UMA METODOLOGIA HEURÍSTICA BASEADA EM GRASP, VND E VNS PARA A RESOLUÇÃO DO PROBLEMA DE DIMENSIONAMENTO DE REDES IP

Carlos Frederico Marcelo da Cunha Cavalcanti
Marcone Jamilson Freitas Souza
Fernanda Sumika Hojo de Souza
Viviane de Souza Coelho

Universidade Federal de Ouro Preto
Departamento de Computação
Campus Morro do Cruzeiro s/n
Ouro Preto – Brasil 35400-000

{cfmcc, marcone, fersouza, vivianec}@iceb.ufop.br

Resumo

O presente trabalho apresenta uma proposta de formulação e implementação de algoritmos baseados nas técnicas de otimização GRASP (*Greed Randomized Search Procedure*), VND (*Variable Neighborhood Descent*) e VNS (*Variable Neighborhood Search*) para satisfazer a nova geração da Internet, que implementa Qualidade de Serviço e Engenharia de Tráfego. Este contexto surgiu da crescente expansão da Internet e da necessidade de satisfazer a novos requisitos impostos por aplicações mais complexas, tais como transmissões em tempo real, exigindo que caminhos explícitos entre um nó de entrada da rede e um ou mais nós de saída sejam computados. Esta tarefa é também chamada de dimensionamento da rede. Resultados computacionais são apresentados, comprovando que é possível prover uma melhora no dimensionamento da rede através das técnicas propostas.
Palavras-chave: Qualidade de Serviço, Dimensionamento de redes, Engenharia de Tráfego, Metaheurísticas.

Abstract

This work presents a proposal of formulation and implementation of algorithms based on GRASP (Greed Randomized Search Procedure), VND (Variable Neighborhood Descent) and VNS (Variable Neighborhood Search) optimization techniques to satisfy the new Internet generation, which implements Quality of Service (QoS) and Traffic Engineering. Because of the Internet expansion and the necessity to support new requirements demanded by more complex applications, such as real time transmissions, this new context appeared requiring that paths between the ingress and egress nodes are determined. This task is also called network dimensioning. Computational results are presented, proving that it is possible to provide an improvement in the network dimensioning, through the proposed techniques.
Key words: Quality of Service, Network Dimensioning, Traffic Engineering, and Metaheuristics.

1. Introdução

A Internet, como foi proposta inicialmente, mostrou-se muito eficiente para atender aos requisitos do momento histórico em que fora criado, porém, não mais tem conseguido atender aos novos requisitos impostos por aplicações mais complexas que implementam troca de informações em tempo real, além do crescente aumento do número de usuários. Por isso, o atual desafio da Internet vem sendo alocar adequadamente os recursos da rede, para garantir um nível apropriado de qualidade de serviço e o correto funcionamento da rede, mesmo que esta esteja sob um alto nível de utilização.

Em resposta ao crescimento de demanda por qualidade de serviço na Internet, a *Internet Engineering Task Force* (IETF) estabeleceu dois grupos de trabalhos para desenvolver os Serviços Diferenciados (*DiffServ*) [Blake et al., 1998] e Serviços Integrados (*IntServ*) [Braden et al., 1994] com o objetivo de estender a arquitetura dos protocolos da Internet, comumente chamados TCP/IP, em referência aos nomes dos seus dois principais protocolos, para prover qualidade de serviço.

Verificou-se que a questão de prover qualidade de serviço em uma rede IP está estritamente relacionada com a otimização do seu desempenho. Por exemplo, para atender aos requisitos de qualidade de serviço feitos por uma aplicação, poderia ser necessário redirecionar o tráfego não prioritário para outras rotas e redirecionar o tráfego prioritário para as rotas que melhor atendam as restrições e garantias exigidas pela aplicação. Na Internet, tal situação é denominada como uma operação de engenharia de tráfego. Awduche et al. [2002] definem o conceito de engenharia de tráfego como o aspecto da engenharia de rede que lida com a avaliação e otimização de desempenho de uma rede.

Para que os objetivos da engenharia de tráfego sejam atingidos na rede Internet, faz-se necessário definir caminhos explícitos entre os nós de uma rede de tal forma a garantir o mapeamento do tráfego nos diversos enlaces da rede. Na versão corrente da Internet, essa funcionalidade não é encontrada.

Para tal, foi idealizada a arquitetura MPLA (*Multiprotocol Label Architecture*) [Rosen & Callon, 2001] que permite que seja estabelecida uma rota explícita entre um nó de ingresso e um nó de egresso, denominada ER-LSP (*Explicitly Routed- Label Switched Path*) ou simplesmente LSP. Essa arquitetura define um esquema de encaminhamento de pacotes IP baseado em rótulos (*labels*), ao invés de endereços. A implementação do esquema é feita através de um protocolo chamado MPLS (*Multiprotocol Label Switching*) [Rosen & Callon, 2001].

Para que rotas explícitas sejam convenientemente definidas e que atendam aos objetivos da engenharia de tráfego e satisfaçam as restrições impostas, técnicas baseadas em roteamento com restrições (*Constraint-based routing*) são utilizadas. Nesse tipo de roteamento, os caminhos por onde pacotes de informação irão trafegar são determinados considerando-se vários tipos de restrições de tal forma a atender os requisitos de qualidade de serviço exigidos pelas aplicações.

O dimensionamento da rede resolve o problema de encontrar o conjunto de caminhos que satisfaçam os requisitos de qualidade de serviço e de engenharia de tráfego estabelecidos, utilizando para isto os dados relativos a uma topologia, como as características dos enlaces pertences a ela (recursos disponíveis) e um conjunto de especificações de agregados de tráfegos entre o nó de origem e os nós de destino da rede (estimativa de tráfego).

A estimativa do tráfego que irá passar pela rede é uma das entradas necessárias quando se deseja dimensioná-la. Dois modelos de especificações de tráfego são de grande importância dentro deste contexto: o modelo *pipe* (tubo), que descreve o fluxo de tráfego entre um nó de entrada e um nó de saída e o modelo *hose* (funil) que descreve o fluxo entre um nó de entrada e um ou mais nós de saída [Goderis et al., 2000].

Sob a ótica da otimização, o problema de encontrar rotas que satisfaçam a esses requisitos pertence à categoria NP-difícil, o que significa que possui ordem de complexidade combinatorial. Em outras palavras, o esforço computacional para a sua resolução cresce exponencialmente com o tamanho do problema, dado pelo número de nós.

Em termos práticos, isto significa que, na resolução de problemas reais de otimização pertencentes à classe NP-difícil, raramente são encontrados os resultados ótimos. Conseqüentemente, os métodos de solução aplicados a instâncias reais são, em geral, heurísticos, isto é, tais métodos não asseguram que a solução final obtida seja a melhor existente. Dentre esses métodos heurísticos destacam-se as metaheurísticas, as quais, ao contrário das heurísticas convencionais, são providas de mecanismos para escapar de ótimos locais, além de serem de fácil implementação e permitirem incluir, com facilidade, múltiplas restrições no roteamento.

Este trabalho apresenta a formulação e implementação de algoritmos baseados nas técnicas GRASP (*Greedy Randomized Search Procedure*), VND (*Variable Neighborhood Descent*) e VNS (*Variable Neighborhood Search*) para resolver o problema de tráfego na rede Internet gerando uma melhora de desempenho da mesma.

Este trabalho está organizado como segue. A seção 2 descreve o problema abordado. A seção 3 descreve um algoritmo de dimensionamento de redes baseado na arquitetura do projeto TEQUILA. Na seção 4 propõe-se uma nova abordagem para solucionar o problema. Resultados computacionais são apresentados e discutidos na seção 5, enquanto a última seção conclui o trabalho.

2. Descrição do problema

Uma rede é modelada como um grafo direcionado $G=(V,A)$, onde V é um conjunto de nós, representando os roteadores, e A é o conjunto dos arcos representando os enlaces da rede. Um elemento $a_{ij} \in A$ é especificado pelo par $l=(v_{i,ingress}, v_{j,egress})$, onde $v_{i,ingress}$ e $v_{j,egress}$ são os nós onde o fluxo de tráfego entra e sai na rede respectivamente, também denominados nós de ingresso e egresso.

Sendo uma classe de qualidade de serviço h implementada pelo *Per Hop Behaviour* (PHB) [Blake et al., 1998] do tipo h, isto é, pelo seu comportamento em relação a esta, tem-se que cada enlace l possui os seguintes recursos associados com o PHB do tipo h: a capacidade do enlace C^h_l e o atraso do enlace d^h_l .

O atraso total do enlace pode ser decomposto em quatro componentes distintos: processamento, enfileiramento, transmissão e propagação [Bertsekas & Gallager, 1992]. Dependendo das características associadas a um PHB, o atraso assumido pode ser determinístico ou probabilístico, máximo ou médio. Por exemplo, considerando os serviços implementados pelos comportamentos dos tipos EF e AF, num serviço implementado pelo comportamento do tipo EF, o atraso máximo deve ser determinístico e limitado a um valor máximo. Diferentemente, num serviço implementado pelo comportamento do tipo AF, o atraso pode assumir um valor esperado, isto é, com a maior probabilidade de acontecer. Geralmente, a capacidade C^h_l é expressa em Mbps e o atraso d^h_l em milisegundos.

O agregado de ordem (OA^h) corresponde ao conjunto de agregados de tráfego (conjunto de pacotes que recebem um mesmo código de acordo com seu PHB h), que compartilham uma restrição de ordenação, ou seja, uma restrição imposta pela sua prioridade em relação aos demais pacotes. Um tronco de tráfego $TT^h_i \in OA^h$ é um fluxo de tráfego entre um nó de ingresso $v_{TTingress}$ e o conjunto de nós de egresso $V_{TTegress}$, sendo $V_{TTegress} \subseteq (V \setminus v_{TTingress})$. A cada TT^h_i associam-se requisitos de banda de passagem de ingresso mínima e máxima, denotadas respectivamente por $b_{TTingress}$, $\overline{b}_{TTingress}$ e com um atraso fim-a-fim máximo $d_{TT,i}$. O atraso máximo pode ser determinístico ou probabilístico, dependendo

do tipo do PHB h. Para cada nó de egresso $v_{TT,y} \in V_{TTegress}$, a banda de passagem requisitada de egresso é definida como $b_{TT,y}$.

Deve-se determinar no grafo G os caminhos para cada par de nós ingresso-egresso ($v_{TT,ingress}$, $v_{TT,engress}$) dos troncos de tráfego TT^h_i, atendendo aos requisitos, objetivando minimizar a utilização da banda de passagem em todos os enlaces.

Um caminho é um conjunto ER (rota explícita), composto por enlaces l=($v_{i,ingress}$, $v_{j,egress}$), portanto ER $\subseteq A$. Um conjunto de caminhos com o mesmo nó inicial de ingresso ($v_{TT,ingress}$), para um PHB h de qualidade de serviço, define uma árvore T^h_{TT}, cuja raiz é o nó de ingresso.

A abordagem do modelo *hose* é utilizada para determinar de modo eficiente os caminhos para os troncos de tráfego, proporcionando uma melhor utilização dos enlaces do grafo.

Portanto, define-se o problema como: determinar uma árvore T^h_{TT} para cada tronco de tráfego $TT^h_i \in OA^h$, de modo que estas árvores atendam aos requisitos de cada TT^h_i e requisitos de largura de banda disponível em cada enlace C^h_l objetivando minimizar a utilização dos enlaces em todo o grafo.

3. O Algoritmo ND

O algoritmo de dimensionamento ND (*Network Dimensioning*), proposto por Cavalcanti [2000], é o principal algoritmo de alocação de recursos em redes IP baseada na arquitetura proposta pelo projeto TEQUILA (*Traffic Engineering for Quality of Service in the Internet at Large Scale*) [Goderis et al., 2000]. Este projeto tem por objetivo estudar, especificar, implementar e validar um conjunto de definições de serviço e ferramentas de Engenharia de Tráfego para obter garantias quantitativas de QoS fim-a-fim.

O algoritmo ND tem como estratégia a solução de um conjunto de problemas de otimização, um para cada classe de qualidade de serviço. A atual implementação do Módulo de Dimensionamento de Rede está dividida em duas partes. A primeira é responsável por prover a interface com os outros módulos da arquitetura, tratamento das diretivas de política e preparar os dados para o núcleo do módulo. O núcleo do dimensionamento de rede *(ND cor*e), é chamado uma vez por cada classe de qualidade de serviço, pelo bloco dimensionamento de rede e retorna as respectivas rotas explícitas, os LSPs. Esse algoritmo de dimensionamento considera a prioridade das classes de qualidade de serviço, conforme definido pelas diretivas de política. O núcleo utiliza um algoritmo de duas fases. Uma fase de alocação com uma heurística construtiva e uma fase de realocação para eliminar possíveis inviabilidades geradas na primeira fase. É considerada uma inviabilidade as sobrecargas na utilização de um enlace, ou seja, quando a soma das bandas de passagem de todos os troncos de tráfegos alocados em um enlace supera a capacidade deste enlace.

3.1 Fase de alocação do algoritmo ND

Na fase de alocação do algoritmo ND é gerada a solução inicial a partir da alocação individual de cada tronco de tráfego na rede, obedecendo a uma ordenação decrescente em relação ao somatório das bandas de passagem dos nós egressos. A alocação de cada tronco de tráfego é um procedimento heurístico, denotado por *First_Allocate*, baseado em caminhos mínimos entre os nós de ingresso e egresso do fluxo.

Os caminhos de um tronco de tráfego entre os nós de ingresso e egresso são caminhos mínimos na rede e são gerados individualmente para cada dupla. Um pré-processamento na rede é realizado para que os arcos que não possuem banda de passagem disponível para atender aos requisitos não sejam utilizados. Na inexistência de um caminho que atenda aos requisitos entre os nós

de ingresso e egresso são consideradas relaxações no estado presente da rede, como considerar os recursos originais da topologia, para se determinar uma nova alocação.

No caso da topologia original de rede não comportar algum tronco de tráfego por exigência dos requisitos de QoS de um agregado de ordem *OA*, é gerada uma exceção, sinalizando a inviabilidade na alocação da demanda de tráfego naquela topologia de rede.

No modelo *pipe* esses caminhos já são a solução. O uso do modelo *hose* nesta abordagem consiste em considerar fluxos de um tronco de tráfego em um mesmo enlace sendo apenas um, alocando o menor valor entre a capacidade do nó de ingresso e o somatório das capacidades dos nós egressos dos fluxos analisados.

3.2. Fase de realocação do algoritmo ND

O objetivo desta fase é a eliminação de possíveis inviabilidades da solução construída, isto é, caso exista algum arco com alocação maior que sua capacidade então é realizado um procedimento, denotado por *Optimise*, para tentar realocar os troncos de tráfego em outros arcos. Os arcos com as maiores inviabilidades são tratados primeiro, sendo feita apenas uma otimização para cada arco sobrecarregado.

A otimização de um arco consiste em determinar o menor tronco de tráfego, que se realocado, elimina a inviabilidade do arco. Se a sobrecarga for maior do que a maior capacidade alocada no arco ou a realocação deste tronco de tráfego não for possível, é aplicado outro método onde se tenta realocar todos os troncos de tráfego que utilizem o arco. Uma realocação é aceita caso a nova banda de passagem utilizada pelo tronco de tráfego for menor do que a alocação prévia na rede.

Considera-se, ainda, a necessidade de alocar mais recursos que a capacidade corrente do enlace, sendo sempre uma decisão de política tratar esta exceção, que é rejeitar os troncos de tráfego que causaram esta demanda ou aumentar os recursos dos enlaces [Cavalcanti, 2000].

4. Metodologia proposta

4. 1 Algoritmo GRASP aplicado ao Problema de Dimensionamento de Redes IP

O algoritmo proposto, denominado *GRASP_Filtro*, é um algoritmo de duas fases. Na primeira, uma solução é construída por um procedimento baseado na fase de construção da metodologia GRASP [Feo & Resende, 1995], chamado de *First_Allocate_GRASP*. Na segunda fase, a solução construída é submetida a um refinamento através de um método de busca local, no caso, pelas metaheurísticas VND ou VNS [Mladenovic & Hansen, 1997]. A Figura 1 apresenta o pseudo-código do método proposto.

```
procedimento GRASP_Filtro (f(.), g(.), N(.), GraspMax, s)
1.  f* ← ∞;
2.  para (Iter = 1,2,...,GraspMax) faça
3.      First_Allocate_GRASP (g(.),α,s);
4.      se (f(s) < f*) então
5.          s* ← s;
6.          f* ← f(s);
7.      fim-se;
8.  fim-para;
9.  s ← s*
10. BuscaLocal(f(.),N(.),s);
11. s ← s*;
```

```
    12. Retorne s;
fim GRASP_Filtro
```

Figura 1: Pseudo-código do Método *GRASP_Filtro*

Como se observa, o procedimento *First_Allocate_GRASP* é aplicado diversas vezes e apenas a melhor solução construída é refinada, operando como um filtro de soluções (linhas 2 a 8 da Figura 1). Esse procedimento é adotado em função de o custo computacional da busca local ser considerado elevado no problema em questão, tornando proibitivo sua aplicação a cada solução construída.

4.2 Função de Avaliação

O problema é definido como sendo de minimização, ou seja, a solução que apresentar função de avaliação de menor valor é melhor. A função de avaliação adotada foi baseada em hierarquias (com dois níveis), de forma que dois critérios podem ser avaliados. No primeiro nível é computada a soma das bandas de passagem excedentes nos enlaces sobrecarregados e no segundo nível a taxa média de ocupação de todos os enlaces da rede.

Uma função de avaliação hierárquica é composta de diversos níveis que, para comparação entre duas soluções, é considerada a melhor aquela que apresentar melhor valor no nível superior e em caso de empate recorre-se à comparação dos níveis inferiores.

4.3 Fase de Construção: *First_Allocate_GRASP*

Nessa fase, desenvolveu-se o procedimento *First_Allocate_GRASP*, o qual é utilizado para construção de uma solução inicial, sendo uma adaptação do método *First_Allocate* proposto em Cavalcanti [2000], descrito na seção 3.1. Ele recebe como parâmetros o agregado de ordem (*oa*), o grafo (*g*) e o valor *alfa_GRASP* do procedimento GRASP, o qual define o tamanho da lista restrita de candidatos (*LRC*). Nesta fase, a solução é construída elemento a elemento, de forma iterativa. A cada iteração da fase de construção, os troncos de tráfego ainda não alocados de um mesmo agregado de ordem são colocados em uma lista de candidatos (*LC*), seguindo um critério de ordenação relativo à soma das bandas de passagem dos nós egressos. O tamanho da lista de candidatos (*LC*) é relativo ao número total de troncos de tráfego (TT) do agregado de ordem em questão. A partir desse número, uma lista restrita de candidatos é calculada em função do parâmetro *alfa_GRASP*, que determina o grau de gulosidade do método. No próximo passo, uma posição aleatória (*p*) da *LRC* é sorteada, para que o TT desta posição *p* seja alocado. É feita a alocação do TT escolhido, que passa a não pertencer mais a *LRC*, e o tamanho da *LC* é diminuído de uma unidade. O método prossegue da mesma forma, até que todos os TTs deste agregado de ordem tenham sido alocados.

A alocação de um tronco de tráfego é feita por um procedimento heurístico baseado em caminhos mínimos entre os nós de ingresso e egresso do fluxo e são gerados individualmente para cada dupla. Na inexistência de um caminho que atenda aos requisitos entre os nós de ingresso e egresso são consideradas relaxações no estado presente da rede, como considerar os recursos originais da topologia para se determinar uma nova alocação. No caso da topologia original de rede não comportar algum tronco de tráfego por causa dos requisitos de QoS do agregado de ordem, é gerada uma exceção, sinalizando a inviabilidade na alocação da demanda de tráfego naquela topologia de rede.

4.4 Fase de Busca Local

Construída uma solução inicial, foram desenvolvidas duas metodologias para refinar essa solução, a saber:

4.4.1 Método VND (*Variable Neighborhood Descent*)

No método VND, proposto por Mladenovic & Hansen [1997], explora-se o espaço de soluções através de trocas sistemáticas de estruturas de vizinhança. Uma nova solução é aceita somente quando for de melhora em relação à solução corrente e retorna-se à primeira estrutura quando isto acontece. A melhora é definida pela função de avaliação de cada solução.

Para a aplicação desse método, foram desenvolvidas duas estruturas de vizinhança, N^1 e N^2, sendo estas baseadas na realocação dos troncos de tráfego relativos aos arcos sobrecarregados de cada solução *s*. Elas diferem entre si somente pelo número de arcos sobrecarregados a serem otimizados. A seguir, é detalhado como são explorados os vizinhos em cada uma dessas vizinhanças.

a) Busca na Vizinhança N^1

Nessa estrutura de vizinhança, um vizinho é gerado otimizando-se um único arco sobrecarregado da solução corrente. Dessa forma, uma solução terá tantos vizinhos quantos forem os arcos sobrecarregados que esta solução possui. Cada vizinho é encontrado escolhendo-se um arco sobrecarregado da solução corrente, e realocando um tronco de tráfego (TT) que utiliza esse enlace, visando retirar sua sobrecarga, se for possível. Para tanto, é escolhido o TT que utilize a menor banda de passagem do enlace, mas que retire a sobrecarga do mesmo. Caso nenhum dos TTs retire a sobrecarga, é escolhido o TT que ocupe o valor mais alto de banda de passagem do enlace.

Para uma solução *s*, é gerado um vizinho *s'* e calculada sua função de avaliação (*fo*). A variação de energia (*delta*) é obtida pela diferença entre o valor da função de avaliação da solução corrente (*fo_corrente*) e o da solução vizinha (*fo_vizinho*). Caso esse *delta* seja melhor que o *melhor_delta* encontrado até então, ele é armazenado como sendo o melhor *delta*. Também são armazenados o *melhor_arco* (que representa o arco que deve ser otimizado), o *melhor_tt* (que representa o tronco de tráfego que deve ser realocado) e o *melhor_vertice* (que representa o vértice de saída do *melhor_arco*). Gerados todos os vizinhos de forma iterativa, é feita o movimento real de mudança de solução, a partir dos valores de melhora armazenados. Para a nova solução, é repetido o mesmo procedimento, até que não seja mais encontrada nenhuma solução de melhora.

b) Busca na Vizinhança N^2

O procedimento de busca na vizinhança N^2 funciona da mesma forma que na vizinhança N^1, com a diferença no número de arcos sobrecarregados otimizados. Na vizinhança N^2, um vizinho é gerado otimizando-se dois arcos sobrecarregados da solução corrente. Os arcos são, então, combinados dois a dois e, dessa forma, o número de vizinhos de uma dada solução é dado por $(n^2-n)/2$, sendo *n* o número de arcos sobrecarregados.

4.4.2) VNS (*Variable Neighborhood Search*)

O Método de Pesquisa em Vizinhança Variável (*Variable Neighborhood Search*, VNS), proposto por Mladenovic & Hansen [1997], também explora o espaço de soluções através de trocas sistemáticas de estruturas de vizinhança. Esse método se difere do método VND por gerar vizinhos da solução corrente de forma aleatória, prevenindo assim a ciclagem, situação que pode ocorrer se alguma regra determinística for usada.

O método VNS desenvolvido utiliza as mesmas estruturas de vizinhança implementadas para o método VND.

A primeira análise sobre a solução inicial s_0 é feita a partir da estrutura de vizinhança N^1, indicada pelo valor da variável *tipo_viz*. A cada iteração, um vizinho qualquer *s'* é gerado através do procedimento *vizinho_qq*, que recebe como parâmetro a solução corrente e o tipo de estrutura de vizinhança corrente (*tipo_viz*). Cada vizinho é encontrado escolhendo-se um arco sobrecarregado da solução corrente, e realocando um TT aleatório que utiliza esse enlace. Se o tipo de vizinhança

corrente for igual a 2, outro arco sobrecarregado e diferente do primeiro sofrerá as mesmas mudanças descritas anteriormente. Esse vizinho é então submetido ao procedimento de busca local VND. Se a solução *s''* encontrada após a busca local for melhor que a solução corrente *s'*, a busca continua de *s''* recomeçando da primeira estrutura de vizinhança. Caso contrário, a busca continuará sendo feita em *s'* a partir da próxima estrutura de vizinhança da solução corrente. Este procedimento é encerrado quando o número de iterações (*num_int*) for maior que número máximo de iterações previamente definido (*VNSMax*).

5. Resultados obtidos e Discussão

O Problema de Dimensionamento de Redes IP, embora seja um problema de importante aplicação, ainda é pouco explorado pela literatura afim. Sendo assim, para avaliar a modelagem heurística proposta neste trabalho, instâncias foram obtidas através do programa NS (*Network Simulator*), utilizando-se o pacote GT-ITM (*Georgia Tech - Internetwork Topology Models*) que contém código capaz de gerar grafos que modelam a estrutura topológica das redes.

A seguir, apresentam-se os resultados computacionais obtidos a partir de testes realizados em cada uma das instâncias geradas. Duas variantes do método proposto *GRASP_Filtro* foram consideradas: *GRASP_Filtro*+VND (algoritmo descrito pela Figura 1, com busca local realizada pelo método VND) e *GRASP_Filtro*+VNS (Algoritmo descrito pela Figura 1, com busca local realizada pelo método VNS). Todos os algoritmos foram implementados na linguagem C e os testes computacionais foram realizados sob o sistema operacional Linux, em um microcomputador com processador K6-2 de 450 Mhz e 198 MB de memória RAM. Os parâmetros do método *GRASP_Filtro*, em suas duas versões, foram definidos de forma empírica a partir de testes. O número de vezes que o método construtivo *First_Allocate_GRASP* é ativado, isto é, *GraspMax*, foi definido como 100, visto que aumentando seu valor não houve melhora significativa na qualidade da solução inicial. O valor *alfa_GRASP* que se mostrou mais adequado foi 0.3. Já o número máximo de iterações VNS utilizado foi *VNSMax* = 15.

Os resultados a seguir foram obtidos utilizando, como teste, uma instância de 12 nós e 15 enlaces. Foram realizadas 30 execuções de cada método, cada qual partindo de uma semente diferente de números aleatórios. Entretanto o mesmo conjunto de sementes foi utilizado em todos os métodos.

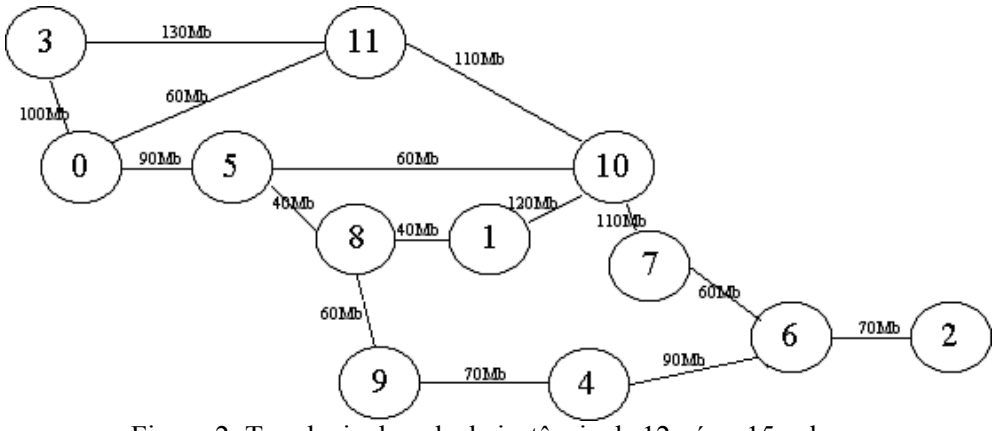

Figura 2: Topologia de rede da instância de 12 nós e 15 enlaces

Na Tabela 1 são apresentados os resultados dos experimentos computacionais. A segunda coluna mostra o melhor resultado obtido a partir da aplicação dos métodos *First_Allocate* e *Optimise* propostos em Cavalcanti [2000], descritos nas seções 3.1 e 3.2 respectivamente. As colunas 3 e 4 referem-se ao melhor resultado e à média dos resultados obtidos com a aplicação dos métodos *GRASP_Filtro*+VND e *GRASP_Filtro*+VNS respectivamente.

A partir da linha 3 da Tabela 1 apresentam-se respectivamente: o tempo de execução, em segundos; os valores da função de avaliação no nível 1 (FO^1), que corresponde à soma das bandas de passagem excedentes nos enlaces sobrecarregados; os valores da função de avaliação no nível 2 (FO^2), que corresponde à taxa média de alocação em todos os enlaces; o número de enlaces sobrecarregados; a taxa média de sobrealocação nos enlaces, em porcentagem, e o desvio dos valores médios de FO^2 encontrados em relação à melhor solução obtida conforme FO^2 pela aplicação de todos os métodos, isto é:

$$\text{Desvio} = \left(\frac{FO^2 Media - FO^2 Melhor}{FO^2 Melhor} \right) \times 100$$

Tabela 1: Resultados obtidos

Métodos Aplicados	***First_Allocate + Optimise***	***GRASP_Filtro* + VND**		***GRASP_Filtro* + VNS**	
	Melhor	**melhor**	**média**	**melhor**	**média**
Tempo de CPU	< 1	1	1,23	1	1,33
FO¹	59	0	48,77	0	48,76
FO²	68,3	57,43	66,41	55,93	66,27
Enlaces sobrecarregados	5	0	3,9	0	3,9
Taxa média de sobrealocação nos enlaces	15	0	17,98	0	17,98
Desvio	-	18,74		18,49	

Observa-se que o método descrito em Cavalcanti [2000], denotado por *First_Allocate+Optimise*, é determinístico, ou seja, sempre que é invocado retorna o mesmo resultado para um dado problema. Dessa forma, a melhor solução encontrada para o problema-teste possui 5 enlaces sobrecarregados, com uma soma de bandas de passagem excedentes de 59 nos enlaces sobrecarregados e uma taxa de alocação média de 68,3% em todos os enlaces.

Combinando a metaheurística *GRASP_Filtro* proposta, e o algoritmo de refinamento VND, obteve-se uma solução que apresenta melhoras em relação ao método *First_Allocate+Optimise*, visto que a soma das bandas de passagem excedentes foi 0, ou seja, não há inviabilidades. Também houve melhora com relação à taxa média de alocação dos enlaces, que passou a ser 48,77%. Já combinando-se a metaheurística *GRASP_Filtro* proposta e o algoritmo de refinamento VNS, obteve-se comportamento semelhante.

Através da comparação dos resultados obtidos na aplicação do algoritmo VNS com os resultados obtidos na aplicação do algoritmo VND, pode-se perceber que não houve melhora significativa. Isto se deve ao fato de que, para esta instância, o VND provavelmente chega em uma solução ótima.

Como apresentado acima, em média, os resultados dos 30 testes realizados para os métodos propostos foram melhores do que o método *First_Allocate+Optimise*, que é totalmente guloso. A porcentagem de casos em que a aplicação de cada método mostrou-se melhor em relação ao método *First_Allocate+Optimise* foi de 96,67% no método VND e de 100% no método VNS.

A fim de verificar a robustez do algoritmo, também foram feitos testes em 100 instâncias geradas a partir de uma topologia de 16 nós, com graus de conectividade variáveis. Dessas 100 instâncias, 20 possuíam matrizes de tráfego na abordagem *pipe* e 80 na abordagem *hose*. Dessa forma, foi possível analisar o desempenho dos algoritmos em situações variadas. Os métodos testados foram *First_Allocate+Optimise, GRASP_Filtro+VND* e *GRASP_Filtro+VNS*.

O Gráfico 1 apresenta, para cada método, os valores obtidos conforme o nível hierárquico FO^1 da função de avaliação de uma solução, que corresponde à soma das bandas de passagem excedentes nos enlaces sobrecarregados. Ele refere-se a um subconjunto de 25 das 100 instâncias tanto no modelo

pipe quanto no *hose*. As instâncias 1, 6, 11, 16 e 21 correspondem ao modelo *pipe* e as demais, ao modelo *hose*. São alocados 240 troncos de tráfego no modelo *pipe* e 16 no modelo *hose*, sendo o ganho de banda de passagem uma das vantagens do último modelo, já que se a taxa de ingresso for especificada em um valor maior do que o necessário para carregar o tráfego dos nós de egresso, o modelo automaticamente ajusta esse valor [Cavalcanti, 2000].

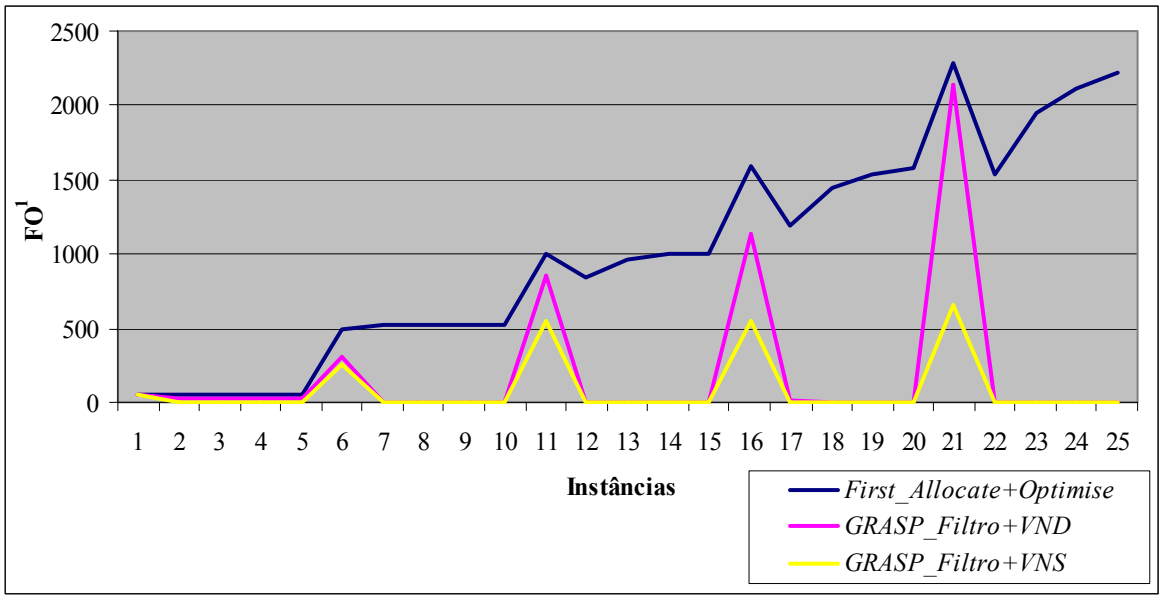

Gráfico 1: Comportamento dos métodos para instâncias com 16 nós e 70 arcos

Nota-se através do gráfico que as inviabilidades, ou enlaces sobrecarregados, foram significativamente reduzidas pelos métodos *GRASP_Filtro+VND* e *GRASP_Filtro+VNS* e ocorrem principalmente no modelo *pipe*, em função do alto número de troncos de tráfego que devem ser alocados, se comparado ao modelo *hose*.

Em relação ao total de instâncias testadas, detectou-se enlaces sobrecarregados em 44 delas no método *GRASP_Filtro+VND* e 41 no método *GRASP_Filtro+VNS*. Comparando os resultados obtidos com os do método *First_Allocate+Optimise*, constatou-se que os métodos propostos apresentaram-se melhores em 100% dos casos.

6. Conclusão

O presente trabalho apresenta uma proposta de formulação e implementação de algoritmos baseados nas técnicas de otimização GRASP (*Greed Randomized Search Procedure*), VND (*Variable Neighborhood Descent*) e VNS (*Variable Neighborhood Search*) para satisfazer a nova geração da Internet, que implementa Qualidade de Serviço e Engenharia de Tráfego.

A partir dos resultados apresentados, pode-se concluir que os métodos propostos têm capacidade de encontrar soluções melhores que as produzidas pelo método *First_Allocate+Optimise*, descrito em Cavalcanti [2000], com redução na soma das bandas de passagem excedentes nos enlaces sobrecarregados e conseqüente redução na taxa de alocação média de todos os enlaces.

Constata-se ainda que, conforme esperado, as soluções encontradas pelo método *GRASP_Filtro+VNS* foram melhores ou iguais às soluções do método *GRASP_Filtro+VND*. Entretanto, é importante ressaltar que o tempo de processamento gasto, principalmente em instâncias no modelo *pipe*, são maiores no método *GRASP_Filtro+VNS* se comparado aos outros métodos.

Sendo assim, deve-se fazer uma avaliação entre os resultados obtidos e o tempo computacional gasto por este método para se saber se é ou não vantajoso o seu uso.

Referências Bibliográficas

AWDUCHE, D.; MALCOLM, J.; AGOGBUA, J.; O'DELL, M. & MCMANUS, J., *Requirements for Traffic Engineering over MPLS*, IETF RFC 2702, 1999.

BERTSEKAS, D. & GALLAGER, R., *Data Networks*, Prentice Hall, 1992.

BLAKE, S.; BLACK, D.; CARLSON, M.; DAVIES, E.; WANG, Z. & WEISS, W., *An Architecture for Differentiated Services*, IETF Informational RFC 2475, 1998.

BRADEN, R.; CLARK, D. & SHENKER, S., *Integrated Services in the Internet Architecture: an Overview*, Internet RFC 1633, 1994.

CAVALCANTI, C.F.M.C., *Algoritmo de Dimensionamento de Rede*, Tese de Doutorado, DCC/UFMG, 2000.

FEO, T.A. & RESENDE, M.G.C., Greedy randomized adaptive search procedures, *Journal of Global Optimization*, 6:109-133, 1995.

GODERIS, D. et al., *Functional Architecture Definition and Top Level Design*, Technical Report Tequila Project – Deliverable D1.1, 2000. Disponível em: http://www.ist-tequila.org/deliverables/D1-1.pdf, acesso em janeiro/2004.

MLADENOVIC, N. & HANSEN, P. Variable Neighborhood Search. *Computers and Operations Research*, 24:1097-1100, 1997.

ROSEN, E. & CALLON, R., *Multiprotocol Label Switching Architeture*, RFC 3031, 2001. Disponível em: www.ietf.org/rfc/rfc3031.txt, acesso em janeiro/2004.