Uma Heurística GRASP-RVNS para o Problema de Mapeamento de Redes Virtuais

Samuel Moreira Abreu Araújo, Fernanda Sumika Hojo de Souza
Departamento de Ciência da Computação
Universidade Federal de São João del-Rei (UFSJ)
CEP 36301-360 – São João del-Rei – MG – Brasil
samucacesa@yahoo.com.br, fsumika@ufsj.edu.br

RESUMO

A virtualização de redes mostra-se fundamental no atual cenário de redes de computadores, e vem sendo aplicada como uma alternativa para o problema de "ossificação da internet". A tecnologia de virtualização é considerada um novo paradigma de redes que permite que várias redes virtuais compartilhem o mesmo substrato físico de forma independente. Neste contexto, surge o problema de Mapeamento de Redes Virtuais, que consiste em decidir onde os nós e enlaces virtuais serão hospedados na rede física respeitando suas capacidades. O problema pertence à classe NP-difícil e visando prover boas soluções em tempo viável, uma abordagem heurística baseada em GRASP e RVNS é proposta. Resultados experimentais demonstram a eficiência do método proposto em comparação com métodos da literatura.

PALAVRAS CHAVE. Redes virtuais. Metaheurística. Mapeamento.

Área Principal: Metaheuristicas, Otimização Combinatória, Simulação.

ABSTRACT

Network virtualization is crucial in the current scenario of computer networks, and it has been applied as an alternative for the "internet ossification"problem. Virtualization technology is considered a new paradigm of networks, allowing multiple virtual networks to share a common physical substrate independently. In this context, the Virtual Network Embedding problem arises, which consists in deciding where virtual nodes and links will be accommodated in the physical network while respecting its capabilities. The problem belongs to the NP-hard class, and aiming to provide good solutions in feasible time, a heuristic approach based on GRASP and RVNS is proposed. Experimental results demonstrate the efficiency of the proposed method in comparison to literature methods.

KEYWORDS. Virtual Networks. Metaheuristic. Embedding.

Main Area: Metaheuristics, Combinatorial Optimization, Simulation.

1. Introdução

Com o crescimento acelerado de aplicações para a internet observado atualmente, existe o uso cada vez mais intenso de novos modelos de funcionamento com protocolos diferentes implementados sobre o modelo TCP/IP, que é um modelo eficiente, mas simples quanto ao número de funcionalidades. Considerando que na concepção do protocolo TCP/IP não não foi prevista essa grande variedade de aplicações atuais, foi preciso propor alternativas a fim de prover novas funcionalidades inexistentes na proposta original. Entretanto essas adições de funcionalidades geram um *overhead* muito grande.

Segundo (Zhu and Ammar, 2006) a internet é um meio rígido onde não é possível fazer alterações no núcleo da rede; essa rigidez é conhecida hoje como "ossificação da internet". Como uma alternativa para este problema, foram propostos mecanismos de virtualização de redes, que permitem a implementação de novas funcionalidades, ao se criar uma visão lógica do hardware, de forma que múltiplas redes virtuais com características particulares possam atuar simultaneamente num mesmo substrato físico. Alguns dos principais benefícios do uso da virtualização incluem flexibilidade, escalabilidade, isolamento, redução de custos e segurança.

Com a popularização da "computação em nuvem", na qual é possível utilizar remotamente recursos de processamento, armazenamento e comunicação em servidores ligados à internet, a adoção de Infraestrutura como Serviço (IaaS - *Infrastructure as a Service*) mostra-se como uma base para a arquitetura da internet do futuro. Neste contexto surgem novos papéis para o atual provedor de serviço de internet (Zhu et al., 2008). São eles: os provedores de infraestrutura, que detém a rede física; os provedores de serviço, que oferecem serviços ponto-a-ponto, e por fim os provedores de conectividade, que são responsáveis por instanciar as requisições virtuais sobre as redes físicas.

Neste contexto, o grande desafio para os provedores de serviços é alocar de forma eficiente a infra-estrutura física existente, para atender as requisições virtuais dos clientes. Este problema é denominado Mapeamento de Redes Virtuais (Fischer et al., 2013), ou em inglês, *Virtual Network Embedding* (VNE), e consiste em determinar o mapeamento de requisições virtuais, compostas por nós (roteadores) e enlaces virtuais aninhados sobre nós e enlaces físicos, que compõem a rede física. Do ponto de vista da aplicação real, o problema tem natureza *online*, uma vez que as requisições não são conhecidas a *priori*, podendo chegar ao provedor na medida que os clientes decidem contratar tais serviços. Assim, a tomada de decisão de aceitação ou rejeição da requisição deve ser realizada de maneira automática, de acordo com a disponibilidade de recursos no momento. É importante ressaltar que mesmo conhecendo as requisições previamente (cenário *offline*), o problema de mapeamento ótimo ainda é considerado NP-difícil, sendo redutível ao problema de separação de multi-caminhos (Andersen, 2002).

Diversas variações são admitidas para o problema VNE. Em sua versão mais básica, são consideradas capacidades de processamento e largura de banda de transmissão, nos nós e enlaces físicos, respectivamente. De forma similar, as requisições de redes virtuais (VNs) possuem demandas de processamento associadas a cada nó virtual e de largura de banda de transmissão nos enlaces. Cada nó virtual de uma VN deve ser mapeado em um único nó do substrato físico que deve estar dentro do raio de mapeamento (D^V) do nó virtual em questão, sendo que um mesmo nó físico não deve hospedar mais de um nó de uma mesma VN. Por outro lado, um enlace virtual pode ser mapeado em mais de um enlace físico, isto é, pode ser mapeado em um caminho físico. Para tanto, é necessário que todos os enlaces pertencentes ao caminho possuam os recursos demandados.

A Figura 1 representa o mapeamento de duas redes virtuais no mesmo substrato físico. O substrato de rede é composto por cinco nós e seis enlaces com capacidades associadas respectivamente. A requisição VN_1 é composta por três nós virtuais e dois enlaces virtuais, com demandas de capacidade associadas, enquanto a requisição VN_2 é dada por três nós virtuais e três enlaces virtuais. Como pode ser observado, as duas requisições puderam ser mapeadas no mesmo substrato, sem extrapolar a capacidade do mesmo.

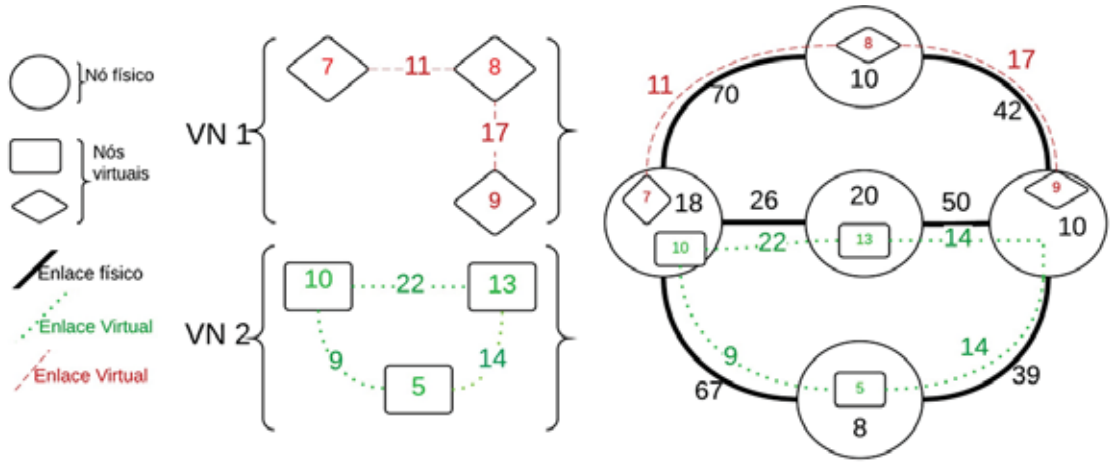


Figura 1: VNE com duas redes virtuais mapeadas em um único substrato físico.

Segundo pesquisas recentes na área (Cheng et al., 2011; Chowdhury et al., 2009), é observado na prática que em casos onde o mapeamento *online* é empregado, um número bastante elevado de rejeições é verificado na chegada de requisições de redes virtuais. Tais rejeições são atribuídas à escassez temporária de recursos, que não permite a alocação da nova demanda no substrato físico. Contudo, de acordo com análises mais detalhadas realizadas em (Luizelli et al., 2013), observou-se que as capacidades dos recursos de forma global não encontravam-se escassas, ao contrário, uma alta taxa de recursos ociosos podia ser observada, mas tais recursos estavam impossibilitados de serem utilizados devido à alocação prévia de alguns poucos recursos críticos já saturados, que inviabilizavam novos atendimentos. Diante disto, ressalta-se a importância de uma alocação adequada das demandas recentes, visando a prover maior disponibilidade para as demandas futuras.

O problema VNE pode ser formulado como um modelo de Programação Linear Inteira (PLI) e resolvido à otimalidade através de pacotes de otimização que implementam algoritmos *branch-and-bound* e *branch-and-cut*. Entretanto, é esperado que apenas instâncias com baixo número de nós e enlaces sejam passíveis de solução em tempo computacional aceitável. Considerando o caráter *online* do problema, soluções exatas não se mostram as mais adequadas, justificando assim a necessidade de abordagens heurísticas para serem empregadas na prática.

Este trabalho está organizado da seguinte forma. A seção 2 apresenta uma breve revisão de literatura, enquanto na seção 3 a descrição formal do problema é introduzida. Na seção 4, a abordagem metaheurística proposta é detalhada. A seção 5 apresenta as métricas avaliadas nos experimentos, enquanto na seção 6, os experimentos computacionais são apresentados e discutidos. Finalmente, a seção 7 conclui o trabalho.

2. Trabalhos Relacionados

Devido ao VNE estar ligado diretamente ao futuro da internet, ele tem sido alvo de interesse de diversos pesquisadores (Zhu and Ammar, 2006; Yu et al., 2008; Chowdhury et al., 2009; Inführ and Raidl, 2011). Segundo (Fischer et al., 2013), o VNE pode ainda ser classificado quanto à reconfiguração das VNs (estático vs. dinâmico), quanto à estratégia do algoritmo (centralizado vs. distribuído) e quanto à resiliência a falhas (conciso vs. redundante). Essa taxonomia categoriza os diversos trabalhos da literatura.

O modelo estático não faz a realocação de VNs, ou seja, uma vez alocadas elas perduram no mesmo local até o fim de sua vida útil. Já no modelo dinâmico, é permitido alterar a alocação de VNs previamente alocadas, ao custo da reconfiguração, mas com o ganho de melhorar a fragmentação do substrato, como estudado em (Fajjari et al., 2011) e (Sun et al., 2013).

Os modelos centralizado e distribuído estão relacionados ao funcionamento do algoritmo de mapeamento. No modelo centralizado, uma única entidade é responsável pelo mapeamento.

Esta visão global permite que soluções melhores sejam geradas, mas em contra partida possui um custo de processamento muito grande. O modelo distribuído divide a tarefa de mapeamento por várias entidades na rede, mas possui um *overhead* gerado pela troca de mensagens entre elas. A maior vantagem do modelo distribuído é a escalabilidade, em cenários de larga escala.

Falhas são comuns em cenários de redes, podendo ser tratadas a fim de prover resiliência quando um nó do substrato ou enlace tenha seu funcionamento comprometido (Rahman and Boutaba, 2013). No modelo consiso, os recursos alocados no substrato são exatamente aqueles requisitados pelas VNs, ou seja, nenhum recurso adicional é reservado. Por outro lado, modelo redundantes reservam recursos para eventuais falhas que possam ocorrer. Este modelo geralmente aumenta a rejeição de novos atendimentos em função da reserva de recursos.

A fim de tornar o cenário de redes mais real, algumas restrições adicionais podem ser consideradas no VNE. Em (Inführ and Raidl, 2011), os autores consideram restrições de localização geográfica para o mapeamento de nós, atraso máximo permitido na comunicação de dois nós virtuais e capacidade máxima de roteamento de um nó físico. As restrições de localização garantem por exemplo, que um serviço comercial de jogos *online* possua nós em diferentes capitais, e as restrições de atraso garantem uma comunicação de melhor qualidade em termos de atraso máximo.

Heurísticas foram propostas em (Zhu and Ammar, 2006), com o objetivo de alcançar balanceamento de carga e minimização de carga nos nós e enlaces da rede física. Os autores propõem duas abordagens: atribuição de redes virtuais sem reconfiguração e atribuição de redes virtuais com reconfiguração. Resultados computacionais demonstraram que o algoritmo proposto tem um desempenho melhor do que o algoritmo *least load* da literatura e para a abordagem com reconfiguração, foi demonstrado que a reconfiguração de apenas um subconjunto das redes virtuais que são mais críticas alcançam a maioria dos benefícios de reconfiguração dinâmica, mantendo um baixo custo.

A abordagem proposta por (Yu et al., 2008) é voltada ao projeto do substrato de rede, com a finalidade de facilitar a aplicação de algoritmos de mapeamento. As características adotadas pelos autores permitem que o substrato de rede possa atender um enlace virtual através de mais de um caminho físico, além de empregar migração de caminhos para otimizar periodicamente a utilização do substrato de rede. Simulações demonstraram que a divisão de rotas e migração de caminhos possibilitam o substrato de rede satisfazer um conjunto mais amplo de redes virtuais.

Em (Chowdhury et al., 2009), dois algoritmos foram propostos para o VNE com restrições de localização: o D-ViNE (*Deterministic VN Embedding*) e o R-ViNE (*Randomized VN Embedding*). O algoritmo D-ViNE, aborda o problema de forma determinística para o mapeamento dos nós escolhendo os nós a serem mapeados de acordo com a solução gerada por um modelo relaxado denominado VNE_LP_RELAX; já R-ViNE usa um mapeamento de nós aleatório. Para o mapeamento dos enlaces são usadas abordagens diferentes que são testadas nos dois algoritmos; a primeira abordagem utiliza um mapeamento por caminho mínimo, mas não leva em conta seu nível de saturação quanto ao fluxo presente nele. Em outra abordagem o fluxo pode ser partido em mais de um caminho, mas que se unem ao mesmo destino, adotando *multicommodity flows* (MCFs). Segundo os testes computacionais, o modelo que permite a divisão de fluxo em diferentes enlaces aumenta o nível de aceitação de requisições feitas, pois gera alternativas a gargalos de enlaces, mas deixa o modelo fora de contexto visto que tal ação não ocorre em um cenário real.

Abordagens metaheurísticas foram propostas em (Fajjari et al., 2011; Zhang et al., 2013). Um algoritmo baseado na metaheurística Colônia de Formigas é apresentado em (Fajjari et al., 2011), no qual um conjunto de formigas artificiais constroem soluções em paralelo e retornam a melhor solução encontrada após um número de iterações. Em (Zhang et al., 2013), um algoritmo de enxame de partículas é proposto, no qual cada partícula representa uma possível solução e melhora sua posição iterativamente através de modificações no mapeamento de nós e enlaces.

A abordagem estudada neste trabalho é classificada como estática para a reconfiguração das VNs, centralizada para a estratégia do algoritmo, concisa para a resiliência a falhas, *online* para

o conhecimento das VNs e sem permitir o mapeamento de um enlace virtual em múltiplos caminhos físicos. Devido à similaridade da abordagem escolhida com o trabalho proposto por (Chowdhury et al., 2009), este foi escolhido como base comparativa para os experimentos computacionais.

3. Definição Formal do Problema

O problema de mapeamento de redes virtuais pode ser modelado representando a rede física através de um grafo não direcionado ponderado $G^P = (N^P, E^P)$, onde N^P e E^P correspondem ao conjunto de nós e enlaces do substrato físico, respectivamente. Cada nó $n^P \in N^P$ do substrato está associado a uma capacidade de CPU $c(n^P)$ e sua localização $l(n^P)$ em um sistema de coordenadas geográficas. Cada enlace do substrato $e^P \in E^P$ está associado a uma capacidade de largura de banda $b(e^P)$.

De forma similar, uma requisição de rede virtual é modelada como um grafo $G^V = (N^V, E^V)$, com suas respectivas demandas de CPU, localização preferencial e largura de banda. Adicionalmente, um parâmetro D^V indica a distância máxima que um nó virtual pode ser mapeado a partir de sua localização preferencial. Seja $d(l(n^V), l(n^P))$ a função que retorna a distância euclidiana entre a localização geográfica de dois nós. É importante ressaltar que no cenário *online*, cada requisição virtual possui ainda um tempo de chegada e um tempo de vida.

Uma solução para o VNE consiste em encontrar um mapeamento $f : G^V \rightarrow G^P$, no qual $\forall\ n^V \in N^V\ \exists\ n^P \in N^P : c(n^V) \leq c(n^P), d(l(n^V), l(n^P)) \leq D^V$ e $\forall\ e^V \in E^V\ \exists\ e^P \in E^P : b(e^V) \leq b(e^P)$. Caso o mapeamento seja possível, considera-se que a requisição foi aceita; caso contrário, a requisição é rejeitada.

4. Heurística GRASP-RVNS

Devido ao caráter *online* do problema de mapeamento de redes virtuais, soluções exatas podem não ser aplicáveis na prática. Dessa forma, abordagens heurísticas são as mais indicadas em função de seu baixo custo computacional, ao preço da não garantia da otimalidade. Como apresentado na seção 2, diversos trabalhos propuseram heurísticas para o VNE, mas sua maioria baseado em critérios gulosos. A abordagem proposta baseada em GRASP (*Greedy Randomized Adaptative Search Procedure*) (Feo and Resende, 1995) e RVNS (*Reduced Variable Neighborhood Search*) (Mladenović and Hansen, 1997) visa combinar a criação de soluções diversificadas e de boa qualidade com um refinamento explorando diferentes vizinhanças desenvolvidas para o problema.

A metaheurística GRASP consiste em um algoritmo iterativo de multi-partida, dividido em duas fases: construção e busca local. A fase de construção, na qual uma solução é gerada elemento a elemento, é caracterizada por um componente probabilístico que define uma lista restrita de candidatos para a escolha do próximo elemento a ser inserido na solução. Além disso, esse processo de seleção é baseado em uma função adaptativa gulosa, que estima o benefício da seleção de cada um dos elementos. A fase de busca local visa alcançar melhores soluções em torno da solução construída. A melhor solução encontrada ao longo de todas as N iterações GRASP realizadas é retornada como resultado.

Algoritmo 1 GRASP-RVNS

```
1: f* = +∞;
2: for i ← 1 to N do
3:     s ← Construtivo(α)
4:     s' ← RVNS (s)
5:     if f(s') < (f*) then
6:         s* ← s'
7:         f* ← f(s')
8:     end if
9: end for
```

O RVNS é uma variação do VNS, onde a etapa de busca local não é realizada. A cada iteração do VNS, parte-se da solução corrente para obter uma solução vizinha aleatória dentro de uma vizinhança $k \in K$. Esta solução vizinha é então submetida a uma busca local. Se a solução gerada for melhor do que a solução corrente, a busca continua a partir dela, retornando-se à primeira estrutura de vizinhança. Caso contrário, continua-se a busca a partir da próxima estrutura de vizinhança $k + 1 \in K$. A condição de parada pode ser definida como um número máximo de iterações, ou tempo de CPU. A escolha do RVNS foi em função do cenário *online* do problema, onde o tempo de resposta do algoritmo deve ser o menor possível.

O Algoritmo 1 ilustra o funcionamento do método, que é empregado na resolução do problema de minimização do desbalanceamento de carga, conforme a função objetivo explicada na seção 5 .

4.1. Heurística Construtiva

A heurística construtiva proposta cria uma solução viável para o problema e é baseada na decomposição do problema em duas etapas:

- Mapeamento de nós: dada uma requisição G^V, deve-se encontrar um conjunto de nós físicos que possuam capacidade de CPU e localidade disponíveis para o mapeamento. Para cada nó virtual deve-se fazer o mapeamento sobre uma nó físico, sendo que um nó físico nunca deve hospedar mais de um nó virtual de uma mesma VN.
- Mapeamento de enlaces: dado um mapeamento de nós válido, para cada enlace virtual de G^V, deve-se encontrar um caminho no substrato que liga os nós físicos onde os nós do enlace virtual foram mapeados.

As duas etapas são tratadas sequencialmente, de forma que se o mapeamento de nós não obtiver sucesso, a VN é rejeitada sem que esta chegue na etapa de mapeamento de enlaces. O Algoritmo 2 ilustra o funcionamento das duas etapas da construção.

Algoritmo 2 Construtivo (α)

```
1:  for all n^V ∈ G^V do
2:     LRC ← calcula-LRC(n^V, α)
3:     n^P ← escolhe-elemento(LRC)
4:     ret ← mapeia(n^V, n^P)
5:     if ret = false then
6:        ERRO no mapeamento de nós
7:        return false
8:     end if
9:  end for
10: for all e^V ∈ G^V do
11:    n_1^P ← 1º nó físico do enlace e^V
12:    n_2^P ← 2º nó físico do enlace e^V
13:    ret ← Dijkstra(n_1^P,n_2^P)
14:    if ret = false then
15:       ERRO no mapeamento de enlaces
16:       return false
17:    end if
18: end for
19: return true
```

O mapeamento de nós virtuais é realizado sequencialmente, sendo que para cada nó virtual da VN que chega é gerada uma lista de nós candidatos (LC) a mapeamento válido. A partir

desta lista e do parâmetro α do método GRASP, é criada uma lista restrita de candidatos (LRC), com elementos que atendam os critérios:

- $\{n^P \in \text{LRC} \mid h(n^P) \geq h_{max} - \alpha(h_{max} - h_{min})\}$
- $h(n^P) = R(n^P) \sum_{e^P \in E^P(n^P)} R(e^P)$, onde $R(n^P)$ é a capacidade residual de CPU do nó físico n^P e $R(e^P)$ é a capacidade residual de cada enlace e^P ligado ao nó n^P.

A função $h(n^P)$ gera o valor de saturação do nó em termos da capacidade residual de CPU do nó junto a largura de banda residual dos enlaces que incidem sobre o próprio nó. Os parâmetros h_{max} e h_{min} são respectivamente os valores máximo e mínimo de $h(n^P)$ encontrados para cada nó físico dentro do D^V do nó virtual. Assim o mapeamento de nó virtual é priorizado nos nós físicos menos saturados.

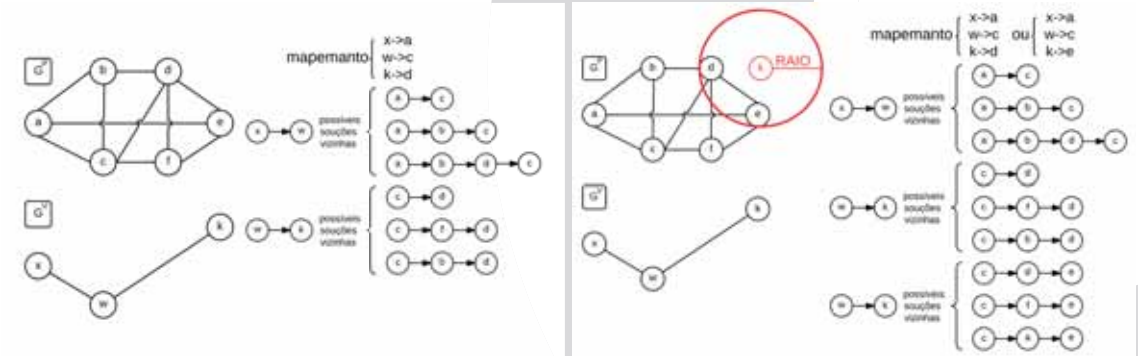
O mapeamento de enlaces virtuais é realizado iterativamente, tal que para cada enlace virtual um caminho na rede física é gerado. Foram utilizadas duas abordagens para a geração de rotas: caminho mínimo em número de saltos e caminho mínimo em utilização dos enlaces. Em ambos os casos, a rota é gerada através do algoritmo de Dijkstra, onde os pesos nas arestas são unitário e $b(e^P)/R(e^P)$, respectivamente. O algoritmo Dijkstra foi escolhido por sua eficiência considerando-se o tempo de processamento e qualidade da solução. Ele resolve o problema do menor caminho entre dois vértices, em grafos ponderados com pesos das arestas não negativos (Dijkstra, 1959).

4.2. Estruturas de Vizinhança

Para a aplicação da metaheurística RVNS, foram elaboradas duas vizinhanças baseadas na realocação dos mapeamentos de enlaces e nós virtuais. Estes movimentos ou trocas na solução corrente são realizados apenas dentro da região viável de soluções, se o movimento causar algum tipo de inviabilidade ele não é concluído, voltando-se a posição anterior a troca.

A vizinhança N^1 consiste no remapeamento de um enlace virtual. Dessa forma, o mapeamento do enlace é removido do substrato físico, restaurando os recursos consumidos pelo mesmo. Uma nova rota é gerada para mapear este enlace. Para evitar que a mesma rota removida seja gerada, um enlace físico aleatório pertencente à rota antiga é removido do grafo. Caso nenhuma rota seja gerada, o mapeamento antigo é restaurado. A Figura 2(a) ilustra esta estrutura de vizinhança.

A vizinhança N^2 explora o remapeamento de um nó e seus enlaces. Caso existam dentro do raio de mapeamento do nó virtual, alternativas de mapeamentos viáveis, o movimento de vizinhança também pode ocorrer na troca de nó, e consequentemente na troca dos enlaces virtuais ligados a este nó. Esta estrutura de vizinhança é ilustrada na Figura 2(b).



(a) Vizinhança N^1 com troca somente de enlace (b) Vizinhança N^2 com troca de nó e enlace

Figura 2: Estruturas de Vizinhança N^1 e N^2

5. Métricas de Avaliação

Para realizar a comparação entre a abordagem proposta e outras abordagens, é necessário estabelecer as métricas avaliadas nos experimentos computacionais. As métricas utilizadas neste trabalho são definidas a seguir, adotando-se as mesmas estabelecidas em (Chowdhury et al., 2009).

A primeira métrica avaliada e considerada a principal neste trabalho é a taxa de aceitação das requisições. Esta métrica é dada pela razão do número de VNs aceitas pelo número de VNs que chegaram na unidade de tempo em questão. Esta taxa pode ser expressa por:

$$\sum_{V \in VN^t} \frac{z^{G^V}}{|VN^t|}, \tag{1}$$

onde VN^t é o conjunto de VNs que chegaram até o tempo t e $z^{G^V} = 1$ indica que a requisição G^V foi atendida.

Duas métricas importantes do ponto de vista do provedor de serviços dizem respeito à receita gerada pela aceitação das VNs e pelo custo para realizar o mapeamento. Assim, enquanto uma VN encontra-se ativa, ela gera uma receita e um custo ao provedor de serviços, sendo que quando seu tempo de vida expira, ambos valores deixam de ser computados.

A receita é expressa em função dos recursos demandados pela própria VN:

$$\sum_{e^V \in E^V} b(e^V) + \sum_{n^V \in N^V} c(n^V) \tag{2}$$

O custo é expresso em função dos recursos gastos na infraestrutura física:

$$\sum_{e^V \in E^V} \sum_{e^P \in E^P} b(e^V) x_{e^P}^{e^V} + \sum_{n^V \in N^V} c(n^V), \tag{3}$$

onde $x_{e^P}^{e^V} = 1$ indica se o enlace virtual e^V utiliza o enlace físico e^P.

As duas últimas métricas avaliadas estão relacionadas ao balanceamento de carga na rede física. As taxas de utilização de nós e enlaces são importantes para detecção de possíveis gargalos e esgotamento de recursos em pontos da rede, que consequentemente levam ao aumento do número de rejeições das VNs.

A taxa média de utilização de nós é dada pela média das razões entre utilização do nó por capacidade do nó em um dado tempo t e pode ser expressa por:

$$\frac{1}{|N^P|} \sum_{n^P \in N^P} \frac{\sum_{n^V \in N^V} c(n^V) y_{n^P}^{n^V}}{c(n^P)}, \tag{4}$$

onde $y_{n^P}^{n^V} = 1$ indica se o nó virtual n^V está hospedado no nó físico n^P.

A taxa média de utilização de enlaces é dada pela média das razões entre utilização do enlace por capacidade do enlace em um dado tempo t, sendo expressa por:

$$\frac{1}{|E^P|} \sum_{e^P \in E^P} \frac{\sum_{e^V \in E^V} b(e^V) x_{e_P}^{e^V}}{b(e^P)}. \tag{5}$$

Finalmente, a função objetivo adotada na abordagem proposta foi definida visando o balanceamento de carga na rede, a fim de evitar gargalos e aumento no número de rejeições. Em função do número de rejeições devido à falha no mapeamento de enlaces ser maior do que devido à falha em nós, priorizou-se apenas minimizar a taxa de utilização dos enlaces. Assim, a função objetivo considerada pode ser dada por:

$$\sum_{e^P \in E^P} [T(e^P) * 100]^{T(e^P)}, \text{ onde } T(e^P) = \frac{\sum_{e^V \in E^V} b(e^V) x_{e^P}^{e^V}}{b(e^P)} \quad (6)$$

6. Experimentos Computacionais

Nesta seção são apresentados os resultados computacionais obtidos através da abordagem proposta e variações e um algoritmo da literatura. Os testes foram realizados em um computador Intel Core i5 3ª geração, com 8GB de RAM DDR3, utilizando o sistema operacional Ubuntu $14.04.2$ LTS. Os algoritmos foram testados através de 10 execuções e a dispersão dos dados apresentada com um intervalo de confiança de 95%.

6.1. Cenário de Simulação

As instâncias[1] utilizadas foram propostas por (Chowdhury et al., 2009). A topologia de rede do substrato foi gerada aleatoriamente com 50 vértices, usando a ferramenta GT-ITM em grid (25×25). Cada par de vértices do substrato foi conectado aleatoriamente com uma probabilidade de 0.5. Os recursos de CPU e de largura de banda dos nós e dos enlaces do substrato correspondem a números reais uniformemente distribuídos entre 50 e 100.

As requisições de VNs chegam segundo uma distribuição Poisson com uma taxa de 4 VNs a cada 100 unidades de tempo para a instância de maior porte. Para a instância de menor porte, a taxa é de 5 VNs a cada 1.000 unidades de tempo. Cada VN tem um tempo de vida com distribuição exponencial com média de $\mu = 1.000$ unidades de tempo. O número de vértices de uma VN é determinado aleatoriamente entre 2 e 10, por uma distribuição uniforme. O grau médio de conectividade de um VN foi fixado em 50%.

Os requisitos de CPU dos nós virtuais estão distribuídos uniformemente entre 0 e 20 e os requisitos de largura de banda dos enlaces virtuais entre 0 e 50. Nós virtuais também são localizados no grid (25×25). As simulações consideram 50.000 unidades de tempo nas operações do sistema.

6.2. Abordagens Avaliadas

Foram avaliadas dez variações da abordagem proposta e uma abordagem da literatura. O algoritmo GRASP-RVNS possui seis versões, de acordo com os valores do parâmetro α utilizado e o método empregado no mapeamento de enlaces. O valor do parâmetro α do método GRASP foi variado para $\alpha \in \{0.1, 0.3, 0.6\}$, denotados respectivamente por L1, L3 e L6. O método empregado no mapeamento de enlaces varia entre peso unitário ou fixo (PF) e peso baseado na utilização do enlace, ou peso variável (PV). O valor do parâmetro N foi definido com valor 4, uma vez que valores superiores não apresentaram melhoras nos resultados. As outras quatro variações não incluem o GRASP, mas uma construção gulosa baseada nas escolhas do primeiro nó disponível para mapeamento (PP) e nó com mais recursos sobrando (MR). Ambos utilizam o RVNS como refinamento e variação no método de mapeamento dos enlaces (PF ou PV). Finalmente, O algoritmo da literatura avaliado corresponde ao D-ViNE-LB (Chowdhury et al., 2009), que prioriza o balanceamento de carga, em sua abordagem que não permite separação em múltiplos caminhos. Como este algoritmo é determinístico, não é necessária mais do que uma execução do mesmo.

6.3. Análise de Desempenho

A Figura 3 apresenta a média do número de VNs aceitas ao final da simulação de 10 execuções de todas as abordagens consideradas. Dois cenários são apresentados; o primeiro de baixa densidade, com chegada de 250 VNs e o segundo de alta densidade, com chegada total de 2000 VNs. É possível observar que as abordagens GRASP-RVNS mostram-se superiores nos dois cenários avaliados. A abordagem D-ViNE-LB foi similar às variações gulosas avaliadas.

No primeiro cenário as variações PF e PV foram similares e o valor de $\alpha = 0.6$ obteve os melhores resultados. No segundo cenário, as variações PF mostraram-se superiores e o valor de

[1] http://www.mosharaf.com/ViNE-Yard.tar.gz/

α não obteve grande impacto. De forma geral, a variação GR_RVNS_L6_PF obteve os melhores resultados, demonstrando que uma LRC mais diversificada leva a soluções de boa qualidade na construção e que o peso fixo dos enlaces visando minimizar o número de saltos gera menos uso dos enlaces, melhorando o número de aceitações.

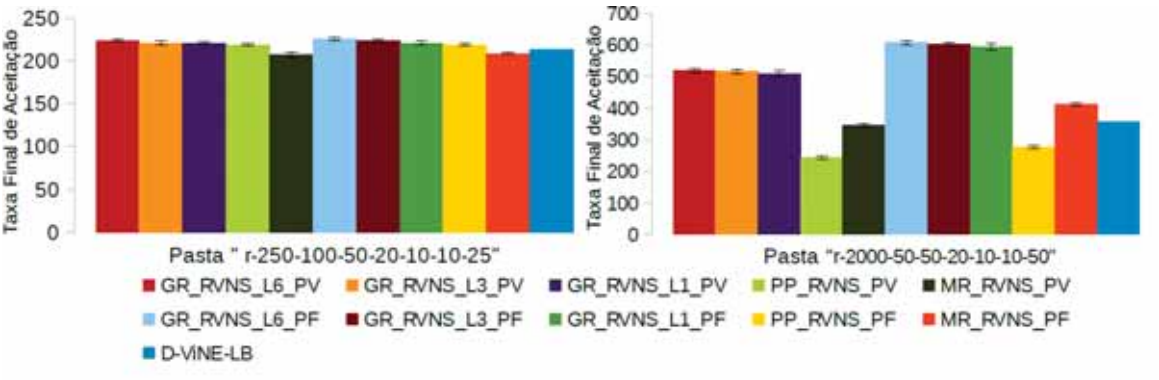


Figura 3: Número de VNs aceitas: (a) Chegada de 250 VNs, (b) Chegada de 2000 VNs

A baixa dispersão dos dados com intervalo de 95% de confiança demostra a robustez do algoritmo proposto, como pode ser observado na Figura 3.

As Figuras 4, 5, 6, 7 e 8 apresentam as métricas descritas na seção 5 para uma única execução dos métodos considerando o segundo cenário (de maior densidade). O eixo X apresenta o tempo de simulação entre 1.000 unidades de tempo até 50.000 unidades de tempo.

Analisando a taxa de aceitação ao longo do tempo (Figura 4), é possível observar que as variações GR_RVNS_L6_PF e GR_RVNS_L6_PV foram significativamente superiores às demais variações. O algoritmo D-ViNE-LB mostrou-se superior apenas à variação gulosa PP.

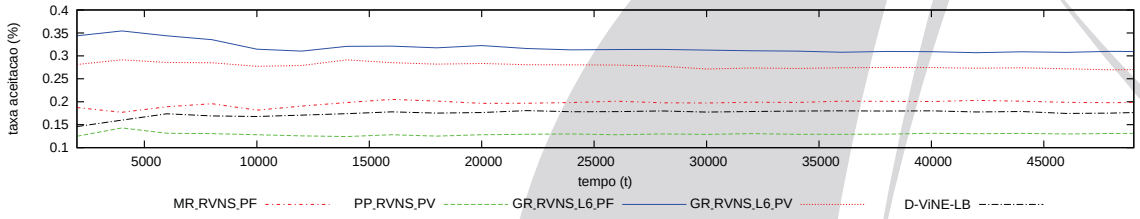


Figura 4: Taxa de aceitação de VNs (seção 5, equação 1)

A receita gerada pela aceitação das VNs pode ser observada na Figura 5, sendo que esta é diretamente relacionada com a taxa de aceitação das VNs. Assim, a maior receita é obtida pela variação GR_RVNS_L6_PF.

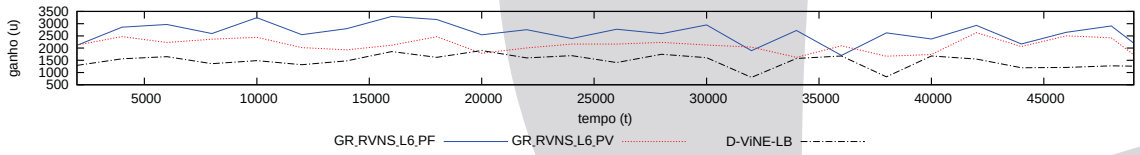


Figura 5: Receita gerada pelas VNs aceitas (seção 5, equação 2)

De acordo com a Figura 6, que ilustra o custo gerado para o mapeamento, as duas variações obtiveram um desempenho próximo. É possível notar que a variação GR_RVNS_L6_PF foi ligeiramente superior quanto a economia dada pelo custo de alocação, devido a esta priorizar

sempre a menor distancia em número de saltos entre dois nós físicos. Consequentemente, esta estratégia acarretará um custo menor. O algoritmo D-ViNE-LB apresenta um menor custo devido ao seu maior número de rejeições.

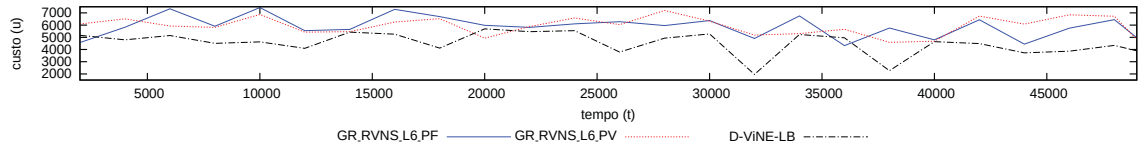


Figura 6: Custo gerado pelas VNs aceitas (seção 5, equação 3)

As taxas médias de utilização de nós (Figura 7) e de enlaces (Figura 8) também são diretamente relacionadas com a taxa de aceitação. O algoritmo D-ViNE-LB apresenta as menores taxas em função de sua maior rejeição das VNs. Em relação à taxa de utilização dos nós, embora similares, é possível observar que o GR_RVNS_L6_PF está ligeiramente acima do que o GR_RVNS_L6_PV. Por outro lado, quanto à taxa de utilização dos enlaces, o GR_RVNS_L6_PF mostra-se inferior, por utilizar menos enlaces para mapear os enlaces virtuais. Destaca-se ainda que a taxa de utilização dos enlaces é superior à taxa de utilização dos nós, indicando que o mapeamento de enlaces geralmente é o gargalo para a rejeição das VNs.

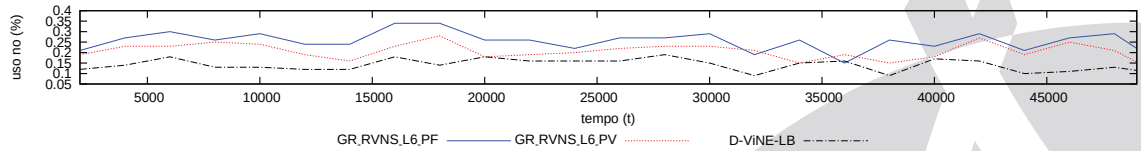


Figura 7: Taxa média de utilização de nós do substrato físico (seção 5, equação 4)

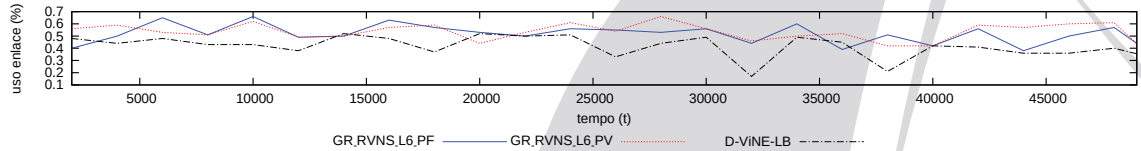


Figura 8: Taxa média de utilização de enlaces do substrato físico (seção 5, equação 5)

7. Conclusões

Virtualização de redes enfrenta diversos desafios a serem superados. Este estudo buscou colaborar com novas soluções para o problema de mapeamento de redes virtuais, visando melhorar os serviços para atendimento das demandas dos usuários. Foi proposta uma abordagem heurística baseada em GRASP e RVNS que monstrou-se mais eficiente no mapeamento de redes virtuais em comparação a versões gulosas e baseada em arredondamento como o D-ViNE-LB. De acordo com os experimentos computacionais realizados, o GR_RVNS_L6_PF apresentou as maiores taxas de aceitação e receita gerada pelas VNs mapeadas. Além disso, por priorizar o mapeamento de enlaces através de rotas com menos saltos, seu custo de mapeamento é menor do que a variação GR_RVNS_L6_PV, onde o custo das arestas é baseado na sua utilização.

Como trabalhos futuros, pretende-se avaliar o comportamento dos algoritmos quando as requisições não chegam ao provedor de serviços individualmente, mas quase ao mesmo tempo. Isso implica na execução do algoritmo para um lote de VNs a serem processadas juntas, como em janelas de tempo. Esse conhecimento prévio de um lote de VNs pode auxiliar na decisão do melhor mapeamento a ser realizado. Outra diretriz a ser investigada diz respeito a cenários de falhas,

inerentes a sistemas reais. A solução para um mapeamento resiliente a falhas exige a reserva de recursos adicionais, que deve ser otimizada, para evitar a rejeição de novas VNs em função deste mecanismo.

Agradecimentos

Os autores agradecem ao CNPq (446350/2014-1) e à FAPEMIG (PIBIC/FAPEMIG/UFSJ) pelo apoio no desenvolvimento deste trabalho.

Referências

Andersen, D. G. (2002). Theoretical approaches to node assignment. Available at: `http://www.cs.cmu.edu/~dga/papers/andersen-assign.ps`. Unpublished Manuscript.

Cheng, X., Su, S., Zhang, Z., Wang, H., Yang, F., Luo, Y., and Wang, J. (2011). Virtual network embedding through topology-aware node ranking. *SIGCOMM Comput. Commun. Rev.*, 41(2):38–47.

Chowdhury, N. M. M. K., Rahman, M. R., and Boutaba, R. (2009). Virtual network embedding with coordinated node and link mapping. In *INFOCOM*, pages 783–791. IEEE.

Dijkstra, E. W. (1959). A note on two problems in connexion with graphs. *Numerische Mathematik*, 1(1):269–271.

Fajjari, I., Aitsaadi, N., Pujolle, G., and Zimmermann, H. (2011a). Vne-ac: Virtual network embedding algorithm based on ant colony metaheuristic. In *ICC*, pages 1–6. IEEE.

Fajjari, I., Aitsaadi, N., Pujolle, G., and Zimmermann, H. (2011b). Vnr algorithm: A greedy approach for virtual networks reconfigurations. In *GLOBECOM*, pages 1–6. IEEE.

Feo, T. A. and Resende, M. G. (1995). Greedy randomized adaptive search procedure. *Journal of Global Optimization*, 6:109–133.

Fischer, A., Botero, J. F., Beck, M. T., and de Meer, H. (2013). Virtual network embedding: A survey. *IEEE Communications Surveys & Tutorials*, pages 71–97.

Inführ, J. and Raidl, G. R. (2011). Introducing the virtual network mapping problem with delay, routing and location constraints. In Pahl, J., Reiners, T., and Voß, S., editors, *INOC*, volume 6701 of *Lecture Notes in Computer Science*, pages 105–117. Springer.

Luizelli, M., Bays, L., Buriol, L., Barcellos, M., and Gaspary, L. (2013). Caracterizando o impacto de topologias no mapeamento de redes virtuais. In *Anais do XXXI Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos*, pages 75–88, Brasília, DF, Brasil.

Mladenović, N. and Hansen, P. (1997). Variable neighborhood search. *Computer and Operations Research*, 24(11):1097–1100.

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

Sun, G., Yu, H., Anand, V., and Li, L. (2013). A cost efficient framework and algorithm for embedding dynamic virtual network requests. *Future Gener. Comput. Syst.*, 29(5):1265–1277.

Yu, M., Yi, Y., Rexford, J., and Chiang, M. (2008). Rethinking virtual network embedding: substrate support for path splitting and migration. *Computer Communication Review*, 38(2):17–29.

Zhang, Z., Cheng, X., Su, S., Wang, Y., Shuang, K., and Luo, Y. (2013). A unified enhanced particle swarm optimization-based virtual network embedding algorithm. *International Journal of Communication Systems*, 26(8):1054–1073.

Zhu, Y. and Ammar, M. (2006). Algorithms for assigning substrate network resources to virtual network components. *INFOCOM 2006. 25th IEEE International Conference on Computer Communications*, pages 1–12.

Zhu, Y., Zhang-Shen, R., Rangarajan, S., and Rexford, J. (2008). Cabernet: Connectivity architecture for better network services. In *Proceedings of the 2008 ACM CoNEXT Conference*, CoNEXT '08, pages 64:1–64:6, New York, NY, USA. ACM.