (*Tree REduction and LIne Solution*, TRELIS) é proposta para reduzir o tamanho do problema, através da seleção de um número reduzido de nós na topologia para posicionar as SFCs. Os resultados mostram que a heurística é efetiva e economiza até 15% da energia dos servidores, ainda garantindo o funcionamento dos serviços provisionados se um servidor falhar.

O trabalho está estruturado da seguinte forma. A Seção 2 apresenta os trabalhos relacionados. Na Seção 3, discutem-se os conceitos e estratégias utilizados no trabalho. O problema de otimização é formulado na Seção 4. A Seção 5 descreve a TRELIS, enquanto a Seção 6 apresenta os experimentos realizados. Finalmente, a Seção 7 conclui o trabalho e aponta direções futuras.

2. Trabalhos Relacionados

O posicionamento de SFCs tem sido considerado em várias frentes de pesquisa. Em [Luizelli et al., 2015], é formalizado o problema de posicionamento de SFCs, com o objetivo de reduzir o número de VNFs alocadas e comparando valores de atraso fim a fim entre a solução proposta e infraestruturas tradicionais. O trabalho de [Sahhaf et al., 2015] considera SFCs formadas por serviços descritos em alto nível, que são decompostos em VNFs de baixo nível também ligadas em SFCs. Por exemplo, um *firewall* pode incluir uma VNF de filtragem *web* e uma de bloqueio de *malware*. O estudo de [Bari et al., 2016] trata do problema de orquestração de VNFs para reduzir despesas operacionais (*Operational Expenditures*, OPEX), minimizando os custos de implementação, consumo de energia e penalidades por violações em Acordos de Nível de Serviço (*Service Level Agreement*, SLA), considerando o consumo de energia como parte dos custos. O trabalho de [Mehraghdam et al., 2014] também se destina à alocação de recursos em NFV, mapeando as requisições de VNFs na rede para vários objetivos, como minimizar a latência nos enlaces criados, maximizar a largura de banda restante nos enlaces físicos ou minimizar o número de sítios com VNFs instanciadas. Em [Moens e De Turck, 2014], também é proposto um problema de posicionamento de SFCs, porém voltado para cenários híbridos, com *middleboxes* dedicados e VNFs instanciadas em servidores genéricos. O estudo de [Ma et al., 2015] aborda o problema de alocação de VNFs considerando a variação de tráfego, já que as VNFs podem tanto inserir tráfego em uma SFC quanto consumi-lo. Em [Clayman et al., 2014], é proposta uma arquitetura baseada em um elemento orquestrador que aloca os serviços de rede em nós virtuais. Embora os artigos citados tratem de diversos aspectos no posicionamento de SFCs, esses trabalhos de NFV não consideram resiliência e os serviços são interrompidos em caso de falha. Assim, a contribuição deste artigo é considerar a resiliência, além de minimizar o consumo de energia dos servidores.

Considerando a resiliência, uma estratégia baseada em replicação também é utilizada em [Couto et al., 2015], em que são utilizadas réplicas de centro de dados em redes geodistribuídas para prover resiliência em cenários de desastre, visando a sobrevivência do conteúdo armazenado. Já em [Rahman e Boutaba, 2013], é formulado um problema de alocação de recursos para Mapeamento de Redes Virtuais (*Virtual Network Embedding*, VNE) e são considerados aspectos de resiliência, já que o provisionamento das redes virtuais leva em conta falhas de enlace para incentivar a sobrevivência da rede, diferente deste artigo, que considera falhas de servidor. O problema de VNE é mais simples que o de NFV, pois o mapeamento ocorre em um nível, das redes virtuais para a infraestrutura. Em NFV, são dois níveis, das SFCs para as VNFs e dessas para a infraestrutura [Luizelli et al., 2015]. O VNE pode ser visto como um precursor do posicionamento em NFV.

3. Conceitos de Resiliência e Estratégia Adotada

O modelo de falha adotado considera que um servidor se torna inoperante na ocorrência de uma falha, ou seja, falhas silenciosas do tipo *fail-stop* [Tanenbaum e Van Steen, 2007]. Quando um servidor falha, suas VNFs e seus enlaces físicos tornam-se imediatamente inoperantes. Assim, as SFCs que atravessam o servidor que falhou são particionadas. Se houver algum servidor ligado ao restante da rede apenas por um enlace com servidor que falhou, esse também fica inalcançável. No entanto, considera-se que esse servidor isolado continua funcionando, apenas perdendo a conectividade com o restante da rede. As VNFs instanciadas nesse servidor continuam

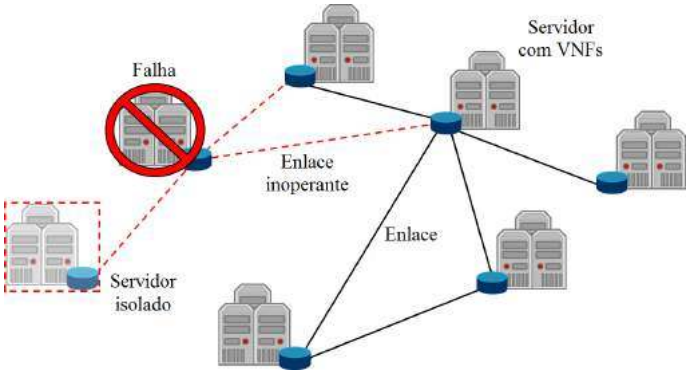


Figura 2. Elementos de um SRG são afetados pela mesma falha.

operacionais e podem ainda ser utilizadas nas SFCs, contanto que as mesmas não dependam de outro segmento da rede. A Figura 2 ilustra essas situações.

Como a falha de um servidor pode afetar outros elementos da rede, utiliza-se o conceito de Grupo de Risco Compartilhado (*Shared Risk Group*, SRG) [Couto et al., 2014]. Um SRG é composto por todos os elementos em uma determinada região afetados pela mesma falha. Na Figura 2, o SRG associado ao servidor que falhou é ilustrado pelos enlaces ligados a ele, tracejados, que ficam inalcançáveis quando a falha ocorre. O servidor dentro das linhas tracejadas também está inalcançável para o restante da rede, mas continua operacional, portanto não falha.

No modelo proposto, considera-se a falha de qualquer um dos servidores. Essa falha é mais crítica do que uma falha de enlace, já que quando um servidor fica inoperante seus enlaces também falham. O número de réplicas necessárias para tolerar k falhas silenciosas, do tipo *fail-stop*, é dado por $n = k + 1$ [Tanenbaum e Van Steen, 2007]. Apenas uma falha ($k = 1$) é considerada para simplificar o modelo, embora o mesmo possa ser expandido para tolerar mais falhas. Assim, para cada VNF operacional na SFC, deve haver uma réplica, isto é, uma VNF de *backup*. Essa réplica deve ter igual capacidade para poder substituir a VNF operacional correspondente em caso de falha, além de enlaces alternativos para preservar a integridade da SFC.

A Figura 3 mostra como a replicação de VNFs é aplicada a uma SFC e como os enlaces são construídos entre pares de VNFs. No lado esquerdo, é representada a SFC que o cliente solicita e, no lado direito, a SFC que deve ser alocada para garantir a resiliência. As linhas tracejadas indicam VNFs e enlaces virtuais de *backup*, enquanto as linhas cheias representam a SFC principal. Sendo assim, para cada VNF operacional há uma VNF de *backup* correspondente e, para cada enlace virtual da SFC principal, há três enlaces replicados interligando as VNFs que conectam. As VNFs dentro dos círculos são as funções de rede encadeadas que um usuário solicita ao seu provedor, como *firewall*, *proxy* e tradução de endereços [Mijumbi et al., 2015]. Já as VNFs em quadrados são as instâncias de *backup* e permitem que o serviço continue sendo provido, mesmo que um servidor da rede se torne inoperante. Portanto, para tornar a SFC resiliente na ocorrência de uma falha de servidor, a estratégia adotada consiste na replicação de cada VNF operacional solicitada em servidores diferentes dos das operacionais. Por exemplo, mesmo que as três VNFs 0, 1 e 2 da Figura 3 estejam hospedadas no mesmo servidor e ele falhe, as VNFs de *backup* das três continuam ativas e a SFC pode ser provida pelos enlaces $(0', 1')$ e $(1', 2')$.

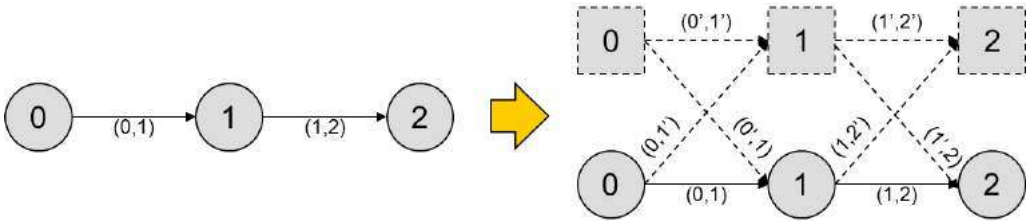


Figura 3. Exemplo de uma SFC qualquer com replicação de VNFs.

4. Formulação do Problema de Alocação com Replicação de SFCs

O problema de otimização formulado tem como objetivo reduzir o consumo de energia nos servidores, aproveitando as capacidades ociosas das VNFs operacionais. Além disso, o problema replica VNFs, garantindo resiliência para uma falha *fail-stop*.

A redução da quantidade de VNFs devido ao compartilhamento também pode reduzir o consumo de energia. Além disso, reduz-se o OPEX, já que os gastos com despesas de energia elétrica e manutenção de *hardware* e *software* diminuem. Embora necessitem estar alocadas e com recursos suficientes para seu funcionamento, as VNFs de *backup* permanecem inativas até que sejam eventualmente ativadas, utilizando técnicas de *cold standby*. Assim, apenas o consumo de energia das VNFs operacionais é considerado.

A Tabela 1 lista as variáveis, parâmetros e conjuntos utilizados na formulação do problema, bem como as notações utilizadas e suas descrições. A formulação é composta pela função objetivo e por restrições específicas.

Função objetivo. O modelo de energia é baseado em [Bari et al., 2016]. Nesse modelo, o consumo de energia pode variar entre o valor mínimo ($\mathcal{E}_{min_i}$), referente ao estado de *idle*, até o máximo ($\mathcal{E}_{max_i}$), que corresponde a 100% de uso do servidor. A Equação 1 é a função objetivo do problema MILP, que visa minimizar a parcela variável do consumo de energia, já que o consumo em *idle* é considerado fixo. Em seguida, são apresentadas as restrições para solução do problema e as equações de domínio de variáveis.

$$\text{minimizar} \sum_{i\in\mathcal{I}} \sum_{m\in\mathcal{M}} \sum_{j\in\mathcal{J}_m} (\mathcal{E}_{max_i} - \mathcal{E}_{min_i}) \cdot \frac{\mathcal{V}_{m,j}}{\mathcal{S}_i} \cdot y_{i,m,j} \tag{1}$$

$$\text{sujeito a} \sum_{m\in\mathcal{M}} (\sum_{j\in\mathcal{J}_m} \mathcal{V}_{m,j} \cdot y_{i,m,j} + \sum_{r\in\mathcal{R}_m} \mathcal{K}_{m,r} \cdot z_{i,m,r}) \leq \mathcal{S}_i \quad \forall i \in \mathcal{I} \tag{2}$$

$$\sum_{q\in\mathcal{Q}} \mathcal{F}_{m,q} \cdot a_{i,m,q} \leq \sum_{j\in\mathcal{J}_m} \mathcal{V}_{m,j} \cdot y_{i,m,j} \quad \forall i \in \mathcal{I},\ m \in \mathcal{M} \tag{3}$$

$$\sum_{i\in\mathcal{I}} a_{i,m,q} = \begin{cases} 1, & \text{se } F_{m,q} \geq 0 \\ 0, & \text{se } F_{m,q} = 0 \end{cases} \quad \forall m \in \mathcal{M},\ q \in \mathcal{Q} \tag{4}$$

$$\sum_{i\in\mathcal{I}} y_{i,m,j} \leq 1 \quad \forall m \in \mathcal{M},\ j \in \mathcal{J}_m \tag{5}$$

Tabela 1. Notações utilizadas no problema.

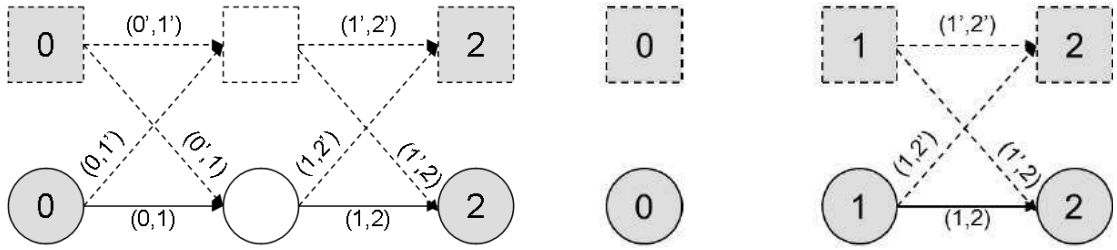
Notação	Descrição	Tipo
$\mathcal{I}$	Servidores disponíveis	Conjunto
$\mathcal{M}$	Tipos de VNF disponíveis	Conjunto
$\mathcal{J}_m$	Instâncias disponíveis para a VNF m	Conjunto
$\mathcal{R}_m$	Instâncias de *backup* disponíveis para a VNF m	Conjunto
$\mathcal{Q}$	Requisições de serviço na forma de SFCs	Conjunto
$\mathcal{U}$	Enlaces virtuais ligando apenas VNFs operacionais	Conjunto
$\mathcal{U}'$	Enlaces virtuais ligando uma VNF operacional e uma de *backup*, (u, v') e (u', v), ou duas VNFs de *backup*, (u', v')	Conjunto
$\mathcal{E}_{max_i}$	Consumo de energia máximo do servidor i	Parâmetro
$\mathcal{E}_{min_i}$	Consumo de energia mínimo do servidor i	Parâmetro
$\mathcal{S}_i$	Capacidade dos recursos computacionais do servidor i	Parâmetro
$\mathcal{V}_{m,j}$	Recursos computacionais utilizados pela instância j da VNF do tipo m	Parâmetro
$\mathcal{K}_{m,r}$	Recursos computacionais utilizados pelo *backup* r da VNF do tipo m	Parâmetro
$\mathcal{F}_{m,q}$	Recursos computacionais da VNF do tipo m requeridos pela SFC q	Parâmetro
$\mathcal{B}_{i,n}$	Largura de banda do enlace físico (i, n)	Parâmetro
$\mathcal{W}_{q,u,v}$	Largura de banda solicitada pela SFC q no enlace virtual (u, v)	Parâmetro
$\mathcal{T}_q$	Atraso total permitido pela SFC q	Parâmetro
$\mathcal{D}_{i,n}$	Atraso de propagação associado ao enlace físico (i, n)	Parâmetro
$\mathcal{P}_m$	Atraso de processamento associado à VNF do tipo m	Parâmetro
m_q^o , m_q^f	Primeira VNF (o) de uma SFC q e última VNF (f) de uma SFC q	Parâmetro
m_q^u , m_q^v	VNF que está no nó u ou no nó v de um enlace virtual (u, v) qualquer da SFC q	Parâmetro
$y_{i,m,j}$	Variável binária indicando se a instância j da VNF do tipo m está alocada no servidor i	Variável
$z_{i,m,r}$	Variável binária indicando se o *backup* r da VNF do tipo m está alocada no servidor i	Variável
$a_{i,m,q}$	Variável binária indicando se a SFC q solicita uma VNF do tipo m no servidor i	Variável
$c_{i,m,q}$	Variável binária indicando se a SFC q solicita um *backup* da VNF do tipo m no servidor i	Variável
$b_{i,n,q,u,v}$	Variável binária indicando se a SFC q solicita um enlace operacional (u, v) no enlace físico (i, n)	Variável
$d_{i,n,q,u,v}$	Variável binária indicando se a SFC q solicita um enlace de *backup* (u, v) no enlace físico (i, n)	Variável
$x_{i,n,q,u,v}$	Variável binária indicando se algum dos enlaces de *backup* (u, v'), (u', v) ou (u', v') da SFC q está no enlace físico (i, n)	Variável
$g_{i,q,u,v}$	Variável binária indicando se o enlace virtual qualquer (u, v) da SFC q está no SRG afetado por uma possível falha no servidor i	Variável

Restrições de alocação de VNFs. A Inequação 2 assegura que o consumo de recursos computacionais de todas as VNFs, operacionais e de *backup*, em um servidor não ultrapasse a capacidade do mesmo. De forma similar, a Inequação 3 garante que o total de recursos demandados por todas as SFCs que utilizem uma VNF não ultrapasse a capacidade dela. A Equação 4 afirma que, se uma SFC solicita recursos de uma VNF específica, então deve haver algum servidor para alocar essa VNF, não podendo haver uma requisição de serviço não atendida. A Inequação 5 afirma que uma instância de uma determinada VNF não pode ser atribuída a mais de um servidor simultaneamente.

$$\sum_{q \in \mathcal{Q}} \sum_{(u,v) \in \mathcal{U}} \mathcal{W}_{q,u,v} \cdot (b_{i,n,q,u,v} + x_{i,n,q,u,v}) \leq \mathcal{B}_{i,n} \quad \forall i, n \in \mathcal{I} \tag{6}$$

$$\sum_{i \in \mathcal{I}} (\sum_{n \in \mathcal{I}} \sum_{(u,v) \in \mathcal{U}} \mathcal{D}_{i,n} \cdot b_{i,n,q,u,v} \; + \sum_{m \in \mathcal{M}} \mathcal{P}_m \cdot a_{i,m,q}) \leq \mathcal{T}_q \quad \forall q \in \mathcal{Q} \tag{7}$$

Restrições de alocação de enlaces. A Inequação 6 assegura que a largura de banda total utilizada pelos enlaces virtuais em um determinado enlace físico não ultrapasse a banda disponível no mesmo. Vale notar que, como é possível que dois ou três enlaces de *backup* de um mesmo enlace virtual operacional atravessem um *link* físico em comum, a reserva de banda ocorreria para cada um dos dois ou três enlaces de *backup*. Isso não é necessário, já que, em caso de falha, basta que seja reservada a capacidade de apenas um desses enlaces



(a) VNF principal e *backup* compartilham servidor. (b) Os quatro enlaces caem ao mesmo tempo.

Figura 4. Dois tipos de quebra na SFC.

de *backup*, ao invés de alocar desnecessariamente o dobro ou o triplo desse valor. A Inequação 7 garante que o atraso total em uma SFC não exceda o atraso máximo aceitável daquela SFC. Esse atraso é dado pela soma do atraso de processamento de cada VNF requerida e o atraso de propagação em cada um dos enlaces físicos que a SFC atravessa.

$$\sum_{q \in \mathcal{Q}} \mathcal{F}_{m,q} \cdot c_{i,m,q} \leq \sum_{r \in \mathcal{R}_m} \mathcal{K}_{m,r} \cdot z_{i,m,r} \quad \forall i \in \mathcal{I},\ m \in \mathcal{M} \tag{8}$$

$$\sum_{i \in \mathcal{I}} c_{i,m,q} = \begin{cases} 1, & \text{se } F_{m,q} \geq 0 \\ 0, & \text{se } F_{m,q} = 0 \end{cases} \quad \forall m \in \mathcal{M},\ q \in \mathcal{Q} \tag{9}$$

$$a_{i,m,q} + c_{i,m,q} \leq 1 \quad \forall i \in \mathcal{I},\ m \in \mathcal{M},\ q \in \mathcal{Q} \tag{10}$$

$$\sum_{i \in \mathcal{I}} z_{i,m,r} \leq 1 \quad \forall m \in \mathcal{M},\ r \in \mathcal{R}_m \tag{11}$$

$$0 \leq 3 \cdot x_{i,n,q,u,v} - d_{i,n,q,u,v'} - d_{i,n,q,u',v} - d_{i,n,q,u',v'} \leq 2\, \forall i, n \in \mathcal{I},\ q \in \mathcal{Q},\ (u,v) \in \mathcal{U}' \tag{12}$$

Restrições de alocação de VNFs de *backup*. A Inequação 8 e a Equação 9 têm o mesmo propósito que a Inequação 3 e a Equação 4, respectivamente, porém se destinam às VNFs de *backup*. A Inequação 10 impede que uma VNF operacional e o *backup* associado a ela estejam no mesmo servidor, o que poderia levar ao problema mostrado na Figura 4(a), no qual a falha de um servidor quebra a SFC e sua réplica. A Inequação 11 tem a mesma função da Inequação 5, porém voltada para as VNFs de *backup*. A Inequação 12 representa a função lógica OU entre as três variáveis de enlace de *backup*, que é utilizada na Inequação 6 para alocar a requisição de banda de apenas um dos três enlaces de *backup* quando eles estão no mesmo enlace físico.

$$\sum_{n \in \mathcal{N}} (b_{i,n,q,u,v} - b_{n,i,q,u,v}) = a_{i,m_q^o,q} - a_{i,m_q^f,q} \quad \forall i \in \mathcal{I},\ q \in \mathcal{Q},\ (u,v) \in \mathcal{U} \tag{13}$$

$$0 \leq a_{i,m_q^u,q} + a_{i,m_q^v,q} - 2 \cdot b_{i,i,q,u,v} \leq 1 \quad \forall i \in \mathcal{I},\ q \in \mathcal{Q},\ (u,v) \in \mathcal{U} \tag{14}$$

$$b_{i,n,q,u,v} + b_{n,i,q,u,v} \leq 1 \quad \forall i, n \in \mathcal{I},\ q \in \mathcal{Q},\ (u,v) \in \mathcal{U} \tag{15}$$

$$d_{i,n,q,u,v} + d_{n,i,q,u,v} \leq 1 \quad \forall i, n \in \mathcal{I},\ q \in \mathcal{Q},\ (u,v) \in \mathcal{U}' \tag{16}$$

$$\sum_{n \in \mathcal{I}} b_{i,n,q,u,v} \leq 1 \quad \forall i \in \mathcal{I},\ q \in \mathcal{Q},\ (u,v) \in \mathcal{U} \tag{17}$$

$$\sum_{i \in \mathcal{I}} b_{i,n,q,u,v} \leq 1 \quad \forall n \in \mathcal{I},\ q \in \mathcal{Q},\ (u,v) \in \mathcal{U} \tag{18}$$

$$\sum_{n \in \mathcal{I}} d_{i,n,q,u,v} \leq 1 \quad \forall i \in \mathcal{I},\ q \in \mathcal{Q},\ (u,v) \in \mathcal{U}' \tag{19}$$

$$\sum_{i \in \mathcal{I}} d_{i,n,q,u,v} \leq 1 \quad \forall n \in \mathcal{I},\ q \in \mathcal{Q},\ (u,v) \in \mathcal{U}' \tag{20}$$

Restrições de formação de enlaces. A Equação 13 especifica que o somatório de todos os fluxos que entram e saem de um servidor é igual ao somatório de todos os fluxos que surgem nele e se destinam a ele. A Inequação 14 afirma que um enlace virtual interno deve ser formado se duas VNFs operacionais de uma SFC estiverem no mesmo servidor. Esse enlace, por ser estabelecido internamente no servidor, possui banda infinita e atraso zero. As Inequações 15 e 16 asseguram que um enlace virtual, operacional ou de *backup*, só pode estar alocado em um sentido do enlace físico. Já as Inequações 17 e 18 afirmam que um enlace virtual operacional só pode sair ou chegar, respectivamente, a um servidor por um único enlace físico. As Inequações 19 e 20 fazem o mesmo para os de *backup*.

$$\sum_{n \in \mathcal{N}} (d_{i,n,q,u,v'} - d_{n,i,q,u,v'}) = a_{i,m_q^o,q} - c_{i,m_q^f,q} \quad \forall i \in \mathcal{I},\ q \in \mathcal{Q},\ (u,v') \in \mathcal{U}' \tag{21}$$

$$\sum_{n \in \mathcal{N}} (d_{i,n,q,u',v} - d_{n,i,q,u',v}) = c_{i,m_q^o,q} - a_{i,m_q^f,q} \quad \forall i \in \mathcal{I},\ q \in \mathcal{Q},\ (u',v) \in \mathcal{U}' \tag{22}$$

$$\sum_{n \in \mathcal{N}} (d_{i,n,q,u',v'} - d_{n,i,q,u',v'}) = c_{i,m_q^o,q} - c_{i,m_q^f,q} \quad \forall i \in \mathcal{I},\ q \in \mathcal{Q},\ (u',v') \in \mathcal{U}' \tag{23}$$

$$0 \leq a_{i,m_q^u,q} + c_{i,m_q^v,q} - 2 \cdot d_{i,i,q,u,v'} \leq 1 \quad \forall i \in \mathcal{I},\ q \in \mathcal{Q},\ (u,v') \in \mathcal{U}' \tag{24}$$

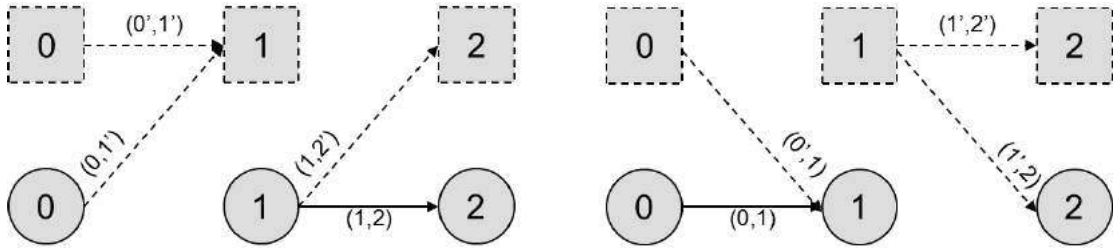
$$0 \leq c_{i,m_q^u,q} + a_{i,m_q^v,q} - 2 \cdot d_{i,i,q,u',v} \leq 1 \quad \forall i \in \mathcal{I},\ q \in \mathcal{Q},\ (u',v) \in \mathcal{U}' \tag{25}$$

$$0 \leq c_{i,m_q^u,q} + c_{i,m_q^v,q} - 2 \cdot d_{i,i,q,u',v'} \leq 1 \quad \forall i \in \mathcal{I},\ q \in \mathcal{Q},\ (u',v') \in \mathcal{U}' \tag{26}$$

As Equações 21, 22 e 23 têm a mesma função que a Equação 13, estando relacionadas aos enlaces de *backup* que partem de uma VNF operacional para um *backup*, que partem de um *backup* para uma VNF operacional e que ligam duas VNFs de *backup*, respectivamente. Da mesma forma, as Inequações 24, 25 e 26 formam um paralelo com a Inequação 14, criando enlaces virtuais internos de *backup* quando estão em sequência, no mesmo servidor, uma VNF operacional e o *backup* da próxima, um *backup* e a VNF operacional seguinte e duas VNFs de *backup*, respectivamente.

$$0 \leq (1 + (2 \cdot (n-1))) \cdot g_{i,q,u,v} - \sum_{n \in \mathcal{N}} (b_{i,n,q,u,v} + b_{n,i,q,u,v}) + b_{i,i,q,u,v} \leq 2 \cdot (n-1)$$
$$\forall i \in \mathcal{I},\ q \in \mathcal{Q},\ (u,v) \in \mathcal{U} \tag{27}$$

$$0 \leq (1 + (2 \cdot (n-1))) \cdot g_{i,q,u,v} - \sum_{n \in \mathcal{N}} (d_{i,n,q,u,v} + d_{n,i,q,u,v}) + d_{i,i,q,u,v} \leq 2 \cdot (n-1)$$
$$\forall i \in \mathcal{I},\ q \in \mathcal{Q},\ (u,v) \in \mathcal{U}' \tag{28}$$



(a) Enlaces chegam ao *backup* e saem da principal. (b) Enlaces chegam à principal e saem do *backup*.

Figura 5. Quebra na SFC por uma descontinuidade espelhada.

$$g_{i,q,u,v} + g_{i,q,u,v'} + g_{i,q,u',v} + g_{i,q,u',v'} \leq 3 \quad \forall i \in \mathcal{I},\ q \in \mathcal{Q},\ (u,v) \in \mathcal{U} \cup \mathcal{U}' \tag{29}$$

$$g_{i,q,u,v} + g_{i,q,u',v} + g_{i,q,v',z} + g_{i,q,v',z'} \leq 3 \quad \forall i \in \mathcal{I},\ q \in \mathcal{Q},\ (u,v),(v,z) \in \mathcal{U} \cup \mathcal{U}' \tag{30}$$

$$g_{i,q,u,v'} + g_{i,q,u',v'} + g_{i,q,v,z} + g_{i,q,v,z'} \leq 3 \quad \forall i \in \mathcal{I},\ q \in \mathcal{Q},\ (u,v),(v,z) \in \mathcal{U} \cup \mathcal{U}' \tag{31}$$

Restrições de resiliência e SRGs. As Inequações 27 e 28 definem uma função lógica OU que indica se um determinado enlace virtual pertence ao SRG de um servidor, ou seja, se uma possível falha no servidor em questão implica na queda do enlace virtual. Assim, sempre que um enlace virtual chegar, sair ou for um enlace interno em um determinado servidor, ele estará contido no grupo de enlaces afetados pela falha naquele servidor. Vale ressaltar que um enlace virtual pertence a tantos SRGs distintos quantos forem os servidores pelos quais ele passar. Assim como a Inequação 10, as Inequações 29, 30 e 31 também garantem que o serviço não seja interrompido. A Inequação 29 define que os quatro enlaces virtuais relacionados, isto é, o enlace operacional e seus três enlaces de *backup*, não podem estar simultaneamente no mesmo SRG, o que poderia levar ao problema mostrado na Figura 4(b). Nas Inequações 30 e 31, para um dado enlace virtual qualquer (u, v), (v, z) é o enlace seguinte, tendo a VNF em v como origem, ao invés de destino. Essas inequações evitam uma ruptura espelhada na SFC. Isso ocorre quando os dois enlaces que chegam a uma VNF de *backup* e os dois enlaces que partem de sua VNF operacional correspondente e vice-versa ficam inacessíveis ao mesmo tempo, como mostrado na Figura 5.

Domínio das variáveis. Todas as variáveis utilizadas são binárias, como visto na Tabela 1. Assim, devido à limitação de espaço, as equações de domínio de variáveis foram omitidas.

5. Heurística TRELIS (*Tree REduction and LIne Solution*)

Dada a complexidade do problema MILP formulado, este trabalho propõe uma heurística para aumentar a escalabilidade no posicionamento de SFCs. A ideia fundamental é reduzir o número de variáveis. De acordo com a Tabela 1, o número de servidores da infraestrutura é um índice presente em todas as variáveis e, sendo assim, reduzir a quantidade de servidores que podem hospedar VNFs diminui o número de variáveis e de restrições do problema. Em linhas gerais, a TRELIS recebe a rede original e remove enlaces que fecham ciclos, transformando-a em uma *spanning tree* mínima, contendo os nós da rede original. Então, a TRELIS cria uma rede em linha a partir da árvore para ser usada no problema de posicionamento de SFCs. No exemplo da Figura 6, uma topologia é submetida à TRELIS, que remove os ciclos da rede e forma uma linha com 8 servidores.

O Algoritmo 1 detalha a TRELIS. Da linha 1 até a 4, são definidos os valores iniciais das variáveis. O consumo de energia é iniciado com E_{max}, que representa o maior

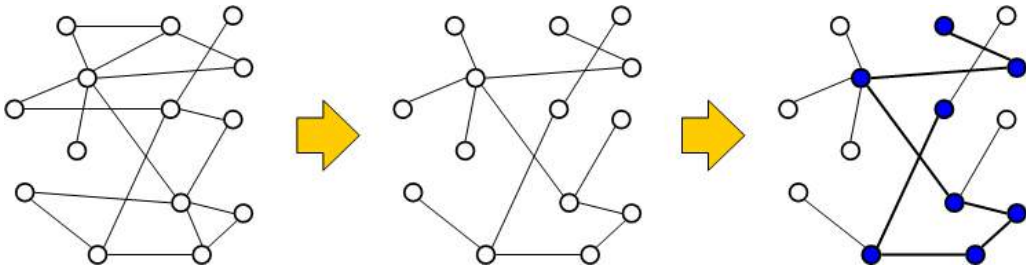


Figura 6. Remoção de ciclos e seleção de nós em uma linha feita pela TRELIS.

gasto possível de energia na rede. Para resolver o problema, a TRELIS calcula inicialmente a quantidade mínima de servidores da rede em linha necessária para a alocação das VNFs operacionais e de *backup*, com base nos recursos disponíveis. Por exemplo, se cada servidor é capaz de hospedar apenas 8 VNFs e as requisições precisam de 64 VNFs, então são necessários, no mínimo, 8 servidores. Esse tamanho mínimo é representado por $minun$. Na linha 5, a *spanning tree* mínima é formada a partir da rede original. Na linha 6, a função `melhoresCaminhosTodosPares` encontra todos os caminhos mais curtos entre cada par de servidores, formando linhas de diferentes tamanhos. Então, seleciona-se um tamanho i, a partir de 2 servidores, e comparam-se todas as linhas com i nós. A melhor rede é aquela com o maior somatório de largura de banda de todos os enlaces. Assim, há uma melhor rede para cada i e são essas as linhas usadas para resolver o problema. O maior valor de i é igual ao tamanho da maior rede em linha possível para a rede inicial.

A TRELIS itera da linha 7 até a 21, selecionando, na função `extrairLinha` (linha 11), uma rede em linha com $minun$ servidores. Essa rede é escolhida a partir dos caminhos calculados na linha 6 e o valor $minun$ é incrementado a cada iteração. Na função `resolverProblema` (linha 12), a rede em linha escolhida é então fornecida como entrada do problema formulado na Seção 4. Em cada iteração, calcula-se o ganho na economia de energia em relação à iteração anterior. Se o $ganho$ for menor que ϵ, a TRELIS termina a execução e escolhe o posicionamento da última iteração, já que, embora o resultado não seja suficientemente melhor a ponto de justificar uma nova iteração, ele ainda é um resultado melhor que o da penúltima iteração. Se o $ganho$ for menor que zero, escolhe-se o posicionamento da penúltima iteração, nas linhas 22 a 26, já que esse resultado é melhor que o da última iteração. Isso garante que o resultado final obtido pela TRELIS seja sempre o melhor dentre todos os resultados obtidos nas iterações anteriores, evitando que o último resultado seja escolhido caso ele seja pior que o anterior.

6. Avaliação dos Resultados

Para avaliar a solução ótima e a TRELIS, utiliza-se o IBM CPLEX como ferramenta de otimização. São realizados dois experimentos para SFCs mais curtas e mais longas, com 2 e 4 VNFs, respectivamente. No primeiro experimento, são considerados dois cenários, com servidores de capacidade $\mathcal{S}_i$ = 1000 $\forall i \in \mathcal{I}$ e $\mathcal{S}_i$ = 4000 $\forall i \in \mathcal{I}$. No segundo experimento, são considerados servidores com $\mathcal{S}_i$ = 4000 $\forall i \in \mathcal{I}$ e $\mathcal{S}_i$ = 8000 $\forall i \in \mathcal{I}$. Aumenta-se a capacidade para as SFCs longas pois a quantidade de recursos requerida aumenta por haver mais VNFs. Utilizam-se tipos diferentes de VNF para formar as SFCs. Para cada VNF, aloca-se $\mathcal{V}_{m,j}$ = 125 unidades de capacidade de um servidor. Cada requisição de VNF nas SFCs consome $\mathcal{F}_{m,q}$ = 100 unidades de recurso de uma VNF. Os

Algoritmo 1: Heurística TRELIS

Entrada: Rede original $redeOriginal$, Requisições de SFCs $requisicoes$, Ganho mínimo ϵ.
Saída: Posicionamento das SFCs $melhorPosicionamento$, Consumo de energia $energia$

$minun = \texttt{arredondar}(\frac{2 \cdot nrVnfsRequisitadas}{nrMedioDeVnfsPorServidor})$;
$ganho = \epsilon + 1$;
$energia = E_{max}$;
$posicionamento = Vazio$;
$arvore = \texttt{spanningTree}(redeOriginal)$;
$melhoresLinhasPossiveis = \texttt{melhoresCaminhosTodosPares}(arvore)$;
enquanto $ganho > \epsilon$ **faça**
    $ganhoAnterior = ganho$;
    $energiaAnterior = energia$;
    $posicionamentoAnterior = posicionamento$;
    $rede = \texttt{extrairLinha}(minun, melhoresLinhasPossiveis)$;
    $energia, posicionamento = \texttt{resolverProblema}(rede)$;
    se $posicionamento \neq Vazio$ **então**
        $ganho = energiaAnterior - energia$;
    senão
        $ganho = ganhoAnterior$;
        $energia = energiaAnterior$;
        $posicionamento = posicionamentoAnterior$;
    fim
    $minun$ += 1;
fim
se $ganho > 0$ **então**
    $melhorPosicionamento = posicionamento$;
senão
    $melhorPosicionamento = posicionamentoAnterior$;
fim

valores de energia dos servidores foram retirados de [Bari et al., 2016]. Utiliza-se como rede física as redes RNP, GEANT e RENATER, empregando a mesma topologia, banda e latência de [Couto et al., 2015]. Emprega-se nos experimentos uma máquina com processador Intel Xeon E3-1270 V2 @ 3.50GHz e 32 GB de memória RAM DDR3 1333MHz.

6.1. Tempos de execução da solução ótima

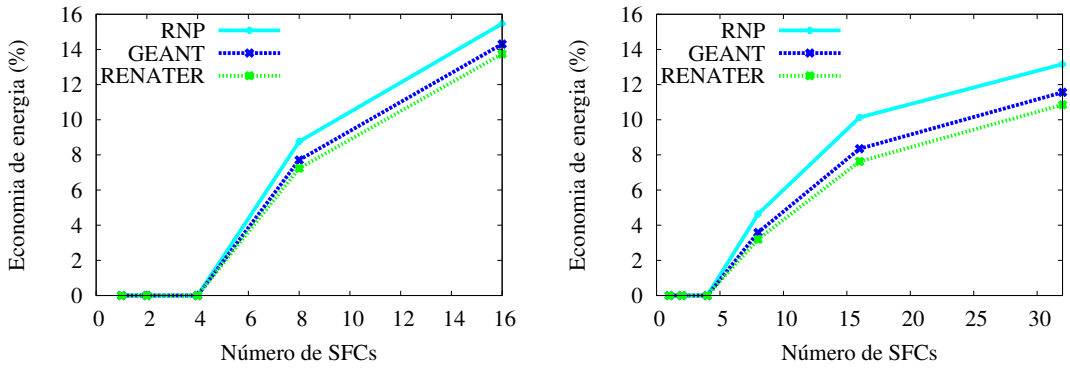
No problema de otimização, a quantidade de nós da topologia tem grande impacto na quantidade de variáveis do problema. As redes da RNP, GEANT e RENATER possuem, respectivamente, 28, 41 e 48 nós. A Tabela 2 mostra o tempo de execução de acordo com o número de SFCs solicitadas. A execução só é possível para até 4 SFCs. A partir de 8 SFCs, o CPLEX não alcança uma solução e consome toda a memória da máquina utilizada. Observa-se que, para apenas 4 SFCs, a solução ótima com a topologia da RENATER chega a quase um minuto de execução, devido às suas 49 mil variáveis e 64 mil restrições, justificando a proposta de uma heurística que reduza o tamanho problema. Por isso, apenas o experimento com SFCs mais curtas foi realizado para a solução ótima.

6.2. Resultados da TRELIS

Os tempos de execução da TRELIS, não apresentados em sua totalidade por questões de espaço, mostram que a heurística obtém soluções em menos tempo. Por exemplo, para 4 SFCs e servidores com capacidade $\mathcal{S}_i$ = 1000, todas as redes apresentam tempos

Tabela 2. Tempos de execução da solução ótima em segundos ($\mathcal{S}_i$ = 1000).

	Número de SFCs		
Topologia	**1**	**2**	**4**
RNP	0.8626	2.3002	7.9395
GEANT	4.6154	8.6275	43.0281
RENATER	5.6197	15.4545	59.6814



(a) Servidores com capacidade $\mathcal{S}_i = 1000$.

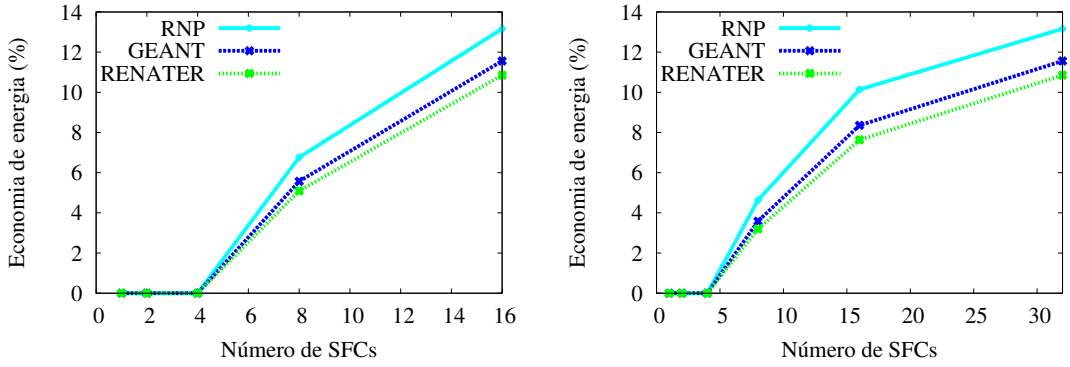
(b) Servidores com capacidade $\mathcal{S}_i = 4000$.

Figura 7. Economia de energia ao compartilhar VNFs por SFCs mais curtas.

de execução inferiores a 0.1 segundo. Além disso, é possível solucionar o problema para mais SFCs. Com servidores de capacidade 1000, são posicionadas até 16 SFCs em menos de 25 segundos. Para os de capacidade 4000, são posicionadas até 32 SFCs em menos de 3 segundos. Com mais capacidade, são necessários menos servidores na rede em linha, resolvendo o problema em menos tempo.

As Figuras 7(a) e 7(b) mostram a economia de energia com SFCs curtas devido ao compartilhamento de VNFs. A economia de energia é definida como a porcentagem de redução do consumo de energia no caso com compartilhamento em relação ao caso original, isto é sem compartilhamento de VNFs, como mostra a Equação 32. Com até 4 SFCs, não há economia de energia devido aos valores de capacidade das VNFs serem muito próximos à capacidade requerida pelas SFCs, de modo que a alocação até esse ponto é feita como no caso sem compartilhamento, já que os fragmentos de capacidade das VNFs ainda são capazes de atender a uma SFC. Como a rede e os servidores considerados são homogêneos, o resultado da heurística coincide com o da solução ótima sem perda de generalidade. Assim, os resultados ótimos foram omitidos. Além disso, com uma rede homogênea, o valor de ϵ utilizado foi E_{max} para que o Algoritmo 1 retorne assim que achar uma solução. Da mesma forma, como os servidores são homogêneos, a diferença no consumo de energia entre as três topologias se deve à quantidade de servidores disponíveis. Como a RENATER possui mais servidores que as outras duas, são mais servidores em *idle* consumindo energia, o que faz com que ela tenha valores menores de economia. A partir de 4 SFCs, como mostram as Figuras 7(a) e 7(b), há economia de energia com a heurística proposta. Além disso, quanto mais SFCs são requisitadas, mais energia é economizada, já que há mais VNFs sendo compartilhadas.

$$Economia = \frac{Consumo_{original} - Consumo_{compartilhamento}}{Consumo_{original}} \cdot 100\% \qquad (32)$$



(a) Servidores com capacidade $\mathcal{S}_i = 4000$. (b) Servidores com capacidade $\mathcal{S}_i = 8000$.

Figura 8. Economia de energia ao compartilhar VNFs por SFCs mais longas.

Para SFCs mais longas, cujos resultados são mostrados na Figura 8, o comportamento da economia de energia é semelhante. Isso mostra que o tamanho das SFCs não impacta significativamente na economia.

7. Conclusões e Direções Futuras

Este estudo verificou que é possível desenvolver mecanismos no contexto de NFV que permitem o posicionamento energeticamente eficiente de SFCs, através do compartilhamento de VNFs. Além disso, devido à característica virtual das funções de rede, as instâncias de *backup* posicionadas permanecem desligadas durante a operação normal da rede. Assim, é possível oferecer resiliência sem prejudicar a eficiência energética.

A inovação da abordagem é tratar do posicionamento ótimo de SFCs com resiliência a uma falha de servidor. Os resultados mostraram que é possível reduzir em até 15% o consumo de energia nos servidores ao compartilhar VNFs, além de suportar até uma falha silenciosa de servidores, evitando a interrupção do serviço fornecido pelas SFCs através de VNFs de *backup*. A escalabilidade limitada da solução ótima levou ao desenvolvimento da heurística TRELIS, que consiste em reduzir o número de servidores do problema. Como o número de servidores impacta na quantidade de todas as variáveis consideradas, reduzi-lo se mostrou eficiente para alocar mais SFCs e reduzir os tempos de execução foi capaz de alocar quatro vezes mais SFCs em um quarto do tempo. Assim, foi possível alocar até 32 SFCs. Esses números de SFCs posicionadas são razoáveis, pois representam o posicionamento de um lote de SFCs para uma execução da TRELIS.

Como direções futuras, destacam-se a extensão do problema para mais de uma falha de servidor e para falhas específicas de enlace, além de testar configurações heterogêneas de servidores. Além disso, o problema formulado considera requisições de SFCs em lote, alocadas em uma única execução. Na prática, requisições individuais podem ser realizadas sob demanda. O problema de otimização e a heurística podem ser adaptados para alocar as requisições individuais ou em grupos de SFCs, atualizando os parâmetros do problema à medida que seus recursos são consumidos pelas SFCs.

Referências

Bari, M. F., Chowdhury, S. R., Ahmed, R., Boutaba, R. e Duarte, O. C. M. B. (2016). Orchestrating virtualized network functions. *Aceito para publicação em IEEE Tran-*

sactions on Network and Service Management.

Clayman, S., Maini, E., Galis, A., Manzalini, A. e Mazzocca, N. (2014). The dynamic placement of virtual network functions. Em *2014 IEEE Network Operations and Management Symposium (NOMS)*, p. 1–9. IEEE.

Couto, R. S., Secci, S., Campista, M. E. M. e Costa, L. H. M. K. (2014). Network design requirements for disaster resilience in IaaS clouds. *IEEE Communications Magazine*, 52(10):52–58.

Couto, R. S., Secci, S., Campista, M. E. M. e Costa, L. H. M. K. (2015). Otimização do posicionamento de servidores físicos em centros de dados resilientes a desastres. Em *XXXIII Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos (SBRC)*, p. 417–430.

Han, B., Gopalakrishnan, V., Ji, L. e Lee, S. (2015). Network function virtualization: Challenges and opportunities for innovations. *IEEE Communications Magazine*, 53(2):90–97.

Herrera, J. G. e Botero, J.-F. (2016). Resource allocation in NFV: A comprehensive survey. *IEEE Transactions on Network and Service Management*, 13(3):518–532.

John, W., Pentikousis, K., Agapiou, G., Jacob, E., Kind, M., Manzalini, A., Risso, F., Staessens, D., Steinert, R. e Meirosu, C. (2013). Research directions in network service chaining. Em *IEEE SDN for Future Networks and Services (SDN4FNS)*, p. 1–7. IEEE.

Luizelli, M. C., Bays, L. R., Buriol, L. S., Barcellos, M. P. e Gaspary, L. P. (2015). Piecing together the nfv provisioning puzzle: Efficient placement and chaining of virtual network functions. Em *2015 IFIP/IEEE International Symposium on Integrated Network Management (IM)*, p. 98–106. IEEE.

Ma, W., Medina, C. e Pan, D. (2015). Traffic-aware placement of NFV middleboxes. Em *IEEE Global Communications Conference (GLOBECOM)*, p. 1–6.

Mehraghdam, S., Keller, M. e Karl, H. (2014). Specifying and placing chains of virtual network functions. Em *IEEE 3rd International Conference Cloud Networking (CloudNet)*, p. 7–13.

Mijumbi, R., Serrat, J., Gorricho, J.-L., Bouten, N., De Turck, F. e Boutaba, R. (2015). Network function virtualization: State-of-the-art and research challenges. *IEEE Communications Surveys & Tutorials*, 18(1):236–262.

Moens, H. e De Turck, F. (2014). Vnf-p: A model for efficient placement of virtualized network functions. Em *10th International Conference on Network and Service Management (CNSM) and Workshop*, p. 418–423.

Rahman, M. R. e Boutaba, R. (2013). SVNE: Survivable virtual network embedding algorithms for network virtualization. *IEEE Transactions on Network and Service Management*, 10(2):105–118.

Sahhaf, S., Tavernier, W., Rost, M., Schmid, S., Colle, D., Pickavet, M. e Demeester, P. (2015). Network service chaining with optimized network function embedding supporting service decompositions. *Computer Networks*, 93:492–505.

Tanenbaum, A. S. e Van Steen, M. (2007). *Distributed systems*. Prentice-Hall.