Paralelização de um Algoritmo para Alocação de Redes Virtuais em GPUs

Mateus Boiani[1], **Guilherme Koslovski**[1]

[1]Ciência da Computação – Universidade do Estado de Santa Catarina (UDESC)

mateuz.boiani@labp2d.joinville.udesc.br, guilherme.koslovski@udesc.br

Resumo. *Um dos desafios dos Provedores de Serviços (SPs) é alocar recursos físicos para hospedar Redes Virtuais (VNs), um problema NP-Difícil. Dentre as soluções propostas na literatura destaca-se a solução heurística VITreeM, que utiliza premissas que ditam o ponto de partida inicial para alocação. Este artigo propõe a paralelização do algoritmo VITreeM utilizando Unidades Gráficas de Processamento (GPU) considerando todos os pontos de partida possíveis.*

1. Introdução

A virtualização de redes de computadores surgiu como uma das tecnologias promissoras para o futuro da Internet [Fischer et al. 2013], difundindo o provisionamento de redes virtuais sobre um conjunto de recursos físicos. Uma rede virtual é um conjunto de elementos (nós de conexão e canais de comunicação) alocados sobre uma infraestrutura física. Sobre uma rede física é possível alocar várias redes virtuais com diferentes características, fornecendo assim uma abstração que permite o desenvolvimento de novos protocolos e serviços [Fischer et al. 2013, Chowdhury et al. 2012].

A virtualização permite a provedores de serviço (SP, *Service Provider*) criarem redes virtuais (VN, *Virtual Network*) a fim de oferecer serviços customizados para usuários através do compartilhamento de recursos de um ou mais provedores de infraestrutura física (InP, *Infrastructure Provider*) [Chowdhury et al. 2012]. Porém, um dos principais desafios enfrentados por SPs é alocar de forma eficiente recursos físicos para hospedar VNs. Este desafio é conhecido como *Virtual Network Embedding* – VNE – e é um problema pertencente à classe NP-Difícil. Diversas soluções heurísticas para VNE surgiram na literatura [Fischer et al. 2013]. Dentre as soluções existentes, o algoritmo VITreeM é uma solução heurística baseada em árvores [de Oliveira and Koslovski 2015] que limita o espaço de busca aplicando abstrações na representação dos dados. Entretanto, acaba se tornando ineficiente em instâncias extensas do problema.

O uso de CPUs tem se mostrado ineficiente para obter soluções em tempo computacional aceitável. Especificamente, unidades gráficas de processamento (GPU, *Graphics Processing Unit*) foram desenvolvidas para acelerar a execução de operações matriciais e vetoriais. Tais operações são aplicáveis no algoritmo VITreeM. Assim, o presente trabalho tem como objetivo paralelizar o algoritmo VITreeM com foco em GPUs. O trabalho esta organizado como segue: a Seção 2 apresenta a definição formal da alocação de redes virtuais e o algoritmo VITreeM, enquanto a Seção 3 apresenta a proposta de paralelização.

2. Alocação de Redes Virtuais

Uma VN é uma composição temporária de elementos virtuais (roteadores, *switches* e enlaces) sobre elementos de rede físicos. InPs oferecem recursos virtualizados através de

interfaces programáveis para SPs, que por sua vez, utilizam os recursos disponibilizados por diversos InPs para criar e distribuir VNs. Desta forma, é possível compartilhar os recursos de um ou mais InPs entre múltiplos usuários (também pode prover a outros SPs), oferecendo serviços customizados [Zhang et al. 2014, Chowdhury et al. 2012].

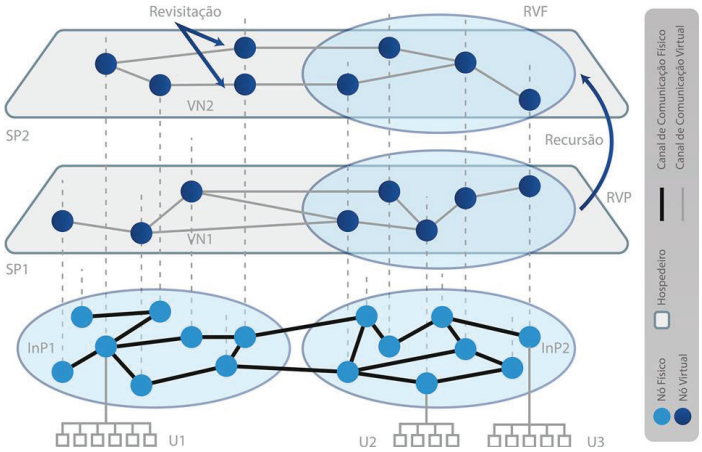

Figura 1. Cenário de Execução de Redes Virtuais.

A Figura 1 apresenta dois InPs (InP1 e InP2) provendo recursos para um SP (SP1). Por sua vez, o SP1 cria uma rede virtual (VN1) e disponibiliza para outro SP (SP2). Neste cenário, SP2 utiliza recursos oriundos de outro SP para criar a rede virtual VN2. A criação da rede virtual VN2 é feita através de recursão. Uma recursão ocorre quando uma ou mais redes virtuais são geradas a partir de outra rede virtual compondo uma hierarquia com relação de pai-filho (RVP-RVF). A propriedade de revisitação permite que um nó físico hospede múltiplos nós virtuais de uma rede virtual.

Um dos principais desafios que um provedor de redes virtuais enfrenta é a decisão de quais recursos físicos serão utilizados para hospedar as requisições. As requisições possuem restrições nos nós e canais de comunicação, que devem ser mapeados em um provedor que possui recursos finitos [Chowdhury et al. 2012]. Cada nó físico está associado a uma capacidade de processamento e cada canal de comunicação está associado a uma largura de banda disponível.

A alocação de redes virtuais pode ser formalizada como um problema de mapeamento de grafos. Redes físicas podem ser modeladas como grafos não direcionados: os vértices representam os nós (*switches* e roteadores) e as arestas os canais de comunicação. VNs também são descritas como grafos, sendo os pesos de nós e arestas a especificação das capacidades solicitadas.

A alocação de VNs sobre uma infraestrutura física é formalmente descrita como segue: dada uma infraestrutura física $I = (N, E, C)$ sendo N um conjunto de dispositivos físicos, E o conjunto de canais de comunicação, e $C(a)$ representando um vetor de recursos disponíveis de forma que $a \in N \cup E$; uma requisição de rede virtual pode ser descrita como $I_{rv} = (N_{rv}, E_{rv}, C_{rv})$ sendo que N_{rv} representa o conjunto de dispositivos virtuais e E_{rv} o conjunto de canais de comunicação virtuais. C_{rv} é aplicado para recursos virtuais indicando a capacidade requisitada (*e.g.*, largura de banda, capacidade de processamento de *switches*). O mapeamento de VNs sobre uma infraestrutura física pode ser descrito como $M = I_{rv} \rightarrow (M_N, M_P)$ no qual $M_N : n^i \in N_{rv} \rightarrow n \in N$ é um mapeamento de dispositivos virtuais sobre dispositivos físicos e $M_P : e^i \in E_{rv} \rightarrow P$ é o

mapeamento de canais virtuais de comunicação sobre caminhos físicos de comunicação. Um mapeamento $M(I_{rv})$ é considerado válido se para todo dispositivo virtual ou canal virtual de comunicação de I_{rv} o mapeamento não excede as capacidades de I, ou seja, $\forall n^i \in N_{rv}, C_{rv}(n^i) \leq C(M_N(n^i))$ e $\forall e^i \in E_{rv}, C_{rv}(e^i) \leq min(C(l) \forall l \in P)$.

Como consequência da complexidade do problema, um número de heurísticas para solucionar o VNE foi proposto na literatura especializada. Entretanto, diversos algoritmos acabam se tornando ineficientes em grafos extensos. Em suma, os trabalhos exploram formulações tradicionais de teoria dos grafos que podem ser otimizadas com técnicas de programação paralela.

2.1. O Algoritmo VITreeM

O algoritmo VITreeM foi selecionado como algoritmo alvo do estudo devido aos resultados obtidos em análises anteriores [de Oliveira and Koslovski 2015]. O algoritmo encontrou soluções eficientes na visão do provedor de serviço ao diminuir a fragmentação do *data center*. Ainda, a taxa de aceitação de requisições foi superior a algumas das soluções propostas na literatura. VITreeM incorpora conceitos e abordagens previamente consolidadas em trabalhos de VNE para definição dos pontos de partida e organização das topologias (representação em árvores).

Inicialmente em VITreeM, a requisição de rede virtual e a infraestrutura física são representadas como grafos, possibilitando desta forma a existência de ciclos. Uma das etapas do algoritmo é a transformação deste grafo para a forma de árvores. Um grafo G é dito na forma de árvore quando satisfaz as condições de conectividade, isso significa que existe um caminho entre dois vértices quaisquer de G. Além disso, G deve ser acíclico.

A solução apresentada por VITreeM é decomposta em quatro etapas: *(i)* inicialmente é realizada a transformação dos grafos de entrada em árvores; *(ii)* a etapa seguinte consiste em definir os nós de partida para alocação; *(iii)* a terceira etapa aplica a heurística de *caixas* para simplificar a representação dos recursos impactando na diminuição do espaço de busca; e *(iv)* ocorre o mapeamento das árvores.

3. Proposta de Paralelização do VITreeM usando GPU

VITreeM utiliza a definição dos pontos de partida para iniciar a comparação dos elementos das árvores. Ocorre que o ponto de partida influencia diretamente no resultado da alocação, e ainda, uma alocação mal efetuada poderá influenciar em uma tentativa de alocação futura. Desta forma, utilizando técnicas de programação paralela e o poder computacional ofertado pelas GPUs através da plataforma CUDA, o presente trabalho aumentará as possibilidades de alocação investigadas por VITreeM. Todos os possíveis pontos de partida serão analisados em paralelo. A Figura 2 apresenta o fluxo do algoritmo VITreeM e a proposta de paralelização do mesmo. Em CPU será executada toda tarefa de inicialização do algoritmo, leitura de arquivo representando infraestrutura física e virtual, a etapa (i) descrita na Seção 2.1 que consiste em transformar os grafos de entrada em árvores, etapa (ii) que deverá preparar todos os pontos de partida possíveis para invocar a função de *kernel* responsável pela etapa (iv) do algoritmo (parte que irá de fato ser executada na GPU). A função de *kernel* executará o mapeamento das árvores de acordo com um ponto de partida passado por referência em sua chamada. Assim teremos n *threads* executando em paralelo na GPU, cada qual executando o mapeamento da mesma requisição de alocação iniciando em pontos de partida diferentes.

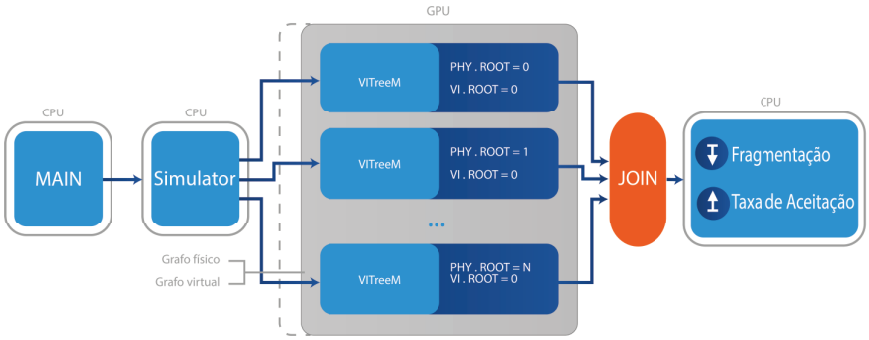

Figura 2. Proposta de Paralelização do Algoritmo VITreeM usando GPU.

Terminado todos mapeamentos pode-se então ser efetuada a análise de fragmentação do *data center* e custo de alocação para cada ponto de partida testado. Ao final da análise será possível, não somente, obter a melhor alocação possível utilizando o algoritmo VITreeM, mas também, determinar se os pontos de partida utilizados no algoritmo original e consolidados na literatura são de fato a melhor opção.

4. Considerações Finais

A alocação de recursos físicos para hospedar VNs é um dos desafios em foco atualmente, que pertencente a classe NP-Difícil. Uma das soluções existentes na literatura é o algoritmo VITreeM, que inicia uma tentativa de alocação por pontos de partida consolidados em trabalhos de VNE. Conforme apresentado na Seção 3, o ponto de partida influencia diretamente no resultado da alocação e também em alocações futuras. O presente trabalho propõe otimizar o resultado de uma ou mais alocações utilizando o algoritmo VITreeM. A otimização será feita combinando técnicas de programação paralela com o uso de GPU, buscando o aumento da taxa de aceitação e diminuição da fragmentação do *data center*.

Referências

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

de Oliveira, R. and Koslovski, G. (2015). VITreeM - um algoritmo baseado em árvores para alocação de infraestruturas virtuais. In *XXXIII Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos (SBRC)*.

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

Zhang, S., Qian, Z., Wu, J., Lu, S., and Epstein, L. (2014). Virtual network embedding with opportunistic resource sharing. *IEEE Trans. Parallel Distrib. Syst.*, 25(3):816–827.