PUCRS

FACULDADE DE INFORMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO
EM CIÊNCIA DA COMPUTAÇÃO

RÔMULO REIS DE OLIVEIRA

HEURÍSTICAS PARA MAPEAMENTO DE REDES VIRTUAIS DE SINCRONIA HÍBRIDA

Porto Alegre
2015

PÓS-GRADUAÇÃO - *STRICTO SENSU*

Pontifícia Universidade Católica
do Rio Grande do Sul

PONTIFÍCIA UNIVERSIDADE CATÓLICA DO RIO GRANDE DO SUL
FACULDADE DE INFORMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

HEURÍSTICAS PARA MAPEAMENTO DE REDES VIRTUAIS DE SINCRONIA HÍBRIDA

RÔMULO REIS DE OLIVEIRA

Dissertação apresentada como requisito parcial à obtenção do grau de Mestre em Ciência da Computação na Pontifícia Universidade Católica do Rio Grande do Sul.

Orientador: Prof. Fernando Luís Dotti

Porto Alegre
2015

Dados Internacionais de Catalogação na Publicação (CIP)

O48h Oliveira, Rômulo Reis de
Heurísticas para mapeamento de redes virtuais de sincronia híbrida / Rômulo Reis de Oliveira. – Porto Alegre, 2015.
61 p.

Diss. (Mestrado) – Fac. de Informática, PUCRS.
Orientador: Prof. Dr. Fernando Luís Dotti.

1. Informática. 2. Máquinas Virtuais. 3. Redes de Computadores. 4. Heurística (Informática). I. Dotti, Fernando Luís. II. Título.

CDD 004.65

Ficha Catalográfica elaborada pelo
Setor de Tratamento da Informação da BC-PUCRS

Pontifícia Universidade Católica do Rio Grande do Sul
FACULDADE DE INFORMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

TERMO DE APRESENTAÇÃO DE DISSERTAÇÃO DE MESTRADO

Dissertação intitulada "Heurísticas para Mapeamento de Redes Virtuais de Sincronia Híbrida" apresentada por Rômulo Reis de Oliveira como parte dos requisitos para obtenção do grau de Mestre em Ciência da Computação, aprovada em 24/04/2015 pela Comissão Examinadora:

Prof. Dr. Fernando Luís Dotti– Orientador — PPGCC/PUCRS

Prof. Dr. Tiago Coelho Ferreto– — PPGCC/PUCRS

Prof. Dr. Elias Procópio Duarte Júnior– — UFPR

Homologada em 10/09/2015, conforme Ata No. 016 pela Comissão Coordenadora.

Prof. Dr. Luiz Gustavo Leão Fernandes
Coordenador.

PUCRS
Campus Central
Av. Ipiranga, 6681 – P32– sala 507 – CEP: 90619-900
Fone: (51) 3320-3611 – Fax (51) 3320-3621
E-mail: ppgcc@pucrs.br
www.pucrs.br/facin/pos

HEURÍSTICAS PARA MAPEAMENTO DE REDES VIRTUAIS DE SINCRONIA HÍBRIDA

RESUMO

As redes virtuais de sincronia híbrida surgiram da combinação entre a virtualização de redes, a qual permite a coexistência de várias redes virtuais no mesmo substrato físico compartilhado fornecendo infraestrutura de maneira flexível e econômica, e arquitetura de redes com sincronia parcial, essa relevante em sistemas distribuídos para construir sistemas confiáveis. Um dos principais desafios em virtualização de redes é o mapeamento eficiente dos recursos virtuais na rede de substrato, pois é um problema de complexidade NP-Difícil. Ao considerar a sincronia dos recursos virtuais e físicos, se torna mais difícil efetuar esse mapeamento, inviabilizando o cálculo da solução ótima em ambientes reais. Sendo assim, abordagens heurísticas são necessárias para encontrar soluções semi-ótimas de maneira mais rápida. Neste trabalho são adaptadas quatro abordagens heurísticas para efetuar o mapeamento de redes virtuais de sincronia híbrida. Para avaliar o desempenho dessas heurísticas foram efetuados dois conjuntos de experimentos. No primeiro conjunto de experimentos são comparadas as soluções ótimas e as respectivas soluções semi-ótimas, os resultados indicaram que a eficiência das heurísticas são melhores quando as requisições de redes virtuais são menores, além disso houveram alguns custos de soluções semi-ótimas equivalentes ao custo de mapeamento da solução ótima. O segundo conjunto de experimento avalia o desempenho das heurísticas utilizando um substrato de rede mais próximo do contexto real e um maior número de requisições de redes virtuais. Os resultados desse segundo experimento demonstram que mesmo com um número maior de requisições de redes virtuais e um substrato maior, as soluções foram calculadas em tempo aceitável.

Palavras Chave: Virtualização de redes, redes virtuais, mapeamento de redes virtuais, redes virtuais de sincronia híbrida.

HEURISTICS FOR HYBRID SYNCHRONY VIRTUAL NETWORKS EMBEDDING

ABSTRACT

Hybrid synchrony virtual networks arose by combining network virtualization, which allows the co-existence of several virtual networks in the same shared physical substrate, providing infrastructure in a flexible and economic way, with partial synchrony network architecture, which is relevant in distributed systems in order to build reliable systems. One of the main challenges in network virtualization is the efficient mapping of virtual resources in the substrate network, since it is a NP-Hard complexity problem. When considering the synchrony of virtual and physical resources it becomes more difficult to map, making it unfeasible to calculate the optimal solution in real environments. Thus, heuristic approaches are necessary for finding semi-optimal solutions faster. In this work, four heuristics for mapping hybrid synchrony virtual networks are adapted. In order to evaluate these heuristics, two sets of experiments were executed. In the first set is compared the optimal solutions with their respective semi-optimal solutions, the results show the heuristics' efficiency are better when the virtual network requests are smaller, furthermore there were some semi-optimal solution mapping costs equivalent to the optimal solution mapping cost. The second set of experiments evaluates the heuristics performance using a physical substrate closer to real context and a larger number of virtual network requests. The results of this second set of experiments demonstrate that even with a larger number of virtual requests and a larger substrate, the solutions were computed in acceptable time.

Keywords: Network virtualization, Virtual network, virtual network embedding, Hybrid synchrony virtual networks.

LISTA DE FIGURAS

LISTA DE TABELAS

SUMÁRIO

1. INTRODUÇÃO

Virtualização de redes tem sido um tema bastante estudado nos últimos anos, pois permite prover infraestrutura de maneira econômica e flexível, viabilizando, por exemplo, o experimento de novos protocolos em larga escala, além disso, essa tecnologia também pode ser facilmente integrada nas arquiteturas de redes atuais.

Sincronia é um fator importante em sistemas distribuídos, pois ambientes onde o tempo máximo de processamento e entrega de mensagens são conhecidos facilitam o desenvolvimento de sistemas distribuídos. Porém estes ambientes são mais caros, visto que são compostos somente por componentes que podem fornecer tais garantias. Por isso é comum que os ambientes de redes não forneçam garantias temporais, entretanto, certos problemas são impossíveis de se resolver sem tais garantias na presença de falhas, como o problema do consenso distribuído [16]. Para contornar esse problema, o conceito de sincronia parcial foi proposto [7].

Uma variedade de aplicações com diferentes requisitos podem se aproveitar da virtualização de redes, visto que os componentes da infraestrutura oferecida pelos provedores de infraestrutura podem apresentar características ou propriedades distintas, proporcionando um ambiente com os requisitos necessários para a execução destas aplicações. Hasan et al. (2014) [20] propôs as redes virtuais de sincronia híbrida, esse tipo de rede virtual é composta por subconjuntos de nodos e enlaces que podem fornecer garantias temporais, permitindo que vários tipos de aplicações, que necessitam de sincronia em algum momento, sejam executados sobre essa infraestrutura.

O mapeamento dos recursos virtuais nos recursos físicos de maneira eficiente é um dos principais desafios em virtualização de redes. Em razão de se tratar de um problema de complexidade NP-Difícil, encontrar a solução ótima deste problema necessita de um tempo computacional significativo. Em redes virtuais de sincronia híbrida esse problema é dificultado visto que existem dois subconjuntos de componentes físicos e virtuais a serem considerados no mapeamento. Portanto abordagens heurísticas se fazem necessárias, para viabilizar um mapeamento semi-ótimo em tempo computacional aceitável. Por isso, neste trabalho são adaptadas quatro abordagens heurísticas para que estas resolvam o mapeamento de redes virtuais de sincronia híbrida.

A primeira heurística utiliza o algoritmo clássico *Worst Fit* como base para o mapeamento dos nodos, enquanto a segunda heurística utiliza o algoritmo clássico *Best Fit*. Já a terceira heurística, constrói uma árvore de alcançabilidade, a qual é utilizada para definir a ordem de mapeamento dos nodos virtuais e mapeá-los de maneira mais próximos fisicamente. A quarta heurística é uma melhoria incremental da terceira heurística, onde passa a ser considerado além de capacidade de CPU disponível nos nodos físicos, se há também capacidade de largura de banda disponível nos enlaces dos nodos físicos durante o mapeamento dos nodos virtuais.

Para validar as heurísticas propostas foram realizados dois conjuntos de experimentos. O primeiro conjunto de experimentos visa comprar a solução ótima e as soluções semi-ótimas encon-

tradas pelas heurísticas, já o segundo conjunto de experimentos visa comprar as soluções heurísticas em um cenário com um número maior de recursos virtuais e físicos.

O resto desta dissertação está dividida da seguinte maneira: O capítulo 2 apresenta uma base teórica sobre virtualização de redes e o estado da arte do mapeamento de redes virtuais, assim como redes virtuais de sincronia híbrida e o mapeamento de redes virtuais de sincronia híbrida. O capítulo 3 apresenta quatro abordagens heurísticas adaptadas para o mapeamento de redes virtuais de sincronia híbrida. O desempenho dessas heurísticas é analisado por meio dos experimentos apresentados no capítulo 4. Por fim são feitas as considerações finais da dissertação e discutido os trabalhos futuros no capítulo 5.

2. REFERENCIAL TEÓRICO

Este capítulo introduz o conceito de virtualização de redes e o motivo pelo qual esse tema tem atraído a atenção nos últimos anos (Seção 2.1). Na Seção 2.2 é discutido o mapeamento eficiente das redes virtuais no substrato físico, o qual é um dos principais desafios em virtualização de redes. Após isso, na Seção 2.3 são descritas as características e a implicação em se adotar uma arquitetura de sistemas distribuídos com garantia temporal, sem garantia temporal ou uma arquitetura híbrida. Também são apresentadas as redes virtuais de sincronia híbrida. Já na Seção 2.4 é apresentado o estado atual das pesquisas sobre o mapeamento desse tipo de rede virtual. Por último, na seção 2.5 são apresentados e discutidos os trabalhos relacionados.

2.1 Virtualização de Redes

O sucesso da Internet depende de um conjunto de tecnologias distintas de infraestrutura, estas necessárias para executar uma diversidade de protocolos e aplicações distribuídas. Embora nas últimas décadas a Internet tenha demonstrado sucesso ao suportar várias aplicações distribuídas e uma variedade de tecnologias de rede, seu crescimento e popularidade têm resultado no aumento do escopo do uso da Internet e também expôs algumas deficiências da arquitetura atual, como segurança, estabilidade e controle de roteamento e garantias de qualidade de serviço [30].

Alterações na arquitetura da Internet são fundamentais para a eliminação dessas deficiências e para permitir a criação de novos serviços. Porém a popularidade da Internet, juntamente com o fato da arquitetura atual ser comandada por um grande número de Provedores de Serviço de Internet (Internet Service Providers - ISPs), são os maiores impedimentos para a evolução da mesma. Qualquer alteração necessita de um acordo entre várias entidades, o que restringe as alterações na arquitetura da Internet a simples atualizações incrementais, sendo raras e onerosas as implementações de novas tecnologias de rede [3, 30].

Esse cenário de impedimento na evolução na arquitetura da Internet é denominado pela literatura como o problema da ossificação da Internet e segundo vários autores [3, 10, 14, 30] a virtualização de redes é a provável solução para esse problema. Ambientes de virtualização de redes permitem a coexistência de múltiplas redes virtuais na mesma rede de substrato (ou rede física), tornando possível, por exemplo, testes em larga escala de novas arquiteturas, protocolos e serviços [10].

Para permitir o fornecimento de maneira simples e flexível de redes virtuais sob demanda, é adotado o modelo de infraestrutura como um serviço (Infrastructure as a Service – IaaS). O que implica em um maior desacoplamento no ambiente de rede através da separação dos Provedores de Serviço de Internet em duas novas entidades: Provedor de Infraestrutura (Infrastructure Providers - InPs), que seria responsável por gerenciar a infraestrutura física, fazendo a alocação e desalocação dos recursos, além de garantir o isolamento das redes virtuais; e o Provedor de Serviço (Service

Providers - SPs), que teria como função criar redes virtuais sobre recursos físicos oferecidos por múltiplos provedores de infraestrutura e oferecer serviços de rede fim-a-fim [3, 14].

Tendo como base o modelo de IaaS para redes virtuais, Schaffrath et al. (2009) [28] separou o gerenciamento das regras de negócio, por meio da identificação de três papeis principais na camada de Provedor de Serviço (Figura 2.1): Provedor de Rede Virtual (Virtual Network Provider – VNP), responsável por organizar os recursos virtuais de um ou mais provedores de infraestrutura em uma topologia virtual; Operador de Rede Virtual (Virtual Network Operator – VNO), responsável pela instalação e gerenciamento das redes virtuais conforme as necessidades do Provedor de Serviço, este responsável por oferecer serviços por meio das redes virtuais.

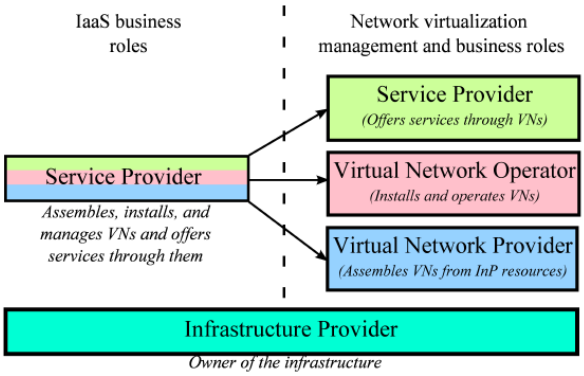

Figura 2.1 – Futuro modelo de negócio da Internet. Fonte: Fischer et al. (2013)[15]

Uma rede virtual é basicamente um conjunto de nodos virtuais conectados por um conjunto de enlaces virtuais que formam uma topologia virtual. Cada nodo virtual é hospedado em um nodo físico, já cada enlace virtual pode ser hospedado por um conjunto de recursos físicos. Além disso, um ambiente de virtualização de redes deve permitir a coexistência de múltiplas redes virtuais no mesmo substrato físico, além de permitir que um recurso físico hospede vários recursos virtuais. O ambiente também deve ser recursivo, ou seja, permitir que uma rede virtual seja oferecida pelo InP aos SPs, sendo herdadas pelas redes virtuais filhas as características e limitações da infraestrutura virtual pai [10].

A figura 2.2 demonstra um ambiente de virtualização de redes, onde há dois Provedores de Infraestrutura, os quais são vistos de maneira distinta pelos Provedores de Serviço. Também ocorre recursão, pois uma das redes virtuais é mapeada sobre outra rede virtual.

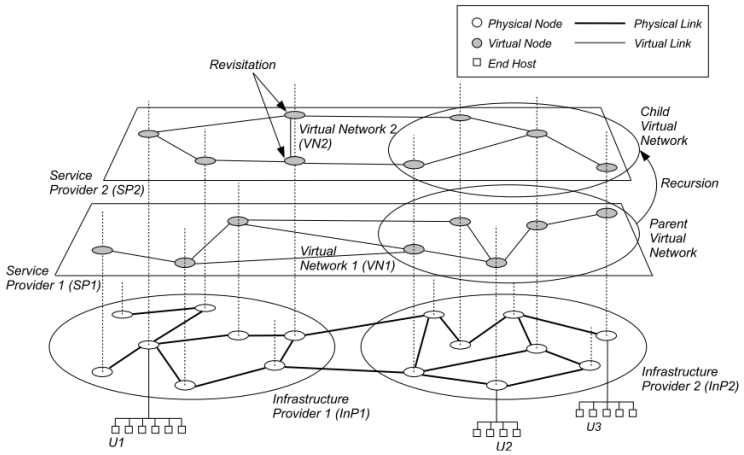

Figura 2.2 – Ambiente de virtualização de rede. Fonte: Chowdhury e Boutaba (2009)[10]

2.2 Mapeamento de Redes Virtuais

A alocação de maneira eficiente dos recursos físicos para suportar múltiplas requisições de criação de redes virtuais é importante para permitir o aumento do número de redes virtuais coexistentes e consequentemente baixar o custo de operação. Porém, segundo Haider et al. (2009) [18] o principal desafio em virtualização de rede é justamente o mapeamento eficiente de recursos virtuais em uma rede de substrato. Isso ocorre quando o provedor de infraestrutura, ao receber uma requisição de rede virtual, aloca os recursos físicos de maneira mais eficiente possível, respeitando as propriedades ou requisitos de cada nodo e enlace virtual e a capacidade disponível dos recursos físicos. Para que isso seja possível, o provedor de infraestrutura deve conhecer a topologia da infraestrutura que ele gerencia, assim como o estado atual de cada recurso, como a capacidade atual disponível de largura de banda de cada enlace ou memória disponível em cada nodo.

O mapeamento de redes virtuais pode ser dividido em dois subproblemas: mapeamento dos nodos virtuais, onde os nodos devem ser alocados nos nodos físicos e mapeamento dos enlaces virtuais, onde os enlaces virtuais que conectam esses nodos virtuais devem ser mapeados em caminhos para conectar os correspondentes nodos no substrato [15].

A alocação ótima de recursos com requisitos nos nodos e enlaces virtuais, é um problema de complexidade NP-Difícil [2], podendo ser representado através de modelos de programação inteira mista (Mixed Integer Programming - MIP). Segundo Chowdhury e Boutaba (2009) [10] em ambientes de virtualização de rede, há quatro propriedades chaves, descritas a seguir, que dificultam ainda mais o mapeamento de redes virtuais:

- **Diversas topologias:** As redes virtuais podem ter qualquer topologia.
- **Limitações dos recursos:** Cada rede virtual possui requisitos específicos de recurso que o mapeamento deve respeitar e satisfazer.
- **Requisições online:** Em um ambiente ideal, o algoritmo de mapeamento deve ser capaz de resolver as requisições conforme elas chegam.
- **Controle de admissão:** Tendo como princípio que os recursos do substrato físico são limitados, é necessário rejeitar ou colocar em uma fila as requisições que não podem ser satisfeitas em determinado momento, também há possibilidade de bloquear a chegada de novas requisições quando não houver recursos suficientes, além disso, é necessário fazer a reserva dos recursos assim que uma requisição for aceita.

Alguns modelos para encontrar soluções ótimas de mapeamento foram propostas [20, 21], porém, encontrar uma solução ótima de um problema NP-Difícil exige um tempo computacional significativo, então, para diminuir o tempo necessário para se obter a solução, uma ou mais dessas propriedades são tratadas de maneira flexível ou simplesmente ignoradas. Como, por exemplo, a

adoção de um cenário com um único provedor de infraestrutura, ou o tratamento de requisições do provedor de serviço de maneira *offline*, ou seja, todas as requisições já são conhecidas.

Mesmo tratando de maneira flexível uma ou mais das quatro propriedades chaves, a complexidade deste problema continua sendo NP-Difícil. Por este motivo, diversos algoritmos que adotam técnicas de heurísticas têm sido propostos [20, 21, 24, 34, 11, 6, 36], cada um para um contexto específico. A heurística procura encontrar uma solução válida para um problema a um custo computacional aceitável e não uma solução ótima.

De acordo com Fischer et al. (2013) [15], todas as abordagens de mapeamento de redes virtuais podem ser categorizadas como:

- **Dinâmico ou Estático:** Uma abordagem dinâmica permite que ocorram alterações na rede de substrato, como inclusão de novos nodos e enlaces ou aumento na capacidade dos recursos. Também são permitidas alterações nas redes virtuais, as quais podem ainda ser parcialmente ou completamente migradas ou excluídas. Já uma abordagem estática não permite qualquer tipo de alteração na rede de substrato ou nas redes virtuais.

- **Centralizado ou Distribuído:** Em uma abordagem centralizada há somente uma entidade responsável por efetuar o mapeamento. Isso exige que a entidade tenha conhecimento global de todo o sistema, ou seja, conhecer o estado atual de cada recurso oferecido pelos múltiplos servidores de infraestrutura. Isso pode ser visto como uma vantagem, porém, ter o processamento de mapeamento centralizado em uma única entidade pode ser perigoso em caso de falhas. Além disso, essa abordagem tende a ter maiores problemas de escalabilidade. Por outro lado, em uma abordagem distribuída há várias entidades responsáveis pelo mapeamento. Nesse caso, não necessariamente todas as entidades devem conhecer todas as redes de substrato ofertadas pelos provedores de infraestrutura.

- **Conciso ou Redundante:** Um mapeamento conciso tem como foco utilizar o mínimo de recursos necessários para efetuar o mapeamento. Isso permite que um número maior de redes virtuais sejam alocadas na rede de substrato. Entretanto, não há garantias de recuperação quando ocorre falha na rede de substrato. Ao contrário de um mapeamento redundante, que utiliza mais recursos que o necessário para efetuar um mapeamento. Os recursos redundantes podem ser utilizados quando ocorra alguma falha na rede de substrato. Mapear um enlace virtual em múltiplos caminhos ou ter instâncias duplicadas de nodos virtuais são exemplos de cenários com uma abordagem redundante.

Cada categoria é independente uma da outra, um algoritmo de mapeamento pode ser estático, centralizado e conciso, por exemplo.

2.3 Redes Virtuais de Sincronia Híbrida

Um sistema distribuído é composto por um conjunto de processos autônomos, estes, conectados por meio de uma rede, a qual permite que os processos coordenem suas atividades e compartilhem recursos do sistema, de maneira com que o usuário perceba um único sistema [29].

A arquitetura adotada por um sistema distribuído é fortemente vinculada a hipóteses sobre o ambiente em que o mesmo será implementado. Um fator importante levado em consideração é a sincronia fornecida pela infraestrutura física. Isso ocorre porque em ambientes síncronos há limites superiores conhecidos de tempo para execução de processos e entregas de mensagens. Já em ambientes assíncronos, o tempo de execução de processos e entrega de mensagens não são conhecidos.

Ao considerar um ambiente assíncrono, o desenvolvimento do sistema distribuído se torna mais complexo. Embora um sistema desenvolvido para ser executado em um ambiente assíncrono seja compatível com um ambiente síncrono, o contrário não ocorre. Em um ambiente assíncrono também não é possível distinguir um processo falho de um processo lento [7]. Além disso, Fischer et al. (1985) [8] provou a impossibilidade do consenso distribuído na presença de falha. Tal fato impede a garantia de que alguns problemas sejam resolvidos em ambientes assíncronos, como os problemas dos generais bizantinos, da consistência interativa ou do *multicast* totalmente ordenado e confiável.

Um ambiente síncrono, embora facilite o desenvolvimento dos algoritmos e seja possível distinguir falha no processo de processo lento, é composto somente com componentes que fornecem garantias temporais, o que torna a implementação desse tipo de ambiente muito caro ou até mesmo inviável. A necessidade de os processos compartilharem recursos de processamento, canais de comunicação e acesso à rede também contribuem para que frequentemente os ambientes de sistemas distribuídos sejam assíncronos.

Uma maneira de resolver o problema da impossibilidade do consenso distribuído é adotar uma arquitetura parcialmente síncrona, que além de possibilitar a garantia da solução do problema de consenso, [13], tem uma infraestrutura menos custosa que um ambiente totalmente síncrono. Isso ocorre porque partes do ambiente se comportam de forma síncrona e partes de forma assíncrona.

Cristian e Fetzer (1999) [12] propõem um modelo com sincronia temporizada, onde os componentes do ambiente de rede podem ter seus comportamentos alternados entre síncrono e assíncrono, permitindo então a variação da sincronia com o tempo. Já Veríssimo (2006) [31] apresenta o modelo *wormhole*, que explora a dimensão espacial para fornecer sincronia híbrida.

Devido à facilidade de alocar recursos, redes virtuais é uma maneira flexível para oferecer ambientes de sincronia parcial. Sendo que esse tipo de ambiente desejado para possibilitar a construção de algoritmos distribuídos eficientes e com menor grau de complexidade, se comparado a ambientes totalmente assíncronos, e economicamente mais atrativos, se comparados a ambientes totalmente síncronos. [20]

Vista a relevância deste cenário, Hasan et al. (2014)[20] apresentou a combinação de sincronia híbrida com redes virtuais, que resulta em uma abstração de um novo tipo de rede virtual denominado de redes virtuais de sincronia híbrida (Hybrid Synchrony Virtual Network - HSVN), onde, as redes virtuais possuem subconjuntos de nodos e enlaces que têm limites máximos conhecidos de tempo para processo e comunicação.

Hasan et al. (2014) [20] ainda ilustrou um detector perfeito de falha $\mathcal{P}$ implementado em uma rede virtual de sincronia híbrida, o que reforça a relevância deste tipo de rede virtual. Detectores de falhas são relevantes na construção de sistemas distribuídos confiáveis, visto que o problema o problema de consenso distribuído pode ser resolvido com eles. Além disso, é possível adaptar uma abordagem de detecção de falhas para solucionar outros problemas relevantes, como eleição [25] ou detecção de predicados [17].

A principal atração das redes virtuais de sincronia híbrida está em ter acesso à uma infraestrutura com sincronia parcial de forma econômica. Como as redes virtuais são alocadas sob demanda e substrato físico é compartilhado, vários usuários podem usufruir do mesmo recurso físico, além disso, cada usuário pode definir as configurações da rede virtual conforme suas necessidades.

2.4 Mapeamento de Redes Virtuais de Sincronia Híbrida

Efetuar o mapeamento de redes virtuais de sincronia híbrida é mais difícil por considerar a sincronia dos recursos, em relação ao mapeamento de redes virtuais mais comuns na literatura, que consideram somente largura de banda e capacidade de CPU. Hasan et al. (2014) [20] propôs um modelo matemático, no formato de *Mixed Integer Program* (MIP), para encontrar a solução ótima de mapeamento. Esse modelo respeita os requisitos de sincronia e ao mesmo tempo minimiza a utilização de recursos síncronos, considerando também o CPU e largura de banda dos componentes das redes virtuais e do substrato físico.

A entrada de dados do modelo acima citado é composta por um conjunto de redes virtuais de sincronia híbrida e uma rede de substrato com sincronia parcial. Tanto os componentes das redes virtuais quanto do substrato físico têm como parâmetro a sincronia, podendo estes serem síncronos ou assíncronos. Os nodos virtuais têm uma demanda de CPU e os enlaces virtuais têm uma demanda de largura de banda. Os nodos físicos têm uma capacidade de CPU, enquanto os enlaces físicos têm uma capacidade de largura de banda.

O mapeamento é feito de maneira *offline* e a solução ótima deste modelo satisfaz os requisitos dos recursos virtuais sem exceder os limites dos recursos físicos. Além disso, os enlaces virtuais podem ser mapeados tanto para um único enlace físico, quanto para um caminho físico, composto por um conjunto de enlaces físicos, sendo que os nodos entre os enlaces desse caminho são intermediadores, funcionando como roteadores. Também não é permitido mapear mais de um nodo virtual em um nodo físico de uma mesma rede virtual. Essa regra limita o número de nodos virtuais da mesma rede afetados por falhas nos nodos físicos. Os resultados obtidos por esse modelo também

indicam que ele tende a efetuar o mapeamento de forma balanceada, evitando o sobrecarregamento dos recursos físicos.

Já em Hasan et al. (2014) [19] foi investigado um cenário onde o substrato físico não tem distinção de sincronia entre os recursos, sendo possível ser definido quais recursos deverão ser síncronos de maneira mais eficiente para efetuar o mapeamento de um conjunto de redes virtuais. Para isso, foi proposto um modelo matemático no formato de *Mixed Integer Program* que tem como parâmetro de entrada um substrato físico sem atributo de sincronia definido e um conjunto de redes virtuais. A solução encontrada informa o atributo de sincronia dos recursos físicos e o mapeamento dos elementos das redes virtuais. Assim, é possível configurar subconjuntos de recursos do substrato físico para serem síncronos, em um número mínimo suficiente para que seja possível efetuar o mapeamento. Esta nova proposta procura eliminar a existência de recursos físicos síncronos inutilizados, mas pressupõe-se que os recursos podem ser dinamicamente configurados para se comportarem de forma síncrona ou assíncrona.

Como citado anteriormente, o problema para o mapeamento de redes virtuais é de complexidade NP-Difícil, exigindo um tempo computacional significativo para encontrar a solução, o que é inviável em ambientes reais. Sendo assim, abordagens heurísticas são necessárias para que o mapeamento seja feito de maneira mais rápida, mesmo que a solução encontrada não seja ótima. Por esse motivo foram implementadas e adaptadas quatro abordagens heurísticas para efetuar o mapeamento de redes virtuais de sincronia híbrida.

2.5 Trabalhos relacionados

Mapeamentos de redes virtuais em redes físicas são extensamente estudados e uma visão geral destes pode ser encontrada em [15]. Os mapeamentos variam em sua definição conforme os objetivos das redes e do mapeamento em questão. Por exemplo, [36, 34, 24] consideram questões topológicas no mapeamento, já [33, 27, 8] consideram resiliência e [4] segurança. Em algumas propostas, também foram assumidas algumas hipóteses sobre o ambiente para minimizar o espaço do problema. [36, 24], por exemplo, tratam as requisições de redes virtuais de maneira *offline*, onde todas as requisições de redes virtuais já são previamente conhecidas.

Embora haja várias propostas para calcular a solução ótima do mapeamento de redes virtuais em um substrato físico [34, 11, 1, 9], o uso de heurísticas é importante neste contexto devido à complexidade computacional para calcular a solução ótima. Neste trabalho, considera-se trabalhos diretamente relacionados às propostas de mapeamento de redes virtuais que consideram tempo de atraso, como em [35], ou que a rede virtual seja composta por nodos ou enlaces de diferentes tipos, assim como o substrato pode ser composto por diferentes subconjuntos de componentes, como por exemplo, em [4, 5].

Como citado anteriormente, neste contexto específico enquadra-se [35] que propôs um algoritmo heurístico para o mapeamento de redes virtuais orientadas a serviços de *multicast* sujeitos

a atrasos e variações de atraso. Nessa arquitetura há um recurso emissor e vários recursos receptores. Jogos online e teleconferência são exemplos de aplicações que se beneficiariam desse tipo de rede virtual, pois nessas aplicações é esperado que as mensagens sejam entregues dentro de um período máximo de tempo e que a diferença entre os tempos de entrega nos múltiplos receptores seja mínima.

Zhang et al. (2010), em [35], apresentam um modelo matemático cujo o objetivo é minimizar o uso de recursos físicos necessários para efetuar o mapeamento. Também considerando a capacidade e demanda de CPU, largura de banda dos recursos físicos e virtuais, além do atraso entre o emissor e cada receptor, assim como a variação de atraso entre os receptores. Posteriormente, propõem uma heurística com base no modelo matemático, tendo como entrada uma requisição de criação de rede virtual com os seguintes parâmetros: recurso emissor, lista dos recursos receptores, capacidade de CPU dos nodos, capacidade de largura de banda dos enlaces, atraso máximo entre o emissor e receptor e a variação de atraso máximo permitido entre os receptores. Primeiramente são calculados k caminhos mais curtos entre o emissor e cada receptor, depois são filtrados a fim de localizar o melhor resultado.

Bays et al. (2012), em [4], também apresentaram um trabalho relacionado, pois o mesmo permite que as redes virtuais demandem um nível de segurança, que é fornecido pelo substrato físico. Há três níveis de segurança: *(i) end-to-end cryptography* requer que nodos virtuais específicos sejam mapeados em nodos físicos que tenham suporte à criptografia e descriptografia, *ii) point-to-point cryptography* necessita que alguns enlaces virtuais sejam mapeados em um caminho físico composto somente por nodos intermediadores (roteadores) que sejam capazes de criptografar e descriptografar, permitindo que a aplicação pode encriptar o cabeçalho do protocolo. *iii) non-overlapping networks* requer que todos os nodos e enlaces não compartilhem o substrato físico alocado com outras redes virtuais. Assim, existem nodos e enlaces virtuais com demandas específicas de segurança que são providas por elementos também específicos da rede de substrato.

Bays et al. (2012), em [4], também apresentaram um trabalho relacionado, pois o mesmo permite que as redes virtuais demandem um nível de segurança, que é fornecido pelo substrato físico. Há três níveis de segurança: *(i) end-to-end cryptography* requer que nodos virtuais específicos sejam mapeados em nodos físicos que tenham suporte à criptografia e descriptografia, *ii) point-to-point cryptography* necessita que alguns enlaces virtuais sejam mapeados em um caminho físico composto somente por nodos intermediadores (roteadores) que sejam capazes de criptografar e descriptografar, permitindo que a aplicação pode encriptar o cabeçalho do protocolo. *iii) non-overlapping networks* requer que todos os nodos e enlaces não compartilhem o substrato físico alocado com outras redes virtuais. Assim, existem nodos e enlaces virtuais com demandas específicas de segurança que são providas por elementos também específicos da rede de substrato.

Ainda em [4] é proposto um modelo matemático no formato de *Integer Linear Programming* (ILP) que satisfaz os requisitos de segurança e ao mesmo tempo otimiza a utilização dos recursos em um tempo aceitável, tratando as requisições de maneira *offline*. Assim como em [35], os componentes virtuais possuem demandas de largura de banda e capacidade de CPU e os componentes físicos possuem uma capacidade máxima de CPU e largura de banda.

Em outro trabalho [5] é proposta uma abordagem heurística que trata requisições de maneira *online* para as redes virtuais propostas em [4]. Essa heurística constrói uma solução inicial pelo mapeamento semi-aleatório dos nodos virtuais nos nodos físicos e depois ocorre o mapeamento dos enlaces virtuais por meio do algoritmo de caminho mais curto de Dijkstra. Então essa solução inicial é avaliada pelo custo do mapeamento dos enlaces, após isso, uma busca interativa por soluções é iniciada, sendo essa busca concluída quando um número máximo de interações ocorrer ou o custo de mapeamento dos enlaces for melhor que o um valor predeterminado. A cada interação um nodo virtual é movido para outro nodo físico e todos os enlaces deste nodo virtual são realojados.

Esse trabalho se difere dos mencionados principalmente pelo pressuposto adotado, já que *i)* Um subconjunto dos recursos do substrato físico oferecem garantia temporal, assim como um subconjunto de recursos das redes virtuais tem como requisito garantia temporal. Diferentemente de [35], onde todos os recursos físicos e virtuais têm garantias ou demandas por garantias temporais; *ii)* O espaço do problema não é minimizado como em [36, 24], pois as requisições são tratadas de maneira *online*, assim como os recursos são considerados limitados, necessitando de controle de admissão; *iii)* A terceira heurística considera a topologia da rede virtual na etapa do mapeamento dos nodos por meio da construção de uma árvore de alcançabilidade, diferentemente de [36, 34, 24]; *vi)* A quarta heurística proposta, além de considerar a topologia durante o mapeamento dos nodos virtuais, também considera se os nodos físicos têm enlaces com largura de banda suficiente para mapear os enlaces do nodo virtual.

3. HEURÍSTICAS PARA MAPEAMENTO DE RVSH

Este capítulo apresenta na seção 3.1 as notações e definições de variáveis utilizadas em quatro algoritmos heurísticos para o mapeamento de RVSH, os quais são apresentados nas seções 3.2, 3.3, 3.4 e 3.5 e possuem três características em comum, sendo eles centralizados, somente uma entidade executa o processo de mapeamento, estáticos, não há alteração no mapeamento de uma rede virtual após seu mapeamento e concisos, busca alocar somente os recursos necessários, sem redundâncias [15]. As quatro propostas têm como principais objetivos minimizar a utilização dos enlaces e a alocação de recursos síncronos. Também deve ser minimizado o tempo de execução necessário para se obter uma solução em relação ao modelo matemático proposto em Hasan et al. (2014) [20]. Este capítulo apresenta na seção 3.1 as notações e definições de variáveis utilizadas em quatro algoritmos heurísticos para o mapeamento de RVSH, os quais são apresentados nas seções 3.2, 3.3, 3.4 e 3.5 e possuem três características em comum, sendo eles centralizados, somente uma entidade executa o processo de mapeamento, estáticos, não há alteração no mapeamento de uma rede virtual após seu mapeamento e concisos, busca alocar somente os recursos necessários, sem redundâncias [15]. As quatro propostas têm como principais objetivos minimizar a utilização dos enlaces e a alocação de recursos síncronos. Também deve ser minimizado o tempo de execução necessário para se obter uma solução em relação ao modelo matemático proposto em Hasan et al. (2014) [20].

Todos algoritmos apresentados neste capítulo possuem duas etapas, sendo que na primeira etapa é efetuado o mapeamento dos nodos virtuais sobre os nodos físicos. Para um nodo físico ser elegível como hospedeiro, o mesmo deve ter capacidade de CPU disponível suficiente para hospedar um nodo virtual, o qual tem uma demanda de CPU, além de respeitar os atributos de sincronia. Um nodo virtual síncrono pode ser mapeado somente em nodos físicos síncronos, já os nodos virtuais assíncronos, podem ser mapeados tanto em nodos físicos síncronos, quanto assíncronos. Também não é permitido que mais de um nodo virtual da mesma rede virtual seja mapeado para o mesmo nodo físico, pois isso aumentaria o impacto caso ocorresse um erro ou falha no nodo físico.

Após todos os nodos virtuais de uma rede virtual serem mapeados com sucesso, inicia-se a segunda etapa, onde ocorre o mapeamento dos enlaces virtuais sobre os enlaces físicos. Um enlace físico é considerado elegível para hospedar um enlace virtual somente quando o enlace físico tem uma capacidade de largura de banda disponível suficiente para hospedar o enlace virtual, o qual tem uma demanda de largura de banda. Além disso, a sincronia do enlace virtual deve ser respeitada, pois, os enlaces virtuais síncronos só podem ser mapeados em enlaces físicos síncronos, já enlaces virtuais assíncronos, podem ser mapeados tanto para enlaces físicos síncronos, quanto assíncronos. Cada enlace virtual pode ser mapeado tanto para um único enlace físico, quanto para um conjunto de enlaces físicos. Como citado anteriormente, um dos objetivos é economizar recursos síncronos (nodos e enlaces), por serem mais caros, sendo assim, recursos virtuais assíncronos somente são mapeados em recursos físicos síncronos, caso não haja recursos físicos assíncronos elegíveis disponíveis.

3.1 Definição de estruturas de representação

O substrato físico pode ser representado por um grafo não direcional $G_s = (N_s, L_s)$. Esse grafo é composto por um conjunto de nodos físicos N_s conectados por meio de um conjunto de enlaces físicos L_s. Cada $n_s \in N_s$ tem uma capacidade limitada de CPU, assim como, cada $l_s \in L_s$ tem uma capacidade limitada de largura de banda. A sincronia dos recursos é obtida pela função booleana Sync(n_s) para os nodos e Sync(l_s) para os links, ambas funções retornam *TRUE* quando os recursos forem síncronos e *FALSE* para recursos assíncronos.

Uma requisição de criação de redes virtuais de sincronia híbrida, HSVNRequest, é composta por um conjunto de redes virtuais, onde cada rede virtual de sincronia híbrida pode ser representada por um grafo não direcional $G_v^k = (N_v^k, L_v^k)$, onde, N_v^k é um conjunto de nodos virtuais conectados por enlaces virtuais contidos no conjunto L_v^k. O subscrito v indica se tratar de uma rede virtual e o sobrescrito k é o número identificador da rede virtual.

Cada $n_v^k \in N_v^k$ tem um custo de CPU para ser mapeado, assim como, cada $l_v^k \in L_v^k$ tem um custo de largura de banda para ser mapeado.A sincronia dos recursos virtuais pode ser verificada pela função booleana Sync(n_v^k) para os nodos virtuais e Sync(l_v^k) para os enlaces virtuais, ambas funções retornam *TRUE* para recursos síncronos e *FALSE* para recursos assíncronos.

3.2 Heurística HSVN-WorstFit

Em outros trabalhos que exploram heurísticas no mapeamento de redes virtuais, como em Liu et al. (2011) [23] e Li et al. (2013) [22], é adotado um algoritmo base para que seja possível efetuar comparações com os algoritmos propostos. Esse algoritmo base é composto por duas etapas. Na primeira etapa ocorre o mapeamento dos nodos virtuais utilizando o algoritmo clássico *Worst Fit*, sendo considerado o custo de CPU do nodo virtual. Ou seja, cada nodo virtual de uma rede virtual, é mapeado para o nodo físico com maior capacidade de CPU disponível.

Na segunda etapa é efetuado o mapeamento dos enlaces virtuais utilizando o algoritmo do caminho mais curto de Dijkstra, alocando cada enlace virtual em um caminho composto por enlaces físicos com largura de banda disponível suficiente para hospedar o enlace virtual. O algoritmo heurístico *Worst Fit* foi adaptado para suportar RVSH. Uma das principais características desse algoritmo é efetuar um mapeamento balanceado, por meio de uma distribuição mais igualitária dos nodos virtuais, evitando o sobrecarregamento dos nodos físicos, quando estes têm a mesma capacidade de CPU disponível inicialmente.

No algoritmo base HSVN-WorstFit (*Algorithm* 1), é dado como parâmetro de entrada um substrato físico com sincronia parcial $G_s = (N_s, L_s)$ e um conjunto de requisições de criação de redes virtuais de sincronia híbrida HSVNRequest. Para cada $G_v^k \in HSVNRequest$, é chamado o procedimento MapNodesWorstFit(N_v^k, G_s), demonstrado no algoritmo MapNodesWorstFit (*Algorithm* 2).

Algorithm 1 HSVN-WorstFit($G_s, HSVNRequest$)

1: **for all** $G_v^k = (N_v^k, L_v^k) \in HSVNRequest$ **do**
2: **if MapNodesWorstFit(**N_v^k, G_s**) ==** $rejected$ **then**
3: **FreePhysicalResources(**G_v^k, G_s**)**;
4: **else**
5: **if MapLinks(**L_v^k, G_s**) ==** $rejected$ **then**
6: **FreePhysicalResources(**G_v^k, G_s**)**;
7: **end for**

Nessa primeira etapa de mapeamento, os nodos virtuais serão alocados de acordo com sua ordem de chegada, seguindo o modelo de fila (First In, First Out - FIFO). O algoritmo MapNodesWorstFit recebe como entrada um conjunto de nodos virtuais de sincronia híbrida e o grafo $G_s = (N_s, L_s)$ que representa o substrato físico. Para cada $n_v^k \in N_v^k$, é verificada a sincronia do mesmo. Caso o nodo virtual seja síncrono, é chamado o procedimento FindSyncWorstFit(n_k^v, G_s), o qual retornará o nodo físico com maior capacidade de CPU disponível, que não tenha hospedado nenhum nodo virtual da mesma rede virtual e que tenha capacidade de CPU suficiente para hospedar n_v^k. Quando for possível localizar um recurso físico elegível, é efetuada sua alocação por meio do procedimento AllocateNode($n_k^v, physicalNode$), o qual é responsável também por subtrair o custo de n_v^k da capacidade de CPU do nodo hospedeiro. Caso não haja um nodo físico que se encaixe nesses requisitos, será retornado a mensagem *"rejected"*.

Algorithm 2 MapNodesWorstFit(N_v^k, G_s)

1: **for all** $n_v^k \in N_v^k$ **do**
2: $physicalNode \leftarrow null$;
3: **if Sync(**n_v^k**) ==** $TRUE$ **then**
4: $physicalNode \leftarrow$ **FindSyncWorstFit(**n_v^k, G_s**)**;
5: **else**
6: $physicalNode \leftarrow$ **FindAsyncWorstFit(**n_v^k, G_s**)**;
7: **if** $physicalNode == null$ **then**
8: $physicalNode \leftarrow$ **FindSyncWorstFit(**n_v^k, G_s**)**;
9: **if** $physicalNode == null$ **then**
10: **return** $rejected$;
11: **else**
12: **AllocateNode(**$n_v^k, physicalNode$**)**;
13: **end for**
14: **return** $accepted$;

Quando o nodo virtual for assíncrono, é chamado o procedimento FindAsyncWorstFit(n_k^v, G_s), esse procedimento é semelhante ao FindSyncWorstFit(n_k^v, G_s), porém, nesse caso, será procurado um recurso físico assíncrono. Caso não haja um recurso físico elegível, é chamado o procedimento FindSyncWorstFit(n_k^v, G_s), pois como citado anteriormente, um nodo virtual síncrono pode ser alocado em um nodo físico síncrono. Sendo assim, o procedimento FindAsyncWorstFit(n_k^v, G_s) é chamado primeiro a fim de economizar recurso síncrono. Caso ambos procedimentos não encontrem um candidato elegível, será retornado a mensagem *"rejected"*.

Quando o algoritmo HSVN-WorstFit recebe a mensagem *"rejected"* do procedimento MapNodesWorstFit(N_v^k, G_s), significa que não foi possível efetuar o mapeamento de todos os nodos da rede virtual, sendo assim, é chamado o procedimento FreePhysicalResources(G_k^v, G_s). Esse procedimento desaloca os recursos que foram reservados para o mapeamento dessa rede virtual. Entretanto, quando a mensagem recebida for diferente de *"rejected"*, será iniciado a segunda etapa do mapeamento pela chamada do algoritmo MapLinks (*Algorithm* 3).

MapLinks tem como função mapear os enlaces da rede virtual, esse algoritmo tem como entrada o substrato físico, $G_s = (N_s, L_s)$ e um conjunto de enlaces virtuais, L_v^k, com seus respectivos nodos virtuais de origem e destino e os hospedeiros desses nodos virtuais. Para cada enlace virtual é verificado se o mesmo é síncrono ou assíncrono por meio da função Sync(l_v^k). Quando o enlace for síncrono, será chamada a função FindSyncPath(l_v^k, G_s). Essa retorna o menor caminho físico síncrono, que liga os nodos físicos que hospedam os nodos virtuais de origem e destino desse enlace, com largura de banda disponível suficiente para hospedar o enlace virtual.

Algorithm 3 MapLinks(L_v^k, G_s)

```
1: for all l_v ∈ L_v^k do
2:     path ← null;
3:     if Sync(l_v^k) == TRUE then
4:         path ← FindSyncPath(l_v^k, G_s);
5:     else
6:         path ← FindPath(l_v^k, G_s);
7:     if path == null then
8:         return rejected;
9:     else
10:        AllocatePath(l_v^k, path, G_s);
11: end for
12: return accepted;
```

Quando o enlace for assíncrono, será chamada a função FindPath(l_v^k, G_s), essa função se difere da função FindSyncPath(l_v^k, G_s), pois, encontra o caminho mais curto, podendo ser esse caminho composto por enlaces físicos tanto síncronos quando assíncronos. Entretanto, enlaces síncronos possuem um valor maior, sendo assim, o caminho retornado por essa função só vai conter enlaces físicos síncronos caso não haja um caminho assíncrono, ou o caminho assíncrono seja composto por um número de enlaces assíncrono significativamente maior que de enlaces síncronos, tornando-se mais caro.

Não há como definir o quanto mais caro é uma ambiente síncrono em relação a um ambiente assíncrono. Os valores de um ambiente síncrono e assíncrono são variáveis, dependendo do valor do mercado e de outras configurações do ambiente. Nesse caso, foi adotado como hipótese que os enlaces síncronos são dez vezes mais caros que os enlaces assíncronos, porém esse valor é variável e pode ser alterado.

Após ser localizado um caminho válido, o mesmo será mapeado, sendo seu custo subtraído dos enlaces físicos hospedeiros através da função AllocatePath($l_v^k, path, G_s$). Caso não haja um caminho válido, l_v^k não será mapeado e será retornado a mensagem *"rejected"*.

Quando o algoritmo HSVN-WorstFit recebe a mensagem *"rejected"* do procedimento MapLinks(L_s^k, G_s), significa que não foi possível efetuar o mapeamento de todos os links da rede virtual, sendo assim, é chamado o procedimento FreePhysicalResources(G_k^v, G_s), esse procedimento, como descrito anteriormente, desaloca os recursos que foram reservados para o mapeamento dessa rede virtual. Quando a mensagem recebida for diferente de *"rejected"* significa que a rede virtual foi mapeada com sucesso.

Para facilitar o entendimento do comportamento do algoritmo HSVN-WorstFit, é ilustrado o mapeamento das redes virtuais 1 e 2 em uma rede de substrato, estes apresentados na Figura 3.1. Os nodos são representados por eclipses identificados por uma letra maiúscula. O valor dentro dos nodos representa a demanda de CPU, no caso das redes virtuais, ou a capacidade de CPU disponível dos nodos do substrato físico. Os enlaces são identificados por letras minúsculas, sendo que, os enlaces virtuais tem uma demanda de largura de banda fixado em 10 e os enlaces físicos tem uma capacidade máxima de largura de banda disponível inicialmente fixada em 30.

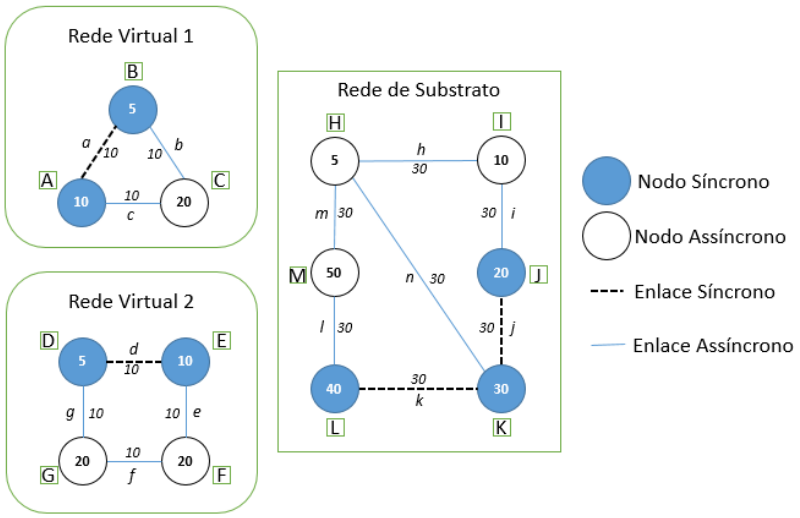

Figura 3.1 – Redes virtuais e Rede de substrato

Como citado anteriormente, o mapeamento é efetuado seguindo o modelo de fila. Sendo assim, a Rede Virtual 1 será mapeada primeiramente, como pode ser visualizado na Etapa 1 da Figura 3.2. Nessa primeira etapa, foi mapeado, nesta ordem, o nodo síncrono A no nodo físico síncrono L, pois este possuía a maior capacidade de CPU disponível. Assim como, o nodo síncrono B é mapeado no nodo síncrono K e o nodo virtual assíncrono C é mapeado no nodo físico assíncrono N.

As letras identificadores dos nodos e enlaces da rede de substrato foram omitidos na Figura 3.2. Nesta Figura são apresentados somente os identificadores dos componentes virtuais mapeados na rede de substrato, a capacidade de CPU disponível nos nodos físicos e a capacidade de largura de banda disponível nos enlaces físicos.

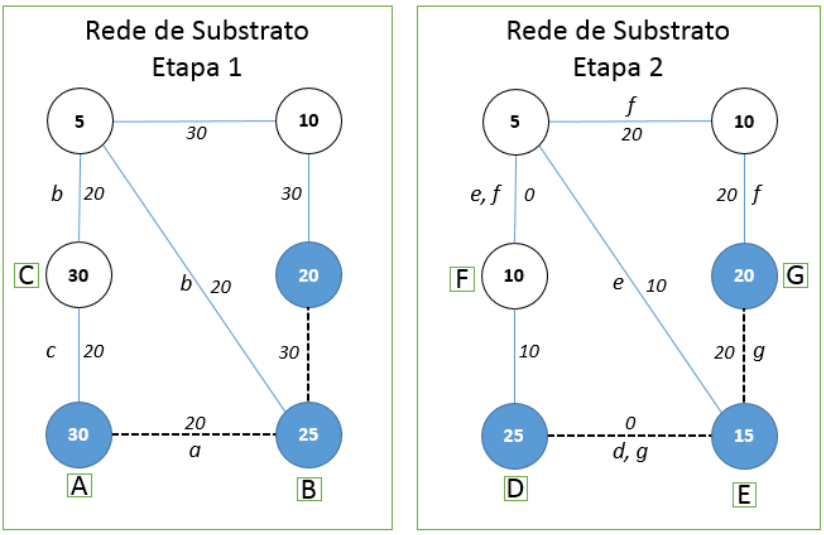

Figura 3.2 – Etapas do mapeamento utilizando HSVN-WorstFit

Após o mapeamento de todos os nodos com sucesso, inicia-se o mapeamento dos enlaces virtuais. O enlace virtual síncrono *a* é mapeado no enlace físico síncrono *k*, por ser o caminho mais curto entre os nodos físicos hospedeiros dos nodos virtuais A e B. Já o enlace virtual assíncrono *b* foi mapeado nos enlaces físicos assíncronos *m* e *n*. Para encerrar o mapeamento da Rede Virtual 1, o enlace virtual *c* é mapeado no enlace físico *l*, também por ser o caminho mais curto.

Após o mapeamento da Rede Virtual 1, inicia-se o mapeamento da Rede Virtual 2, ilustrado na Etapa 2 da Figura 3.2. Primeiramente é efetuado o mapeamento do nodo virtual síncrono D no nodo físico síncrono L, visto que o mesmo tinha a maior capacidade de CPU disponível. O mesmo ocorre com o nodo virtual síncrono E, mapeado para o nodo físico síncrono K e com o nodo virtual assíncrono F, mapeado no nodo físico assíncrono M. Já o último nodo virtual a ser mapeado, o nodo assíncrono G, foi mapeado no nodo físico síncrono J, pois não havia um nodo físico assíncrono com capacidade de CPU disponível suficiente para hospedar o nodo G. Embora o nodo físico síncrono L tivesse a maior capacidade de CPU disponível, o mesmo já estava a hospedar o nodo virtual D, pertencente a mesma rede virtual, por isso ele não foi considerado elegível para hospedar o nodo virtual G.

Após todos os nodos da Rede Virtual 2 serem mapeados, inicia-se o mapeamento dos enlaces virtuais. O enlace virtual síncrono *d* é mapeado no enlace síncrono *k*, visto ser o caminho mais curto entre os hospedeiros dos nodos virtuais D e E. O enlace virtual assíncrono *e* foi mapeado para os enlaces físicos assíncronos *m* e *n*, por ser o caminho mais curto composto somente por enlaces assíncronos. Assim como, o enlace virtual assíncrono *f* foi mapeado para os enlaces físicos assíncronos *h*, *i* e *m*. já o enlace virtual assíncrono *g* foi mapeado nos enlaces físicos síncronos *j* e *k*, sendo esse o único caminho possível, visto que o enlace físico *m* estava com sua capacidade de largura de banda esgotada.

3.3 Heurística HSVN-BestFit

O segundo algoritmo heurístico é uma adaptação do algoritmo *Best Fit* para efetuar o mapeamento dos nodos virtuais. Nesse caso, o algoritmo *Best Fit* mapeia cada nodo virtual de uma rede virtual no nodo físico com a menor valor de CPU disponível, mas suficiente para hospedar o nodo virtual. Um efeito da abordagem *Best Fit* nesse cenário é a concentração dos nodos virtuais alocados em menos nodos físicos o que favorece, por exemplo, a economia de energia em *data centers*. De qualquer forma, esse trabalho não tem como foco a economia de energia, sendo isso só um exemplo de cenário possível.

No algoritmo heurístico HSVN-BestFit (*Algorithm* 4), foi trocada a chamada da função MapNodesWorstFit(N_v^k, G_s), linha 2 do algoritmo HSVN-WorstFit, para MapNodesBestFit(N_v^k, G_s) (*Algorithm* 5), linha 2 do algoritmo HSVN-BestFit. Esse procedimento é similar ao MapNodesWorst-Fit (*Algorithm* 2), porém utiliza a função FindSyncBestFit(n_v^k, G_s) (linhas 4 e 8) para buscar um nodo físico síncrono com a menor capacidade disponível, mas suficiente para efetuar o mapeamento. Também é utilizado a função FindAsyncBestFit(n_v^k, G_s) (linha 6) para buscar um nodo físico assíncrono com a menor capacidade disponível, mas suficiente para efetuar o mapeamento. Já o mapeamento dos enlaces utiliza o o algoritmo MapLinks (*Algorithm* 3) já descrito anteriormente.

Algorithm 4 HSVN-BestFit($G_s, HSVNRequest$)

```
1: for all G_v^k = (N_v^k, L_v^k) ∈ HSVNRequest do
2:     if MapNodesBestFit(N_v^k, G_s) == rejected then
3:         FreePhysicalResources(G_v^k, G_s);
4:     else
5:         if MapLinks(L_v^k, G_s) == rejected then
6:             FreePhysicalResources(G_v^k, G_s);
7: end for
```

Algorithm 5 MapNodesBestFit(N_v^k, G_s)

```
1: for all n_v^k ∈ N_v^k do
2:     physicalNode ← null;
3:     if Sync(n_v^k) == TRUE then
4:         physicalNode ← FindSyncBestFit(n_v^k, G_s);
5:     else
6:         physicalNode ← FindAsyncBestFit(n_v^k, G_s);
7:         if physicalNode == null then
8:             physicalNode ← FindSyncBestFit(n_v^k, G_s);
9:     if physicalNode == null then
10:        return rejected;
11:    else
12:        AllocateNode(n_v^k, physicalNode);
13: end for
14: return accepted;
```

Assim como no algoritmo anterior, quando HSVN-BestFit (*Algorithm* 4) recebe a mensagem *"rejected"* do procedimento MapLinks(L_s^k, G_s), significa que não foi possível efetuar o mapeamento de todos os enlaces da rede virtual, sendo assim, é chamado o procedimento FreePhysicalResources (G_k^v, G_s), esse procedimento efetua a desalocação dos recursos que foram reservados para o mapeamento dessa rede virtual.

Para facilitar o entendimento do comportamento do algoritmo HSVN-BestFit, é ilustrado o mapeamento das redes virtuais 1 e 2 na rede de substrato da Figura 3.3. Os nodos são representados por eclipses e identificados por uma letra maiúscula. O valor dentro dos nodos representa a demanda de CPU, no caso das redes virtuais, ou a capacidade de CPU disponível dos nodos do substrato físico. Os enlaces são identificados por letras minúsculas, sendo que, os enlaces virtuais têm uma demanda de largura de banda fixado em 10 e os enlaces físicos tem uma capacidade máxima de largura de banda disponível inicialmente fixada em 30.

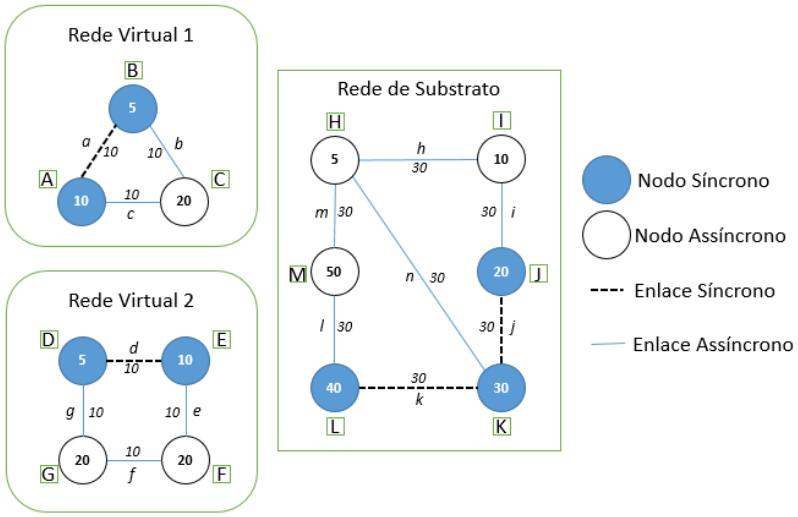

Figura 3.3 – Redes virtuais e Rede de substrato

Assim como o algoritmo HSVN-WorstFit, o mapeamento é efetuado seguindo o modelo de fila. Sendo assim, a Rede Virtual 1 será mapeada primeiramente, como pode ser visualizado na Etapa 1 da Figura 3.4. Nessa primeira etapa, iniciando o mapeamento pelo nodo virtual síncrono A, há três candidatos síncronos a serem considerados, sendo o nodo físico J eleito como hospedeiro, já que este nodo físico tem o valor de CPU disponível mais próximo do valor de demanda de CPU do nodo A.

As letras identificadores dos nodos e enlaces da rede de substrato foram omitidos na Figura Figura 3.4. Nesta Figura são apresentados somente os identificadores dos componentes virtuais mapeados na rede de substrato, a capacidade de CPU disponível nos nodos físicos e a capacidade de largura de banda disponível nos enlaces físicos.

Os nodos físicos síncronos L e K são candidatos a hospedarem o nodo virtual síncrono B. O nodo J não é considerado, pois, o mesmo já hospeda um nodo da mesma rede virtual que o nodo virtual B. Por possuir o valor de CPU disponível mais próximo do valor de demanda do nodo virtual B, o nodo físico K é eleito como hospedeiro. Assim como o nodo virtual assíncrono C, que é

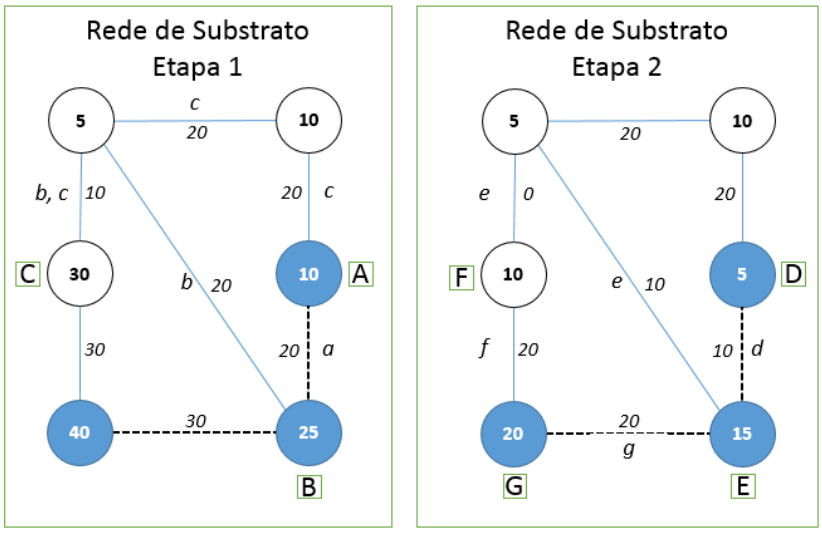

Figura 3.4 – Etapas do mapeamento utilizando HSVN-BestFit

mapeado no nodo assíncrono M, visto este ser o único candidato assíncrono elegível, pois os demais nodos físicos assíncronos não tem capacidade de CPU disponível suficiente para hospedar o nodo virtual C.

O mapeamento dos enlaces virtuais da Rede Virtual 1, inicia-se após a conclusão de mapeamento de todos os nodos virtuais desta rede virtual. Primeiramente é mapeado o enlace virtual síncrono *a* no no enlace físico síncrono *j*, por ser o caminho síncrono mais curto. Já o enlace virtual assíncrono *b*, mapeado nos enlaces físicos assíncronos *m* e *n*, e o enlace virtual assíncrono *c*, mapeado nos enlaces físicos assíncronos *m*, *h* e *i*, foram mapeados de forme que economizassem recurso físico síncrono, visto que o enlace virtual *b* poderia ter sido mapeado nos enlaces *k* e *l* e o enlace virtual *c* poderia ter sido mapeado nos enlaces físicos *j*, *n* e *m* ou *j*, *k* e *l*.

O mapeamento da Rede Virtual 2, apresentado na Etapa 2 da Figura 3.4, inicia-se pelo mapeamento do nodo virtual síncrono D, tendo como candidatos elegíveis os nodos físicos J, K e L. Sendo que desses candidatos, o nodo eleito foi K, por ter o valor de CPU disponível mais próximo do valor de demanda de CPU do nodo virtual D. O mesmo ocorre com o nodo virtual síncrono E, mapeado no nodo físico síncrono K, e o nodo virtual assíncrono F, mapeado no único nodo físico assíncrono com capacidade de CPU disponível suficiente para hospedá-lo. Entretanto, o nodo virtual assíncrono G não teve candidatos assíncronos com capacidade de CPU disponível suficiente para efetuar seu mapeamento. Sendo assim, o mesmo foi mapeado para o nodo físico síncrono J, o qual era o único disponível com capacidade de CPU suficiente para hospedá-lo. Note que o nodo físico L não era um candidato, pois já estava a hospedar um nodo da mesma rede virtual.

Como todos os nodos da Rede Virtual 2 foram mapeados com sucesso, é efetuado o mapeamento dos enlaces virtuais. O enlace virtual síncrono *d* é mapeado no enlace físico síncrono *j*, por este ser o caminho mais curto síncrono entre os nodos físicos hospedeiros dos nodos virtuais D e E. O enlace virtual assíncrono *e* por sua vez, é mapeado nos enlaces físicos assíncronos *m* e *n*, visto que o outro caminho mais curto possível (*f* e *g*) era composto por um enlace síncrono.

O enlace virtual assíncrono *f* é mapeado no enlace físico *l*, enquanto o enlace físico *g* é mapeado nos enlaces físicos *k*, *n*, *h* e *i*, utilizando um enlace síncrono, pois, o enlace físico *m* não possuía largura de banda suficiente e o outro caminho possível, composto pelos nodos *k* e *j*, ocupava mais de um recurso síncrono, sendo assim mais caro.

3.4 Heurística HSVN-Virtual Tree Topology (HSVN-VTT)

Ambos algoritmos apresentados anteriormente não consideravam a topologia ou a distância dos nodos virtuais ao efetuar o mapeamento dos mesmos, o que impacta no custo final do mapeamento. Esse impacto ocorre, pois, nodos virtuais vizinhos na rede virtual podem ser mapeados em nodos físicos distantes, obrigando o mapeamento de um único enlace virtual em vários enlaces físicos, consumindo assim mais recursos.

Por esse motivo, foi desenvolvida a heurística HSVN-VTT (*Algorithm* 6), a qual considera a distância entre os nodos virtuais durante o mapeamento dos mesmos. Diversos trabalhos na literatura levam em consideração, no momento do mapeamento dos nodos virtuais para nodos físicos, não somente as demandas dos nodos, mas também as demandas da topologia da rede virtual e a possibilidade de seu atendimento na topologia física. Cheng et al. (2011) [9] e Liu et al. (2011) [23] consideram a quantidade de enlaces, a largura de banda dos vários enlaces e a CPU de cada nodo, durante a escolha de mapeamento de nodos, para aumentar as chances dos nodos virtuais serem mapeados em nodos físicos, cujos enlaces físicos comportem a demanda dos enlaces virtuais do nodo mapeado em nodos físicos mais próximos, baixando o custo dos mapeamento dos enlaces virtuais. Já em Li et al. (2013) [22], além de ser considerado estes elementos, também é considerada ainda a distância em número de enlaces intermediários entre dois nodos, para a decisão de mapeamento dos mesmos.

Li et al. (2013) [22] propõe um algoritmo de mapeamento de redes virtuais também dividido em duas etapas onde no início da primeira etapa é construída uma árvore de alcançabilidade com base na rede virtual. Esta árvore é utilizada para efetuar o mapeamento dos nodos. Esta mesma estratégia foi transposta para o caso de HSVN. Sendo assim, essa heurística tem como base uma parte do modelo apresentado em Li et al. (2013) [22].

Algorithm 6 HSVN-VTT($G_s, HSVNRequest$)

```
1: for all G_v^k ∈ HSVNRequest do
2:     tree ← BuildTree(G_v^k);
3:     root ← GetRoot(tree);
4:     if MapFromTheRoot(root, tree, G_s) == rejected then
5:         FreePhysicalResources(G_v^k, G_s);
6:     else
7:         if MapLinks(L_v^k, G_s) == rejected then
8:             FreePhysicalResources(G_v^k, G_s);
9: end for
```

No algoritmo HSVN-VTT (*Algorithm* 6), para cada requisição de criação de rede virtual, $G_v^k \in HSVNRequest$, que foi informado como parâmetro de entrada, é construída uma árvore de alcançabilidade, por meio da função BuildTree(G_v^k), onde a raiz será o nodo virtual com a maior demanda de CPU, sendo seus filhos os nodos diretamente ligados à raiz ordenados de maneira decrescente por demanda de CPU. Os demais nodos são mapeados na árvore de maneira recursiva em profundidade, escolhendo o filho com maior demanda de CPU. A árvore estará completa assim que todos os nodos virtuais fizerem parte da dela. A Figura 3.5 demonstra a árvore obtida a partir de uma requisição de rede virtual. É importante salientar que essa árvore é usada somente na etapa do mapeamento dos nodos virtuais, como será descrito nos próximos parágrafos.

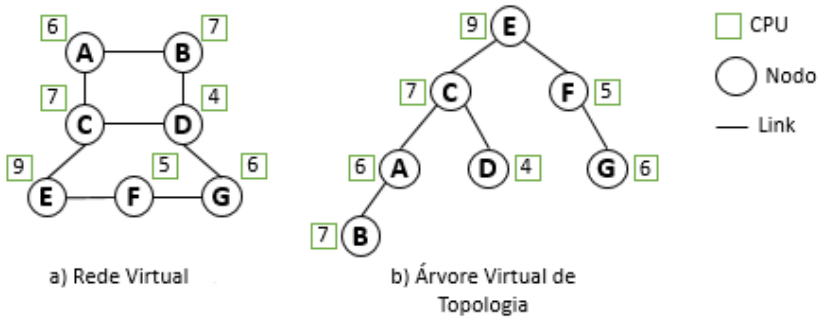

Figura 3.5 – Construção da árvore virtual de topologia a partir de uma rede virtual. Fonte: [22]

Após a construção da árvore é iniciado o processo de mapeamento dos nodos virtuais. Por meio da função GetRoot($tree$) é obtido o nodo virtual com maior demanda de CPU, o qual é a raiz da árvore criada. Então é chamado o procedimento MapFromTheRoot($root, tree, G_s$), representado pelo *Algorithm* 7.

No algoritmo MapFromTheRoot, primeiramente é efetuado o mapeando do nodo virtual que está na raiz da árvore, por esse motivo ele foi dado como parâmetro de entrada desse procedimento. O nodo raiz é mapeado de maneira diferenciada, visto que ele não possui pai. Para evitar gargalos, esse nodo virtual é mapeado para o nodo físico com maior capacidade de CPU disponível e suficiente para hospedar o nodo virtual em questão, sendo assim, o nodo hospedeiro é obtido pela função FindSyncWorstFit($root, G_s$) ou FindAsyncWorstFit($root, G_s$), ambas descritas anteriormente, apenas foram alterados os parâmetros de entrada. Como pode ser notado o requisito de sincronia continua sendo considerado, sendo que, quando o nodo raiz for síncrono o mesmo só poderá ser mapeado em um nodo físico síncrono, mas se o nodo raiz for assíncrono, ele somente será mapeado em um nodo físico síncrono caso não haja um nodo físico assíncrono elegível.

Caso nenhum nodo do substrato físico seja elegível para efetuar o mapeamento do nodo raiz (linha 10), é retornada a mensagem *"rejected"* (linha 11). Caso o mapeamento do nodo raiz seja efetuado com sucesso, inicia-se o mapeamento de cada filho por meio do procedimento MapNodesBasedOnTree($n_v^k, tree, G_s$) (*Algorithm* 8). Esse procedimento tem como entrada o nodo virtual que pretende-se efetuar o mapeamento, a árvore de alcançabilidade e a rede de substrato. A função Children(n_v^k) retorna o conjunto de nodos filhos de n_v^k ordenados de maneira decrescente de CPU.

Algorithm 7 MapFromTheRoot($root, tree, G_s$)

```
1: candidateHost ←null;
2: if Sync(root) == TRUE then
3:     candidateHost ← FindSyncWorstFit(root, G_s);
4: else
5:     candidateHost ← FindAsyncWorstFit(root, G_s);
6:     if (candidateHost) == null then
7:         candidateHost ← FindSyncWorstFit(root, G_s);
8:     end if
9: end if
10: if candidateHost == null then
11:     return rejected;
12: else
13:     if Children(root) not equal ∅ then
14:         for all n_v^k ∈ Children(root) do
15:             if MapNodesBasedOnTree(n_v^k, tree, G_s) == rejected then
16:                 return rejected;
17:             end if
18:         end for
19:     end if
20: end if
```

O processo de mapeamento dos filhos ocorre em pré-ordem, ou seja, seram mapeados os nodos virtuais seguindo um percurso em profundidade com base na árvore criada. Para cada nodo virtual a ser mapeado, é calculado o fator NF para cada nodo físico candidato a hospedar o nodo virtual, isso ocorre pela chamada da função CalcNF($n_v^k, tree, G_s$), lembrando que o nodo pai já tem seu mapeamento decidido. A equação do fator de seleção do nodo (Node Selection Factor - NF) não foi alterada com relação a [22], ele é calculado com base na capacidade máxima de CPU do nodo físico candidato, e na distância entre o hospedeiro do nodo virtual pai, já decidido, e o nodo físico candidato em questão, conforme a equação (1).

$$NF = MaxCPU(n_s) * (1/Dis(Host(Parent(n_v^k)), n_s)); \tag{3.1}$$

Seja $n_s \in N_s$ um nodo físico candidato a hospedar o nodo virtual $n_v^k \in N_v^k$; $Dis(n_s, n'_s)$ a distância entre o nodo físico n_s e o nodo físico n'_s dada pelo número de enlaces entre n_s e n'_s; $Host(n_v^k)$ o nodo físico que hospeda o nodo virtual n_v^k; $Parent(n_v^k)$ o nodo virtual pai de n_v^k na árvore, a equação (1) resulta em um fator NF diretamente proporcional à capacidade de CPU do nodo físico candidato e inversamente proporcional à distância do nodo físico candidato ao nodo físico que hospeda o pai do nodo virtual em mapeamento.

O nodo virtual é mapeado no nodo físico com o maior NF, mas que tenha capacidade de CPU disponível suficiente para hospedar o nodo virtual. Assim, como no mapeamento do nodo raiz, a sincronia também é considerada. Nodos virtuais síncronos são mapeados somente em nodos síncronos, já os nodos virtuais assíncronos são mapeados em nodos síncronos somente se não hou-

Algorithm 8 MapNodesBasedOnTree($n_v^k, tree, G_s$)

```
1: candidateHost ←null;
2: CalcNF(n_v^k, tree, G_s);
3: if Sync(n_v^k) == true then
4:     candidateHost ← FindSyncCandidate(n_v^k, G_s);
5: else
6:     candidateHost ← FindAsyncCandidate(n_v^k, G_s);
7:     if (candidateHost) == null then
8:         candidateHost ← FindSyncCandidate(n_v^k, G_s);
9:     end if
10: end if
11: if candidateHost == null then
12:     return rejected;
13: else
14:     if Children(n_v^k) not equal ∅ then
15:         for all n_v^k ∈ Children(n_v^k) do
16:             if MapNodesBasedOnTree(n_v^k, tree, G_s) == rejected then
17:                 return rejected;
18:             end if
19:         end for
20:     end if
21: end if
```

verem nodos físicos elegíveis. Caso não seja localizado um nodo do substrato que não forneça os requisitos do nodo virtual, a rede virtual é rejeitada e os recursos físicos já reservados para essa rede virtual são liberados por meio do procedimento FreePhysicalResources(G_v^k, G_s) (linha 5 do algoritmo HSVN-VTT).

A segunda etapa do mapeamento ocorre somente quando todos os nodos virtuais são mapeados com sucesso, nessa etapa inicia-se o mapeamento dos enlaces virtuais, por meio do Algoritmo MapLinks (*Algorithm* 3), já descrito anteriormente. Caso não seja encontrado um nodo físico ou um caminho elegível, a rede virtual é rejeitada e os recursos antes reservados para a mesma são liberados.

Para facilitar o entendimento do comportamento do algoritmo HSVN-VTT, é ilustrado o mapeamento das redes virtuais 1 e 2 na rede de substrato da Figura 3.6. Os nodos são representados por eclipses identificados por uma letra maiúscula. O valor dentro dos nodos representa a demanda de CPU, no caso das redes virtuais, ou a capacidade de CPU disponível dos nodos do substrato físico. Os enlaces são identificados por letras minúsculas, sendo que, os enlaces virtuais tem uma demanda de largura de banda fixado em 10 e os enlaces físicos tem uma capacidade máxima de largura de banda disponível inicialmente fixada em 30.

Assim como nos demais algoritmos deste trabalho, o mapeamento das redes virtuais é efetuado seguindo o modelo de fila. Sendo assim, a Rede Virtual 1 será mapeada primeiro. Ao receber uma rede virtual para mapear, esse algoritmo monta uma árvore de alcançabilidade referente à esta rede virtual. Essa árvore é construída da seguinte maneira: Primeiramente é selecionado o

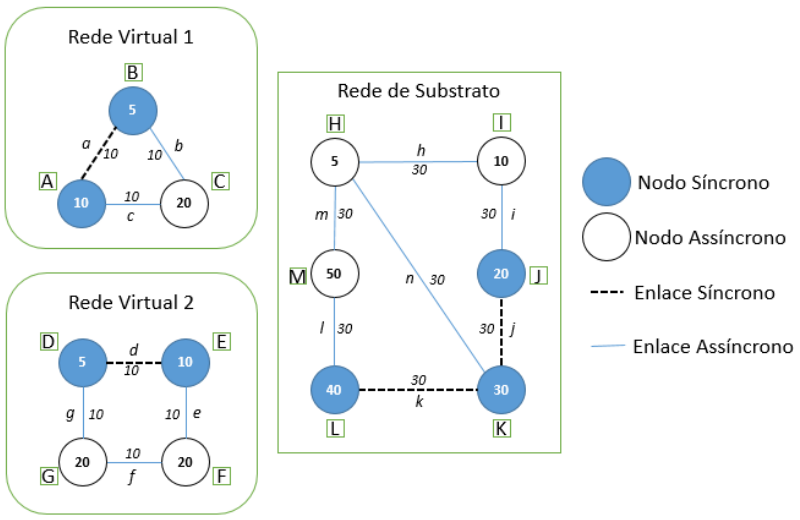

Figura 3.6 – Redes virtuais e Rede de substrato

nodo virtual com maior demanda de CPU para ser a raiz desta árvore, portanto, o nodo virtual C é definido como nodo raiz da Rede Virtual 1. Após a definição do nodo raiz, são adicionados os nodos filhos, sendo estes, os nodos que estão diretamente ligados ao nodo raiz por um enlace virtual da rede virtual. Como os nodos A e B estão diretamente conectados ao nodo C, ambos são adicionados na árvore como seus filhos. Então, os nodos recém adicionados são ordenados de maneira decrescente com base na sua demanda de CPU. Após todos os nodos de uma rede virtual serem incluídos na árvore, ela é dada como concluída. A árvore montada a partir da Rede Virtual 1 é ilustrada na Figura 3.7.

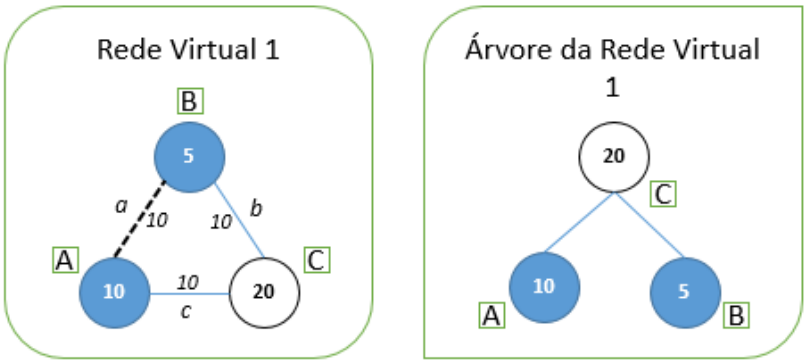

Figura 3.7 – Árvore de alcançabilidade construída a partir da Rede Virtual 1

Após a árvore de alcançabilidade da Rede Virtual 1 ser concluída, inicia-se o processo de mapeamento dos nodos virtuais. A ordem do mapeamento é definida por uma busca em profundidade na árvore, então é mapeamento primeiro a raiz e depois seus filhos da esquerda para a direita.

O nodo virtual C é mapeado de maneira distinta dos demais por ser o nodo raiz. Sendo ele um nodo virtual assíncrono, seu mapeamento deve priorizar nodos físicos assíncronos. Como citado anteriormente, a raiz é mapeada para o nodo físico com maior capacidade de CPU disponível. Então, o nodo virtual C é mapeado no nodo físico assíncrono M, sendo este o único nodo físico assíncrono com capacidade de CPU disponível suficiente para hospedá-lo.

Para efetuar o mapeamento do nodo virtual síncrono A é calculado o fator NF para cada nodo da rede de substrato. O cálculo é feito com base na capacidade máxima de CPU de cada nodo físico e na distância do nodo físico M que hospeda o nodo virtual C, o qual é pai do nodo virtual A na árvore, em relação aos demais nodos físicos. Os valores do fator NF para o mapeamento deste nodo virtual podem ser conferidos na na Etapa 1 da Figura 3.8, assim como o valor da distância entre o nodo hospedeiro de C e os demais nodos. O maior fator NF foi obtido pelo nodo físico síncrono L, sendo este o hospedeiro do nodo virtual A. O mesmo ocorre com o nodo virtual síncrono B, o qual é mapeado no nodo físico síncrono K, já que que este obteve o maior fator NF, como pode ser verificado na Etapa 2 da Figura 3.8.

Tanto na Figura 3.8, quanto na Figura 3.10, os grafos que representam as etapas de mapeamento, apresentam os valores do fator NF obtido em cada etapa. Também é apresentada a distância entre o nodo físico que hospeda o nodo virtual pai e os demais nodos candidatos. Também é apresentado o identificador dos nodos virtuais que foram mapeados. Já o grafo que representa o estado final do mapeamento de uma rede virtual, são apresentados a largura de banda e capacidade de CPU disponíveis na rede de substrato. Também são apresentados os identificadores dos nodos e enlaces virtuais que foram mapeados.

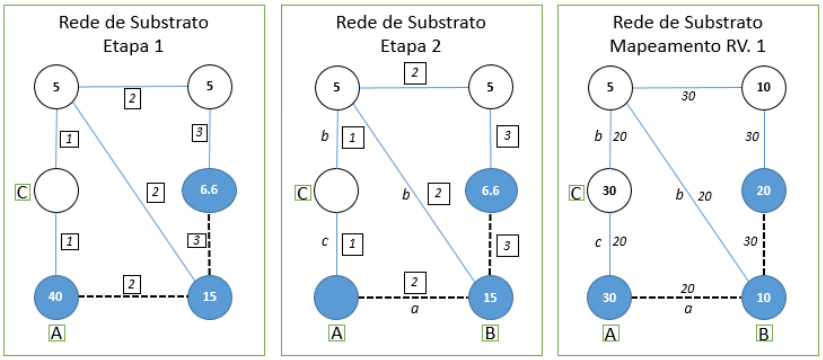

Figura 3.8 – Etapas do mapeamento utilizando HSVN-VTT

Ao concluir o mapeamento de todos os nodos virtuais com sucesso, inicia-se o mapeamento dos enlaces virtuais da Rede Virtual 1, a partir desta etapa a árvore de alcançabilidade não é mais utilizada. Note que a árvore é utilizada apenas para auxiliar na ordenação de mapeamento dos nodos virtuais e para calcular o fator NF, indicando quem é o pai de um nodo virtual.

O enlace síncrono *a* é mapeado no enlace físico síncrono *k* e o enlace virtual assíncrono *c* é mapeado para o enlace físico *l*, por serem o caminho mais curto entre os nodos físicos que hospedam os nodos virtuais de origem e destino desses enlaces virtuais. Já o enlace virtual *b* é mapeado nos enlaces físicos assíncronos *m* e *n*, visando economizar recursos síncronos.

A árvore de alcaçabilidade também é construída para mapear a Rede Virtual 2. O nodo assíncrono F é definido como a raiz da árvore por ter a maior demanda de CPU. Embora o nodo virtual G tenha o mesmo valor de demanda de CPU, ele não é selecionado como raiz. Isso ocorre, pois, como pode ser verificado na Figura 3.8, os identificadores dos nodos são dados por letras do alfabeto, sendo essa a ordem em que os nodos são processados pelo algoritmo. Quando o F é

pré-definido como raiz, ele pode ser substituído somente se um nodo virtual tivesse maior demanda de CPU.

Após a definição do nodo raiz, inicia-se a inclusão dos demais nodos na árvore. Como os nodos E e G estão diretamente conectados ao nodo F, estes são adicionados à árvore como filhos do nodo F. Após a inclusão destes nodos, eles são ordenados por demanda de CPU. Visto que a construção da árvore ainda não foi concluída, pois o nodo D ainda não foi adicionado à árvore, inicia-se o mapeamento dos filhos do nodo raiz seguindo a ordem da esquerda para a direita (busca em profundidade), sendo assim, primeiramente são mapeados os filhos do nodo G. Sendo o nodo D o único nodo ainda não adicionado à árvore que está diretamente conectado ao nodo G, ele é adicionado na árvore como filho do nodo G. Então o processo de construção da árvore é completado. Caso houvessem outros nodos não conectados diretamente ao nodo G e que ainda não haviam sido adicionados à árvore, seria iniciado o mapeamento dos filhos do nodo E.

Concluído o processo da construção da árvore da Rede Virtual 2, ambos representados na Figura 3.9, inicia-se o processo de mapeamento dos nodos virtuais. A ordem em que os nodos são mapeados é definido pelo modelo de busca em profundidade na árvore construída. O nodo virtual assíncrono F, raiz da árvore, é mapeado no nodo físico assíncrono M, mesmo que o mesmo tenha a maior capacidade de CPU disponível, ele foi eleito como hospedeiro, pois era o único nodo físico assíncrono com capacidade de CPU disponível para hospedar o nodo virtual F.

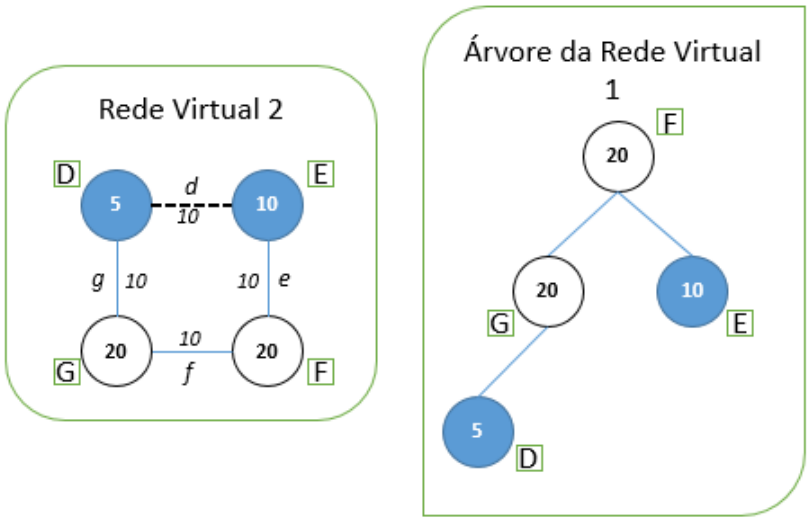

Figura 3.9 – Árvore de alcançabilidade construída a partir da Rede Virtual 2

Após o mapeamento do nodo raiz, é calculado o fator NF para efetuar o mapeamento do nodo virtual assíncrono G. Tanto o valor do fator NF de cada nodo físico, quanto a distância do nodo hospedeiro do nodo pai de G, são apresentados na Etapa 1 da Figura 3.10, onde é possível verificar que o nodo físico síncrono L possui o maior fator NF. Então o nodo virtual assíncrono G é mapeado para o nodo físico síncrono L, mesmo este sendo síncrono, já que os demais nodos assíncronos não possuíam capacidade de CPU disponível suficiente para hospeda o nodo virtual.

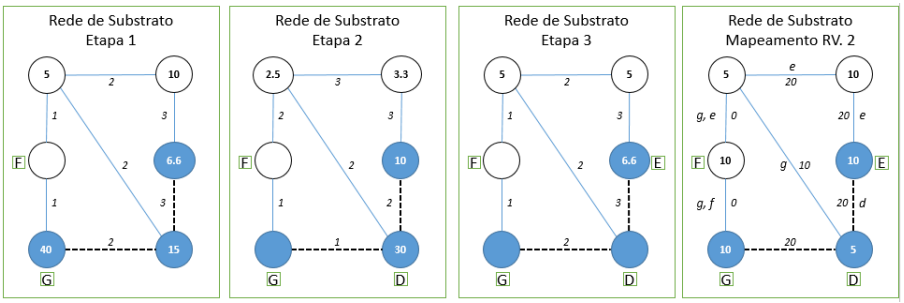

Figura 3.10 – Etapas do mapeamento utilizando HSVN-VTT

O nodo virtual síncrono D, após o cálculo do fator NF, apresentado na Etapa 2 da Figura 3.10, é mapeado no nodo físico síncrono K, visto que o mesmo obteve o maior fator NF, sendo este o mesmo motivo do nodo virtual síncrono E ser mapeado no nodo físico J, como pode ser observado na Etapa 3 da Figura 3.10.

Como citado anteriormente, a árvore de alcançabilidade não é mais utilizada após o mapeamento dos nodos virtuais. Os enlaces virtuais são mapeados de acordo com a topologia original da rede virtual. O enlace virtual síncrono *d* é mapeado no enlace físico síncrono *j*, pois o mesmo é o caminho mais curto entre os hospedeiros dos nodos virtuais de origem e destino do enlace virtual. Já o enlace virtual assíncrono *e* foi mapeado para os enlaces físicos *m* e *n* com o objetivo de economizar recursos síncronos. O enlace virtual assíncrono *f* é mapeado no enlace físico assíncrono *l*, por ser o caminho mais curto entre os hospedeiros dos nodos virtuais de origem e destino do enlace virtual. Por fim, o enlace virtual assíncrono *g* é mapeado para os enlaces físicos assíncronos *h*, *i* e *m* para economizar recursos síncronos.

3.5 Heurística HSVN-Virtual Tree Topology & Bandwidth (HSVN-VTTBW)

As heurísticas apresentadas até esta seção têm uma deficiência em comum durante a etapa de mapeamento dos nodos. Elas não considerem para cada nodo físico, sua capacidade total de largura de banda disponível e não fazem uma análise de disponibilidade de largura de banda por enlace físico para hospedar um determinado nodo virtual e as demandas de cada um de seus enlaces virtuais. Com isso, uma maior taxa de rejeição acontece na fase de mapeamento de enlaces, já que o mapeamento de nodos pode ter considerado nodos físicos com enlaces não capazes de suportar as demandas dos respectivos enlaces virtuais.

Com o objetivo de aumentar a taxa de aceitação do algoritmo heurístico anterior, foi incluída nesse algoritmo uma nova regra para que um nodo físico seja considerado elegível. Essa regra define que para que um nodo físico seja elegível para mapear um nodo virtual, a largura de banda disponível nos enlaces deste nodo físico deve se suficiente para que os enlaces do nodo virtual sejam mapeados. Essa regra é implementada nas funções FindSyncWorstFit e FindAsyncWorstFit do *Algorithm* 7 e nas funções FindSyncCandidate e FindSyncCandidate do *Algorithm* 8, as quais retornam um candidato válido após analisarem cada nodo físico.

Em HSVN-VTTBW, as funções modificadas verificam também se a soma da largura de banda dos enlaces síncronos do nodo físico candidato é maior ou igual a soma da largura de banda dos enlaces virtuais síncronos do nodo virtual, assim como a soma da largura de banda de todos os enlaces conectados ao nodo físico é maior ou igual a soma de todos os enlaces assíncronos do nodo físico. Após isso é verificado se há no mínimo um enlace físico síncrono com largura de banda suficiente para hospedar cada enlace virtual síncrono, assim como, também é verificado se há no mínimo um enlace físico com largura de banda suficiente para hospedar cada enlace virtual do nodo virtual.

4. AVALIAÇÃO DE DESEMPENHO

Neste capítulo é apresentada uma avaliação das heurísticas propostas, as quais foram implementadas na linguagem Java 7. As métricas utilizadas para tal avaliação são descritas na seção 4.1. Já na seção 4.2 são descritas as ferramentas e parâmetros utilizados no primeiro experimento e na sessão 4.3 a apresentação e discussão dos resultados deste experimento. O segundo experimento é apresentado na sessão 4.4, sendo a apresentação e discussão de seus resultados efetuados na sessão 4.5.

4.1 Métricas de Avaliação

Nesta seção são descritas as diferentes métricas utilizadas na avaliação das heurísticas propostas. Métricas são necessárias para avaliar a performance ou qualidade do mapeamento, permitindo a comparação entre diferentes abordagens. Fischer et al. (2013) [15] também identificou um conjunto de métricas utilizadas para avaliar o mapeamento de redes virtuais. Entre estas, com exceção do fator de ineficiência e utilização dos recursos síncronos, as seguintes são relevantes para RVSH:

- **Taxa de aceitação** A taxa de aceitação se dá pela divisão entre o número de redes virtuais mapeadas com sucesso e o número de redes virtuais que foram requisitadas.
- **Custo de mapeamento** O custo de mapeamento das redes virtuais é obtido pela soma de CPU e largura de banda dos enlaces da rede de substrato que foram utilizados para mapear um conjunto de redes virtuais. No caso de ambos experimentos apresentados nesse trabalho, o custo de CPU necessário para o mapeamento sempre será mínimo, sendo assim, nesse trabalho o custo será somente a soma de largura de banda utilizada para efetuar o mapeamento.
- **Utilização de recursos síncronos** A utilização dos recursos síncronos verifica quanto de recurso síncrono foi alocado, isso permite verificar quanto de recursos síncronos foram alocados a mais ou até desnecessariamente.
- **Fator de ineficiência** Como ocorre rejeição no mapeamento de algumas redes virtuais em alguns experimentos, pode ser injusto efetuar uma comparação entre as heurísticas somente pelo custo de mapeamento. O fator de ineficiência visa fazer uma comparação mais justa entre as heurísticas. Esse fator é obtido pela divisão entre a soma da largura de banda dos recursos físicos utilizados no mapeamento e a soma da largura de banda mínima necessária para mapear as redes virtuais que foram mapeadas com sucesso. Sendo que a solução ótima tem taxa de ineficiência 1.
- **Tempo de execução** O tempo de execução é o tempo necessário para o mapeamento ser efetuado. Essa métrica é relevante neste trabalho, pois o principal objetivo da utilização

de abordagens heurísticas é reduzir o tempo computacional necessário para encontrar uma solução válida.

4.2 Primeiro experimento

O primeiro experimento compara as soluções encontradas pelos algoritmos heurísticos propostos neste trabalho com a solução ótima obtida pelo modelo proposto em Hasan et al. (2014) [20]. Os dados e o ambiente de execução para esse experimento são os mesmos utilizados em Hasan et al. (2014) [20], visando uma comparação mais justa dos resultados. Sendo assim, o substrato físico e as redes virtuais foram randomicamente geradas pela ferramenta BRITE[26] seguindo o modelo Waxman [32]. Os experimentos foram executados em um computador com CPU de 4 cores de 1.60 GHz, e 2 GB de memória RAM. Foram executados vinte experimentos, divididos em três grupos, A, B e C. Cada grupo com requisições de redes virtuais com tamanho total de 10, 20 e 30 de nodos respetivamente.

Tabela 4.1 – Parâmetros de configuração dos experimentos

Propriedades dos Experimentos				
Grupo : Tamanho da Rede Virtual	A: 10 nodos B: 20 nodos C: 30 nodos			
Cenário	1	2	3	4
Nodos do Substrato Físico	25			
Banda do Substrato Físico	Uniformente distribuído entre 1Gbps e 3Gbps			
CPU do Substrato Físico	100			
Banda das Redes Virtuais	Uniformente distribuído entre 100Mbps e 1Gbps			
CPU das Redes Virtuais	Respectivamente 10, 15 e 25% da capacidade de CPU do substrato físico			
Sincronia do Substrato Físico	30%			100%
Sincronia das Redes Virtuais	0%	30%	60%	x%

Em todos os experimentos desta sessão, o substrato físico é composto por 25 nodos, sendo a largura de banda dos enlaces distribuída entre 1 e 3 Gbps. Nos cenários 1, 2 e 3 o substrato físico é composto por 30% de recursos síncronos, já no cenário 4, 100% dos recursos são síncronos. As redes virtuais são compostas por 3, 4 ou 5 nodos cada, sendo que cada nodo possui respectivamente 10%, 15% e 25% de CPU. A largura de banda das redes virtuais foi distribuída entre 100Mbps e 1Gbps. A sincronia das redes virtuais varia de acordo com o grupo, sendo 0% no cenário 1, 30% no cenário 2, 60% no cenário 3, já no cenário 4 a sincronia é referenciada como x%, pois, neste cenário, o custo de mapeamento será independente da sincronia das redes virtuais, visto que o substrato físico será totalmente síncrono.

4.3 Resultados do primeiro experimento

O primeiro fator a ser avaliado é a taxa de aceitação, a qual pode ser visualizada nos gráficos de custo de largura de banda dos grupos A, B e C (Figura 4.1, Figura 4.2, Figura 4.3).

Para diminuir a poluição visual dos gráficos, a taxa de aceitação só é exibida quando a mesma for inferior à 100%.

As taxas de aceitação das soluções ótimas foram 100%, demonstrando que é possível efetuar o mapeamento de todas as requisições deste conjunto de experimentos. A heurística HSVN-BestFit apresentou o melhor desempenho nesse critério, obtendo taxa de aceitação inferior a 100% apenas no experimento C4. Já a heurística HSVN-WorstFit teve a pior taxa de aceitação, tendo mapeado com sucesso todas as requisições de apenas um terço dos experimentos. E embora a heurística HSVN-VTTBW tenha obtido uma taxa de aceitação superior à heurística HSVN-VTT no experimento C2, ambas tiveram taxa de aceitação inferior à 100% no grupo C de experimentos.

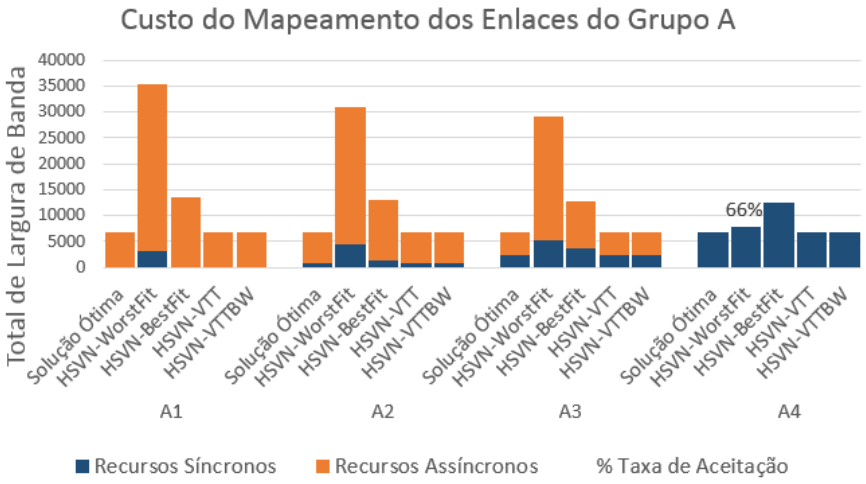

Figura 4.1 – Custo de mapeamento de cada cenário do grupo A

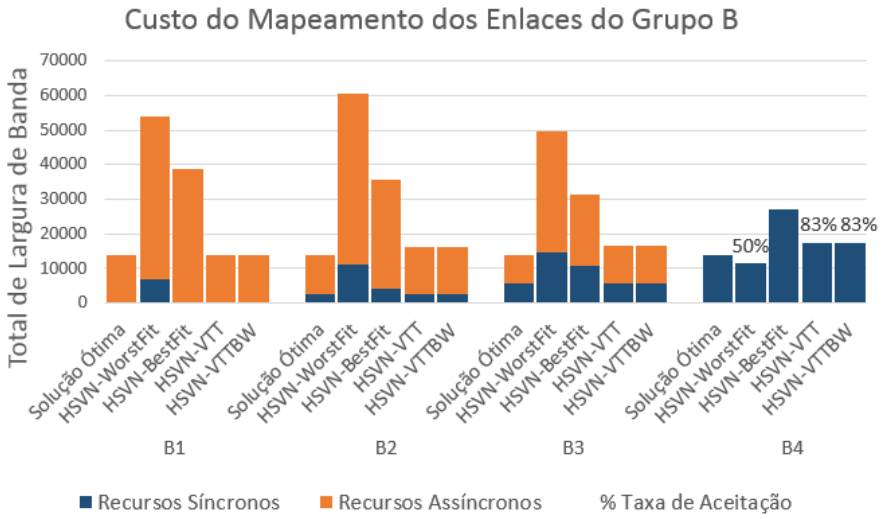

Figura 4.2 – Custo de mapeamento de cada cenário do grupo B

O segundo fator avaliado foi a utilização de recursos síncronos, pois um dos objetivos das heurísticas propostas é minimizar o consumo desse tipo de recurso. Visto que não há requisição de redes virtuais com demanda de sincronia no cenário 1 e observando os gráficos do custo de

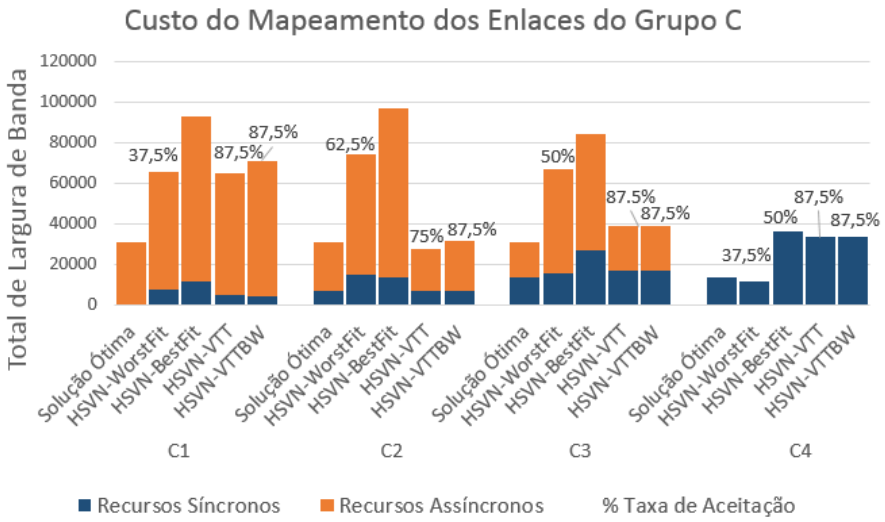

Figura 4.3 – Custo de mapeamento de cada cenário do grupo C

largura de banda dos grupos A, B e C (Figura 4.1, Figura 4.2, Figura 4.3), é possível verificar que a as soluções ótimas do cenário 1 não utilizam recursos síncronos, demonstrando que não é necessária a utilização desse tipo de recurso para efetuar o mapeamento. Entretanto, todas as soluções encontradas pela heurística HSVN-WorstFit fazem utilização desnecessária de recursos síncronos, enquanto as demais heurísticas somente utilizam recursos síncronos desnecessários no grupo C, cenário 1, onde as requisições de redes são maiores.

Quanto ao custo de mapeamento, a heurística HSVN-WorstFit apresentou os custos mais altos, chegando a custar mais de 500% o custo da solução ótima no experimento A1 com 100% de taxa de aceitação. Já as heurísticas HSVN-Tree e HSVN-TreeBW apresentaram os menores custos, inclusive custos mínimos, equivalentes aos custos da solução ótima, em todos os experimentos do grupo A e no experimento B1. Enquanto a heurística HSVN-BestFit apresentou um custo superior em todos os experimentos, porém a melhor taxa de aceitação. Por isso, uma comparação entre as heurísticas com base somente nos custos de mapeamento, sem considerar a taxa de aceitação, pode ser injusto.

Para fazer uma comparação mais justa entre as heurísticas, foi calculado o fator médio de ineficiência de cada algoritmo por grupo (Figura 4.4). Nos experimentos desta sessão a solução ótima obteve sempre fator de ineficiência 1, que é a menor taxa de ineficiência.

O fator médio de ineficiência da heurística HSVN-BestFit aumentou conforme o aumento das redes virtuais, indicando que tal heurística não é a mais adequada para grandes demandas de requisições de redes virtais. Já as heurísticas HSVN-VTT e HSVN-VTTBW apresentam no grupo A as mesmas taxas de ineficiências que o modelo ótimo. O que significa que é possível obter-se soluções com custo igual às soluções ótimas com essa heurística. Nos grupos B e C a média de ineficiência apresenta um aumento, mas de qualquer forma ambas as heurísticas obtiveram as menores taxas de ineficiência. No grupo C a heurística HSVN-VTT obteve uma maior ineficiência

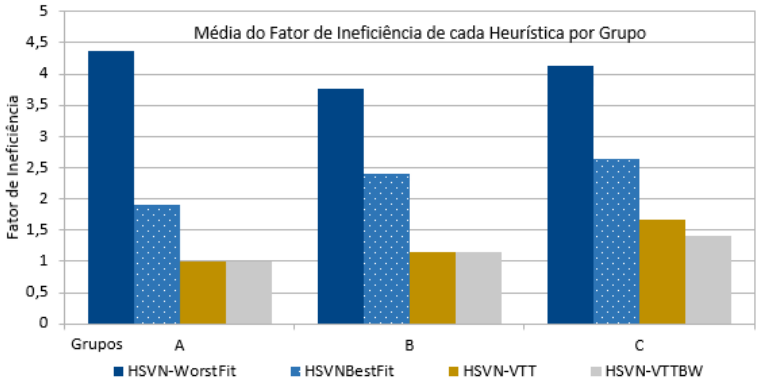

Figura 4.4 – Média de ineficiência de cada heurística por grupo

do que a heurística HSVN-VTTBW, além de uma menor taxa de aceitação, o que indica que que o HSVN-VTTBW tende a ter melhor desempenho para para efetuar o mapeamento em redes maiores.

O último item a ser avaliado é o tempo de execução. Como pode ser verificado na Tabela 4.2 do tempo de execução, o modelo de Hasan et al. (2014)[20] chegou a levar mais de 58 minutos para encontrar uma solução ótima, enquanto todas as heurísticas aqui propostas encontram uma solução em menos de 1 segundo.

Tabela 4.2 – Tempo necessário para obter a solução ótima de mapeamento, em minutos. Fonte: Hasan et al. (2014) [20]

Grupo	Exp.1	Exp.2	Exp.3
A	0.85	0.40	0.31
B	37.55	1.64	10.26
C	58.34	27.12	4.47

4.4 Segundo experimento

Este experimento compara as soluções encontradas pelas heurísticas descritas anteriormente. Também é analisado o comportamento dessas heurísticas ao efetuar o mapeamento de um número maior de redes virtuais em um substrato físico composto por um número maior de nodos e enlaces, além desse substrato ter adotando a topologia Torus para conectar seus nodos. Pelas redes virtuais e o substrato físico serem maiores neste segundo experimento, não é viável obter a solução ótima, devido ao tempo de execução necessário para a solução ótima ser calculada.

As redes virtuais foram randomicamente geradas pela ferramenta BRITE[26] seguindo o modelo Waxman[32]. Os experimentos foram executados em um computador com CPU de 4 núcleos de 2.2 GHz e 8GB de RAM. Foram executados 16 experimentos divididos em três grupos, A, B e C. Cada grupo com requisições de redes virtuais com tamanho total de 80, 100 e 120 nodos respectivamente. As redes virtuais são compostas por conjuntos de 4, 6 e 10 nodos, sendo que cada nodo possui uma demanda de CPU. O valor da demanda de CPU dos nodos virtuais foi distribuído de maneira incremental entre os nodos de cada rede virtual, sendo 20 o valor inicial. Por exemplo,

em uma rede virtual composta por 4 nodos, cada nodo terá respectivamente 20, 21, 22, e 23 de demanda de CPU. Os enlaces virtuais têm 1Gbps de demanda de largura de banda.

Tabela 4.3 – Parâmetros de configuração dos experimentos

Propriedades dos Experimentos				
Grupo : Tamanho da Rede Virtual	A: 80 nodos B: 100 nodos C: 120 nodos			
Cenário	1	2	3	4
Nodos do Substrato Físico	100			
Banda do Substrato Físico	10Gbps			
CPU do Substrato Físico	100			
Banda das Redes Virtuais	1Gbps			
CPU das Redes Virtuais	Incremental a partir de 20			
Sincronia do Substrato Físico	30%			
Sincronia das Redes Virtuais	0%	30%	60%	100%

O substrato físico é composto por 100 nodos com 100 unidades de capacidade inicial de CPU, onde 30% destes nodos são síncronos. A topologia de rede Torus foi adotada para ser mais próximo de um ambiente real do que uma rede randomicamente gerada, nesse caso, foi usado como base a topologia dos computadores da lista TOP 500de computadores. Os enlaces da rede de substrato tem capacidade de 10Gbps. Os nodos físicos síncronos são conectados aos seus vizinhos também síncronos por um enlace síncrono, conforme ilustrado na Figura 4.5.

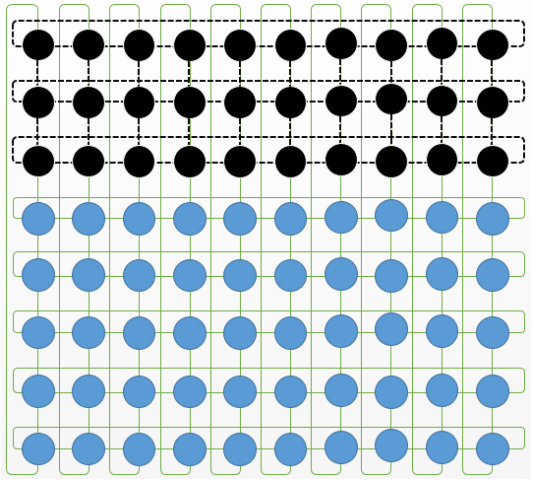

Figura 4.5 – Representação da rede de substrato do segundo experimento

A sincronia das redes virtuais varia de acordo com cada cenário. O cenário 1 é composto por somente redes virtuais assíncronas. No cenário 2 as redes virtuais têm 30% dos nodos e enlaces síncronos e 60% no cenário 3. Já o quarto cenário é composto somente por redes virtuais totalmente assíncronas.

4.5 Resultados do segundo experimento

A taxa de aceitação é o primeiro fator a ser analisado. Para diminuir a poluição visual dos gráficos, a taxa de aceitação só é exibida quando a mesma é inferior a 100%, conforme pode ser

observado nas Figuras 4.6, 4.7 e 4.8. Como não foram calculadas as soluções ótimas para esse experimento, não há como garantir se é possível efetuar o mapeamento de todas as redes virtuais de cada grupo e cenário. Então é possível que em alguns experimentos, algumas redes virtuais tenham sido rejeitadas por não haver como mapear elas.

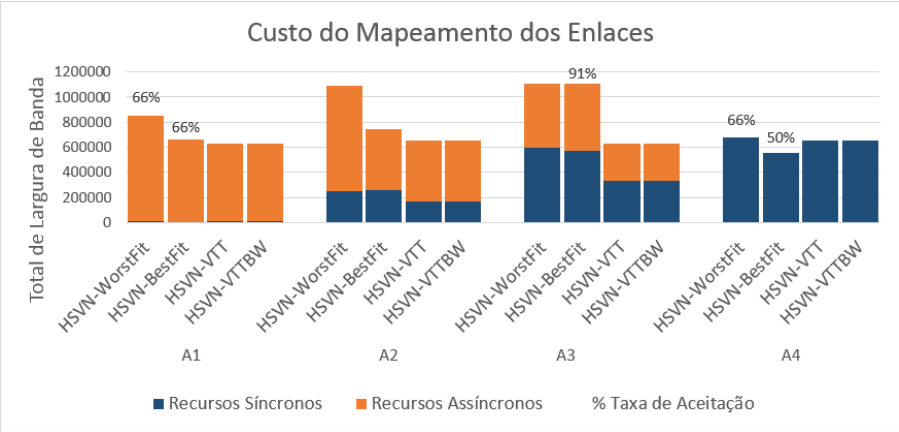

Figura 4.6 – Custo de mapeamento de cada cenário do grupo A

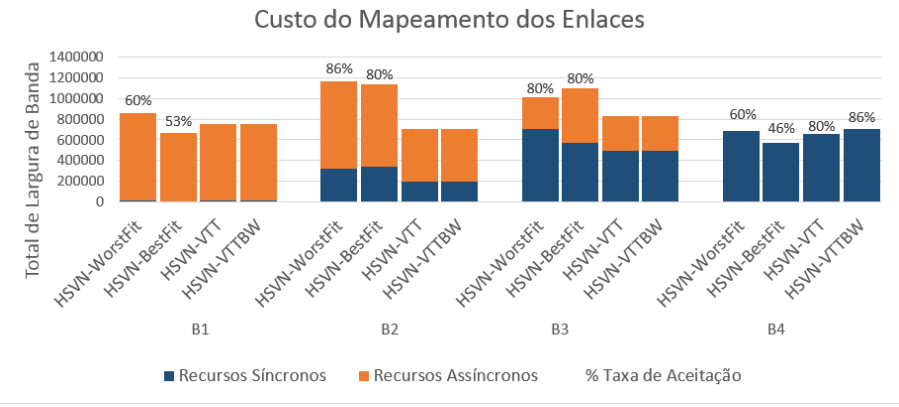

Figura 4.7 – Custo de mapeamento de cada cenário do grupo B

Ao contrário dos resultados do primeiro experimento, seção 4.3, a heurística HSVN-BestFit, embora tenha obtido a mesma taxa de aceitação que o que a heurística HSVN-WorstFit em alguns experimentos, não teve taxa de aceitação maior que as demais heurísticas, conseguindo efetuar o mapeamento de todas as redes virtuais somente no experimento A2 (Grupo A, Cenário 2). Sendo assim, HSVN-BestFit teve o pior desempenho nesse fator. Isso ocorreu pelo fato deste algoritmo concentrar o mapeamento dos nodos virtuais nos mesmo nodos físico, o que implica em um sobrecarregamento dos mesmos e dos seus enlaces físicos. Já a heurística HSVN-WorstFit mapeia as redes virtuais de forma mais espalhada, o que acaba sobrecarregando menos os nodos, porém acabou mapeando nodos virtuais vizinhos em nodos físicos distantes, sobrecarregando os enlaces físicos.

A heurística HSVN-VTTBW obteve as melhores taxas de aceitação, aceitando 6% a mais de redes virtuais que a heurística HSVN-VTT nos experimentos B4 e C4. Isso ocorreu, pois a heurística HSVN-VTTBW verifica se há enlaces e banda disponível suficiente no nodos físico candidato

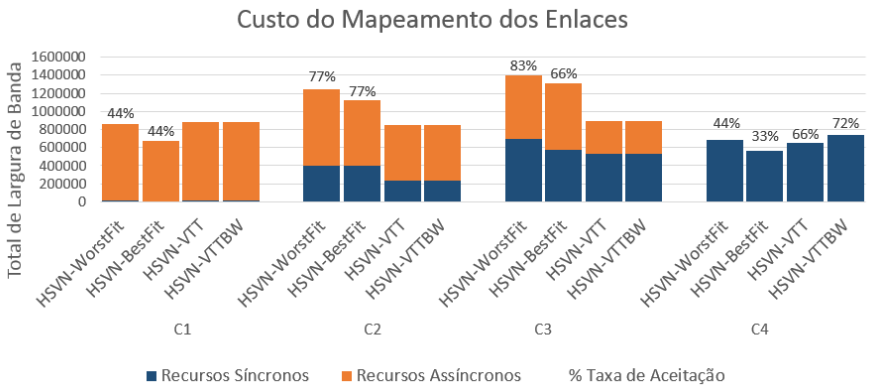

Figura 4.8 – Custo de mapeamento de cada cenário do grupo C

a hospedeiro para mapear os enlaces virtuais do nodo virtual, diminuindo a possibilidade do nodo virtual ser mapeado em um nodo físico sem largura de banda para efetuar o mapeamento dos enlaces virtuais.

O segundo fator a ser analisado é a utilização de recursos síncronos. Visto que o custo de recursos síncronos utilizados no cenário 1 não é suficiente para que seja percebido nos gráficos, os custos do cenário 1 são apresentados na Tabela 4.4. Como pode ser observado, apenas a heurística HSVN-BestFit não utilizou recurso síncrono nos experimentos do cenário 1. Por ter a mesma taxa de aceitação que HSVN-WorstFit nos experimentos A1 e C1, pode-se afirmar que apresentou um melhor desempenho nestes dois experimentos. Isso não significa que esta heurística seja melhor que as demais heurísticas em economia de recursos síncronos, pois teve as menores taxas de aceitação. Já as heurísticas HSVN-VTT e HSVN-VTTBW, obtiveram o melhor desempenho neste quesito em todos os experimentos.

Tabela 4.4 – Custo de mapeamento dos enlaces no cenário 1

Custo do mapeamento dos enlaces no cenário 1		
Grupo A	**Recursos Síncronos**	**Recursos Assíncronos**
HSVN-WorstFit	9216	839680
HSVN-BestFit	0	663552
HSVN-VTT	12288	614400
HSVN-VTTBW	12288	614400
Grupo B	**Recursos Síncronos**	**Recursos Assíncronos**
HSVN-WorstFit	11264	843776
HSVN-BestFit	0	670720
HSVN-VTT	12288	743424
HSVN-VTTBW	12288	743424
Grupo C	**Recursos Síncronos**	**Recursos Assíncronos**
HSVN-WorstFit	11264	843776
HSVN-BestFit	0	670720
HSVN-VTT	12288	874496
HSVN-VTTBW	12288	874496

Outro fator a ser avaliado é o custo de mapeamento dos enlaces virtuais. Como pode ser observado nas Figuras nas Figuras 4.6, 4.7 e 4.8, o custo das soluções obtidas pelos algoritmos HSVN-VTT e HSVN-VTTBW é o mesmo em quase todos os experimentos. Apenas no experimento B4 e C4 a heurística HSVN-VTTBW teve um custo superior, causado pela taxa de aceitação maior.

Sendo assim, ambos tiveram melhor desempenho que as demais heurísticas nesta métrica. As heurísticas HSVN-WorstFit e HSVN-BestFit obtiveram um custo de mapeamento mais alto que as demais heurísticas, mesmo tendo uma taxa de aceitação menor. Ou seja, essas heurísticas alocaram mais recursos para efetuar o mapeamento de um número menor de redes virtuais em relação a HSVN-VTT e HSVN-VTTBW.

Como citado anteriormente, efetuar uma comparação levando em consideração somente taxa de aceitação ou custo de mapeamento pode ser injusto, pois uma abordagem que tem maior taxa de aceitação vai obter uma solução com um custo mais alto, isso nos melhores casos. Sendo assim, para fazer uma comparação mais justa entre as heurísticas, foi calculado o fator médio de ineficiência por grupo. Conforme pode ser observado no gráfico da Figura 4.9, a heurística HSVN-WorstFit foi a abordagem menos eficiente, chegando a custar no mínimo 240% a mais de recursos além do necessário, sendo que tanto HSVN-WorstFit quanto HSVN-BestFit, chegaram a alocar aproximadamente 370% mais recursos do que o necessário, em alguns experimentos. Entretanto, assim como no experimento anterior, a heurística HSVN-BestFit ainda apresentou uma maior eficiência que HSVN-WorstFit.

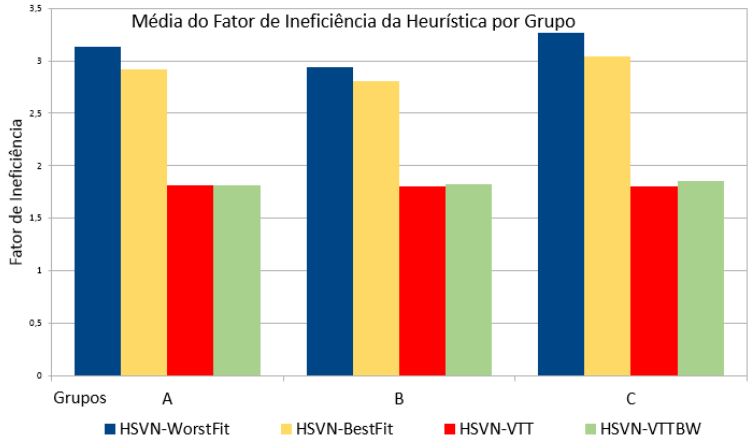

Figura 4.9 – Média de ineficiência de cada heurística por grupo

A heurística mais eficiente foi HSVN-VTT, sendo que no experimento B2, o custo de mapeamento foi 64% mais alto que o custo mínimo para efetuar o mapeamento e no experimento C4 foi alocado 92% a mais de recursos do que o mínimo necessário. Já a heurística HSVN-VTTBW foi menos eficiente devido aos experimentos B4 e C4, onde utilizou respectivamente 94% e 104% a mais de recursos do que o mínimo necessário para efetuar o mapeamento. Isso ocorre porque HSVN-VTTBW acaba mapeando nodos mais distantes para poder aceitar mais redes.

É importante destacar que a média de ineficiência das heurísticas HSVN-VTT se manteve estável mesmo com o aumento do número de redes virtuais, já a heurística HSVN-VTTBW teve um aumento de 1% na taxa de ineficiência média no Grupo B e 4% na taxa de ineficiência média no Grupo C, indicando que sua eficiência piore conforme o aumento de requisições.

O último item avaliado é o tempo de execução. Conforme apresentado na Tabela 4.5, as heurísticas HSVN-WorstFit e HSVN-BestFit encontraram uma solução de maneira mais rápida que as

demais heurísticas, porém as soluções encontradas pelas heurísticas HSVN-VTT e HSVN-VTTBW são melhores em questão de custo de mapeamento e taxa de aceitação.

Tabela 4.5 – Tempo médio necessário para obter a solução de mapeamento (milissegundos)

Tempo médio de execução			
	Grupo A	Grupo B	Grupo C
HSVN-WorstFit	312	355	395
HSVN-BestFit	290	349	372
HSVN-VTT	5013	6251	7382
HSVN-VTTBW	5028	6255	7395

5. CONSIDERAÇÕES FINAIS

A partir do surgimento de redes virtuais de sincronia híbrida, tornou-se necessário explorar diferentes abordagens para efetuar o mapeamento deste tipo de rede virtual, pois, embora tenha sido proposto um modelo matemático para calcular a solução ótima de mapeamento, encontrar a solução ótima é um problema de complexidade NP-Difícil, o que dificulta ou até mesmo inviabiliza calcular a solução ótima em ambientes reais.

Por esse motivo foram adaptados e implementados quatro algoritmos heurísticos para efetuar o mapeamento deste tipo de rede virtual. Para avaliar o desempenho dos algoritmos foram analisados os resultados obtidos pela execução de dois experimentos. O primeiro experimento adotava um substrato de rede e redes virtuais menores, para que fosse possível efetuar uma comparação entre as soluções semi-ótimas e ótima. Já no segundo experimento é adotado um substrato físico com topologia de rede Torus e composto por um número maior de componentes, também é utilizado um número maior de redes virtuais, permitindo analisar o comportamento dos algoritmos quando a demanda é maior.

5.1 Contribuições da pesquisa

Redes virtuais de sincronia híbrida são uma subclasse de problemas, onde redes virtuais e físicas têm subconjuntos de nodos e enlaces com características diferentes, dando prioridade ao mapeamento para um destes subconjuntos. Há outros casos similares relatados na literatura, onde nodos e enlaces possuem, por exemplo, diferentes níveis de segurança ou confiabilidade. Sendo assim, as heurísticas propostas podem ser úteis em uma classe maior de problemas, onde há recursos que apresentam características distintas importantes para o problema em questão.

Então, as principais contribuições desta pesquisa são os quatro algoritmos heurísticos adaptados para suportar redes virtuais de sincronia híbrida, pois ainda não haviam sido exploradas abordagens heurísticas para esse tipo de rede virtual. Todas as heurísticas propostas encontraram soluções válidas em tempo computacional aceitável. Em alguns experimentos com redes virtuais menores, as heurísticas HSVN-VTT e HSVN-VTTBW encontraram soluções com o mesmo custo de mapeamento que a solução ótima. A heurística HSVN-VTTBW também apresentou um aumento na taxa de aceitação em relação a heurística HSVN-VTT e a heurística HSVN-BestFit teve a melhor taxa de aceitação no primeiro experimento, indicando que em alguns cenários esse algoritmo pode ser viável em alguns cenários.

5.2 Pesquisas Futuras

Redes virtuais de sincronia híbrida são um assunto recente, sendo o mapeamento um dos vários tópicos a serem estudados. Mapeamento de forma distribuída, migração de recursos virtuais mapeados no substrato e algoritmos que efetuam o mapeamento em uma única etapa são exemplos de abordagens de mapeamento a serem exploradas para esse tipo de rede virtual. Também podem exploradas diferentes hipóteses sobre o ambiente, como por exemplo, a adoção de diferentes tipos de topologia de rede no substrato físico ou o tratamento das requisições de redes virtuais de maneira *offline*. Os algoritmos heurísticos propostos neste trabalho também podem receber melhorias incrementais, como considerar a largura de banda dos enlaces durante o mapeamento dos nodos virtuais nos algoritmos HSVN-WorstFit e HSVN-BestFit. Outra melhoria que poderia ser aplicada, nesse caso em HSVN-VTTBW, é a verificação da existência de um caminho válido entre o nodo físico candidato a mapear um nodo virtual e o nodo hospedeiro do nodo virtual pai.

REFERÊNCIAS BIBLIOGRÁFICAS

[1] Alkmim, G.; Batista, D.; da Fonseca, N. "Optimal mapping of virtual networks". In: Global Telecommunications Conference (GLOBECOM 2011), 2011 IEEE, 2011, pp. 1–6.

[2] Andersen, D. G. "Theoretical approaches to node assignment", *Computer Science Department*, 2002, pp. 86.

[3] Anderson, T.; Peterson, L.; Shenker, S.; Turner, J. "Overcoming the internet impasse through virtualization", *Computer*, vol. 38–4, Abr 2005, pp. 34–41.

[4] Bays, L. R.; Oliveira, R. R.; Buriol, L. S.; Barcellos, M. P.; Gaspary, L. P. "Security-aware optimal resource allocation for virtual network embedding". In: Proceedings of the 8th International Conference on Network and Service Management, 2012, pp. 378–384.

[5] Bays, L. R.; Oliveira, R. R.; Buriol, L. S.; Barcellos, M. P.; Gaspary, L. P. "Efficient, online embedding of secure virtual networks", Dissertação de Mestrado, Universidade Federal do Rio Grande do Sul, 2013.

[6] Botero, J.; Hesselbach, X.; Fischer, A.; Meer, H. "Optimal mapping of virtual networks with hidden hops", *Telecommunication Systems*, vol. 51–4, 2012, pp. 273–282.

[7] Chandra, T. D.; Toueg, S. "Unreliable failure detectors for reliable distributed systems", *Journal of the ACM*, vol. 43–2, 1996.

[8] Chen, Y.; Li, J.; Wo, T.; Hu, C.; Liu, W. "Resilient virtual network service provision in network virtualization environments". In: Parallel and Distributed Systems (ICPADS), 2010 IEEE 16th International Conference on, 2010, pp. 51–58.

[9] Cheng, X.; Su, S.; Zhang, Z.; Wang, H.; Yang, F.; Luo, Y.; Wang, J. "Virtual network embedding through topology-aware node ranking", *SIGCOMM Comput. Commun. Rev.*, vol. 41–2, Abr 2011, pp. 38–47.

[10] Chowdhury, N.; Boutaba, R. "Network virtualization: state of the art and research challenges", *Communications Magazine, IEEE*, vol. 47–7, July 2009, pp. 20–26.

[11] Chowdhury, N. M. K.; Rahman, M. R.; Boutaba, R. "Virtual network embedding with coordinated node and link mapping". In: IEEE INFOCOM, 2009, pp. 783–791.

[12] Cristian, F.; Fetzer, C. "The timed asynchronous distributed system model", *IEEE Trans. on Parallel and Distributed Systems*, vol. 10–6, 1999.

[13] Dwork, C.; Lynch, N.; Stockmeyer, L. "Consensus in the presence of partial synchrony", *Journal of the ACM (JACM)*, vol. 35–2, 1988, pp. 288–323.

[14] Feamster, N.; Gao, L.; Rexford, J. "How to lease the internet in your spare time", *SIGCOMM Comput. Commun. Rev.*, vol. 37–1, Jan 2007, pp. 61–64.

[15] Fischer, A.; Botero, J.; Till Beck, M.; de Meer, H.; Hesselbach, X. "Virtual network embedding: A survey", *Communications Surveys Tutorials, IEEE*, vol. 15–4, Fourth 2013, pp. 1888–1906.

[16] Fischer, M. J.; Lynch, N. A.; Paterson, M. S. "Impossibility of distributed consensus with one faulty process", *J. ACM*, vol. 32–2, Abr 1985.

[17] Gartner, F. C.; Kloppenburg, S. "Consistent detection of global predicates under a weak fault assumption". In: The 19th IEEE Symposium on Reliable Distributed Systems, 2000.

[18] Haider, A.; Potter, R.; Nakao, A. "Challenges in resource allocation in network virtualization". In: 20th ITC Specialist Seminar, 2009, pp. 20.

[19] Hasan, R.; Machado Mendizabal, O.; Reis De Oliveira, R.; Dotti, F. "A study on substrate network synchrony demands to support hybrid synchrony virtual networks". In: Computer Networks and Distributed Systems (SBRC), 2014 Brazilian Symposium on, 2014, pp. 344–352.

[20] Hasan, R.; Mendizabal, O.; Dotti, F. "Hybrid synchrony virtual networks: Definition and embedding". In: ICN 2014, The Thirteenth International Conference on Networks, 2014, pp. 104–110.

[21] Houidi, I.; Louati, W.; Ameur, W. B.; Zeghlache, D. "Virtual network provisioning across multiple substrate networks", *Computer Networks*, vol. 55–4, 2011, pp. 1011–1023.

[22] Li, X.; Guo, C.; Wang, H.; Li, Z.; Yang, Z. "A constraint optimization based virtual network mapping method". In: 2012 International Conference on Graphic and Image Processing, 2013, pp. 87683X–87683X.

[23] Liu, J.; Huang, T.; Chen, J.-y.; Liu, Y.-j. "A new algorithm based on the proximity principle for the virtual network embedding problem", *Journal of Zhejiang University SCIENCE C*, vol. 12–11, 2011, pp. 910–918.

[24] Lu, J.; Turner, J. "Efficient Mapping of Virtual Networks onto a Shared Substrate", Relatório Técnico, Washington University in St. Louis, 2006.

[25] Matsui, H.; Inoue, M.; Masuzawa, T.; Fujiwara, H. "Fault-tolerant and self-stabilizing protocols using an unreliable failure detector", *IEICE Trans. on Information and Systems*, vol. 83–10, 2000.

[26] Medina, A.; Lakhina, A.; Matta, I.; Byers, J. "Brite: Boston university representative internet topology generator". Capturado em: http://www.cs.bu.edu/brite.

[27] Rahman, M.; Boutaba, R. "Svne: Survivable virtual network embedding algorithms for network virtualization", *Network and Service Management, IEEE Transactions on*, vol. 10–2, June 2013, pp. 105–118.

[28] Schaffrath, G.; Werle, C.; Papadimitriou, P.; Feldmann, A.; Bless, R.; Greenhalgh, A.; Wundsam, A.; Kind, M.; Maennel, O.; Mathy, L. "Network virtualization architecture: Proposal and initial prototype". In: Proceedings of the 1st ACM Workshop on Virtualized Infrastructure Systems and Architectures, 2009, pp. 63–72.

[29] Schneider, F. B. "Distributed systems (2nd ed.)". , Mullender, S. (Editor), ACM Press/Addison-Wesley Publishing Co., 1993, cap. What good are models and what models are good?

[30] Turner, J.; Taylor, D. "Diversifying the internet". In: Global Telecommunications Conference, 2005. GLOBECOM '05. IEEE, 2005, pp. 6 pp.–760.

[31] Veríssimo, P. E. "Travelling through wormholes: a new look at distributed systems models", *ACM SIGACT News*, vol. 37–1, 2006.

[32] Waxman, B. M. "Routing of multipoint connections", *Selected Areas in Communications*, vol. 6–9, 1988.

[33] Yu, H.; Anand, V.; Qiao, C.; Sun, G. "Cost efficient design of survivable virtual infrastructure to recover from facility node failures". In: Communications (ICC), 2011 IEEE International Conference on, 2011.

[34] Yu, M.; Yi, Y.; Rexford, J.; Chiang, M. "Rethinking virtual network embedding: Substrate support for path splitting and migration", *SIGCOMM Comput. Commun. Rev.*, vol. 38–2, 2008, pp. 17–29.

[35] Zhang, M.; Wu, C.; Jiang, M.; Yang, Q. "Mapping multicast service-oriented virtual networks with delay and delay variation constraints". In: Global Telecommunications Conference (GLOBECOM 2010), 2010 IEEE, 2010.

[36] Zhu, Y.; Ammar, M. "Algorithms for assigning substrate network resources to virtual network components". In: INFOCOM 2006. 25th IEEE International Conference on Computer Communications. Proceedings, 2006, pp. 1–12.

Pontifícia Universidade Católica do Rio Grande do Sul
Pró-Reitoria de Graduação
Av. Ipiranga, 6681 - Prédio 1 - 3º. andar
Porto Alegre - RS - Brasil
Fone: (51) 3320-3500 - Fax: (51) 3339-1564
E-mail: prograd@pucrs.br
Site: www.pucrs.br