3.
DESCRIÇÃO E SOLUÇÃO DO PROBLEMA

O problema que almejamos solucionar é o da determinação de Q caminhos ópticos cujos nós origem e destino são pré-definidos. Deseja-se estabelecê-los de modo que o custo total (soma dos custos associados a cada caminho) seja mínimo. Como caminhos distintos que possuem enlaces ópticos iguais não podem compartilhar o mesmo comprimento de onda, devemos levar em consideração que conversões de comprimentos de onda em determinados nós podem ocorrer. Esse custo também deve ser computado no custo total da rota.

Define-se como caminho de luz, (do inglês, *lightpath*) a transmissão totalmente óptica, onde não há conversão de comprimento de onda durante todo o percurso entre o nó fonte e o nó destino. É claro que o emprego destes caminhos de luz reduz os custos de processos eletrônicos, aumenta a confiabilidade e garante a qualidade de serviço, mas, por outro lado, existem algumas limitações físicas que impossibilitam o encaminhamento dos sinais somente através de caminhos de luz. Essas limitações são: o número máximo de comprimento de onda disponíveis, o comprimento do enlace, a dispersão do comprimento de onda durante o percurso (é claro que quanto maior é a distância, maior é a dispersão do comprimento de onda) e, a sintonia dos transceptores em cada nó da rede [21].

Como nem sempre é viável a transmissão de sinais por caminhos totalmente ópticos, devido às limitações explicadas no parágrafo anterior, criou-se o conceito de semi-caminho de luz. Um semi-caminho de luz é obtido com encadeamento de diversos caminhos de luz com diferentes comprimentos de onda. Então, em alguns nós da rede, haveria conversão de comprimento de onda, mas, em geral, não em todos. Pode-se avaliar, pela própria definição de caminho de luz, que um caminho de luz é um caso especial do semi caminho de luz, onde o número de nós que fazem conversão de comprimentos de onda é zero.

O objetivo deste trabalho é apresentar um algoritmo heurístico que otimize o roteamento dos semi-caminhos de luz entre os nós de origem e de destino,

levando-se em consideração o menor custo entre eles. Para isso, serão levados em consideração dois custos:

i) o custo de percorrer um determinado enlace em um comprimento de onda;
ii) o custo de conversão de comprimento de onda em um nó intermediário, quando houver troca de comprimento do mesmo.

Note que se levássemos em consideração somente o primeiro custo, o algoritmo para resolver o problema de roteamento de comprimentos de luz seria reduzido ao simples algoritmo de se achar a menor distância entre os nós fonte e destino. Mas se considerarmos o custo de conversão de comprimentos de onda, o problema não é tão trivial [27].

Considere uma rede óptica descrita por um grafo orientado G = (V, E), onde cada aresta representa um enlace óptico pelo qual podem trafegar M comprimentos de onda pertencentes ao conjunto Λ. Assume-se em princípio que cada nó de G possui conversores ópticos que podem converter sinais ópticos de qualquer comprimento de onda para qualquer outro. Matematicamente podemos definir os custos definidos em i) e ii) da seguinte maneira:

- ω: E x $\Lambda \longrightarrow R^+ \cup \{0\}$, onde ω (e,λ) é o custo de utilização do comprimento de onda λ na aresta e;
- c: V x $E^2 \longrightarrow R^+ \cup \{0\}$, onde c_i (*a,b*) é o custo de conversão do comprimento de onda *a* para o comprimento de onda *b* no vértice i.

Um semi-caminho de luz entre os vértices *r* e *s* é descrito por:
$P_{r,s} = \{(e_1, \lambda_1), (e_2, \lambda_2), ..., (e_k, \lambda_k)\}$, onde:

- $O(e_1) = r$, $D(e_k) = s$, $D(e_i) = O(e_{i+1})$ para $i \in \{1, 2, ..., k-1\}$
- $\lambda_i \in \Lambda$ para todo $i \in \{1, 2, ..., k\}$

O(e) é a origem do enlace e, D(e) é o destino do enlace e, r é a origem da rota e s o destino da rota. O comprimento de onda deve pertencer ao conjunto de

comprimentos de onda Λ, já definido anteriormente. Se $\lambda_1 = \lambda_2 = ... = \lambda_k$, então o semi caminho de luz é na verdade um caminho de luz.

Quando o comprimento de onda λ_i não está disponível para um determinado enlace, o custo ω (e, λ_i) associado ao mesmo neste enlace é infinito. Por outro lado, o custo de conversão entre dois comprimentos de onda, c_i (a, a) iguais é zero.

O custo total do percurso entre os nós origem (*r*) e destino (*s*) é dado pela seguinte expressão:

$$C\,[P_{r,s}] = \sum_{i=1}^{k} \omega\,(e_i, \lambda_i) + \sum_{i=1}^{k-1} C_{D_{e_i}}\,(\lambda_i, \lambda_{i+1})$$

O primeiro somatório fornece os custos de percorrer os enlaces em comprimentos de onda específicos e o segundo somatório fornece os custos de conversões de comprimentos de onda nos nós intermediários. Note que somente os comprimentos de onda diferentes contribuem para o segundo somatório, pois se λ_i for igual a λ_{i+1}, o custo de conversão é zero, conforme já dito anteriormente [26].

Vale lembrar mais uma vez que caminhos distintos não podem compartilhar o mesmo comprimento de onda em um mesmo enlace óptico.

O algoritmo proposto deve achar o melhor caminho de luz entre os nós origem e destino, minimizando o custo C[P].

Não é proibido que um semi caminho de luz visite um mesmo nó mais de uma vez, com diferentes comprimentos de onda, mas espera-se que isto não aconteça. Na prática, se um semi caminho de luz passa por um mesmo nó da rede com o mesmo comprimento de onda, essa segunda passagem será descartada pelo algoritmo, já que o mesmo é feito para otimizar os custos e é óbvio que isto acarretará um custo final maior para a rota proposta. Esse é o motivo pelo qual não se exclui isto na definição do algoritmo.

No nosso caso, serão apresentados, a seguir, dois algoritmos que determinarão os Q semi caminhos de luz entre os nós origem e destino, visando minimizar a soma dos custos C[P] a ele associados.

Para o desenvolvimento dos algoritmos mencionados, é conveniente transformar o grafo original em um outro que denominaremos de grafo aumentado e cujo objetivo é permitir que o problema desejado (que envolve os custos de conversão de comprimento de onda) possa ser resolvido por métodos de busca de caminhos ótimos em grafos. É esse grafo aumentado que será o "descritor" da rede óptica para todos os algoritmos aqui considerados.

No item a seguir, apresenta-se o método de geração de grafo aumentado.

3.1
Metodologia de conversão de grafo original para grafo aumentado

A metodologia de conversão do grafo original para grafo aumentado consiste em considerar a conversão de comprimento de onda como arestas deste novo grafo, conforme descrito abaixo e indicado em [21]:

i) Seja N* = LN vértices, onde L é o número de comprimentos de onda e N é o número de nós do grafo original;

ii) Arrume estes vértices em uma matriz com L linhas e N colunas. Cada coluna corresponde aos nós da rede original e cada linha corresponde à um comprimento de onda;

iii) Na i-ésima linha, i = 1, 2, ..., L, desenhe uma linha direta da coluna j para a coluna h, aonde existirem o enlace do nó j para h da rede e quando λ_i estiver disponível neste enlace. Coloque os custos - ω (e_k, λ_i) - correspondentes a utilização deste comprimento nesse enlace nesta linha. Isso representa a rede com respectivo comprimento de onda e seus custos;

iv) Na coluna j, j = 1, 2, ..., N, desenhe uma linha direta da linha i para k, se no nó j a conversão de comprimento de onda do λ_i para o λ_k estiver disponível. Coloque os custos - c_j (λ_i, λ_k) - da conversão de um

comprimento de onda para outro. Isto representa a rede com a possível conversão de comprimento de onda e seus respectivos custos.

Pelo mapeamento acima descrito, o custo mínimo associado à passagem de um sinal entre os nós s e t corresponde à menor soma dos números representados na matriz pelos caminhos, horizontais e verticais, entre as colunas s e t.

Conforme já descrito, os dados dessa matriz servirão de dados de entrada para os algoritmos que fornecerão o melhor caminho entre os nós fonte e destino, visando o custo mínimo.

Para implementar esse grafo aumentado, pode-se descrevê-lo matematicamente da seguinte forma:

O grafo original é composto de vértices e arestas, onde:

$G = (V, E)$, $V = \{1, 2, ..., N\}$ e
$E = \{e_1, e_2, ..., e_M\}$, $e_i = (v_1^{(i)}, v_2^{(i)})$ $\quad v_j^{(i)} \in V$

Dado um número de comprimentos de onda L, o grafo aumentado também será composto de vértices e arestas e será representado por:
$G^* = (V^*, E^*)$, onde o número de vértices é N* e o número de arestas é M*.

Existem $N^* = N.(L+1)$ vértices divididos em dois grupos assim descritos:

- Vértices tipo I – vértices artificiais numerados de 1 a N para manter a coerência com os vértices do grafo original.

$$v^*_i = i \text{ para } i \in \{1, 2, ..., N\}$$

- Vértices tipo II – vértices gerados pelo par (vértice, comprimento de onda). Existem em quantidade igual à N . L e indexado da forma:

$$v^*_{N+(r-1).L+s} = (r,s) \text{ para } r \in \{1, 2, ..., N\} \text{ e } s = \{1, 2, ..., L\}$$

Existem, ainda, três tipos de arestas:

- Arestas tipo I - são arestas verticais que representam os pares (vértice, comprimento de onda). Existem em quantidades M . L e se escrevem como:

$$e^*_{(r-1).L+s} = (e_r,s) \text{ para } r \in \{1, 2, ..., M\} \text{ e } s = \{1, 2, ..., L\}$$

- Arestas tipo II – são arestas horizontais que representam a transição (vértice, comprimento de onda original) para (vértice, comprimento de onda comutado), se comprimento de onda original for diferente do comprimento de onda comutado. Existem em quantidade N.L.(L-1) e se escrevem como:

$$e^*_{M.L+(u-1).L.(L-1)+g(r,\,s)} = (u, r, s) \text{ para } u \in \{1, 2, ..., N\} \text{ e } r, s \in \{1, 2, ...,L\}$$

- Arestas tipo III – são 2.N.L arestas representando arestas bidirecionais de custo nulo, ligando o vértice artificial i a cada um dos vértices (i, j) para j em {1, 2, ..., L} e são indexadas por:

$$e^*_{M.L+N.L.(L-1)+(i-1).L+j} = (i, j) \text{ para } i \in \{1, 2, ..., N\} \text{ e } j \in \{1, 2, ..., L\}$$

O número de arestas que terá o grafo aumentado é representado por:

$$M^* = M.L + N.L(L-1) + 2.N.L = M.L + N.L\,(L+1) = L\,[\,M + N.(L+1)]$$

Um exemplo gráfico do grafo aumentado associado a um grafo original utilizando-se dois comprimentos de onda é apresentado na figura 3.1.

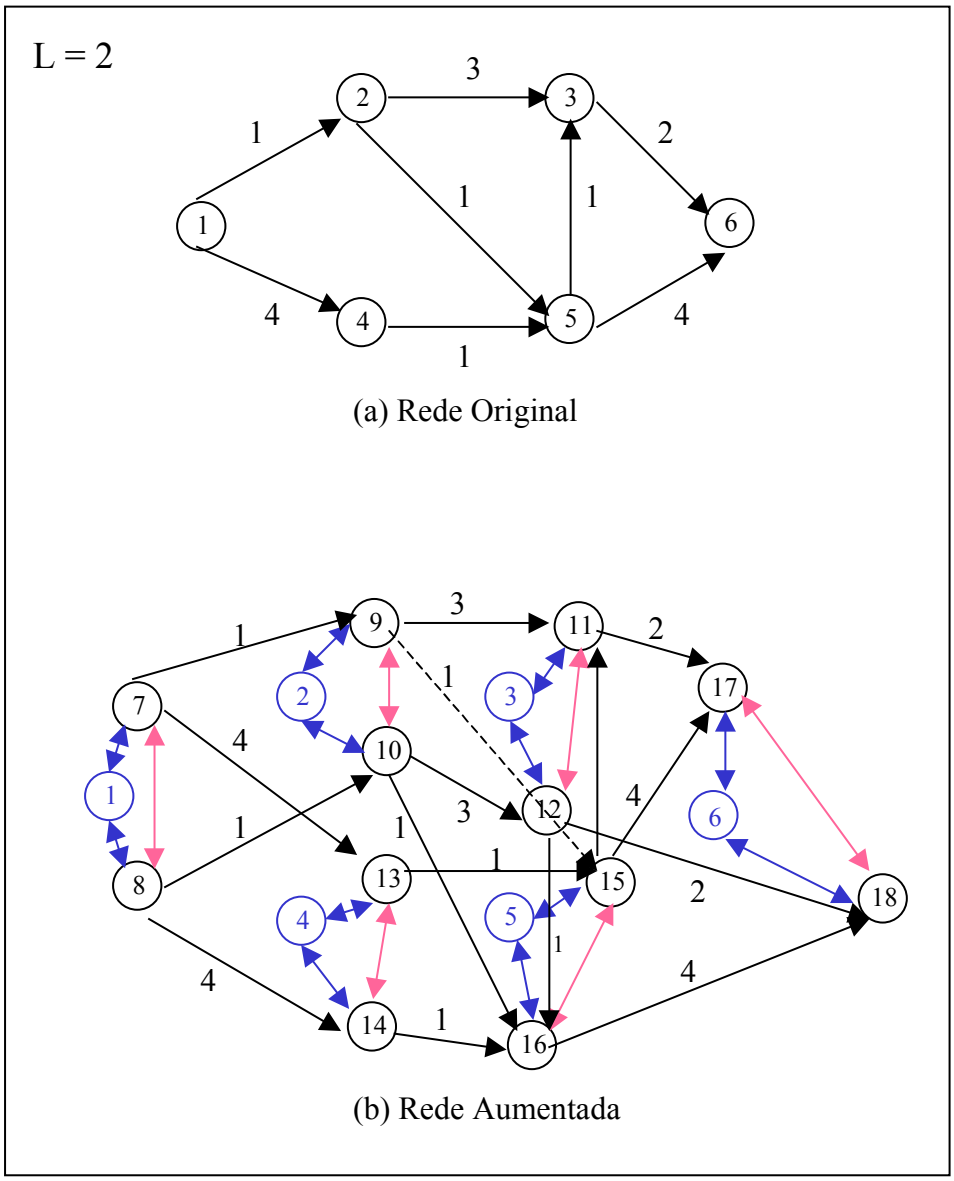

Figura 3.1: Exemplo de grafo aumentado com L = 2.

Note que as arestas associadas as linhas coloridas de azul possuem custo zero, por se tratarem de ligações não físicas, já que se tratam do mesmo nó, ou seja, por exemplo, os nós 1, 7 e 8 são o mesmo e correspondem ao nó 1 da rede original. Todas as linhas coloridas do gráfico não são ligações físicas. Cada nó do grafo original se transformou em 3 no grafo aumentado: o original, o que representa o custo do enlace no comprimento de onda λ_1 e o que representa o custo do enlace no comprimento de onda λ_2. Para esse exemplo acima representado (figura 3.1) foram utilizados somente dois comprimentos de onda distintos. Se mais de dois comprimentos de onda fossem definidos, mais nós teria o grafo aumentado e, consequentemente, maior seria a representação gráfica do mesmo. Muitas vezes,

isso pode ser um obstáculo para a solução dos problemas de RWA através de algoritmos. Mas isso será discutido posteriormente.

Um programa de conversão de grafo original para grafo aumentado foi produzido para gerar os dados necessários aos programas seguintes. Esse programa gerará dois arquivos que servirão de entrada para os dois algoritmos descritos nos próximos itens. Esses arquivos possuem informações de enlaces com seus respectivos custos.

3.2
Algoritmo de determinação das rotas ótimas

O algoritmo de determinação de rotas ótimas é baseado no método de programação linear inteira.

A programação linear é um planejamento de atividades que pode ser representado matematicamente, onde todas as funções são lineares.

Para uma melhor compreensão de um problema de P.L. (Programação Linear) enunciamos a seguir alguns conceitos e expressões que vão ser usados ao longo deste item [22 e 23]:

- Função objetivo – é uma função linear que vamos otimizar, maximizando-a ou minimizando-a;

- Variáveis de decisão – são as varáveis cujos valores deseja-se conhecer. Todas as funções associadas ao problema dependem dessas varáveis;

- Variáveis de folga – são as variáveis que se usa para transformar as inequações em equações. É importante salientar que cada variável de folga está associada a uma restrição;

- Restrições – são condições (representadas por equações ou inequações lineares) que se impõem ao modelo dado. Existem dois tipos de restrições:

- Restrições do problema – são restrições do tipo ≤ , = ou ≥ , que se relacionam com uma ou mais variáveis do problema;
- Restrições de não negatividade – são aquelas onde as desigualdades das variáveis de decisão são sempre do tipo $x_1 \geq 0, ..., x_n \geq 0$;

- Forma padrão (standard) – quando as restrições de um problema de P.L. são apresentadas na forma de equações;

- Forma canônica – quando as restrições de um problema de P.L. são apresentadas na forma de inequações;

- Solução – é qualquer conjunto de valores para as variáveis que satisfaça as restrições;

- Solução admissível (solução possível) – é qualquer especificação de valores para as variáveis que satisfaça as restrições do problema e as condições de não negatividade;

- Solução ilimitada (do inglês, unbounded) – é aquela em que a função objetivo pode crescer (no caso da maximização) ou decrescer (no caso da minimização), indefinidamente, tendo em conta todas as restrições do problema;

- Solução ótima – é aquela que maximiza ou minimiza a função objetivo sobre toda a região admissível;

- Região admissível – é o conjunto de todas as soluções admissíveis.

Uma vez que os problemas de P.L. determinam o planeamento ótimo de atividades, ou seja, um plano ótimo que representa a melhor solução entre todas as soluções possíveis, a P.L. é uma técnica de planejamento que tem-se constituindo como uma das mais poderosas em quase todos os ramos das atividade humanas. É

um método de maximizar (ou minimizar) uma função linear (designada função objetivo), definida num dado conjunto convexo, tendo em conta que as variáveis estão sujeitas a restrições lineares. Estas restrições podem ser do tipo ≤, = ou ≥ e as variáveis são reais não negativas [22 e 23]. Se todas as variáveis são restritas a assumirem valores inteiros, então a P.L. é denominada de Programação Linear Inteira (P.L.I.) e, é um caso particular da P.L..

Os problemas são representados matematicamente por modelos que podem ser apresentados na forma padrão ou na forma canônica (já definidas anteriormente). Para o nosso estudo, o problema de determinação de rotas com alocação de comprimentos de onda, visando minimizar os custos totais das rotas, pode ser descrito da seguinte forma: Sejam $R_i = (u_i, v_i)$ com $i \in \{1, 2, ..., Q\}$ e $u_i, v_i \in V$ (conjunto de vértices do grafo original) as rotas que se deseja implementar. Define-se a variável F_r^s como 1 se a aresta r do grafo aumentado (que varia de 1 a M*) pertence a rota s (que varia de 1 a Q) e como 0, caso contrário.

Seja W_r o custo associado a aresta r. O objetivo do problema é o de minimizar o custo total associado aos Q enlaces ópticos. Assim a função objetivo pode ser expressa por:

$$Z = \sum_{r=1}^{M^*} \sum_{s=1}^{Q} W_r . F_r^s$$

Note que, pela definição do problema acima, trata-se de um problema P.L.I. de Q.M* variáveis binárias de decisão.

Este problema está sujeito as seguintes restrições:

Restrições de grau

i) $\sum_{s=1}^{Q} F_r^s \leq 1$ para todo $s \in \{1, 2, ..., M^*\} - U$ onde U é o conjunto de arestas que envolvem os nós do tipo III .

Consequentemente, existem M* - 2.N.L restrições deste tipo;

ii) Para cada nó s, seja Ω(s) o conjunto de pares ordenados de enlaces (a, b) de E* tais que D[a] = O[b] = s. Existem N* destes conjuntos. Então para cada nó s e rota r, tem-se:

$$\sum_{(u,v)\in\Omega(s)} F_u^r - F_v^r = \begin{cases} -1 & \text{se } s = u_r \\ +1 & \text{se } s = v_r \\ 0 & \text{caso contrário} \end{cases}$$

Note que o número de tais restrições é Q.N*. Assim, o número total de restrições é M* + Q.N* - 2.N.L

A primeira restrição nos garante que uma aresta seja usada somente por, no máximo, uma rota. Já a segunda restrição garante a continuidade física da rota.

Através dessa formulação de P.L.I. é possível a obtenção de solução do problema de rotas ótimas em redes ópticas cujo modelamento se encaixa no aqui descrito.

Podemos utilizar diversos "solvers" de problemas deste tipo, mas a maioria deles é de custo bastante elevado. Utilizamos o GLPK – GNU Linear Programming Kit, um software de domínio público e que se encontra na *Internet* no endereço www.gnu.org/software/glpk/glpk.html.

É interessante discutir neste ponto a complexidade computacional da busca de solução para o problema de interesse por meio de P.L.I. Os números de variáveis (Var) e de restrições (R) dependem diretamente dos números de nós e de comprimentos de onda do grafo original. Pelas fórmulas já apresentadas nos itens 3.1 (para M* e N*) e 3.2 (para R e Var), tem-se:

$$\begin{cases} \text{Var} = \text{Q.N.}(\text{L}^2 + \alpha.L + 1) \\ \text{R} = \text{N.}[\text{L}^2 + [\alpha + (\text{Q} - 2)].\text{L} + 1] \end{cases}$$

onde:

- N = número de nós do grafo original;

- M = número de arestas do grafo original;
- Q = número de rotas;
- L = número de comprimentos de onda;
- α (vazão) = M/N.

Olhando-se o número de variáveis por número de rotas e nós do grafo original, tem-se a expressão quadrática da forma $L^2 + \alpha.L + 1$. O mesmo acontece com o número de restrições por número de nós do grafo original. Este problema pode ultrapassar a capacidade do "solver" se existirem muitos comprimentos de onda disponíveis na rede, assim como se existirem muitos nós. Esse é o principal problema deste tipo de solução.

A título de ilustração, o GLPK tem para o caso contínuo, uma limitação aproximada de 5000 variáveis e 8000 restrições. Entretanto, quando o problema é de natureza inteira, esse limite cai para aproximadamente 800 variáveis e igual número de restrições. Esses valores são facilmente alcançáveis com rede de pequenas dimensões.

Para ilustrarmos o crescimento do número de restrições e variáveis neste problema com o aumento de nós e números de comprimentos de onda de uma rede, apresenta-se um gráfico abaixo.

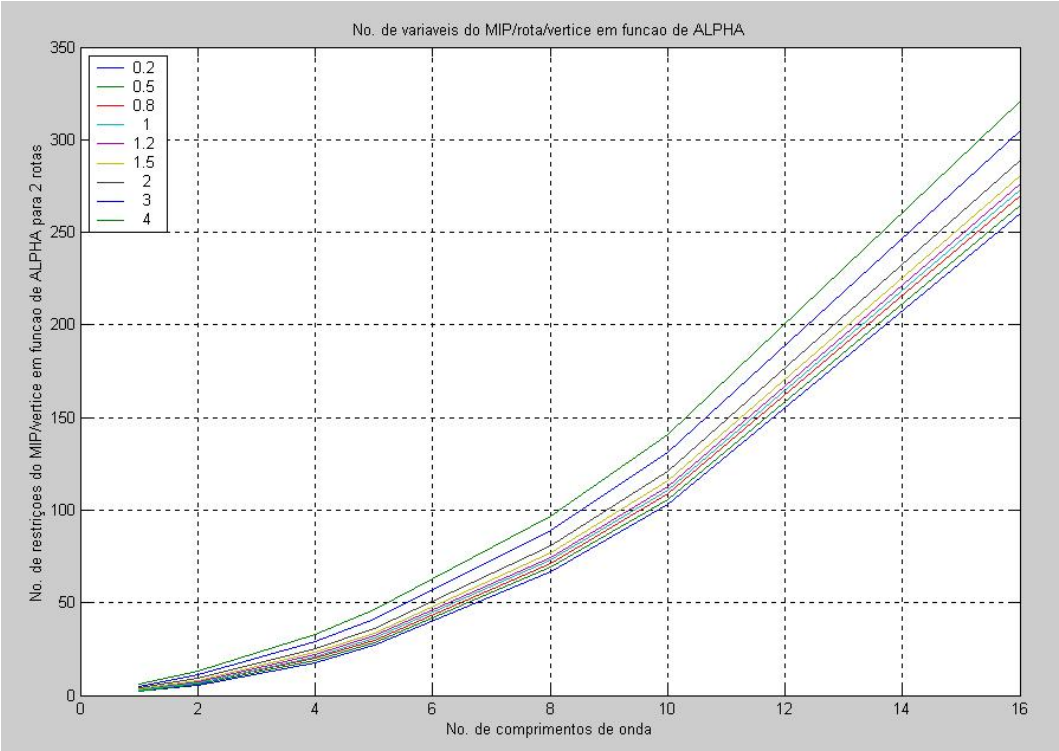

Figura 3.2: Dependência do número de restrições e variáveis em função do número de comprimentos de onda

Note que a rede do gráfico é uma rede pequena (duas rotas – descrita no gráfico do lado esquerdo). Mas, mesmo assim, se aumentarmos o número de comprimentos de onda, o aumento do número de restrições para o problema aumenta significativamente.

3.3
Algoritmo proposto

Para que se torne possível a solução do problema de interesse em redes com maior número de nós e comprimentos de onda, sem que haja um número excessivo de variáveis e restrições, outro algoritmo foi desenvolvido.

O algoritmo aqui apresentado é de natureza seqüencial e, portanto, é de natureza sub-ótima, pois a seqüência de execução das rotas influenciará na solução final do problema.

A idéia central deste algoritmo é o estabelecimento das rotas de forma seqüencial. As rotas são ordenadas por algum critério e a seguir é estabelecido para a rota 1 a

solução de menor custo. Como a rota 2 no grafo aumentado não pode compartilhar nenhum enlace já escolhido pela rota 1, basta que quando da escolha da rota 2, todos os enlaces utilizados pela rota 1 tenham seus custos modificados para infinito. Assim, a escolha da rota 2 ótima nunca conterá enlaces utilizados na rota 1.

Este procedimento é repetido quando da escolha da rota 3. Os custos associados aos enlaces usados nas rotas anteriores são modificados para infinito.

A repetição para os Q enlaces é, então, executada, gerando uma solução viável sub-ótima para o nosso problema.

O procedimento proposto acima possui dois aspectos que podem fazer com que as rotas sub-ótimas produzidas tenham custo total muito elevado em relação ao procedimento de solução das rotas ótimas. São eles a saber:

i) o fato de se escolher uma única opção para a rota i, quando as rotas de índices 1 a (i-1) forem determinadas;
ii) a ordem com que as rotas são geradas.

O aspecto mencionado em i) pode ser facilmente contornável pela utilização de mais de uma opção de rota. Isto implica que se deve escolher os K melhores caminhos para uma dada rota. Este problema já foi abordado na literatura [24], [25] e um dos seus desdobramentos desagradáveis diz respeito à possibilidade de geração de "loops" (rotas em que se passa mais de uma vez pelo mesmo nó) nos caminhos ótimos a exceção do primeiro.

Existem métodos para geração dos K melhores caminhos sem repetição de nós, porém estes algoritmos tem complexidade da ordem de O ($K.n^3$), enquanto que os que admitem repetição de nós, são da ordem O (K.n.logn), onde n é o número de nós do grafo.

Dado a grande diferença de complexidade computacional dos dois casos, usaremos métodos que admitem repetição de nós.

É importante salientar que o nosso problema é particularmente sensível a gerar caminhos com nós repetidos, uma vez que o grafo aumentado possui arestas de custo zero. Esse problema pode ser contornado fazendo com que, quando os K melhores caminhos entre os nós r e s forem gerados, todos os enlaces entrantes em r e saintes de s tenham custo infinito. Com essas precauções tomadas, em nenhum exemplo investigado observou-se rotas com nós repetidos, embora reconheça-se que a geração de caminhos com nós repetidos não pode ser evitada de todo.

Quanto ao aspecto mencionado em ii), a solução implementada para a ordenação foi no sentido da "mais" para a "menos" crítica, onde por crítico entende-se como custo. Rotas ótimas de custo elevado, quando evitadas tendem a gerar outras de custo mais elevado ainda. Assim, as rotas são geradas na ordem inversa de seus custos ótimos individualizados.

Utilizou-se o algoritmo denominado de REA – *Recursive Enumeration Algorithm* [25] para a determinação dos K melhores caminhos entre dois nós, onde é possível a repetição de nós. Esse algoritmo também é de domínio público e encontra-se na *Internet* no endereço www.terra.act.uji.es/REA/.

Todo o procedimento pode ser visualizado em termos de um diagrama de árvore, como mostrado na figura abaixo (figura 3.3), onde cada nó na árvore é identificado a uma rota. Caminhos entre um nó inicial e um nó final indicam uma possível escolha de rotas para o problema.

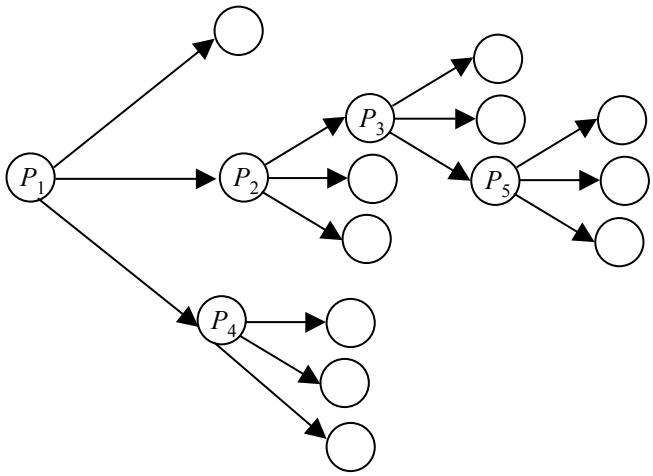

Figura 3.3: Diagrama em árvore

Essa árvore possui K^Q nós terminais, o que significa que existe K^Q caminhos cujos custos precisam ser levantados e investigados [24].

Para evitar o crescimento explosivo e exponencial da árvore de possibilidades de caminhos, algum tipo de limitação em seu crescimento precisa ser imposto. Lembrando que problemas semelhantes são encontrados em diversas áreas das Telecomunicações como decodificação seqüencial, códigos de treliça e outros, pode-se seguir algo a semelhança do algoritmo conhecido como algoritmo M [24]. Em linguagem simples, se numa determinada etapa, o número de nós ultrapassa um valor prefixado M, somente os M melhores nós serão preservados, sendo o resto descartado. É óbvio que neste processo é possível descartar a solução ótima do problema de interesse. Entretanto, a complexidade de memória deste processo é muito menor do que do método exato por Programação Linear Inteira e pode gerar solução que não fique muito distante da ótima.

Em resumo, o algoritmo proposto pode ser descrito da seguinte forma:

1) Seja G = (V, E) o grafo original, ω e c as funções custo já mencionadas e R = $\{R_i = (u_i, v_i)$ i $\in$ 1, 2, ..., G$\}$o conjunto dos pares origem-destino das rotas desejadas;
2) Constrói-se o grafo aumentado G* = (V*, E*) de acordo com o procedimento já descrito;

3) Para cada rota R_i determina-se o seu trajeto de custo mínimo com o grafo G pelo algoritmo de Dijkstra (é caso particular do REA onde K = 1) [25]. Essas rotas são decrescentemente ordenadas de acordo com o seu custo. Assim R_1 passa a ser a rota mais "cara", R_2 a segunda mais "cara" e, assim sucessivamente;

4) Determine os K melhores trajetos, segundo o algoritmo REA, para a rota R_1 e para cada um destes trajetos e gere uma fila com elementos (level, parent, trajeto, custo), onde:

   - Level = nível do nó da árvore correspondente ao trajeto;

- Parent = trajeto de nível anterior já escolhido;
- Trajeto = sequência de nós do grafo aumentado;
- Custo = custo total acumulado até então.

Nesta etapa, todos os nós da árvore tem level = 1 e parent = 0;

5) Construa um ponteiro (PTR) para o primeiro elemento da fila e faça $l_0 = 1$;

6) Enquanto PTR não atingir o fim da fila, faça:

- Selecione o elemento apontado por PTR. Seja l o seu nível;
- Se $l \neq l_0$;
- Modifique os custos associados a G* de modo que todos os enlaces utilizados por algum ancestral desse elemento tenha custo infinito;
- Determine os K melhores caminhos para a rota Rl+1 e armazene cada um desses trajetos na fila na forma de quádrupla discutida em 4);
- Incremente PTR.

7) Dos elementos que sobram na fila, escolha o de menor custo.