UNIVERSIDADE FEDERAL DE CAMPINA GRANDE
CENTRO DE ENGENHARIA ELÉTRICA E INFORMÁTICA
UNIDADE ACADÊMICA DE ENGENHARIA ELÉTRICA

TRABALHO DE CONCLUSÃO DE CURSO

ALGORITMOS DE ALOCAÇÃO DE ROTA E DE COMPRIMENTO DE ONDA EM REDES ÓPTICAS WDM

ALUNO: PAULO RIBEIRO LINS JÚNIOR
ORIENTADOR: PROF. DR. EDMAR CANDEIA GURJÃO

CAMPINA GRANDE, 18 DE AGOSTO DE 2006

TRABALHO DE CONCLUSÃO DE CURSO

ALGORITMOS DE ALOCAÇÃO DE ROTA E DE COMPRIMENTO DE ONDA EM REDES ÓPTICAS WDM

Relatório apresentado como requisito necessário para a disciplina de Projeto de Engenharia Elétrica, obrigatória para conclusão do curso de Engenharia Elétrica da Universidade Federal de Campina Grande.

Aluno: Paulo Ribeiro Lins Júnior

Orientador: Prof. Dr. Edmar Candeia Gurjão

Convidado: Prof. Dr. José Ewerton Pombo de Farias

Biblioteca Setorial do CDSA. Fevereiro de 2021.

Sumé - PB

Agradecimentos

★ À energia que rege o universo e as minúsculas partes que o compõe (seja ela chamada de Deus, Alá, Rah, Tupã, Força ou qualquer outro apelido), por permitir que eu faça parte desse todo tão magnífico;

★ À minha estimada família, que da forma deles sempre me ajudaram e respeitaram a mim e ao meu trabalho. Em especial ao exemplo de força, fé e capacidade de vencer adversidades ao qual carinhosamente chamo de mãe e a grande figura da qual tirei grandes lições e que chamo de pai;

★ Aos meus grandes irmãos e amigos de longa data Jonas Agápito, Jerônimo Silva, Adrian Lívio, Erik Silva, Olímpio Cipriano, Diego Bezerra, José Luís dentre outros sem os quais definitivamente não teria chegado até aqui, e aos mais recentes porém não menos importantes amigos Carlos Danilo, Gilney e Jean, que tornam o dia-dia rotineiro do laboratório mais agradável;

★ À todos os companheiros de luta que durante um ou mais semestres sofreram as provações de ser um aluno de Engenharia Elétrica juntamente comigo;

★ Ao Departamento de Engenharia Elétrica, berço de minha formação, pela oportunidade de crescimento técnico e pessoal. Em especial as figuras de Adail e Rosilda, anjos que tornam nossa passagem pelo curso menos dolorosa;

★ Ao professor Marcelo Sampaio, por ter me dar a oportunidade de trabalhar ao lado de um grande profissional;

★ Ao professor Iguatemi, que iniciou a orientação deste trabalho e sem o qual não o teria concluído;

★ Ao meu orientador Edmar Candeia Gurjão, grande mestre e amigo, por ter sido e ainda ser um guru e uma referência de profissional e ser humano;

★ Aos professores do Departamento de Engenharia Elétrica que cumpriram sua missão como tutores, e me ajudaram a crescer técnico e profissionalmente; também agradeço aos que não cumpriram sua missão como tutores, por terem me dado um ótimo exemplo do que não quero jamais ser.

Sumário

Lista de Figuras

Capítulo 1

INTRODUÇÃO

1.1 Considerações Gerais

Na década de 70 as redes de comunicação eram utilizadas essencialmente para transmissão de voz utilizando comutação de circuito. Qualquer falha ou serviço que viesse a causar modificação na rede, gerava a necessidade de alteração manual da mesma. Muitas dessas ações incluíam o roteamento desses sinais de voz, feitos através da configuração manual dos *switches* pelos administradores da rede.

Com o passar do tempo o número de usuários aumentou e foi incluída a transmissão de dados, aumentando o tráfego na rede e conseqüentemente a necessidade por mais largura de faixa. Surgiu então a necessidade de encontrar um meio de comunicação que pudesse suprir essa necessidade por largura de faixa e velocidade. Para essa finalidade a fibra óptica se mostrou o meio mais eficiente. [SOMANI, 2005]

A primeira geração de redes de fibras ópticas foi utilizada basicamente para aumento da capacidade da rede que a precedeu, sendo que toda a comutação e serviços de rede eram efetuados através de circuitos eletrônicos. Esse tipo de rede mista acabou gerando um gargalo em virtude da nescessidade de conversão O-E-O (óptica - eletro - óptica) que a mesma possuía. A tecnologia predominante de rede física era o SDH/SONET (*Synchronous Digital Hierarchy/Synchronous Optical Network*) e a taxa de transmissão máxima obtida atingiu 40 Gb/s.[SILVEIRA *et al.*, 2003]

Numa segunda geração, objetivando o aumento da capacidade de transmissão passou-se a ter redes totalmente ópticas, baseadas na multiplexação do comprimento de onda do sinal óptico, isto é, nas redes WDM (Wavelength Division Multiplexing). Neste caso, fala-se de taxas de transmissão da ordem de dezenas de Tb/s.

As redes ópticas de segunda geração, comumente denominadas de redes roteadas por comprimento de onda (*Wavelength Routed Networks*), são atualmente as mais usadas nos *backbones* para serviços que demandam alta taxa de transmissão tais como serviços de videoconferência, vídeo sob demanda e de transferência de grandes volumes de dados.

Os usuários destas redes se comunicam através de caminhos ópticos. Esses caminhos ópticos são roteados e comutados pelos nós intermediários através de multiplexadores OADM (*Optical Add-Drop Multiplexer*) e comutadores OXC (*Optical Cross Connect*). Os caminhos ópticos devem ocupar o mesmo comprimento de onda através dos enlaces da rede quando não existirem conversores de comprimento de onda. Esta propriedade é denominada de restrição de continuidade do comprimento de onda.

Um dos pontos mais importantes do projeto e gerência de uma rede óptica roteada a comprimento de onda é o estabelecimento desses caminhos ópticos. Este estabelecimento é feito através da escolha da rota e da alocação dos comprimentos de onda, respeitando-se a propriedade acima referida.

1.2 Objetivo Geral

Este trabalho objetiva estudar e implementar algoritmos de alocação de rota e de comprimento de onda através de um estudo voltado para o problema de RWA (*Routing and Wavelength Assignment*) em redes ópticas WDM roteadas a comprimento de onda.

1.3 Motivação

As redes ópticas WDM tem ganho cada vez mais aceitação como meio de transporte de dados com grandes taxas de transmissão, como dados bancários, redes de pesquisa,

grandes corporações, dentre outras. Um problema importante em redes ópticas WDM é o do roteamento e alocação de comprimentos de onda (RWA). O algoritmo de RWA tem por objetivo selecionar a melhor combinação de rotas e comprimentos de onda, com base em uma função custo pré-determinada, para cada conexão, de forma a maximizar o número de conexões estabelecidas, fazendo com que nenhuma conexão que compartilhe o mesmo enlace use o mesmo comprimento de onda. Dessa forma, um estudo sobre esses algoritmos se torna relevante, tendo em vista a importância dos mesmos para a implementação de redes ópticas mais robustas, e que possuam o mínimo de probabilidade de bloqueio possível, com o objetivo de evitar ao máximo a perda de conexão pelos usuários.

Esse trabalho está organizado de seguinte forma:

- O **Capítulo 1** dá uma visão geral das redes ópticas e apresenta o objetivo e uma justificativa para o trabalho;
- O **Capítulo 2** descreve os algoritmos de roteamento e alocação de comprimento de onda estudados no trabalho;
- O **Capítulo 3** descreve a implementação do algoritmo de RWA escolhido dentre os descritos no capítulo anterior;
- O **Capítulo 4** traz as conclusões e propostas de trabalhos futuros;
- O **Apêndice** traz o código fonte do programa implementado no trabalho.

Capítulo 2

Roteamento e Alocação de Comprimento de Onda

Este capítulo traz um apanhado teórico sobre o tema da alocação de rota e de comprimento de onda em redes ópticas. Ele é a base teórica para a implementação descrita no capítulo seguinte.

2.1 Introdução

Numa rede óptica roteada a comprimento de onda, os usuários finais se comunicam entre si através de canais WDM totalmente ópticos, geralmente referidos como *caminhos ópticos* (*ligthpaths*). Um caminho óptico é usado para suportar uma conexão numa rede óptica WDM roteada a comprimento de onda, como se fosse uma das vias de uma estrada rodoviária. Sendo assim, a cada demanda requerida na rede é estabelecido um caminho óptico entre o nó origem e o nó destino, utilizando um determinado comprimento de onda.

Na ausência de conversores de comprimento de onda, o caminho óptico precisa ocupar o mesmo comprimento de onda em todos os enlaces entre um nó origem e um nó destino da rede. Esta propriedade recebe o nome de *restrição de continuidade do comprimento de onda*, e é muito comum em redes de grande porte, devido principalmente ao alto custo dos conversores de comprimento de onda. Sendo assim, uma conexão só poderá ser estabelecida

numa rede sem conversão de comprimento de onda se o mesmo comprimento de onda estiver disponível em todos os enlaces do caminho óptico (origem – intermediários – destino). A Figura 2.1 ilustra uma rede óptica roteada a comprimento de onda, enfatizando o caminho óptico entre dois nós. Com referência à figura, é habitual considerar que cada *switch* óptico está conectado a um nó de acesso. Como o trabalho não objetiva tratar dos equipamentos da rede e sim de sua lógica, esse conjunto será simplesmente chamado de nó daqui para a frente.

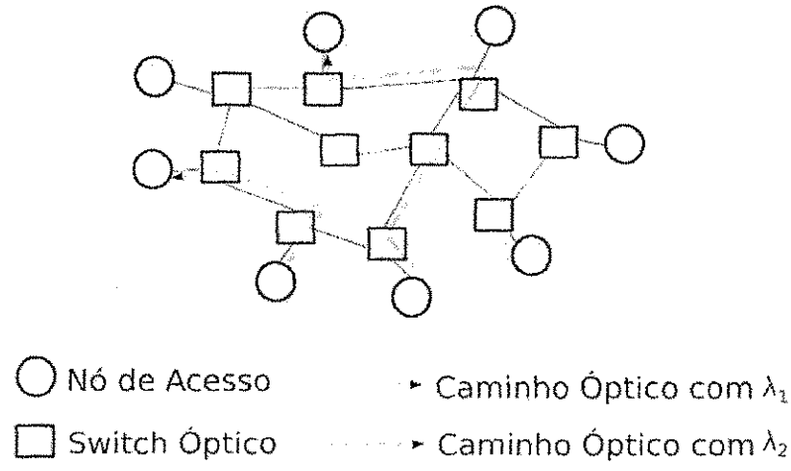

Figura 2.1: Rede óptica roteada a comprimento de onda

2.2 Alocação de Rota e de Comprimento de Onda

Dado um conjunto de conexões, o problema de prover caminhos ópticos à uma rede óptica é chamado de problema de *Alocação de Rota e de Comprimento de Onda* ou simplesmente RWA (*Routing and Assignment Wavelength*). Como o presente trabalho não faz referência à conversão de comprimento de onda, então a cada caminho óptico está associado um único comprimento de onda. Sendo assim, ao se falar em atribuição de comprimento de onda, se fala simultaneamente em alocação de caminho óptico.[ZANG *et al.*, 2000]

Os algoritmos que tratam do problema de alocação de rota e de comprimento de onda (RWA) em uma rede óptica, podem ser classificados em termos do controle da rede, do modo

do estabelecimento da conexão e da forma de modelagem do tráfego utilizado.[PÁDUA, 2001]

O controle da rede pode ser feito de duas formas:

- **centralizado**, onde é suposta a existência de um controle central responsável pelo gerenciamento da rede, mantendo o estado atualizado da rede inteira tal como o estado de ocupação de enlaces e a topologia de nós e enlaces disponíveis ou em falha;
- **distribuído**, onde não existe um controlador central para fazer o roteamento de uma chamada solicitada ou escolher o comprimento de onda a ser utilizado. O gerenciamento e a informação do estado global da rede não ficam disponíveis para os nós ou são disponibilizados apenas após um certo tempo de atraso.

Com relação ao modo de requisição das conexões, os algoritmos podem ser classificados de três formas:

- **estático**, onde o conjunto de solicitações de caminhos ópticos entre pares de nós origem e destino está disponível antecipadamente e o problema consiste em estabelecer os caminhos ópticos de forma a minimizar os recursos alocados pela rede óptica, tais como quantidade de comprimentos de onda ou número de fibras ópticas;
- **incremental ou semi-permamente**, onde os caminhos ópticos são estabelecidos seqüencialmente, e permanecem indefinidamente ou por um longo período de tempo na rede
- **dinâmicos**, onde os caminhos ópticos são estabelecidos na chegada da chamada e liberados após um período finito de tempo. Nos casos semi-permanentes e dinâmicos, o objetivo é minimizar o número de chamadas bloqueadas e/ou a quantidade de recursos utilizados na rede.

Com relação ao modelamento do tráfego, este deve estabelecer o tipo de distribuição de probabilidade que caracteriza a requisição e a duração das chamadas. Comumente são utilizadas as distribuições de Poisson e exponencial negativa para esse modelamento.

A alocação de rota e comprimento de onda para um caminho óptico pode ser feita basicamente de três maneiras:

a) Rota e comprimento de onda selecionados simultâneamente;

b) Rota e comprimento de onda selecionados separadamente, sendo a rota selecionada primeiro e o comprimento de onda depois;

c) Rota e comprimento de onda selecionados separadamente, sendo o comprimento de onda selecionado primeiro e a rota depois.

Alocar rota e comprimento de onda simultâneamente pode se apresentar como um problema muito complexo e não será tratado nesse trabalho. Por esta razão, o problema de RWA pode ser facilitado pela simples separação das funções de roteamento e alocação de comprimento de onda, tratando-os individualmente como sub-problemas.

2.3 Roteamento

A principal função da camada de rede em qualquer rede é determinar o caminho ou os caminhos que o fluxo de dados deve percorrer entre um nó origem e um nó destino. Essa função recebe o nome de roteamento.

A finalidade de um algoritmo de roteamento é simples: dado um conjunto de nós (no caso cada um contendo um roteador) conectados por enlaces, o algoritmo de roteamento descobre a melhor rota, ou o melhor caminho, sendo este geralmente o que possui menor custo. Na prática, os custos de cada enlace podem ser determinados por razões físicas (como comprimento da rota, ou seja, o número de enlaces ou efeitos atenuantes da camada física) ou por razões políticas, (por exemplo, pode existir alguma regra que determine que o roteador X, propriedade de uma empresa qualquer, não deva repassar nenhum tráfego originário de outra empresa).

A formulação de problemas de roteamento geralmente faz uso de uma estrutura denominada *grafo*. Um grafo $G = (N, E)$ é um conjunto de N nós e uma coleção de E arestas, no qual cada aresta é um par de nós do conjunto N. No contexto do roteamento, os nós do grafo representam os pontos de roteamento (no caso os roteadores) – os pontos nos quais são tomadas as decisões de escolha de rota para o tráfego – e as arestas que conectam esses nós

representam os enlaces físicos entre esses nós. Uma abstração gráfica de uma rede é ilustrada na Figura 2.2.

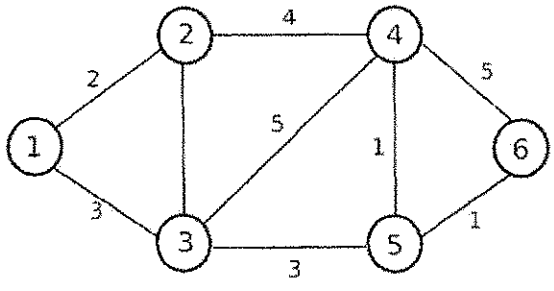

Figura 2.2: Modelo de grafo para redes de comunicações

Como ilustrado na Figura 2.2, uma aresta também possui um valor que representa seu custo. Normalmente o custo de uma aresta pode refletir o tamanho físico do enlace correspondente (por exemplo, um enlace transcontinental pode ter um custo maior que um enlace regional), a banda do enlace ou o custo monetário associado ao mesmo, conforme discutido anteriormente. Nesse trabalho, os custos são atribuídos a cada enlace apenas como dados, sem levar em consideração nenhuma métrica específica. Sendo assim, para qualquer aresta (x, y) em E, denomina-se $c(x, y)$ o custo da aresta entre os nós x e y. Se o par (x, y) não pertencer a E, ou seja, se os citados nós não tiverem nenhuma conexão direta, diz-se que $c(x, y) = \infty$. Caso contrário, considera-se que o nó x é vizinho do nó y. Além disso, no modelamento de redes, sempre considera-se somente grafos não direcionados, isto é, grafos cujas arestas não tem uma direção. Dessa forma a aresta (x, y) é igual à aresta (y, x) e $c(x, y)$ = $c(y, x)$.

Dado que são atribuídos custos às várias arestas (ou enlaces) na abstração do grafo, uma meta natural de um algoritmo de roteamento é identificar a rota ou caminho de menor custo entre fontes e destinos. De uma maneira mais formal, tem-se que uma *rota* ou *caminho* em um grafo $G = (x, y)$ é uma seqüência de nós $(x_1, x_2, ..., x_p)$ tal que cada um dos pares $(x_1, x_2), (x_2, x_3), ..., (x_{p-1}, x_p)$ é uma aresta em E. O custo de um caminho $(x_1, x_2), (x_2, x_3), ..., (x_{p-1}, x_p)$ é simplesmente a soma de todos os custos das arestas ao longo do caminho, ou seja, $c(x_1, x_2) + c(x_2, x_3) + ... + c(x_{p-1}, x_p)$. Dados quaisquer dois

nós x e y, normalmente há muitos caminhos entre os dois, e cada um dos caminhos tem um custo. Um ou mais desses caminhos é o *caminho de menor custo*. Por conseguinte, o problema de menor custo é óbvio: descobrir um caminho entre a origem e o destino que tenha o menor custo. No caso específico de todas as arestas apresentarem o mesmo custo, o caminho de menor custo também é o caminho mais curto, isto é, o caminho com menor número de enlaces entre a origem e o destino.[KUROSE *et al.*, 2006]

2.3.1 Classificação dos algoritmos de roteamento

De um modo geral, os algoritmos de roteamento podem ser classificados como globais ou centralizados, baseado nas informações que cada nós tem do restante da rede.

- **Algoritmo de roteamento global** - calcula a rota de menor custo entre uma origem e um destino usando informações do estado global da rede. Ou seja, o algoritmo considera como dados de cálculo a conectividade entre todos os nós e todos os custos dos enlaces. Isso, por sua vez, exige que essas informações sejam obtidas de alguma forma, antes da realização do cálculo do roteamento. A cálculo em si pode ser executado localmente em algum dos nós (algoritmo global centralizado) ou distribuído em vários nós. Contudo, a principal característica distintiva, nesse caso, é que o algoritmo global tem informações completas sobre conectividade e custo de todos os enlaces da rede. Na prática, algoritmos com informações globais de estado são comumente chamados de *algoritmos de estado de enlace* ou LS (*link state*), tendo em vista que devem estar a par dos custos de cada enlace na rede.

- **Algoritmo de roteamento descentralizado** - calcula a rota de menor custo de modo iterativo e distribuído. Nenhum dos nós tem informação completa sobre os custos de todos os enlaces da rede. Ao invés disso, cada nó começa sabendo apenas os custos dos enlaces diretamente ligados a ele. Entao, por meio de um processo iterativo de cálculo e de troca de informações com seus nós vizinhos, um nó gradualmente calcula o caminho de menor custo até um destino ou um cfonjunto de destinos. A algoritmo de roteamento descentralizado é mais comumente conhecido como *algoritmo de vetor*

de distância ou DV (*distance-vector algorithm*) porque cada nó mantém um vetor de estimativas de custos de um nó até todos os outros da rede.

Uma segunda maneira de classificar algoritmos de roteamento é como estáticos ou dinâmicos. Nos algoritmos de roteamento estáticos, as rotas mudam muito lentamente ao longo do tempo, muitas vezes como resultado de intervenção humana (por exemplo, alteração manual na tabela de roteamento). Já algoritmos de roteamento dinâmicos mudam os caminhos de roteamento à medida que mudam as cargas de tráfego ou a topologia da rede. Um algoritmo dinâmico geralmente é rodado periodicamente como reação direta à mudanças de topologia ou de custos dos enlaces. Ao mesmo tempo que são mais sensíveis à mudança na rede, algoritmos dinâmicos também são mais susceptíveis a problemas como laços de roteamento e oscilação em rotas. [KUROSE *et al.*, 2006]

2.4 Tipos de roteamento

Três tipos de roteamentos podem ser considerados para implementação de redes ópticas WDM, baseado nos algoritmos: roteamento fixo, roteamento alternativo e roteamento adaptativo. A seguir é apresentada uma descrição sobre cada tipo de roteamento: [PÁDUA, 2001][ZANG *et al.*, 2000]

2.4.1 Roteamento fixo

O roteamento fixo é o método mais simples de roteamento, pois sempre escolhe o mesmo caminho para cada demanda origem-destino, caminho este selecionado por algum algoritmo que calcula o caminho mais curto entre dois pontos de um grafo (como o algoritmo de Dijkstra). A conexão entre um par de nós é estabelecida usando-se uma rota pré-determinada. Neste tipo de roteamento, se todos os recursos (comprimentos de onda) são utilizados ao longo do caminho óptico, pode ocorrer bloqueio, o que constitui uma desvantagem desse método. Este método de roteamento também não é tolerante a falhas. Se um enlace falhar, um esquema de roteamento alternativo deve ser implementado dinamicamente.

A Figura 2.3 mostra um exemplo do roteamento fixo do nó 0 ao nó 2.

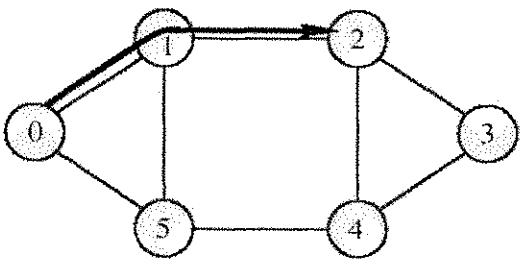

Figura 2.3: Roteamento fixo do nó 0 ao nó 2

2.4.2 Roteamento alternativo

Neste método considera-se a seleção de rotas alternativas à rota mais curta. Em cada nó da rede, deve-se manter uma tabela de roteamento que contêm uma lista ordenada com os números de caminhos fixos para cada nó destino. A ordenação destes nós pode ser feita por qualquer métrica que se escolha, conforme discutido anteriormente, ficando as rotas com menor custo como as primeiras da lista.

Quando uma conexão é requisitada, o nó fonte tenta estabelecer uma conexão com o nó destino através de cada rota usando a tabela de roteamento, começando sempre pela rota de menor custo. Caso a primeira não esteja disponível, a segunda rota mais curta é então utilizada e assim por diante até conseguir uma rota. Caso não seja encontrado nenhum caminho disponível, a requisição é perdida. O roteamento alternativo provê um alto grau de tolerância à falhas nos enlaces reduzindo a probabilidade de bloqueio se comparado ao roteamento fixo.

Ao estabelecer conexões em uma rede WDM roteada em comprimentos de onda é sempre desejável prover algum grau de proteção contra falhas nos nós e nos enlaces da rede através da reserva de alguns comprimentos de onda.

Um enfoque para proteção é configurar dois caminhos ópticos de enlaces disjuntos para qualquer requisição de conexão. Um caminho óptico, denominado caminho óptico principal, será usado para transmitir dados enquanto outro caminho (caminho alternativo)

será usado como *backup* na eventualidade de uma falha no caminho principal.

Uma proteção adicional contra falhas nos nós é obtida escolhendo os caminhos ópticos (principal e alternativos) disjuntos quanto aos nós. O roteamento alternativo provê uma solução simples de proteção, pois a proteção da conexão contra falhas no enlace pode ser feita tomando-se os caminhos alternativos disjuntos ao caminho principal em relação ao enlace e escolhendo um deles como *backup*.

A Figura 2.4 mostra um exemplo de roteamento alternativo entre os nós 0 e 2, mostrando a menor rota (linha cheia) e uma rota alternativa (linha tracejada).

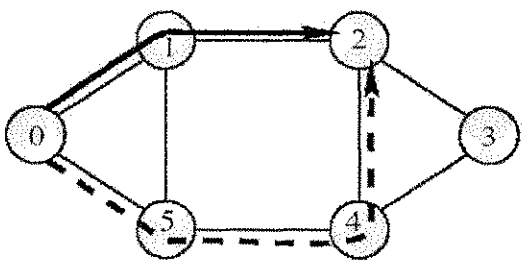

Figura 2.4: Roteamento alternativo do nó 0 ao nó 2, mostrando a menor rota (linha cheia) e uma rota alternativa (linha tracejada)

2.4.3 Roteamento adaptativo

Neste método, a rota entre o nó origem e o nó destino é escolhida dinamicamente em função do estado da rede. O estado da rede é determinado pelo conjunto de todas as conexões em andamento que estão corretamente em progresso.

No roteamento adaptativo de menor caminho, a cada enlace não usado da rede é atribuído um valor de custo (normalmente é um valor unitário) e a cada enlace utilizado, um valor infinito. Os enlaces com conversão de comprimento de onda, quando disponíveis, possuem um custo C_{cc}. Quando uma conexão é requisitada o caminho de menor custo entre o nó origem e o nó destino é calculado dinamicamente. Se mais de um caminho apresenta mesma distância, um deles é escolhido aleatoriamente. Pela escolha adequada do custo C_{cc}, pode-se assegurar que as rotas com conversão de comprimento de onda serão escolhidas

apenas quando os comprimentos de onda para os caminhos ópticos contínuos não estiverem disponíveis. Neste método, a conexão é bloqueada apenas quando não existir uma rota entre os nós origem e nós destino. A grande vantagem do roteamento adaptativo é o seu menor bloqueio de conexão em comparação com os outros roteamentos.

No roteamento adaptativo um esquema de proteção pode ser obtido com o caminho de *backup* sendo estabelecido imediatamente após o caminho principal ter sido estabelecido. O mesmo protocolo de roteamento pode ser usado para se determinar o caminho de *backup*, fazendo-se o custo de seus enlaces iguais a infinito. Pode-se também, no caso do roteamento adaptativo, determinar-se o caminho de *backup* dinamicamente após a falha no caminho principal ter ocorrido.

A Figura 2.5 ilustra o roteamento adaptativo entre o nó 0 e o nó 2.

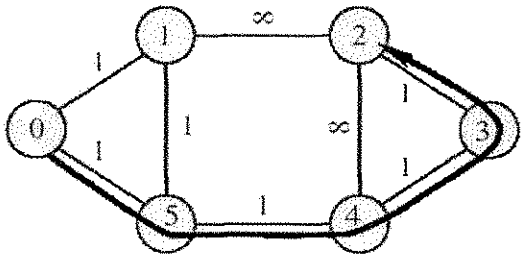

Figura 2.5: Roteamento adaptativo do nó 0 ao nó 2

2.5 Alocação de comprimento de onda

O problema da alocação de comprimento de onda pode ser solucionado de duas formas. Uma é realizando a alocação de comprimento de onda de forma estática, onde se objetiva alocar o menor número de comprimentos de onda para que todos os caminhos ópticos sejam estebelecidos de uma só vez. Para isso, faz-se uso de técnicas de colorimento de grafo. Esse tipo de alocação geralmente é utilizado para redes com tráfego estático. Esta não será utilizada neste trabalho.[ZANG *et al.*, 2000]

Para o caso em que os comprimentos de onda são alocados um de cada vez (tráfego incremental ou dinâmico), métodos heurísticos precisam ser utilizados para atribuir compri-

mentos de onda aos caminhos ópticos.

Dentre as principais heurísticas encontradas na literatura, merecem destaque: [PÁDUA, 2001]

2.5.1 Primeiro Disponível (*Fisrt-Fit* - FF)

O algoritmo "Primeiro Disponível"numera todos os comprimentos, de modo que, quando da procura por comprimentos de onda disponíveis, aqueles de menor número são considerados primeiro que os de número mais alto, ou seja, ele rotula os comprimentos de onda disponíveis de 1 a w nesta ordem, onde w é o número total de comprimentos de onda no enlace. O primeiro comprimento de onda disponível é então selecionado.

Este algoritmo não requer informação global do sistema, e assim seu custo computacional é menor, pois não é necessário pesquisar comprimentos de onda disponíveis em todo o espaço dos comprimentos de onda em cada rota. Atua bem em termos de probabilidade de bloqueio e de imparcialidade de atribuição e na prática é preferido pelo seu pequeno custo computacional e sua baixa complexidade.

2.5.2 Aleatório (*Random* - R)

Este algoritmo inicialmente procura determinar no espaço dos comprimentos de onda o conjunto de todos os comprimentos de onda disponíveis para a rota requisitada. Dentre os comprimentos de onda disponíveis um deles é escolhido aleatoriamente (geralmente com uma probabilidade uniforme). Assim como o FF, o algoritmo R não requer informação global do sistema, possuindo, também um custo computacional menor decorrente dessa característica.

2.5.3 Menos Usado (*Least-Used* - LU)

O algoritmo LU seleciona um comprimento de onda que foi o último a ser utilizado na rede de modo a tentar balancear a carga entre todos os comprimentos de onda. Com isso, facilita a quebra de caminhos ópticos muito longos. Seu desempenho é pior que o do algoritmo R, pois introduz um *overhead* de comunicação adicional, uma vez que necessita

de informações globais da rede para determinar qual foi o último comprimento de onda utilizado. Apresenta também uma necessidade de armazenamento adicional e um maior custo computacional. Não é preferido na prática.

2.5.4 Mais Utilizado (*Most-Used* - MU)

Oposto ao algoritmo LU, o algoritmo MU seleciona o comprimento de onda mais utilizado na rede. Apresenta melhor desempenho que o algoritmo LU e que o algoritmo FF, pois procurar estabelecer as conexões com poucos comprimentos de onda conservando a capacidade ociosa dos comprimentos de onda menos utilizados. Apresenta overhead de comunicação, necessidade de armazenamento e custo computacional similar ao algoritmo LU.

2.5.5 Produto Mínimo (*Min-Product* - MP)

Este algoritmo é utilizado em redes com múltiplas fibras. Para redes com uma única fibra o algoritmo MP se reduz ao algoritmo FF. O algoritmo MP procura organizar os comprimentos de onda na fibra de modo a minimizar o número de redes conectadas com a mesma fibra.

2.5.6 Menos Carregado (*Least-Load* - LL)

Como o algoritmo MP, o algoritmo LL é também projetado para atuar em redes com múltiplas fibras. Este algoritmo seleciona o comprimento de onda que possui maior capacidade residual no link mais carregado ao longo da rota p. Quando utilizado em redes com uma única fibra, a capacidade residual é ajustada em zero ou em um e o algoritmo seleciona o comprimento de onda de menor índice que apresente capacidade residual igual a 1 (com isso se reduz ao algoritmo FF).

O algoritmo LL possui melhor desempenho que os algoritmos MU e FF em termos de probabilidade de bloqueio nas redes com múltiplas fibras.

2.5.7 Soma Máxima (*Max-Sum* - M$\sum$)

O algoritmo da Soma Máxima foi proposto para redes com múltiplas fibras, porém pode ser aplicado também em redes com uma única fibra. Este algoritmo considera todos os possíveis caminhos ópticos na rede com suas rotas pré-definidas e procura maximizar a capacidade caminho restante após o estabelecimento de uma conexão. Assume que a matriz de tráfego (obtida das possíveis requisições de conexão) é definida previamente e que se mantêm estável por um período de tempo.

2.5.8 Perda de Capacidade Relativa (*Relative Capacity Loss* - RCL)

O método RCL está baseado no método M$\sum$. Este método escolhe um comprimento de onda i que minimiza a capacidade relativa de perda. Se um comprimento de onda j ao ser escolhido bloqueia um caminho p_1 e se outro comprimento de onda ao ser escolhido diminui a capacidade dos caminhos p_1 e p_2 , mas não os bloqueia, então o comprimento j deve ser escolhido em detrimento do comprimento i, apesar da capacidade total de perda do comprimento j ser maior que a do comprimento i. O RCL calcula a perda e capacidade relativa para cada caminho em cada comprimento de onda disponível e escolhe o comprimento de onda que minimiza a soma das perdas de capacidade relativa em todos os caminhos. Os métodos MP e RCL podem ser usados com tráfego não-uniforme. O Método RCL é melhor que o método MS nos dois casos (uniforme e não-uniforme).

2.5.9 Perda de Capacidade Relativa Distribuída (*Distributed Relative Capacity Loss* - DRCL)

O método DCRL é implementado usando o algoritmo de Bellman-Ford. Neste algoritmo cada nó permuta tabelas de roteamento com seus nós vizinhos e atualiza sua própria tabela de roteamento. O DRCL introduz em cada nó uma tabela RCL e permite aos nós permutar esta tabela entre si. As tableas RCL são atualizadas de maneira similar às tabelas de roteamento. Cada entrada da tabela RCL é uma tripla composta pelo comprimento de onda w, destino d e pela perda da capacidade relativa $rcl(w, d)$.

Quando uma requisição de conexão chaga ao nó e mais de um comprimento de onda se encontra disponível para o caminho solicitado, o método similarmente aos métodos RCL e M$\sum$ procura trabalhar com estes comprimentos de onda considerando o conjunto de caminhos potenciais para futuras conexões. O método DRCL considera todos os caminhos ópticos do nó- origem da requisição de conexão para todos os outros nós da rede, excluindo o nó-destino da conexão requisitada. O método escolhe o comprimento de onda que minimiza a soma dos $rcl(w,d)$ sobre todos os possíveis destinos d.

2.5.10 Reserva de Comprimento de Onda (*Wavelength Reservation* - WR)

Neste método um dado comprimento de onda em um enlace específico é reservado para o fluxo de tráfego, usualmente tráfego multisalto. Este método reduz a probabilidade de bloqueio para tráfego multisalto, enquanto aumenta a probabilidade de bloqueio para as conexões que atravessam um único enlace da fibra (tráfego salto simples).

2.5.11 Limiar de Proteção (*Protecting Threshold* - PT)

Neste método uma conexão *single-hop* terá atribuído um comprimento de onda apenas se o número de comprimentos de onda disponíveis no enlace estiver acima de um certo nível de limiar. Os métodos PT e WR não especificam quais comprimentos de onda serão escolhidos, mas especificam quais requisições de conexão podem ou não ter um comprimento de onda atribuído em função das condições de uso corrente dos comprimentos de onda. Estes métodos não trabalham sozinhos, mas em conjunto com os outros métodos anteriormente descritos.

Capítulo 3

Implementação do Algoritmo de RWA

Conforme descrito anteriormente, as redes ópticas WDM utilizam uma arquitetura baseada em caminhos ópticos para aproveitar a grande largura de faixa disponível para trasmissão de dados.

Um caminho óptico sem conversão de comprimento de onda é um canal de comunicação estabelecido entre dois nós de uma rede óptica, alocando-se o mesmo comprimento de onda em todos os enlaces do caminho.

Nesta arquitetura o roteamento e a alocação de comprimento de onda aos caminhos ópticos são efetuados procurando otimizar o uso de recursos da rede com o objetivo de diminuir o bloqueio de chamadas e o custo de *hardware* dos comutadores ópticos e meios de transmissão disponíveis, enquanto atendem a requisitos especificados pelos usuários.

O objetivo básico deste trabalho de conclusão de curso foi estudar e implementar algoritmos de alocação de rota e de comprimento de onda. Esse capítulo descreve todo o modelo utilizado bem como os algoritmos tanto para a alocação de rota quanto para a heurística escolhida para a alocação de comprimento de onda, com base no estudo descrito no capítulo anterior.

Na implementação feita neste trabalho, optou-se por um algoritmo de roteamento global, ou seja, onde cada nó conhece o estado atualizado da rede inteira tal como o estado

de ocupação de enlaces e a topologia de nós e enlaces disponíveis. Esses algoritmos tratam de tráfego com características semi-permanente e dinâmica, isto é, com tráfego onde cada requisição é estabelecida seqüêncialmente. Sendo assim, optou-se por trabalhar o problema do RWA como dois sub-problemas, o do roteamento e o da alocação de comprimento de onda, sendo a rota selecionada primeiro e o comprimento de onda depois. O diagrama ilustrado na Figura 3.1 mostra a seqüência de execuções a serem implementadas para a simulação do algoritmo de RWA.

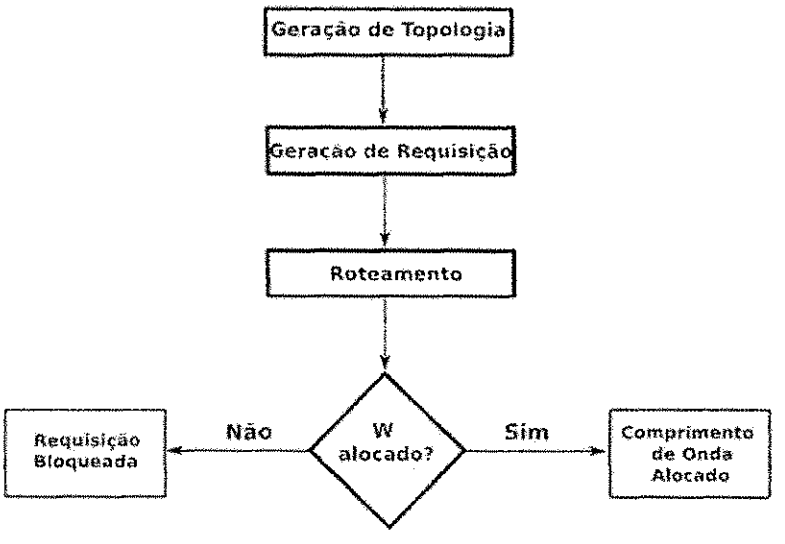

Figura 3.1: Diagrama de funcionamento do algoritmo para roteamento e atribuição de comprimento de onda

3.1 Geração de topologia e de requisições

Na etapa de geração de topologia, o usuário do simulador introduz como parâmetros de inicialização o número de nós, os enlaces e seus respectivos custos. Essa inserção de parâmetros é feita de forma interativa e seqüencial, e estes serão utilizados como dados para a determinação das rotas pelo algoritmo de roteamento.

Em seguida é determinada a requisição ou chamada à qual será alocada uma rota e um comprimento de onda. Nesse ponto, são selecionados um nó origem e um nó destino, para

os quais o algoritmo de RWA atuará. A seleção desses nós pode ser feita propositalmente, ou seja, escolhendo-se dois nós para os quais se queira testar o algoritmo de RWA. Essa escolha dos nós origem e destino também pode ser feita aleatoriamente caso se deseje gerar um número N de requisições e aplicar o algoritmo à elas. No caso, como o tráfego utilizado no modelo é incremental, então cada uma das requisições será gerada, e a rota de menor custo bem como o comprimento de onda serão alocados de forma seqüencial. Assim, se gera a primeira chamada, realiza-se o RWA, depois gera-se a segunda, realiza-se o RWA e assim por diante. Esse número de requisições N pode ser utilizado também como um parâmetro de entrada.

No programa mostrado no Apêndice a geração de topologia é implementada na função `topologia()`.

3.2 Roteamento

O algoritmo de roteamento implementado foi o algoritmo de estado de enlace ou LS (*link state*). Conforme dito, no algoritmo de estado de enlace a topologia da rede e todos os custos dos enlaces são informações disponibilizadas para todos os nós. cada nó pode, portanto, rodar o algoritmo de estado de enlace e calcular o mesmo conjunto de rotas de menor custo como todos os outros nós.[KUROSE *et al.*, 2006]

O algoritmo de estado de enlace utilizado é conhecido como *algoritmo de Dijkstra*, em homenagem a seu inventor. O algoritmo de Dijkstra calcula o caminho de menor custo entre um nó (no caso o nó origem, que será referido como nó u) e todos os outros nós da rede. É um algoritmo iterativo e tem a propriedade de, após a k-ésima iteração, conhecer os caminhos de menor custo para k nós de destino e, dentre os caminhos de menor custo até todos os nós destino, esses k caminhos terão os k menores custos.

A algoritmo de roteamento global consiste em uma etapa de inicialização seguida de um laço. O número de vezes que o laço é rodado é igual ao número de nós da rede. Ao terminar, o algoritmo terá calculado os caminhos mais curtos desde o nó origem u até todos os outros nós da rede.

Sejam as seguintes definições:

- $D(v)$: custo do caminho de menor custo entre o nó origem (u) e o nó destino (v) até essa iteração do algoritmo;
- $p(v)$: nó anterior (vizinho de v) ao longo do caminho de menor custo corrente desde a fonte até v;
- N': subconjunto dos nós; v pertence a N' se o caminho de menor custo entre a origem e v for inequivocadamente conhecido;

O pseudo-código do algoritmo de estado de enlace se encontra listado a seguir.

```
Início
N' = {u}
para todos os nós v
se v for um vizinho de u
então D(v) = c(u,v)
senão D(v) = infinito

Laço:
encontre w não pertencente a N', tal que D(w) é mínimo
adicione w a N'
atualize D(v) para cada vizinho v de w que ainda nao pertença a N'
D(v) = min (D(v), D(w) + c(w,v))

// O novo custo para v é o velho custo para v ou o custo do menor
// caminho conhecido para w mais o custo de w para v
até N' = N
Fim
```

A título de informação, pode-se definir a complexidade desse algoritmo. A complexidade de um algoritmo consiste na quantidade de "trabalho"necessária para a sua execução,

expressa em função das operações fundamentais, as quais variam de acordo com o algoritmo, e em função do volume de dados.

Dados n nós, a complexidade de um algoritmo de roteamento se refere a quantidade de cálculo necessário para efetuar no pior caso o cálculo dos caminhos de menor custo entre os nós origem e destino.

Com relação ao algoritmo apresentado acima, na primeira iteração, é necessário pesquisar todos os n nós para determinar o nó w, que não está em N', e que tem o custo mínimo. Na segunda iteração, é necessário verificar $n-1$ nós para determinaro custo mínimo. Na terceira iteração, verifica-se $n-2$ nós, e assim por diante. Em termos gerais, o número total de nós que é necessário pesquisar em todas as iterações é $n(n+1)/2$. Sendo assim, pode-se afirmar que a complexidade da implementação desse algoritmo de estado de enlace para o pior caso é da ordem de n ao quadrado, ou seja, $O(n^2)$. [KUROSE *et al.*, 2006]

O algoritmo descrito acima encontra todas as menores rotas entre um dado nó orgem e todos os outros nós da rede. Na implementação feita neste trabalho, esse cálculo é feito entre um nó origem e um nó destino especificado, tendo que o objetivo é determinar a rota de menor custo para uma dada requisição entre estes dois nós.

No programa mostrado no apêndice, o algoritmo de roteamento está implementado na função `dijkstra()`.

3.3 Alocação de comprimento de onda

Conforme discutido acima, o algoritmo de roteamento implementado, o algoritmo de estado de enlace, realiza um roteamento fixo, ou seja, ele encontra a menor rota entre um nó origem e um nó destino, sem calcular rotas alternativas. Após definida a rota, segue a etapa de alocação de comprimento de onda.

Nesse trabalho, como o tráfego considerado foi incremental, então os comprimentos de onda são alocados um de cada vez utilizando métodos heurísticos para atribuir os comprimentos de onda aos caminhos ópticos. A heurística escolhida foi a *First-Fit*, descrita no capítulo anterior, tendo como justificativa para a escolha as suas qualidades e facilidade de

implementação.

Inicialmente, o algoritmo *First-Fit* rotula todos os comprimentos de onda, montando assim uma seqüência, que vai de 1 a w nesta ordem, onde w é o número total de comprimentos de onda no enlace. Então, quando da procura por comprimentos de onda disponíveis, aqueles de menor número são considerados primeiro que os de número mais alto, ou seja, ele pesquisa em uma seqüência fixa e tenta alocar o primeiro comprimento de onda onde todos os enlaces estão livres.[TAMASHIRO, 2003]

Uma estrutura necessária para a implementação do algoritmo de alocação de comprimento de onda é a *matriz de ocupação*. A matriz de ocupação é uma matriz de n linhas e m colunas onde n é igual ao número de nós ao quadrado, ou seja, o número total de enlaces que podem existir, considerando a conexão de um dado nó com ele mesmo, e m é o número total de comprimentos de onda, que é um dos parâmetros de entrada a serem definidos pelo usuário na simulação.

Na descrição do algoritmo a seguir, são utilizadas as seguintes definições:

- **Caminho( ):** Vetor contendo a relação de enlaces ópticos que compõe o caminho óptico;
- **comp:** Comprimento do caminho óptico, ou seja, número de enlaces existentes na menor rota entre o nós origem e destino;
- **Ocup (i, j):** Matriz de Ocupação, indexada pelo número do enlace i e pelo comprimento de onda j. O valor Ocup (i, j) = 1 indica que o comprimento de onda j do enlace i está ocupado, enquanto que o valor Ocup (i, j) = 0, qualquer que seja o enlace i, indica que o comprimento de onda j está livre;
- **w:** Índice do comprimento de onda.

Feitas as definições, o estabelecimento do caminho óptico a partir do nó destino utiliza o procedimento descrito pelo algoritmo em pseudo-código descrito abaixo. Este procedimento tenta encontrar uma coluna da matriz de ocupação $Ocup(i, j)$ onde todas as entradas correspondentes aos enlaces da rota selecionada pelo algoritmo de roteamento estão livres.

O contador de comprimentos de onda w é então incrementado para alocar um novo comprimento de onda na seqüência, caso o valor anterior não tenha sido alocado.

```
Ínicio: Algoritmo
//1° passo: cálculo da rota
O caminho mais curto é calculado usando o algoritmo de Dijkstra

//2° passo: procura do comprimento de onda
Alocado = Falso;
w = 1
Enquanto (Alocado não for Verdade) faz
Início: Laço
temp = 0;
para i variando de 1 a comp faz temp = Ocup(Caminho(i),w);
Se temp = 0 então Alocado = Verdade;
Caso contrário faz w = w + 1 (*)
Fim: Laço

//3° passo: atualização da matriz de ocupação
para i variando de 1 a comp faz Ocup(Caminho(i),w) = 1;
Fim: Algoritmo
```

Após escolhido o comprimento de onda para a rota selecionada, passo que acontece no laço mostrado no algoritmo acima, pode-se atualizar a matriz de ocupação mudando o valor de 0 (que indica que o comprimento de onda está livre) para 1 (que indica que o comprimento de onda está ocupado). Isso acontece logo após o fim do laço, conforme mostrado acima. Dessa forma, na próxima requisição, se a rota for composta por algum enlace que compunha a rota anterior, o comprimento de onda alocado para essa segunda requisição deverá ser o seguinte da lista de comprimentos de onda disponíveis.

No caso de se considerar a quantidade de comprimentos de onda limitada, ou seja, de se ter um número máximo de comprimentos de onda, o algoritmo *First-Fit* sofreria uma

pequena alteração. No caso, substituiria-se a linha marcada com um asterisco pelo procedimento abaixo, onde W representa a quantidade máxima de comprimentos de onda. Se o comprimento não for encontrado, ou seja, caso toda a seqüênca de comprimentos de onda seja verificada e nenhum deles esteja disponível para alocação, a solicitação de chamada é bloqueada.

```
Se w < W então w = w + 1
Caso contrário "Requisição de Chamada Bloqueada";
```

Segundo [TAMASHIRO, 2003], este algoritmo tem complexidade de pior caso $O(N^2+NW)$, considerando-se o algoritmo de Dijkstra.

Capítulo 4

Conclusões e Propostas de Trabalhos Futuros

Conforme mencionado, o trabalho objetiva realizar um estudo acerca do problema da alocação de rotas e de comprimentos de onda em redes ópticas WDM, justificado pela importância dessa para o tema de redes ópticas, devido ao grande fluxo de dados que trafega pelos *backbones* atuais.

Esse estudo foi realizado e um dos algoritmos de RWA foi escolhido para ser implementado. Essa escolha foi baseada na facilidade de implementação e condiz com a realidade das redes ópticas reais, que costumeiramente imprementam como heurísticas de alocação de comprimento de onda o *First-Fit* ou o *Random*. O algoritmo implementado realiza o roteamento e aloca os comprimentos de onda de forma satisfatória e rápida, utilizando, mesmo para grandes topologias, com um número de nós superior a 15, e custos variados dos enlaces.

Como sugestão para trabalhos futuros, pode-se fazer uso desse trabalho como ponto de partida para implementações mais rebuscadas, que objetivem, por exemplo, fazer uma avaliação comparativa entre os tipos de roteamento e as diversas heurísticas de alocação de comprimento de onda descritas neste trabalho. Uma sugestão de parâmetro para essa comparação pode ser a avaliação da probabilidade de bloqueio em função da quantidade de comprimentos de onda numa rede óptica dinâmica.

Uma outra possibilidade de trabalho seria a análise destes algoritmos numa rede óp-

tica levando-se em consideração os efeitos da camada física. Esses efeitos podem ser utilizados como métricas para a determinação do custo de cada enlace isoladamente ou em conjunto com outras métricas.

Referências Bibliográficas

[PÁDUA, 2001] PÁDUA, Fabiano João L.; *Proposta de um algoritmo heurístico adaptativo para RWA em redes fotônicas DWDM*, Dissertação de Mestrado, Unicamp,outubro de 2001.

[SOMANI, 2005] SOMANI, Akun K.; *Survavility and Traffic Grooming in WDM Optical Networks*, Cambridge University Press, New York, 2005.

[KUROSE *et al.*, 2006] KUROSE, James F., ROSS Keith W.; *Redes de Computadores e a Internet: Uma Abordagem Top-Down*, 3ª Edição, Editora Addison Wesley, 2006.

[TAMASHIRO, 2003] TAMASHIRO, Silvio Mauro; *Estudo de Algoritmos de Alocação de Rota e Comprimento de Onda em Redes Ópticas*, Dissertação de Mestrado, Unicamp, novembro de 2003.

[ZANG *et al.*, 2000] ZANG, Hui; JUE, Jason P.; MUKHERJEE, Biswanath; *A Review of Routing and Wavelength Assignment Approaches for Wavelength-Routed Optical WDM Networs*, Optical Networks Magazine, janeiro de 2000.

[SILVEIRA *et al.*, 2003] SILVEIRA, Regina Melo; KOVACH, Stephan; CARVALHO, Tereza Cristina Melo de Brito; RUGGIERO, Wilson V.; *Arquitetura, Topologia e Roteamento em Redes Ópticas*, XXI Simpósio Brasileiro de Redes de Computadores, 2003.

Apêndice

Código Fonte da Implementação do Algoritmo de RWA

```
/*
  Nome: Algoritmos de Alocacao de Rota e Comprimento de Onda
  Copyright: GNU
  Autor: Paulo Ribeiro Lins Júnior
  Descrição: TCC
*/

#include <iostream>
#include <cstdlib>
#include <cmath>

using namespace std;


//variáveis globais
int destino, origem, nos = 0;
int custo, *custos = NULL;
int caminho[10];
int Ocup[200][50];
int NumTotalReq, NumLambdaTotal;
int lambda_livre;


//inicialização global das funções
void entrada();
void menu();
void topologia();
void rwa();
void dijkstra(int, int, int, int*);


//função principal
int main(int argc, char **argv )
{
```

```
int i, j;
char opcao[3], l[50];

do {

        menu();

  cin >> opcao;

if ((strcmp(opcao, "d")) == 0) {
topologia();
}


if ((strcmp(opcao, "r") == 0) && (nos > 0) ) {
rwa();
}

} while (opcao != "x");

return 0;
}

//função 'menu'
void menu()
{
    cout << "=======================================\n" << endl;
    cout << "\tAlgoritmos de RWA\n" << endl;
    cout << "\n=======================================\n" << endl;
    cout << "\n" << endl;
    cout << "Opcoes:\n" << endl;
cout << "\t d - Adicionar Topologia\n"
       "\t r - RWA\n"
      "\t CTRL+c - Sair do programa\n" << endl;
    cout << ">>> ";
}


//função 'topologia', que monta a topologia da rede de forma interativa
void topologia()
{
int i, j;

do {
cout << "\nInforme o numero de nos ( no minimo 2 ): ";
cin >> nos ;
cout << endl;
} while (nos < 2 );

if (!custos)
```

```
free(custos);
custos = (int *) malloc(sizeof(int)*nos*nos);
for (i = 0; i <= nos * nos; i++)
custos[i] = -1;

cout << "\n\n" << endl;
    cout << "Entre com os Enlaces:" << endl;
    cout << endl;
do {
do {
cout << "Origem da enlace (entre 1 e " << nos
                << " ou '0' para sair): ";
cin >> origem;

} while (origem < 0 || origem > nos);

if (origem) {
do {

cout << "Destino do enlace (entre 1 e " << nos
                    << ", menos" << origem << "): ";
cin >> destino;

} while (destino < 1 || destino > nos || destino == origem);

do {
cout << "Custo (positivo) do enlace do no " << origem
                    << " , para o no" << destino << "): ";
            cin >> custo;
            cout << endl;
} while (custo < 0);

custos[(origem-1) * nos + destino - 1] = custo;
}

} while (origem);

}


//função 'rwa', que utiliza a função 'dijkstra' para encontrar
//as menores rotas entre dois nós
void rwa()
{

    cout << "\n\n" << endl;
    cout << "Geracao de Requerimentos";
    cout << endl;
    entrada();
```

```
//matriz de ocupação: relaciona os enlaces individualmente com os
//comprimentos de onda um a um, de forma a formar uma matriz
//que será usada no WA. Essa matriz é do tipo - Ocup[numero total
//de enlaces][numero total de lambdas]

int Ocup[nos*nos][NumLambdaTotal];

//O loop 'while' abaixo faz o papel do "tempo de simulação". No caso,
//a simulação roda enquanto a condição NumTotalReq > 0 for verdadeira,
//sendo NumTotalReq um parâmetro de entrada do programa.

while (NumTotalReq > 0)
{

     srand(NumTotalReq);
     int i = 1 + rand() % nos;
     cout << "Origem:" << i << endl;

     srand(NumTotalReq^2);
     int j = 1 + rand() % nos;
     cout << "Destino:" << j << endl;

dijkstra(nos, i,j, custos);
cout << "\n";


    NumTotalReq--;

    }


cout << "<Pressione ENTER para retornar ao menu principal>" << endl;

}


// "sub-função" 'entrada', utilizada na função RWA
void entrada()
{

     cout << "Numero total de requerimentos: ";
     // a variável 'NumTotalReq' representa o numero total de requerimentos
     cin >> NumTotalReq;
     cout << "\n" << endl;
     cout << "Numero de comprimentos de onda total: ";
     // a variável 'NumLambdaTotal' representa o numero total de lambdas
     cin >> NumLambdaTotal;
     cout << "\n" << endl;
}
```

```
//"sub-função" 'dijkstra', utilizada na função RWA:
//implementação do algoritmo de roteamento de Dijkstra
void dijkstra(int nos,int origem,int destino,int *custos)
{
int i,v, cont = 0;
int *ant, *tmp;
int *z;     /* nos para os quais se conhece o caminho minimo */
double min;
double dist[nos]; /* vetor com os custos dos caminhos */


/* aloca as linhas da matriz */
ant = new int(nos);
tmp = new int(nos);
if (ant == NULL) {
cout << "** Erro: Memoria Insuficiente **" << endl;
exit(-1);
}

z = new int(nos);
if (z == NULL) {
cout << "** Erro: Memoria Insuficiente **" << endl;
exit(-1);
}


for (i = 0; i < nos; i++) {
if (custos[(origem - 1) * nos + i] !=- 1) {
ant[i] = origem - 1;
dist[i] = custos[(origem-1)*nos+i];
}
else {
ant[i]= -1;
dist[i] = HUGE_VAL;
}
z[i]=0;
}
z[origem-1] = 1;
dist[origem-1] = 0;

// Laco principal
do {

/* Encontrando o no que deve entrar em z */
min = HUGE_VAL;
for (i=0;i<nos;i++)
if (!z[i])
if (dist[i]>=0 && dist[i]<min) {
min=dist[i];v=i;
```

```
}

/* Calculando as distancias dos novos vizinhos de z */
if (min != HUGE_VAL && v != destino - 1) {
z[v] = 1;
for (i = 0; i < nos; i++)
if (!z[i]) {
if (custos[v*nos+i] != -1 && dist[v] + custos[v*nos+i] < dist[i]) {
    dist[i] = dist[v] + custos[v*nos+i];
ant[i] =v;
    }
         }
}
} while (v != destino - 1 && min != HUGE_VAL);

/* Mostra o Resultado da busca */
cout << "\tDe " << origem << "para " << destino << " : \t" << endl;
if (min == HUGE_VAL) {
cout << "Nao Existe" << endl;
cout << "\tCusto: \t- " << endl;
}
else {
i = destino;
i = ant[i-1];
while (i != -1) {
tmp[cont] = i+1;
cont++;
i = ant[i];
}

for (i = cont; i > 0 ; i--) {
cout << tmp[i-1] << "->" ;
//printf("%d -> ", tmp[i-1]);
}
cout << "" << destino << endl;

cout << "\n\tCusto:" << (int) dist[destino-1] << endl;
}

//alocação de comprimento de onda (first-fit)
int j;

if(cont>0)
{
       cout << "Comprimento do Caminho" << endl;
   for(i=0;i<=cont;i++) cout << caminho[i];
}
        cout << endl;

for(i=0;i<NumLambdaTotal;i++)
```

```
    {
     for(j=0;j<=(cont-1);j++)
     {
       if( Ocup[(caminho[j]-1)*nos + caminho[j+1] - 1][i] != 0 )
             j = cont + 1;
     }

     if(j == cont)//&&(cont>0)
     {
        lambda_livre = i;
        i = NumLambdaTotal + 1;
     }

    }

    if (i == NumLambdaTotal)//&&(cont>0)
{
cout << "Chamada Bloqueada!!" << endl;
}

else
{
    cout << "Comprimento alocado " << lambda_livre+1 << endl << endl;
    }

// atualização da matriz de ocupação após
// alocação do comprimento de onda

for(j=0;j<=(cont-1);j++)
     {
       Ocup[(caminho[j]-1)*nos + caminho[j+1] - 1][lambda_livre] = 1;
     }
}
```

As figuras a seguir ilustram a execução do código-fonte acima descrito.

A Figura 1 mostra o menu iterativo, onde pode-se optar por inserir uma topologia nova ou realizar a alocação de rota e de comprimento de onda de uma topologia já inserida.

```
==========================================================================

        Algoritmos de Alocacao de Rota e Comprimento de Onda

==========================================================================


Opcoes:

        d - Adicionar Topologia
        r - RWA
        CTRL+c - Sair do programa

>>>
```

Figura 1: Menu do programa

A inserção da topologia é feita inserindo-se cada enlace individualmente (nó origem, destino e custo), conforme ilustrado na Figura 2.

```
>>> d

Informe o numero de nos ( no minimo 2 ): 4



Entre com os Enlaces:

Origem do enlace (entre 1 e 4 ou '0' para sair): 1
Destino do enlace (entre 1 e 4, menos 1): 2
Custo (positivo) do enlace do no 1 , para o no 2): 1

Origem do enlace (entre 1 e 4 ou '0' para sair): 1
Destino do enlace (entre 1 e 4, menos 1): 3
Custo (positivo) do enlace do no 1 , para o no 3): 1

Origem do enlace (entre 1 e 4 ou '0' para sair): 2
Destino do enlace (entre 1 e 4, menos 2): 3
Custo (positivo) do enlace do no 2 , para o no 3): 1
```

Figura 2: Inserção de topologia

Depois de inserida a topologia, a alocação de rota e comprimento de onda é aplicada, gerando como resultado a rota e o comprimento de onda escolhidos, conforme ilustrado pela Figura 3, que traz o resultado da simulação repetida três vezes entre os mesmos pares de nós de forma a exemplificar a alteração na tabela de ocupação, conforme descrito no capítulo sobre a implementação dos algoritmos.

```
>>> r


Geracao de Requisições

Numero total de requerimentos: 4

Numero de comprimentos de onda total: 2

Origem: 1; Destino: 4
Rota: 1 -> 3 -> 4
Custo:2
Comprimento alocado : 1

Origem: 1; Destino: 4
Rota: 1 -> 3 -> 4
Custo:2
Comprimento alocado : 2

Origem: 1; Destino: 4
Rota: 1 -> 3 -> 4
Custo:2
Chamada Bloqueada!!
```

Figura 3: Exemplo da alocação de rota e comprimento de onda implementada