Algoritmo Memético Aplicado ao Problema de RWA Estático em WDM com e sem conversão

Tatiane Regina Bonfim[1], **Akebo Yamakami**[1] , **Fabiano J. L. Pádua**[2] , **Edson Moschim**[2]

[1]Departamento de Telemática – Faculdade de Engenharia Elétrica
Universidade Estadual de Campinas – Caixa Postal 6101 – 13081-970 – Campinas, SP

[2]Departamento de Semicondutores Instrumentação e Fotônica
Faculdade de Engenharia Elétrica – Universidade Estadual de Campinas
Caixa Postal 6101 – 13081-970 – Campinas, SP

tatiane,akebo@dt.fee.unicamp.br, padua,moschim@dsif.fee.unicamp.br

Abstract. *In this work we present a memetic algorithm applied to routing and wavelenght assignment (RWA) problem in an optical WDM network with and without conversion, in static case. The algorithm is composed by two procedures: the first one finds a lightpath for routing and the second one assigns wavelenght to each route. We applied a memetic algorithm using the fitness function, which works with the sum of involved connections blocking and distance (cost) in lightpath, to evaluate the solution of the problem. We present simulation and results based on studies of two networks.*

Resumo. *Neste trabalho apresentamos uma proposta de um algoritmo memético aplicado ao problema de roteamento e atribuição de comprimentos de onda (RWA) em uma rede ótica WDM, com e sem conversão, de forma estática. O algoritmo é dividido em dois sub-algoritmos: encontrar um caminho para roteamento e atribuir um comprimento de onda ao mesmo. Aplicamos um algoritmo memético utilizando a função de fitness, que trabalha com a soma do bloqueio das conexões e da distância (custo) envolvidos em um caminho ótico, para avaliar as soluções do problema. Apresentamos simulações e resultados baseados em estudos de duas redes.*

1. Introdução

A técnica de multiplexação por divisão no comprimento de onda (WDM) em fibras óticas vêm crescendo conjuntamente com técnicas de encaminhamento de informações por uma rede, mostrando um futuro promissor em telecomunicações. Em uma rede WDM, os usuários finais se comunicam através de conexões por fibra ótica, denominadas (*lightpaths*). No encaminhamento utilizando comprimentos de onda (WRON), aplica-se as técnicas de reutilização de comprimentos de onda nas fibras óticas ao longo dos enlaces (*links*), por toda a rede. O sistema em conjunto é denotado como redes óticas ou redes fotônicas.

Em redes WDM, um problema importante é o de roteamento e atribuição de comprimentos de onda (RWA - Routing and Wavelenght Assignment), que envolve a seleção da melhor combinação de rota e comprimento de onda para uma determinada conexão (demanda). Diversos algoritmos vêm sendo criados para otimização deste problema e com o intuito de aumentar a quantidade de demandas atendidas na rede [Ahang et al., 2002].

Neste trabalho propomos uma técnica baseada em algoritmos meméticos, os quais usam combinações genéticas em conjunto com uma busca local para encontrar soluções para o problema

de RWA com e sem conversão. Estas soluções são avaliadas por meio de uma função de *fitness*. Usando mutações e cruzamentos genéticos, pode-se recombinar as demandas até que se consiga ter o máximo de demandas simultâneas dentro da rede, com mínima quantidade de comprimento de onda e minimizando o número de bloqueios dentro da rede, cuja topologia é dada.

Em [Ali, 2001], [Ali et al., 2000] foi aplicado um algoritmo genético para resolver o problema de RWA, com o intuito de minimizar o número de bloqueios na rede. Neste trabalho utizamos os mesmos exemplos numéricos de [Ali, 2001] com o objetivo de compararmos os seus resultados com os obtidos pelo algoritmo memético.

Este artigo está dividido em seis Seções. Na Seção 2 apresentamos uma explanação a respeito de redes óticas. Na Seção 3 são apresentados os método de algoritmos genético e memético. O funcionamento do algoritmo aqui proposto é descrito na Seção 4. Na Seção 5 apresentamos as simulações e os resultados. Por último as conclusões são apresentadas na Seção 6.

2. Redes Óticas

Nos dias de hoje o estado da arte das comunicações óticas é a tecnologia WDM empregada em redes óticas [Murthy and Gurusamy, 2002]. Diversos equipamentos vêm sendo desenvolvidos para o melhoramento desta tecnologia, como por exemplo, as chaves óticas (*Optical Crossconnect* – OXC) e os amplificadores óticos (AO). A (Figura 1) mostra um exemplo de um sistema ponto-a-ponto. A rede ótica tem por finalidade a transmissão e recepção de informações através de roteamentos baseado em encaminhamento e atribuição de comprimentos de onda por um caminho ótico. Mais detalhes podem ser vistos em [Murthy and Gurusamy, 2002], [Ramaswami and Sivarajan, 1998].

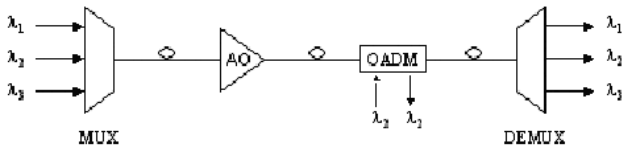

Figura 1: Exemplo de uma comunicação ótica WDM ponto-a-ponto

As redes óticas WDM são formadas por pares de nós origem-destino que se comunicam através de enlaces óticos (EO), também chamados *links*. Uma rota entre um nó origem e destino é chamada de caminho ótico (*lightpath*). Os caminhos óticos podem envolver um ou mais nós da rede. O caminho ótico necessita do uso de um comprimento de onda (um canal), ou seja, cada tipo de informação ou cada usuário está associado a um comprimento de onda específico. Um nó da rede é formado por transmissor/receptor, multiplexador/demultiplexador (MUX/DEMUX), amplificador ótico e Adiciona/Extrai (*Add/Drop* - OADM), o qual tem por finalidade extrair ou adicionar ao sistema um determinado comprimento de onda. As informações chegam pelo OADM, onde são multiplexados os w comprimentos de onda, passam pelo transmissor e, em seguida, conecta-se a um amplificador ótico de linha (AO). Depois passam por um caminho ótico até chegar ao destino. Continuando, o sinal chega ao receptor, o qual tem um demultiplexador (DEMUX), onde os comprimentos de onda são separados novamente e, em seguida, é direcionado ao OADM. Caso possua nós intermediários é usada a chave OXC para chavear a luz de um nó anterior para o próximo nó, que pode ser outro nó intermediário ou até mesmo o nó destino.

Assim, a cada demanda requerida (*requests*) na rede é estabelecido um caminho ótico entre o nó origem e o nó destino, utilizando um determinado comprimento de onda. Dentro das redes óticas

as demandas podem ser encaminhadas pela rede de duas maneiras: usando o mesmo comprimento de onda (sem conversão) ou diferentes comprimentos de onda (com conversão).

Em redes sem conversão de comprimento de onda, um caminho ótico usa o mesmo comprimento de onda em todos os enlaces óticos que o compõem. Esta exigência é chamada de restrição de continuidade de comprimento de onda. Cada enlace ótico tem um número fixo de comprimentos de onda, disponibilizando vários canais, conforme a tecnologia WDM. No RWA sem conversão, duas conexões que compartilhem um *link* comum devem utilizar comprimentos de onda distintos. Em redes com conversão, pode-se manter diferentes comprimentos de onda pelos enlaces das rotas das conexões. Neste tipo de encaminhamento, é necessária a utilização de conversores nos roteadores para converter o comprimento de onda nos nós intermediários. A (Figura 2) apresenta um exemplo de roteamento e atribuição de comprimento de onda sem conversão.

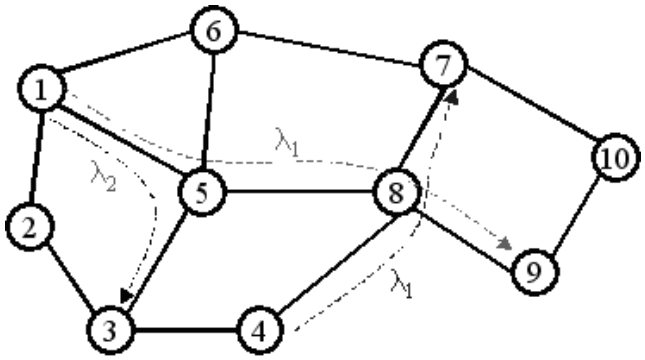

Figura 2: Exemplo de RWA sem conversão

Com o crescimento do uso das redes óticas foi necessário o desenvolvimento de algoritmos que procurem otimizar o atendimento das demandas simultâneas dentro da rede. Com isso tem-se uma maximização de demanda e minimização de bloqueios na rede. Estes algoritmos trabalham com roteamentos e atribuição de comprimentos de onda (RWA) de redes óticas.

Existem duas formas de trabalhar com problemas de RWA: estática, que foi abordada neste trabalho, ou dinamicamente. Algoritmos RWA que trabalham de forma estática, têm como objetivo planejar a distribuição de demandas dentro da rede. No caso dos algoritmos RWA que trabalham de forma dinâmica (*on-line*), não ocorre esta distribuição planejada de demanda, pois cada demanda que chega ao nó origem é tratada individualmente. Em [Ramaswami and Sivarajan, 1995], [Ahang et al., 2002] são mostrados modelos de algoritmos de RWA.

2.1. RWA Estático

Neste trabalho, dado um conjunto D de demandas, procura-se distribuí-las na rede da melhor forma para a obtenção de menor bloqueio. Primeiramente trabalhamos com a rede sem conversão de comprimento de onda e depois com conversão. Observe que não há grande preocupação com o tempo pelo fato de que está sendo feito planejamento na rede. Nos próximos itens apresentamos a formulação matemática para o problema de roteamento e de atribuição de comprimento de onda para redes WDM sem e com conversão.

2.2. Problema de Roteamento e Atribuição de Comprimento de Onda

O objetivo do algoritmo é minimizar a quantidade total de comprimentos de onda necessários para atender um conjunto de conexões de uma dada topologia de rede. A seguir é feita uma formulação para o problema em questão, considerando uma rede WDM sem conversão.

Minimize F_{max}, tal que:

$$
\begin{aligned}
& F_{\max} \geq \sum_{s,d,w} F_{ij}^{sdw},\ \forall i,j \in L\ ;\ \forall s,d \in N\ e\ \forall w; 1 \leq w \leq W \\
& \sum_{i} F_{ij}^{sdw} - \sum_{k} F_{jk}^{sdw} = \begin{cases} -\lambda_{sdw} & se \quad s = j \\ \lambda_{sdw} & se \quad d = j \\ 0 & se \quad c.c. \end{cases} \\
& \sum_{s,d,w} \lambda_{sdw} = \Lambda_{sd}\ ,\ \forall sd \\
& \sum_{s,d} F_{ij}^{sdw} \leq 1\ ,\ \forall w\ ,\ \forall ij \\
& F_{ij}^{sdw}\ \in \{0,1\} \\
& \lambda_{sdw}\ \in \{0,1\}
\end{aligned}
\tag{1}
$$

Onde: N é o conjunto de estações da rede; L é o conjunto de enlaces da rede; W é a quantidade máxima de comprimentos de onda por (i,j). λ_{sdw} é a quantidade de conexões estabelecidas, do nó origem *s* ao nó destino *d*, em um comprimento de onda w; $\sum_{w} \lambda_{sdw} = \Lambda_{sd}$, pois o número de conexões estabelecidas entre o par origem e destino (s,d) não pode ultrapassar a demanda para o par (s,d); Neste problema de atribuição de comprimento de onda, considera-se que 2 ou mais conexões podem ser requisitadas para o mesmo par de origem e destino (s,d), mas que cada uma delas deve utilizar um comprimento de onda distinto e, portanto, $\lambda_{sdw} \in \{0,1\}$; F_{ij}^{sdw} é o número de conexões, estabelecidas do nó *s* ao nó *d*, passando pelo enlace (i,j), e utilizando o comprimento de onda w; $\sum_{s,d} F_{ij}^{sdw} \leq 1$, pois o comprimento de onda em um enlace (i,j) pode ser atribuído para somente um caminho (s,d); $F_{ij}^{sdw} \in \{0,1\}$, que significa que a conexão entre o nó *s* e nó *d*, passando pelo enlace (i,j), estará utilizando o comprimento de onda w, caso seja 1 ou 0, em caso contrário.

A seguir é feita uma formulação para o problema em questão, considerando uma rede WDM com conversão. Em uma rede com conversão, a restrição de continuidade de comprimento de onda pode ser eliminada, e considera-se o uso de conversores de comprimento de onda, nos nós intermediários da rede, para realizarem a conversão dos dados de chegada, que estão utilizando um certo comprimento de onda, para um outro comprimento de onda e posterior encaminhamento para o próximo *link*. Neste tipo de rede, a probabilidade de bloqueio tende a ser menor, pois com o uso de conversores, o número de conflitos de "*lighpaths*" é menor. O uso de conversão de comprimento de onda ainda é muito restrito devido ao seu alto custo e ganhos de desempenho limitados.

Minimize F_{max}, tal que:

$$
\begin{aligned}
& F_{\max} \geq \sum_{s,d} F_{ij}^{sd},\ \forall i,j \in L\ ;\ \forall s,d \in N \\
& \sum_{i} F_{ij}^{sd} - \sum_{k} F_{jk}^{sd} = \begin{cases} -\lambda_{sd} & se \quad s = j \\ \lambda_{sd} & se \quad d = j \\ 0 & se \quad c.c. \end{cases} \\
& \sum_{s,d} F_{ij}^{sd} \leq W,\ \forall i,j \\
& F_{ij}^{sd} \in I
\end{aligned}
\tag{2}
$$

Onde: λ_{sd} é a quantidade de conexões estabelecidas, do nó origem *s* ao nó destino *d*; F_{ij}^{sd} é o número de conexões, estabelecidas do nó *s* ao nó *d*, passando pelo enlace (i,j).

3. Algoritmos Genéticos e Meméticos

Os algoritmos genéticos foram propostos inicialmente por Holland, na década de 70 [Holland, 1973]. Estes algoritmos são baseados nos conceitos da evolução genética, de onde buscou-se importar os mecanismos de evolução, seleção natural das espécies e transmissão da informação genética para os ambientes computacionais.

O algoritmo genético é um método populacional, pois utiliza uma população de indivíduos, na qual cada um representa uma solução para o problema. A cada geração do algoritmo, os indivíduos mais aptos têm maior probabilidade de sobreviverem e, consequentemente, de transmitirem boas qualidades a seus descendentes.

Este método usa o vocabulário da genética. Um **indivíduo** é representado por um cromossomo, que é uma cadeia de genes, que representa uma possível solução para o problema. Um **gen** representa uma certa característica no cromossomo. O valor de um gen é denominado **alelo**. O **genótipo** é o conjunto de genes que irão caracterizar o problema. Uma **população** é um conjunto de indivíduos (cromossomos). A criação de uma população é chamada **geração**. A **função de aptidão** *(fitness function)* representa uma avaliação da qualidade da solução do indivíduo. O processo de reprodução (**crossover**) envolve dois indivíduos (soluções do problema), dando origem a um novo, o qual herdará características genéticas de seus pais. O processo de **seleção** avalia cada indivíduo, selecionando os mais aptos para fazerem parte da próxima geração. A **mutação** é introduzida neste contexto com o único objetivo de introduzir diversidade à população. A **mutação** e o **crossover**, chamados operadores genéticos, são aplicados, geração a geração, de forma que a população sofra um processo evolutivo, gerando indivíduos melhor adaptados.

O algoritmo memético, que foi introduzido por Moscato e Norman [Moscato, 1989], [Moscato and Normam, 1992], [Moscato and Berretta, 1999], é também chamado de algoritmo genético híbrido, pois além de permitir a realização de operações genéticas, utiliza um (ou mais) operador(es) de busca local com o intuito de reforçar o processo de melhoria das soluções. Nesta abordagem híbrida, o algorítmo genético é utilizado para executar a exploração global na população, enquanto o otimizador local é utilizado para melhorar localmente a qualidade das soluções.

Um pseudo-código para o algoritmo memético é apresentado a seguir.

Algorithm 1 *Pseudo-Código do Algoritmo Memético*

Entrada: Matriz de demandas, Tabela de Custos (Distâncias), Número de nós, Número de demandas, Número de gerações;
- Gerar população inicial;
- Avaliar função de fitness dos indivíduos;

for $n = 0$ até $n =$ Número de gerações **do**
- Selecionar indivíduos para realizar operação genética;
- Aplicar crossover;
- Aplicar mutação;
- Avaliar função de fitness dos indivíduos;
- Aplicar busca local;
- Avaliar função de fitness dos indivíduos;
- Selecionar nova população;

end for
Saída: Melhor indivíduo da população; FIM

3.1. Algoritmos Meméticos com População Estruturada

Um dos primeiros passos na implementação de um algoritmo genético ou memético, é definir a representação da solução do problema (representação do cromossomo). Com o intuito de adotarmos uma estrutura de dados eficiente e de rápida manipulação, optamos pelo uso de uma estrutura em árvore ternária com 3 níveis, com um número fixo de 13 agentes [Moscato and Berretta, 1999], conforme a (Figura 3).

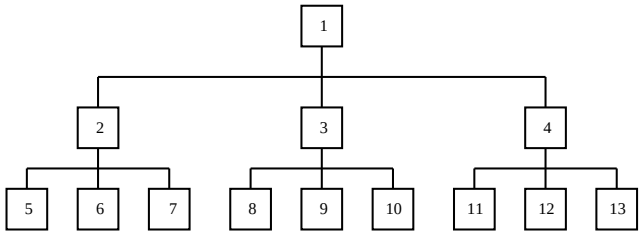

Figura 3: População estruturada em árvore

Nesta estrutura em árvore, a população é formada hierarquicamente por agentes líderes e subordinados. Esta classificação é feita de acordo com o valor da solução que cada agente apresenta. Todo nó raiz possui um valor melhor que os seus nós filhos. Sendo assim, o valor armazenado no agente 1 apresenta a melhor solução encontrada na população.

Podemos verificar que, nesta estrutura, o agente 1 é líder dos agentes 2, 3 e 4. O agente 2 tem como subordinados os agentes 5, 6 e 7. O agente 3 tem como subordinados os agentes 8, 9 e 10 e o agente 4 tem como subordinados os agentes 11, 12 e 13.

Cada agente na população representa dois indivíduos, um chamado de *pocket* e o outro de *current*. A solução que o indivíduo *pocket* possui é sempre melhor que a solução do indivíduo *current*. Quando um *current*, para um dado agente, possui uma solução melhor que o *pocket*, estes valores devem ser trocados.

O indivíduo *pocket* de um líder possui melhor valor que os indivíduos *pocket* de seus agentes subordinados. Quando isto não ocorre, os indivíduos *pocket* destes agentes são trocados de posição. Com estas trocas, garantimos que todo *pocket* será melhor que seu *current*, que todo *pocket* líder será melhor que os indivíduos *pocket* subordinados e que o agente 1 possuirá a melhor solução da população.

A medida tomada para avaliar qual é a melhor solução da população é chamada de função de avaliação ou função de *fitness*. A função de *fitness* é responsável por atribuir um valor referente a qualidade desta solução, indicando o quão próximo um indivíduo está de ser a solução do problema. Assim, a cada indivíduo na população é atribuído um valor, denominado *fitness*, que representa o valor da solução na função objetivo do problema.

Neste tipo de estrutura em árvore, a operação de *crossover* é executada somente entre indivíduos *pocket* e segue o seguinte critério: cada *pocket* líder cruzará com todos os seus indivíduos *pocket* subordinados. O filho resultante deste cruzamento será armazenado no indivíduo *current* do subordinado. Suponha que seja executado um cruzamento entre o *pocket* do agente 1 (líder) e o *pocket* do agente 2 (subordinado), o filho resultante deste cruzamento é armazenado no *current* do agente 2.

A operação de mutação também é executada apenas em indivíduos *pocket*. Neste caso, escolhe-se aleatoriamente um indivíduo *pocket*, com exceção ao *pocket* do agente 1, e efetua-se a mutação. O filho gerado após esta operação é armazenado no indivíduo *current* deste *pocket*.

4. Aplicação do algoritmo memético com população estruturada ao problema de RWA estático

Um pseudo-código para o algoritmo memético com população estruturada, aplicado ao problema RWA estático, é apresentado no *Algorithm 2*.

Algorithm 2 *Pseudo-Código do Algoritmo Memético com população estruturada aplicado ao RWA Estático*

Entrada: Matriz de demandas, Tabela de Custos (Distâncias), Número de nós, Número de demandas, Número de gerações;

- Gerar matriz de demandas;
- Aplicar algoritmo do *Dijkstra* a todas as demandas;
- Gerar população inicial;
- Avaliar função de fitness dos indivíduos;
- Ordenar a árvore;

for $n = 0$ até $n =$ Número de gerações **do**

- Selecionar indivíduos *pocket* líder e subordinado;
- Aplicar crossover;
- Selecionar indivíduo *pocket*;
- Aplicar mutação;
- Avaliar função de fitness dos indivíduos;
- Ordenar a árvore;
- Aplicar busca local;
- Avaliar função de fitness dos indivíduos;
- Ordenar a árvore;

end for

Saída: Melhor indivíduo da população; FIM

4.1. Execução do algoritmo para o problema RWA estático sem conversão

O primeiro passo na execução do algoritmo memético é a geração da matriz de demandas. A matriz de demandas foi gerada de 5 formas: fixa e através das distribuições - Uniforme [1,W], com probabilidade 0.3; Constante W/2; Máxima W e Esparsa Uniforme [0,1], com probabilidade 0.3. W é quantidade máxima de comprimentos de onda.

O segundo passo é avaliação, através do *Djikstra*, dos caminhos possíveis para o roteamento de cada demanda da matriz de demandas e seus respectivos custos (soma das distâncias dos *links* de cada rota).

Um conceito importante e crítico na execução de um algoritmo genético ou memético é a escolha da representação de possíveis soluções para o problema dentro de um cromossomo ou indivíduo.

Para o problema de RWA estático sem conversão, o cromossomo foi codificado por uma matriz D de n linhas e m colunas, onde n é o número de *lambdas* ou comprimentos de onda disponíveis na rede ótica e m é o número de demandas (pares de origem e destino) estáticas que serão passadas pela rede ótica. Cada campo $D_{i,j}$ da matriz, onde $1 \leq i \leq n$ e $1 \leq j \leq m$, é preenchido pelos valores 1 ou 0. O campo $D_{i,j}$ será preenchido com o valor 1 se o comprimento de onda i for alocado a demanda j e 0 para o caso contrário.

O próximo passo na execução do algoritmo memético com população estruturada é a geração da população inicial. Foram gerados 26 indivíduos, 13 indivíduos *pocket* e 13 indivíduos *current*, de

forma aleatória. Cada indivíduo (cromossomo) foi codificado por uma matriz D, conforme detalhado no parágrafo acima. Com o intuito de gerarmos apenas cromossomos factíveis, foi aplicado o critério representado pela Equação 3.

$$\sum_{i=1}^{n} D_{i,j} \leq q_j \ ; \ 1 \leq j \leq m. \tag{3}$$

O parâmetro q_j representa a quantidade de demandas do par origem e destino j.

De acordo com Equação 3, podemos verificar que, para que seja gerado um cromossomo factível, a quantidade de *lambdas* alocados para aquela demanda deve ser no máximo igual a quantidade de demandas existentes para aquele par de origem e destino.

O passo seguinte na execução do algoritmo é a avaliação da função de *fitness*. O critério utilizado para a avaliação das soluções foi a soma do bloqueio com o custo. Foi aplicado um peso de 0.8 para o bloqueio e um peso de 0.2 para o custo, que representa a distância entre o nó origem **s** e o nó destino **d**. O critério utilizado para definir os pesos da função de avaliação foi a prioridade relacionada aos parâmetros bloqueio e custo. Verificamos, através de simulações, que o bloqueio representava maior importância que o custo na classificação de um bom indivíduo e, por isso, recebeu um peso maior que o custo.

Para o cálculo da função de *fitness*, cada indivíduo tem as rotas das demandas e os custos calculados (soma das distâncias dos enlaces da rota pela qual a demanda será roteada) através do algoritmo do caminho mínimo (*djikstra*). O custo total C do indivíduo é a soma dos custos de cada demanda. Tendo as rotas definidas, uma demanda é considerada bloqueada se existir pelo menos 1 (*link*) nesta rota que tente utilizar o mesmo (*lambda*) que outra rota, de outra demanda. O bloqueio B total do indivíduo é a soma dos bloqueios de cada demanda. A função de *fitness* foi calculada, para cada indivíduo, de acordo com a Equação 4. A cada geração, a solução considerada melhor é aquela que apresenta o menor valor de *fitness*.

$$fitnessfunction = 0.8B + 0.2C. \tag{4}$$

O próximo passo na execução do algoritmo é a ordenação da árvore, seguindo os critérios de indivíduos *pocket* e *current*, de forma que o melhor indivíduo (com menor valor de *fitness*) fique na posição do agente 1.

Os passos seguintes, onde são executados os operadores genéticos, são realizados enquanto o critério de parada não for alcançado. O critério de parada utilizado neste trabalho foi o número de gerações. Os operadores genéticos executados neste algoritmo foram o *crossover* uniforme, mutação simples e mutação inversiva.

O *crossover* uniforme é realizado entre indivíduos *pocket* líder e *pocket* subordinado. O filho gerado por este *crossover* é armazenado no indivíduo *current* do subordinado. Nesta operação genética, é gerada aleatoriamente a taxa de pontos de cruzamento (p) ou número de genes que serão permutados entre os cromossomos do *pocket* líder com o do *pocket* subordinado. Em seguida, escolhe-se aleatoriamente a posição dos p genes (números de demandas, $1 \leq j \leq m$) e efetua-se a permuta de toda a coluna da matriz entre os indivíduos *pocket* líder e subordinado.

O *crossover* uniforme é aplicado em uma taxa de cruzamento que varia entre 1 e $m/2$, onde m é número de demandas da rede.

A mutação é realizada apenas em indivíduos *pocket* e o filho gerado é armazenado no *current* deste indivíduo. Na mutação simples, percorre-se todas as colunas (demandas) na matriz do cromossomo e, para cada demanda, escolhe-se duas posições de *lambda* e efetua-se a troca entre elas. O mesmo procedimento é realizado para todos os pares de demanda (s,d).

Na mutação inversiva, percorre-se todas as colunas (demandas) da matriz do cromossomo e, para cada demanda, escolhe-se uma posição de *lambda* e efetua-se a troca entre posições de *lambda* adjacentes.

4.2. Execução do algoritmo para o problema RWA estático com conversão

Os passos 1 (geração da matriz de demandas) e 2 (avaliação dos caminhos para roteamento da demanda) são iguais aos passos 1 e 2 descritos no subitem 4.1.

Para o problema de RWA estático com conversão, o cromossomo foi codificado por um vetor V, de m colunas, onde m é o número total de demandas (conexões) a serem alocadas na rede. Cada campo do vetor V_i, onde $1 \leq i \leq m$, é preenchido por um número de demanda. A posição de demanda no vetor representa a ordem na qual será verificada a possibilidade da demanda ser atendida. Desta forma, o vetor contém a sequência de atendimento das demandas da rede.

O próximo passo na execução do algoritmo memético com população estruturada é a geração da população inicial. Foram gerados 26 indivíduos, 13 indivíduos *pocket* e 13 indivíduos *current*, de forma aleatória. Cada indivíduo (cromossomo) foi codificado por um vetor com uma sequência aleatória de atendimento das demandas.

O passo seguinte na execução do algoritmo é a avaliação da função de *fitness*. O critério utilizado para a avaliação das soluções foi o mesmo descrito no item 4.1, pela equação 4. Para o cálculo da função de *fitness*, cada indivíduo tem as rotas das demandas e os custos calculados (soma das distâncias dos enlaces da rota pela qual a demanda será roteada) através do algoritmo do caminho mínimo (*djikstra*). O custo total C do indivíduo é a soma dos custos de cada demanda. Tendo as rotas definidas, uma demanda é considerada bloqueada se pelos menos um dos *links* de sua rota não conseguir alocar um comprimento de onda. O bloqueio B total do indivíduo é a soma dos bloqueios de cada demanda.

O próximo passo na execução do algoritmo é a ordenação da árvore, seguindo os critérios de indivíduos *pocket* e *current*, de forma que o melhor indivíduo (com menor valor de *fitness*) fique na posição do agente 1.

Os passos seguintes, onde são executados os operadores genéticos, são realizados enquanto o critério de parada não for alcançado. O critério de parada utilizado neste trabalho foi o número de gerações. Os operadores genéticos executados neste algoritmo foram o *crossover OX de 1 ponto*, mutação simples e mutação inversiva.

O *crossover OX* de 1 ponto é realizado entre indivíduos *pocket* líder e *pocket* subordinado. O filho gerado por este *crossover* é armazenado no indivíduo *current* do subordinado.

A execução da operação do *crossover OX* pode ser observada pelo exemplo abaixo, aplicado aos indivíduos *pocket* pai e *pocket* filho descritos da seguinte maneira:

$$\mathbf{V_{pocket-pai}} = \begin{pmatrix} 3 & 0 & 1 & 2 & 4 & 5 \end{pmatrix} \qquad \mathbf{V_{pocket-filho}} = \begin{pmatrix} 1 & 5 & 4 & 3 & 0 & 2 \end{pmatrix}$$

Nesta operação genética, primeiramente é gerado aleatoriamente um ponto de cruzamento (p) entre 1 e m. Suponha que o ponto de *crossover*, escolhido aleatoriamente, seja a posição 3

do vetor. Neste caso, os dois vetores são analisados e, como a sequência 302 do indivíduo *pocket* filho irá para o indivíduo *pocket* pai, o indivíduo *pocket* pai é analisado e, ao serem encontrados os números $3, 0\ e\ 2$, estes são removidos do vetor e os demais valores são deslocados pelo vetor. Como a sequência 245 do indivíduo *pocket* pai irá para o indivíduo *pocket* filho, o indivíduo *pocket* filho é analisado e, ao serem encontrados os números $2, 4\ e\ 5$, estes são removidos do vetor e os demais valores são deslocados pelo vetor. Sendo assim, os vetores *pocket* pai e *pocket* filho ficarão da seguinte forma:

$$\mathbf{V_{pocket-pai}} = \begin{pmatrix} 1 & 4 & 5 & - & - & - \end{pmatrix} \qquad \mathbf{V_{pocket-filho}} = \begin{pmatrix} 1 & 3 & 0 & - & - & - \end{pmatrix}$$

O procedimento final do *crossover OX* de 1 ponto é a inserção das sequências 302 e 245 nos indivíduos *pocket* pai e *pocket* filho, respectivamente. Ao final da execução deste operador genético, os indivíduos filhos gerados serão:

$$\mathbf{V_{current-pai}} = \begin{pmatrix} 1 & 4 & 5 & 3 & 0 & 2 \end{pmatrix} \qquad \mathbf{V_{current-filho}} = \begin{pmatrix} 1 & 3 & 0 & 2 & 4 & 5 \end{pmatrix}$$

A mutação é realizada apenas em indivíduos *pocket* e o filho gerado é armazenado no *current* deste indivíduo. Na mutação simples, escolhe-se duas posições no vetor (duas demandas diferentes) e efetua-se a troca entre elas.

Na mutação inversiva, escolhe-se uma posição no vetor (posição de atendimento da demanda), e efetua-se a troca entre posições de *demanda* adjacentes.

5. Exemplos Numéricos

Aplicamos o algoritmo memético com população estruturada para resolver o problema de RWA estático nas redes óticas apresentadas na (Figura 4), e na (Figura 5). Resolvemos o problema com e sem conversão.

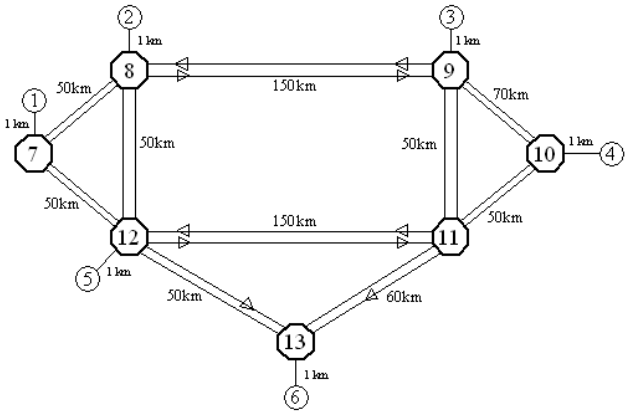

Figura 4: Um exemplo de Rede WDM de 6 nós

Para a Rede de 6 Nós, a geração de demandas foi realizada de 5 formas. Na primeira, foi aplicado ao algoritmo um conjunto de demanda fixas e, nas demais, foram aplicadas as seguintes distribuições: Uniforme [1,w], com probabilidade 0.3; Constante W/2; Máxima W e Esparsa Uniforme [0,1], com probabilidade 0.3. W é quantidade máxima de comprimentos de onda. A rede

da (Figura 4) tem 6 nós externos, que representam os membros receptores e encaminhadores de demandas, e os 7 nós internos representam os roteadores da rede.

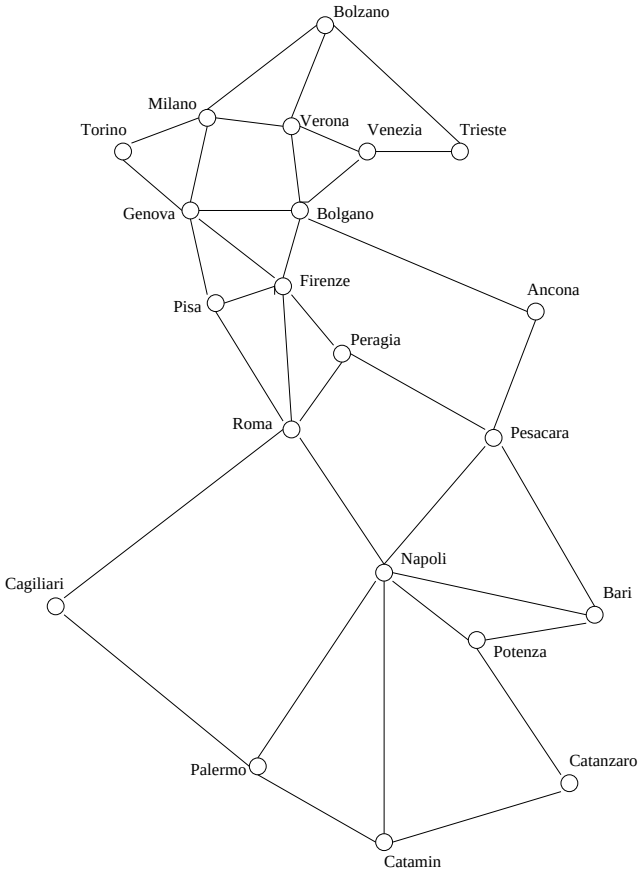

Figura 5: Rede *Italian*

Os parâmetros adotados na execução do algoritmo memético são apresentados na (Tabela 1). Para estabelecer o número de gerações foram executadas várias simulações, com diferentes parâmetros, e alcançamos resultados que não tendem a sofrer mudanças a partir de um certo número de gerações, pois o resultado já se encontra em um ótimo local ou ótimo global.

Parâmetro	*Valor*
Tamanho da população	26
Probabilidade de *crossover*	0.5
Probabilidade de mutação	0.07
W - quantidade de comprimentos de onda	8
Número de gerações	50

Tabela 1: Parâmetros aplicados ao algoritmo memético

Para cada matriz de demanda foi aplicado o algoritmo do caminho mínimo (**dijkstra**) para encontrar a rota com menor custo pelo qual a demanda seria encaminhada. Após encontrado o caminho mínimo e demais caminhos alternativos para cada demanda, os cromossomos da população são gerados e os passos do algoritmo são executados conforme o pseudo-códido descrito pelo *Algorithm 2*. Para o cálculo da função de **fitness**, são calculados o bloqueio e o custo para cada indivíduo. Com intuito de diminuirmos o bloqueio das conexões (demandas), fizemos uma simulação onde, se

a demanda for bloqueada, resolvemos o caminho mínimo novamente para tentar encontrar uma rota pela qual a demanda não seja bloqueada.

Na (Tabela 2) estão apresentados os resultados obtidos pelo algoritmo memético para a Rede da (Figura 4).

	Sem Conversão			Com Conversão		
Matriz de Demandas	*Atendidas*	*Bloqueadas*	*Tempo*	*Atendidas*	*Bloqueadas*	*Tempo*
Fixa	31	8	27.40 seg.	35	4	35.05 seg.
Uniforme	28	22	31.50 seg.	32	18	44.30 seg.
Constante	44	76	88 seg.	48	72	112.30 seg.
Máxima	48	192	191.55 seg.	48	192	208.55 seg.
Esparsa Uniforme	15	0	34.6 seg.	15	0	22 seg.

Tabela 2: Resultados do alg. memético para a rede de 6 Nós com e sem conversão

Em [Ali, 2001], foram aplicados o algoritmo genético (GA) e programação linear (LP) para a resolução do problema RWA estático em redes óticas sem conversão. Para a rede da (Figura 4) foram obtidos os resultados apresentados na (Tabela 3).

	LP	GA
Matriz de Demandas	*Atendidas*	*Atendidas*
Fixa	33	31
Uniforme	33	31
Constante	48	44
Máxima	48	44
Esparsa Uniforme	15	14

Tabela 3: Resultados obtidos em [Ali, 2001]

Podemos analisar, pelas tabelas (Tabela 2) e (Tabela 3), que os resultados obtidos pelo algoritmo memético são, na maioria, melhores que os resultados obtidos pelo algoritmo genético e são bem próximos dos resultados ótimos. Aplicando o algoritmo memético ao problema de RWA, conseguimos alcançar o mesmo valor do LP para as distribuições máxima e esparsa uniforme. Apenas na distribuição uniforme o resultado obtido pelo algoritmo memético foi inferior ao resultado obtido pelo algoritmo genético de [Ali, 2001].

Para a Rede de 21 Nós (Rede *Italian*), a geração de demandas foi realizada de 5 formas, igualmente para o teste com a rede de 6 nós (Figura 4).

Na (Tabela 4) estão apresentados os resultados obtidos pelo algoritmo memético para a Rede *Italian*.

	Sem Conversão			Com Conversão		
Matriz de Demandas	*Atendidas*	*Bloqueadas*	*Tempo*	*Atendidas*	*Bloqueadas*	*Tempo*
Fixa	48	6	545 seg.	54	0	461.5 seg.
Uniforme	135	520	524 seg.	149	506	1833.25 seg.
Constante	158	1522	2144.05 seg.	167	1513	4624.8 seg.
Máxima	162	3198	3255.65 seg.	167	3193	6885.95 seg.
Esparsa Uniforme	92	93	405.70 seg.	133	52	1073.95 seg.

Tabela 4: Resultados do alg. memético para a rede *italian* com e sem conversão

Em [Ali, 2001] , foram aplicados o algoritmo genético (GA) e programação linear (LP) para a resolução do problema RWA estático em redes óticas sem conversão. Para a rede de 21 nós (*Italian Network*) foram obtidos os resultados apenas para a matriz de demandas fixas e estão apresentados na (Tabela 5).

	LP	GA
Matriz de Demandas	*Atendidas*	*Atendidas*
Fixa	54	26

Tabela 5: Resultados obtidos em [Ali, 2001]

Podemos analisar, pelas tabelas (Tabela 4) e (Tabela 5), que os resultados obtidos pelo algoritmo memético são melhores que os resultados obtidos pelo algoritmo genético e são bem próximos dos resultados ótimos.

6. Conclusão

Este trabalho mostrou um algoritmo memético aplicado a Redes Óticas WDM. Utilizando o algoritmo *Dijkstra*, para encontrar o melhor caminho na rede, e trabalhando com o algoritmo proposto para atribuição de comprimentos de onda, obtivemos uma solução para o problema RWA em uma rede ótica WDM sem malha, de forma estática. O algoritmo foi testado para matriz de demandas geradas de 5 formas: fixa e através das distribuições uniforme, constante, máxima e esparsa uniforme. Através de exemplos numéricos, e comparações realizadas com os resultados obtidos pelas simulações do [Ali, 2001], que implementou um algoritmo genético e realizou simulações com programação linear, pudemos avaliar o desempenho do algoritmo memético. Pela análise dos resultados, o algoritmo mostrou-se eficiente pois obteve, na maioria dos testes, resultados melhores que o algoritmo genético e bem próximos dos resultados do método exato. Verificamos, através deste trabalho, que o algoritmo memético tem se comportado melhor que o algoritmo genético e com um tempo de execução menor devido a representação em população estruturada que foi utilizada.

Referências

Ahang, Y., Taira, K., Takagi, H., and Das, S. K. (2002). An efficient heuristic for routing and wavelenght assignment in optical wdm networks. *Proc. IEEE Int. Conf. Communications (ICC 2002)*, pages 2734–2739.

Ali, M. (2001). *Transmission-Efficient Design and Management of Wavelenght-Routed Optical Networks.* MA: Kluwer, norwood edition.

Ali, M., Ramamurthy, B., and Deogun, J. S. (2000). Routing algorithms for all-optical networks with power considerations. *Computer Networks*, 32(5):539–555.

Holland, J. H. (1973). Genetic algorithms and the optimal allocation of trials. *SIAM J. Comput*, 2:88–105.

Moscato, P. (1989). On evolution, search, optimization, genetic algorithms and martial arts: Towards memetic algorithms. Technical Report C3P Report 826, Caltech Concurrent Computation Program.

Moscato, P. and Berretta, R. (1999). *The number partitioning problem: An open challenge for Evolutionary Computation*, chapter 17. McGraw-Hill.

Moscato, P. and Normam, M. (1992). A memetic approach for the travelling salesman problem: implementation of a computational ecology for combinatorial optimization on message-passing systems. *Proc. Intl. Conf. on Parallel Computing and Transportation Applications*.

Murthy, C. S. R. and Gurusamy, M. (2002). *WDM Optical Networks: Concepts, Design and Algorithms*. Prentice Hall PTR, New Jersey.

Ramaswami, R. and Sivarajan, K. N. (1995). Routing and wavelenght assignment in all-optical networks. *IEEE/ACM Transactions on Networking*, 3:489–500.

Ramaswami, R. and Sivarajan, K. N. (1998). *Optical Networks - A Practical Perspective*. Morgan Kaufmann Publishers, San Francisco.