Universidade Estadual de Campinas

Faculdade de Engenharia Elétrica e de Computação

Departamento de Sistemas de Energia Elétrica

Planejamento a Longo Prazo de Sistemas de Transmissão Usando Técnicas de Otimização Combinatorial

Ramón Alfonso Gallego Rendón

Alcir José Monticelli
Orientador

Rubén Augusto Romero Lázaro
Co-orientador

Este exemplar corresponde a redação final da tese defendida por Ramón Alfonso Gallego Rendón e aprovada pela Comissão Julgada em 08/10/97.

Orientador

Tese apresentada à Faculdade de Engenharia Elétrica e de Computação da UNICAMP, como parte dos requisitos exigidos para a obtenção do título de Doutor em Engenharia Elétrica.

Campinas, outubro de 1997

Versão final aprovada
Alcir Monticelli
14/10/97

G135p
32103/BC

UNICAMP
BIBLIOTECA CENTRAL

UNIDADE BC
N.º CHAMADA: T/UNICAMP
G135p
V. Ex.
TOMBO BC/ 32103
PROC. 281/97
C [ ] D [x]
PREÇO R$ 11,00
DATA 18/11/97
N.º CPD

CM-00102677-1

FICHA CATALOGRÁFICA ELABORADA PELA
BIBLIOTECA DA ÁREA DE ENGENHARIA - BAE - UNICAMP

G135p

Gallego Rendón, Ramón Alfonso
Planejamento a longo prazo de sistemas de transmissão usando técnicas de otimização combinatorial / Ramón Alfonso Gallego Rendón.--Campinas, SP: [s.n.], 1997.

Orientadores: Alcir José Monticelli, Rubén Augusto Romero Lázaro.
Tese (doutorado) - Universidade Estadual de Campinas, Faculdade de Engenharia Elétrica e de Computação.

1. Energia elétrica - Transmissão. 2. Otimização combinatória. I. Monticelli, Alcir José. II. Romero Lázaro, Rubén Augusto. III. Universidade Estadual de Campinas. Faculdade de Engenharia Elétrica e de Computação. IV. Título.

Planejamento a Longo Prazo de Sistemas de Transmissão Usando Técnicas de Otimização Combinatorial

Ramón Alfonso Gallego Rendón
FEEC - UNICAMP
Campinas - SP

8 de outubro de 1997

Ao meu pai Alfonso, à memória de
minha mãe Ana Emília, e aos meus
irmãos Alonso, Gladys, Myriam,
Luz Dary, Edda del Socorro,
Juan Angel e Miguel Angel.

Agradecimentos

- Ao professor Alcir José Monticelli pela orientação e profissionalismo demonstrado;
- Ao professor Rubén Romero Lázaro pela amizade e apoio brindado durante minha estadia em Campinas, pelo que fez para torná-la mais agradável e pelo contínuo apoio na orientação da tese;
- Ao colega Sérgio Azevedo de Oliveira pela grande ajuda na correção do texto da tese e boa vontade dispensada;
- Ao professor Ariovaldo Verandio Garcia pela ajuda dispensada especialmente na utilização dos recursos do DSEE;
- Ao professor José Roberto Sanches Mantovani da Unesp (Ilha Solteira) pela assessoria prestada na paralelização do programa MINOS.
- Aos professores do departamento;
- Aos colegas do DSEE;
- A Edna Servidone, secretária do DSEE, pela ajuda dispensada;
- Ao professor Pompilio Tabares Espinosa da Universidad Tecnológica de Pereira (Colômbia), pela amizade e apoio;
- Ao professor Alfonso Alzate Gomez da Universidad Tecnológica de Pereira, pelo incentivo e contínuo apoio a través de correio eletrônico;
- Aos professores da Facultad e Escuela de Eléctrica da Universidad Tecnológica de Pereira pelo incentivo e apoio, assim como a Luz Elena e Adiela secretárias da Facultad de Mecánica e Tecnología Eléctrica respectivamente;

Este trabalho contou com o apoio financeiro do CNPq e do Colciencias (Colômbia).

Resumo

Este trabalho apresenta várias metodologias para resolver o problema de planejamento da expansão a longo prazo de sistemas de transmissão. Após observar os resultados pouco consistentes obtidos com os algoritmos heurísticos construtivos, especialmente quando foram usados na solução de sistemas de grande porte, e os resultados obtidos com os métodos de otimização usando a decomposição de Benders (os quais encontram as soluções ótimas para sistemas de pequeno e médio porte, porém são pouco efetivos quando aplicados na solução de sistemas de grande porte) propos-se metodologias de otimização combinatorial para o estudo deste problema. Assim, estudaram-se quatro metodologias de otimização que foram aplicadas no estudo do planejamento de sistemas de transmissão denominadas "Simulated Annealing" (**SA**), Algoritmos Genéticos (**AG**), Busca Tabu (**TS**) e Algoritmo Híbrido, chamadas metaheurísticas. Cada uma das quatro metodologias apresenta diferentes características na sua implementação, ou seja, em cada uma destas se determinam seus parâmetros de controle, a estrutura de vizinhança e seu próprio algoritmo. Nestes métodos, dada uma configuração, se deseja passar ao melhor vizinho através de entrada e saída de linhas. Em cada uma delas é usado o Modelo de Fluxo de Carga DC para realizar a modelagem da rede elétrica, sendo o mesmo resolvido através de um PL. No processo de elaboração destas metodologias foram incorporados diferentes conceitos para melhorar seus desempenhos, assim foi incorporado o conceito de caminho completo entre barras de carga, geração ou rede existente; e baseada nesta idéia, também foi incorporado o conceito de bloco construtivo no qual os blocos são construídos e conservados e, durante o processo, são transferidos de uma configuração para outra e só são destruídos quando a configuração que os contém é eliminada. Também é usado um algoritmo construtivo como programa inicializador para construir configurações iniciais, denominado metodologia de Garver modificada e usada especialmente em sistemas de grande porte, apresentando resultados de grande qualidade comparados com os obtidos quando as configurações iniciais são obtidas aleatoriamente.

Conteúdo

1 INTRODUÇÃO 1

2 MODELAGEM E RESOLUÇÃO DO PROBLEMA DE PLANEJAMENTO DE SISTEMAS DE TRANSMISSÃO 5

2.1 MODELAGEM DO PROBLEMA 5

2.1.1 Modelo DC 5

2.1.2 Modelo de Transportes 8

2.1.3 Modelo Híbrido 9

2.2 TÉCNICAS USADAS PARA RESOLVER O PROBLEMA DE PLANEJAMENTO 10

2.2.1 Algoritmos Heurísticos Construtivos Usados no Planejamento de Sistemas de Transmissão a Longo Prazo 10

2.2.2 Métodos de Otimização Clássica 13

2.2.3 Métodos de Otimização Combinatorial 15

3 PLANEJAMENTO DE SISTEMAS DE TRANSMISSÃO USANDO "SIMULATED ANNEALING" 17

3.1 INTRODUÇÃO 17

3.2 "SIMULATED ANNEALING": TEORIA 18

3.2.1 Algoritmo de Metrópolis 18

3.2.2 Algoritmo "Simulated Annealing" (SA) 19

3.2.3 Analogia Física e Apresentação da Metaheurística 21

3.2.4 Programa de Resfriamento 21

3.2.5 Convergência 26

3.3 *SA* SEQÜENCIAL APLICADO AO PROBLEMA DE PLANEJAMENTO 26

3.3.1 A Representação do Problema 27

3.3.2 Mecanismo de Transição 29

3.3.3 O Programa de Resfriamento 36

3.3.4 Algoritmo de "Simulated Annealing" Implementado 39

3.4 TESTES EM SISTEMAS PEQUENOS E MÉDIOS E CALIBRAÇÃO DE PARÂMETROS 42

3.4.1 Sistema de 06 Barras de Garver com Redespacho 43

3.4.2 Sistema de 06 Barras de Garver sem Redespacho 43

3.4.3 Sistema Sul Brasileiro com Redespacho 47

3.4.4 Sistema Sul Brasileiro sem Redespacho 50

3.5 TESTES EM UM SISTEMA DE GRANDE PORTE 50

3.5.1 Sistema Norte-Nordeste Brasileiro - Plano 2008 52

3.6 *SA* PARALELO APLICADO AO PROBLEMA DO PLANEJAMENTO . 54

3.6.1 Metodologias Básicas de Paralelização 56

3.6.2 Descrição dos Algoritmos Paralelos Implementados 59

3.6.3 Testes em Sistemas Pequenos, Médios e de Grande Porte 62

3.7 CONCLUSÕES 70

4 PLANEJAMENTO DE SISTEMAS DE TRANSMISSÃO USANDO ALGORITMOS GENÉTICOS (AG) 73

4.1 INTRODUÇÃO 73

4.2 O ALGORITMO GENÉTICO: TEORIA 75

4.2.1 População Inicial 76
4.2.2 Mecanismo de Codificação do Problema 77
4.2.3 Parâmetros de Controle 77
4.2.4 Avaliação do Nível de "Fitness" 78
4.2.5 Seleção e Operadores Genéticos do Ciclo Geracional 79
4.3 O ALGORITMO GENÉTICO BÁSICO APLICADO AO PROBLEMA DE PLANEJAMENTO 81
4.3.1 A Representação e Codificação do Problema 81
4.3.2 O Mecanismo Geracional: Seleção - "Crossover" - Mutação 86
4.3.3 Geração da População Inicial 86
4.3.4 Parâmetros de Controle e Critérios de Parada 88
4.4 IMPLEMENTAÇÕES ESPECIALIZADAS REALIZADAS NO ALGORITMO GENÉTICO 89
4.4.1 Taxa de Mutação Variável 90
4.4.2 Taxa de Mutação Controlada por *SA* 91
4.4.3 Seleção Limitada 91
4.4.4 Implementação dos "Building Blocks" Através dos Caminhos Novos 92
4.4.5 Formas Especializadas de Geração da População Inicial 95
4.4.6 Parâmetro α Variável 96
4.5 ALGORITMO IMPLEMENTADO 96
4.6 TESTES EM SISTEMAS PEQUENOS E MÉDIOS E CALIBRAÇÃO DE PARÂMETROS 97
4.6.1 Sistema de 06 Barras de Garver sem Redespacho 98
4.6.2 Sistema de 06 Barras de Garver com Redespacho 100
4.6.3 Sistema Sul Brasileiro sem Redespacho 101
4.6.4 Sistema Sul Brasileiro com Redespacho 103

4.7 TESTES EM SISTEMAS DE GRANDE PORTE 104

4.7.1 Sistema Norte - Nordeste Brasileiro - Plano 2008 104

4.8 CONCLUSÕES 109

5 PLANEJAMENTO DE SISTEMAS DE TRANSMISSÃO USANDO BUSCA TABU (TS) 110

5.1 INTRODUÇÃO 110

5.2 BUSCA TABU: TEORIA 112

5.2.1 Memória de Curto Prazo e seus Principais Elementos 116

5.2.2 Memória de Longo Prazo e suas Estratégias Associadas 119

5.2.3 Oscilação Estratégica 123

5.2.4 "Path Relinking" 126

5.2.5 Aspectos Importantes na Implementação Prática 128

5.3 BUSCA TABU APLICADO AO PROBLEMA DE PLANEJAMENTO . . 131

5.3.1 Apresentação do Problema 131

5.3.2 Solução do Problema por *TS* 132

5.3.3 O Problema da Estrutura de Vizinhança 136

5.3.4 Características de Memória de Curto e Longo Prazo 137

5.3.5 Intensificação e Diversificação no Algoritmo Implementado 141

5.4 IMPLEMENTAÇÕES ESPECIALIZADAS NO ALGORITMO DE BUSCA TABU 145

5.4.1 Implementação do "Path Relinking" 145

5.4.2 Critério de Parada 146

5.5 ALGORITMO DE BUSCA TABU IMPLEMENTADO 146

5.6 TESTES EM SISTEMAS DE PEQUENO E DE MÉDIO PORTE 148

5.6.1 Sistema de 06 Barras de Garver com Redespacho 148

5.6.2 Sistema de 06 Barras de Garver sem Redespacho 151

5.6.3 Sistema Sul Brasileiro com Redespacho 151

5.6.4 Sistema Sul Brasileiro sem Redespacho 151

5.7 COMENTÁRIOS SOBRE OS RESULTADOS OBTIDOS, INICIANDO COM *UMA* E COM k CONFIGURAÇÕES 156

5.8 TESTES EM SISTEMAS DE GRANDE PORTE 157

5.8.1 Sistema Norte - Nordeste Brasileiro - Plano 2008 157

5.9 CONCLUSÕES 160

6 PLANEJAMENTO DE SISTEMAS DE TRANSMISSÃO USANDO UM ALGORITMO HÍBRIDO 161

6.1 INTRODUÇÃO 161

6.2 ALGORITMO HÍBRIDO - TEORIA 162

6.3 ALGORITMO HÍBRIDO APLICADO AO PROBLEMA DE PLANEJAMENTO 164

6.3.1 Apresentação das Estratégias que Formarão o Algoritmo Híbrido . . 164

6.3.2 A Representação do Problema 165

6.3.3 O Problema da Estrutura de Vizinhança 166

6.3.4 Características de Memória de Curto e Longo Prazo 166

6.3.5 Intensificação e Diversificação 167

6.4 IMPLEMENTAÇÕES ESPECIALIZADAS NO ALGORITMO HÍBRIDO . 168

6.4.1 Mecanismo Básico do Algoritmo: Estratégia A - B - C 168

6.4.2 Uso de Uma População de Configurações 171

6.4.3 Implementação de Configurações Elite 171

6.4.4 Implementação de "Path Relinking" 171

6.4.5 Uso de Estratégias Inspiradas em **SA** 172

6.4.6 Aproveitamento das Características do **SA**, **GA** e **TS** no Problema do Planejamento de Redes de Transmissão 172

6.4.7 Implementação de Versões Híbridas 173

6.4.8 Critério de Parada. 173

6.5 ALGORITMO HÍBRIDO IMPLEMENTADO 174

6.6 TESTES EM SISTEMAS DE PEQUENO E DE MÉDIO PORTE 176

6.6.1 Sistema de 06 Barras de Garver com Redespacho 176

6.6.2 Sistema de 06 Barras de Garver sem Redespacho 178

6.6.3 Sistema Sul Brasileiro com Redespacho 178

6.6.4 Sistema Sul Brasileiro sem Redespacho 178

6.7 TESTES EM SISTEMAS DE GRANDE PORTE 186

6.7.1 Sistema Norte - Nordeste Brasileiro - Plano 2008 186

6.8 CONCLUSÕES 188

7 CONCLUSÕES **189**

A Dados dos Sistemas Testados **198**

A.1 Sistema de 06 Barras de Garver 199

A.2 Sistema Sul Brasileiro de 46 Barras 200

A.3 Sistema Nordeste Brasileiro de 87 Barras 205

B Artigos Publicados **213**

Capítulo 1

INTRODUÇÃO

O problema de planejamento de sistemas de transmissão a longo prazo é um problema de programação não linear inteira mista (PNLIM) que ainda apresenta o fenômeno da explosão combinatorial e que mereceu muita atenção de pesquisadores que usaram uma diversidade de métodos aproximados e de otimização para resolvê-lo. As principais dificuldades na resolução deste problema estão relacionadas com a natureza combinatorial do processo de planejamento que normalmente leva a um número explosivo de alternativas, inclusive no caso de sistemas de médio porte. Além disso, o problema de planejamento apresenta uma estrutura multimodal com um número muito elevado de ótimos locais o que leva a maioria dos métodos aproximados a parar numa solução ótima local e às vezes de pobre qualidade. Para contornar estes problemas foram apresentadas muitas alternativas na literatura especializada.

Neste tipo de planejamento, dada uma configuração inicial e os dados de geração e demanda do horizonte de planejamento (além de outros dados como limites de operação, custos e restrições de investimento), tenta-se determinar o plano de expansão com um custo mínimo, isto é, deve-se determinar *onde* e *que* tipos de novos equipamentos devem ser instalados. Naturalmente este é um subproblema de um caso mais geral, chamado planejamento dinâmico da expansão onde, adicionalmente, deseja-se saber *quando* instalar estes novos equipamentos. Estes resultados preliminares de planejamento posteriormente serão analisados e melhorados testando seu desempenho através de ferramentas de análise mais exatas como fluxo de carga AC, curto circuito, estabilidade transitória, etc.

Na literatura especializada existem três modelos ou formulações matemáticas para realizar a modelagem do problema de planejamento de sistemas de transmissão: modelo DC, modelo de transportes e modelo híbrido. Tradicionalmente, o modelo DC é considerado como sendo o ideal para representar o problema de planejamento sendo os outros modelos versões relaxadas ou simplificadas do modelo DC. Para resolver estes modelos foram usadas várias técnicas que podem ser agrupadas em três grupos: *métodos aproximados*

ou *heurísticos*, *métodos de otimização* e *métodos de otimização combinatorial.*

Os *métodos aproximados* apresentam a vantagem de fornecer soluções de boa qualidade com esforços computacionais pequenos; mas os mesmos raramente encontram a solução ótima de sistemas reais e não fornecem informação da qualidade da solução obtida, isto é, o quão perto da solução ótima se encontra a solução obtida. Entretanto, estes métodos ainda são os mais usados pelas concessionárias de energia elétrica nos trabalhos de planejamento. Os métodos aproximados mais eficientes e conhecidos foram apresentados em [9, 12, 35]. Em [12] e [35] foram usados algoritmos aproximados para resolver problemas de grande porte; são métodos que requerem pouco esforço computacional. Da qualidade da resposta, pode-se dizer que os mesmos apresentam resultados de menor qualidade em relação aos obtidos nos métodos aqui estudados.

Os *métodos de otimização*, geralmente usando técnicas de decomposição matemática, apresentam a característica de que encontram a solução ótima do problema de planejamento e são muito eficientes em sistemas de pequeno e médio porte, mas para sistemas de grande porte ainda apresentam problemas de elevado esforço computacional e de convergência. Os métodos de otimização mais conhecidos foram apresentados em [10, 11, 27, 62, 64, 65].

Os *métodos de otimização combinatorial* como "Simulated Annealing", Algoritmos Genéticos, Busca Tabu, etc., foram aplicados com muito sucesso para resolver muitos problemas do campo da pesquisa operacional e também em alguns problemas de engenharia elétrica. Estes métodos apresentam a característica geral de que convergem em soluções ótimas ou quase ótimas mas com um esforço computacional elevado.

Neste trabalho, estuda-se o planejamento *estático* de sistemas de transmissão empregando para sua resolução várias das técnicas metaheurísticas que mais êxito apresentaram na resolução de problemas do tipo combinatorial; ou seja: "Simulated Annealing" (tanto em sua versão seqüencial, como na paralela), algoritmos genéticos, busca tabu e algoritmo híbrido, este último formado a partir da combinação das melhores características dos métodos mencionados anteriormente.

O "Simulated Annealing" (**SA**), faz uso de conceitos originalmente descritos pela Mecânica Estatística; neste processo físico primeiro é reaquecido o sólido mediante seu aquecimento a uma temperatura elevada; a seguir se vai resfriando lentamente até que as partículas fiquem arranjadas no chamado estado fundamental do sólido. Em cadá nível de temperatura o sólido deve atingir o equilíbrio térmico, se o resfriamento se efetua lentamente; caso contrário, se o resfriamento se efetuar demasiadamente rápido, o sólido ficará com defeitos em sua estrutura.

Os *algoritmos genéticos* (**AG**) são técnicas de busca baseadas no mecanismo da seleção natural e na genética. Na natureza, os indivíduos melhor dotados têm maior chance de sobrevivência e a capacidade de adaptação a um meio mutante é fundamental na

sobrevivência de indivíduos e espécies. Foi baseado nesta lei natural que as espécies sobreviveram e evoluíram na terra. A estrutura do algoritmo genético é formada baseando-se na tentativa de abstração artificial do "algoritmo" de seleção próprio da natureza, com a expectativa de que assim se consigam êxitos similares em relação a capacidade de adaptação a um amplo número de ambientes diferentes.

Busca tabu (**TS**), nasce do campo da inteligência artificial. A busca tabu é um tipo de busca em torno de um ponto que, segundo seu autor, guia o procedimento de busca local para explorar o espaço de soluções além do ótimo local. Assim, a mesma seleciona de modo agressivo o melhor dos movimentos possíveis em cada passo podendo levar a uma degradação da função objetivo; contrário a uma busca local que sempre se move ao redor de um ponto e finaliza em um ótimo local quando não existe uma configuração vizinha melhor. Busca Tabu está baseado na premissa de que a resolução de um problema pode ser considerado inteligente se esse processo incorpora a *memória adaptativa* e a *exploração sensível.*

O algoritmo híbrido é formado por diferentes funções extraídas do **SA**, **AG** e **TS**; é implentado usando **TS** como base e incorporando neste as funções de melhor desempenho observadas em **SA** e **AG**.

A seguir é apresentada a seqüência de desenvolvimento deste trabalho.

No Capítulo II, apresenta-se a modelagem e resolução do problema de planejamento. São apresentadas três formas de modelagem do problema de planejamento, ou seja: modelo DC, modelo de transportes e modelo híbrido. Além disto são apresentadas as técnicas usadas para resolver o problema de planejamento: os métodos aproximados, métodos de otimização clássica e métodos de otimização combinatorial.

No Capítulo III, desenvolve-se o algoritmo "Simulated annealing"(**SA**), tanto na sua versão seqüencial como paralela. Para avaliar o desempenho do algoritmo são testados três sistemas: sistema de 06 barras de Garver [9], sistema Sul brasileiro de 46 barras apresentado em [62] e sistema Norte-Nordeste brasileiro de 87 barras apresentado em [6].

No Capítulo IV, desenvolvem-se os algoritmos genéticos (**AG**). Como no caso anterior, os testes são feitos sobre os três sistemas de pequeno, médio e grande porte.

No Capítulo V, desenvolve-se o algoritmo busca tabu (**TS**). Aqui os testes são feitos sobre os mesmos sistemas que foram mencionados anteriormente.

No Capítulo VI, desenvolve-se um algoritmo híbrido, implementado usando **TS** como base e incorporando neste funções próprias de **SA** e dos **AG**. Para testar este algoritmo são usados os mesmos sistemas que foram mencionados anteriormente.

No Capítulo VII, são feitos comentários dos resultados obtidos e as perspectivas de desenvolvimentos futuros.

No Apêndice A, são apresentados os dados dos sistemas testados e no Apêndice B são anexados os artigos que foram apresentados em congressos do IEEE assim como revistas internacionais, como conseqüência da elaboração da presente tese de doutorado.

Capítulo 2

MODELAGEM E RESOLUÇÃO DO PROBLEMA DE PLANEJAMENTO DE SISTEMAS DE TRANSMISSÃO

2.1 MODELAGEM DO PROBLEMA

2.1.1 Modelo DC

A formulação do problema mostrado abaixo é um problema PNLIM e pertence ao conjunto de problemas chamados *NP-completo* de difícil tratamento, que apresenta o problema da explosão combinatorial pois geralmente existem muitos caminhos candidatos e, além disso, em cada caminho podem ser alocadas várias linhas. Entretanto, quando

$$
\begin{aligned}
Min \quad v = & \sum_{(i,j)\in\Omega} c_{ij} n_{ij} \; + \; \sum_{i} \alpha_i r_i && (2.1)\\
s.a. \quad & B(x+\gamma^{\circ})\theta + g + r = d \\
& (x_{ij}+\gamma_{ij}^{\circ})|\theta_i - \theta_j| \leq (x_{ij}+\gamma_{ij}^{\circ})\overline{\phi}_{ij} \\
& 0 \leq g \leq \overline{g} \\
& 0 \leq r \leq d \\
& 0 \leq n_{ij} \leq \overline{n}_{ij} \\
& \forall (i,j) \in \Omega
\end{aligned}
$$

Descrição das variáveis da formulação:

c_{ij} Custo do ramo (i,j).

x_{ij} Nova susceptância a ser instalada no ramo $(i,j) \in \Omega$.

$B(.)$ Matriz de susceptâncias.

θ Ângulos das tensões nodais.

γ_{ij}^{o} Susceptância inicial no ramo $(i,j) \in \Omega$.

n_{ij} Número de linhas adicionadas, $n_{ij} = \frac{x_{ij}}{\gamma_{ij}}$, sendo γ_{ij} a susceptância nominal de uma linha entre as barras *i-j*.

$\overline{\phi}_{ij}$ Definida pela relação: $\overline{\phi}_{ij} = \frac{\overline{f}_{ij}}{\gamma_{ij}^{o}}$, sendo $\overline{f}_{ij}$ o fluxo máximo de potência permitido que passa pela linha *i-j*.

Ω Conjunto de todos os ramos definidos pelas linhas existentes e as alternativas de expansão.

g Vetor de gerações.

d Vetor de cargas.

$\overline{g}$ Vetor de limites de geração.

r Vetor de geradores fictícios ou artificiais.

α Parâmetro adequado de transformação de unidades.

são especificados os valores dos x_{ij}, isto é, quando é escolhida (ou determinada) uma proposta de investimento ou configuração, a formulação (2.1) se reduz a um simples problema de programação linear cuja única finalidade é verificar a factibilidade da proposta de investimento, isto é, verificar se o investimento proposto produz um corte de carga igual a zero.

Nesta formulação têm-se variáveis contínuas de operação tais como os fluxos nas linhas, os níveis de geração e demanda e a diferença angular nas barras; e variáveis de investimento inteiras, como são os circuitos candidatos à adição.

O problema geral para o planejamento da transmissão em sistemas de energia elétrica, usando o modelo DC, pode ser escrito na seguinte forma:

O problema (2.1), chamado de modelo DC, é considerado como o modelo ideal para ser usado em planejamento de sistemas de transmissão. Este modelo somente leva em

conta as duas leis de Kirchhoff para o sistema elétrico e a capacidade de transmissão das linhas existentes e das candidatas.

O modelo apresentado em (2.1) está ligeramente modificado em relação ao modelo DC convencional. Assim, por exemplo, foi adicionado o termo $\sum_i \alpha_i r_i$ na função objetivo para facilitar o processo de resolução. Quando é escolhido um valor de α relativamente grande, na solução final todos os r_i são iguais a zero. E nestas condições a solução obtida é exatamente igual para a formulação original e a modificada.

A grande vantagem de adicionar o segundo termo na função objetivo é que o problema transformado é sempre factível, pois as sobrecargas no sistema, que eventualmente apareceriam no processo de resolução, são compensadas pelos geradores artificiais alocados em todas as barras de carga. Portanto, é importante levar em conta o conceito de factibilidade deste problema de planejamento. Assim, para uma determinada proposta de investimento x^k (uma configuração) o problema (2.1) é simplesmente um problema de programação linear (PL). Se sua resolução apresenta corte de carga ($r \neq 0$), então a proposta de investimento é infactível para a formulação original mas é factível para a formulação (2.1). Portanto, um algoritmo eficiente que resolva (2.1) deve apresentar uma solução final sem corte de carga para que essa solução seja equivalente à formulação original; já no processo de resolução, eventualmente a formulação (2.1) encontra soluções "infactíveis", mas as mesmas permitem um melhor desenvolvimento do mecanismo de transições e permitem sair mais facilmente de soluções ótimas locais como serão mostrados mais adiante.

No problema de planejamento (2.1), se for especificada uma proposta de investimento $n^k = n_{ij}^k$, o problema resultante é simplesmente um problema de mínimo corte de carga. Assim, uma vez definida uma alternativa de investimento, o problema (2.1) se reduz a um problema de programação linear e pode ser resolvido por qualquer técnica eficiente de PL. O algoritmo de PL, com uma alternativa de investimento definida $n_{ij}^k \rightarrow x_{ij}^k$, resolve o seguinte problema:

$$
\begin{aligned}
Min \quad & w = \sum_i \alpha_i r_i && (2.2)\\
s.a. \quad & B(x+\gamma^o)\theta + g + r = d \\
& (x_{ij}^k + \gamma_{ij}^o)|\theta_i - \theta_j| \leq (x_{ij}^k + \gamma_{ij}^o)\overline{\phi}_{ij} \\
& 0 \leq g \leq \overline{g} \\
& 0 \leq r \leq d
\end{aligned}
$$

O sistema (2.2) é um PL com valores de x_{ij}^k conhecidos ou determinados para uma alternativa de investimento. Portanto, para um x^k escolhido, a solução obtida por um

algoritmo de PL aplicado ao sistema (2.2) indicará simplesmente se o sistema apresenta corte de carga (infactível) ou não (factível) em relação a formulação original. Esta idéia básica é utilizada ao desenvolver os algoritmos de **SA**, **AG**, **TS** e **híbrido** para o problema de planejamento.

2.1.2 Modelo de Transportes

O modelo de transportes foi formulado por Garver em [9] e teve muito sucesso, sendo esta a primeira proposta para planejamento de redes de transmissão que usou programação linear. Esta metodologia consiste basicamente em resolver de maneira aproximada uma versão relaxada do modelo DC. No modelo de Garver, conhecido como *modelo de transportes*, somente se leva em conta a Primeira Lei de Kirchhoff e a capacidade de transmissão das linhas, não sendo levado em conta, portanto, a Segunda Lei de Kirchhoff.

O planejamento de uma rede geralmente apresenta dois problemas: não é conexa e apresenta linhas sobrecarregadas. Nestas condições é impossível resolver um fluxo de carga AC. Para resolver este problema foi proposta esta nova metodologia, formulando um algoritmo baseado em um modelo simplificado da rede. Assim, o modelo de transportes assume a seguinte forma:

$$
\begin{aligned}
Min \quad v = & \sum_{(i,j)\in\Omega} c_{ij} n_{ij} \quad + \sum_{i} \alpha_i r_i \qquad (2.3)\\
\text{s.a.} \quad & Sf + g + r = d \\
& |f_{ij}| - (x_{ij} + \gamma_{ij}^{o})\overline{\phi}_{ij} \leq 0 \\
& 0 \leq g \leq \overline{g} \\
& 0 \leq r \leq d \\
& 0 \leq n_{ij} \leq \overline{n}_{ij} \\
& \forall (i,j) \in \Omega
\end{aligned}
$$

em que S é a matriz de incidência nó-ramo e f_{ij} o fluxo de potência no ramo *i-j*.

O modelo de transportes é uma relaxação do modelo DC pois não leva em conta um conjunto de restrições (Segunda Lei de Kirchhoff) e portanto, uma das desvantagens do modelo de transportes é que a sua solução (mesmo sendo a ótima) pode não ser adequada do ponto de vista do modelo DC, pois para este, a solução do modelo de transportes poderia apresentar um corte de carga inadequado. A vantagem é que o problema (2.3) é um problema linear inteiro misto (PLIM) e relativamente mais fácil de se resolver. Apesar de o modelo de transportes ser uma formulação relaxada, Garver usou um algoritmo

aproximado para resolvê-lo. Entretanto, o algoritmo de Garver foi bem aceito, pois na época não existiam algoritmos melhores para resolver o problema de planejamento da transmissão.

2.1.3 Modelo Híbrido

Este modelo é uma combinação do modelo de transportes e o modelo DC. Neste, trata-se de contornar as desvantagens na solução do modelo de transportes. Adicionou-se ao mesmo somente uma parcela das restrições correspondentes a Segunda Lei de Kirchhoff, isto é, são consideradas como parte da formulação do problema aquelas restrições da Segunda Lei de Kirchhoff correspondentes às linhas existentes e eliminadas as restrições correspondentes aos novos caminhos. A formulação assim definida é conhecida como **modelo híbrido** e foi introduzida pelo próprio Garver em [26]. Assim, o chamado modelo híbrido assume a seguinte forma:

$$
\begin{aligned}
Min \quad v = & \sum_{(i,j)\in\Omega} c_{ij} n_{ij} \; + \; \sum_{i} \alpha_i r_i \qquad (2.4)\\
\text{s.a.} \quad & B_1\theta + S_2 f + g + r = d \\
& |(\theta_i - \theta_j| \leq \overline{\phi}_{ij} \qquad \forall\ (i,j)\ \epsilon\ \Omega_1 \\
& |f_{ij}| \leq (x_{ij} + \gamma^{\circ}_{ij})\overline{\phi}_{ij} \qquad \forall\ (i,j)\ \epsilon\ \Omega_2 \\
& 0 \leq g \leq \overline{g} \\
& 0 \leq r \leq d \\
& 0 \leq n_{ij} \leq \overline{n}_{ij}
\end{aligned}
$$

em que

Ω_1 Representa o conjunto de ramos onde já existem linhas na configuração básica.

Ω_2 Representa o conjunto de ramos candidatos em caminhos onde não existe linha na configuração inicial.

B_1 É a matriz de susceptância para os ramos que pertencem a Ω_1.

S_2 É a matriz de incidência nó-ramo para os ramos que pertencem a Ω_2.

2.2 TÉCNICAS USADAS PARA RESOLVER O PROBLEMA DE PLANEJAMENTO

2.2.1 Algoritmos Heurísticos Construtivos Usados no Planejamento de Sistemas de Transmissão a Longo Prazo

Na literatura especializada são apresentados diversos métodos aproximados que determinam planos de expansão de redes de transmissão a longo prazo. O primeiro método aproximado que apareceu foi o de Garver [9]; neste método se propõe um algoritmo aproximado para encontrar uma boa configuração e não necessariamente a configuração ótima. Estes métodos geralmente são de fácil implementação e requerem pouco esforço computacional; com relação a qualidade da resposta, seu valor fica geralmente afastado da resposta ótima. Este afastamento é maior ao ser resolvidos sistemas de grande porte.

Além do método de Garver, menciona-se o método de Villanasa [26], que é uma modificação do anterior. Sendo que o mesmo trabalha usando o modelo híbrido. Também cabe destacar as metodologias do mínimo esforço usando o modelo DC [12] e o método de mínimo corte de carga, baseado em um programa linear especializado e também usando o modelo DC [35]; metodologias estas empregadas no planejamento dos sistemas de trasmissão das empresas de energia elétrica.

Metodologia de Garver

O algoritmo de Garver [9] é um processo chamado de passo-a-passo onde em cada iteração do algoritmo se toma uma decisão de adicionar uma linha à configuração atual. A linha que é adicionada é aquela que aparece mais sobrecarregada quando o modelo de transportes [equivalente a formulação 2.3] é resolvido.

Esta metodologia considera dois tipos de ligações:

- **Ligações Normais:** Com capacidade de transmissão máximas iguais às linhas reais e custos de transporte iguais às reatâncias dessas linhas.
- **Ligações de Sobrecarga:** São ligações fictícias com capacidades de transmissão ilimitadas e custos de transporte muito superiores aos das linhas normais; estas ligações são colocadas entre todos os nós ou barras nas quais sejam permitidas a construção de novas linhas.

Na metodologia de Garver, todo fluxo que não puder ser transportado pelas ligações normais, fluirão pelas ligações de sobrecarga, pois estas têm capacidades ilimitadas, e só

passarão através das ligações de sobrecarga quando for impossível transportá-los pelas ligações normais, já que estas tem custos muito inferiores. Em cada estágio do processo de planejamento, deve-se resolver um problema de programação linear e logo adicionar um circuito na trajetória de maior sobrecarga. O processo é repetido até eliminar todas as sobrecargas.

A vantagem da metodologia de Garver é a simplicidade na implementação do algoritmo pois ela exige somente soluções sucessivas de programação linear. A maior limitação da metodologia é que ela não garante a obtenção da solução ótima do sistema planejado. Portanto, a metodologia de Garver, em essência, é de naturaleza heurística.

Metodologia de Villanasa-Garver

A metodologia proposta por Villanasa [26] é uma extensão da metodologia de Garver, na qual se adiciona a segunda lei de Kirchhoff na rede existente na formulação apresentada em (2.3), com isto, gera-se a formulação apresentada em (2.4) chamado de modelo híbrido e depois é resolvido usando uma metodologia muito parecida com a metodologia de Garver. Esta metodologia também não garante a otimalidade da solução obtida.

Nesta metodologia, mantêm-se os conceitos de linhas de sobrecarga e, usa-se programação linear para determinar o circuito mais sobrecarregado e, portanto, candidato a adição de um novo circuito.

Metodologia do Mínimo Esforço

Esta metodologia [12] foi desenvolvida na Unicamp e é formulada usando o modelo DC. Esta metodologia faz também um plano de expansão passo-a-passo, isto é, para uma configuração da rede os circuitos são adicionados um a um ou em pequenos grupos. O critério para a adição do próximo circuito é determinado por uma análise de sensibilidade, chamada de mínimo esforço.

Nesta metodologia, a dificuldade de desconexões na rede inicial é contornada adotando-se, superposta à configuração do sistema, uma "rede fictícia" constituída por ligações com susceptâncias igual a, por exemplo, 10^{-4} vezes os valores nominais, colocados em todos os ramos onde são permitidas a construção de novas linhas. A baixa capacidade de transmissão da rede fictícia faz com que estas só sejam utilizadas quando não houver possibilidade de transporte de potência pela rede real.

Nesta emprega-se a relação seguinte, a qual é utilizada como um critério de desempenho na adição de novos circuitos ao sistema.

$$\Delta Z_{ij}^{*} = -\frac{1}{2}(\theta_i^{*} - \theta_j^{*})^2 \Delta\gamma_{ij} \tag{2.5}$$

em que $\Delta\gamma_{ij}$ é a variação da susceptância de um circuito no ramo ij.

Em cada passo do processo de planejamento é adicionado ao sistema aquele circuito que produz o maior impacto na distribuição de fluxos na rede, isto é, aquele que apresenta o maior valor de $|\Delta Z_{ij}|$.

Esta metodologia tem como vantagem a sua rapidez e geralmente apresenta soluções de boa qualidade. A desvantagem é que ela não garante a otimalidade da solução e, às vezes, pode apresentar soluções distantes do ótimo global, pois não existe forma de determinar o quanto perto da solução ótima está a solução encontrada.

A metodologia de mínimo esforço foi usada com sucesso em aplicações práticas com sistemas brasileiros. Baseados nesta metodologia foi projetado um software denominado SINTRA (Síntese de Transmissão) pela UNICAMP em convênio com a ELETROBRÁS.

Metodologia de Mínimo Corte de Carga

O algoritmo de Mínimo Corte de Carga (MCC) [35], usado no planejamento de sistemas de transmissão, é um algoritmo heurístico de tipo construtivo que em cada passo do algoritmo produz a adição de um circuito na configuração base. Estes algoritmos também são conhecidos como algoritmos passo-a-passo, pois em cada iteração deve-se decidir a adição de um circuito na configuração base até que o sistema opere adequadamente, isto é, sem corte de carga.

Em cada iteração do algoritmo de mínimo corte de carga é realizada a adição de um circuito e esse circuito é selecionado de acordo a um índice de desempenho ou índice de sensibilidade.

O índice de sensibilidade que permite encontrar o circuito mais atrativo para adição é determinado pela seguinte relação:

$$IS_{mcc} = (\pi_i - \pi_j)(\theta_i - \theta_j) \tag{2.6}$$

em que π_j é o multiplicador de Lagrange da *j-ésima* restrição do sistema $B\theta + g + r = d$ e os θ_j são os ângulos de tensão de barra obtidos ao resolver (2.1) para a configuração corrente usando um algoritmo de PL.

O índice de sensibilidade é um indicador do impacto que produziria a adição de um circuito no corte de carga de um sistema se o circuito fosse adicionado ao sistema elétrico.

Assim, aquele circuito que possui o maior valor absoluto do índice de sensibilidade deve ser adicionado à configuração base pois é o melhor candidato para produzir uma maior diminuição no corte de carga do sistema.

O algoritmo construtivo apresentado em [35], que apresenta a mesma estrutura do algoritmo de mínimo esforço, apresenta a seguinte estrutura básica:

FASE I: Adição de Circuitos

1. Define-se a configuração base do sistema elétrico e os circuitos candidatos a adição.
2. Calcula-se o corte de carga do sistema, para a configuração base corrente, usando um algoritmo de PL. Se não existe corte de carga, vai para Fase II. Caso contrário, calcula-se o índice de sensibilidade usando (2.6) e ordena-se em uma lista, os circuitos em ordem decrescente do valor absoluto de IS_{mcc}.
3. Adiciona-se o primeiro circuito da lista ordenada à configuração corrente, obtendo-se uma nova configuração base. Volta-se para o passo 2.

FASE II: Retirada de Circuitos

1. Nesta fase, tenta-se retirar circuitos adicionados que se revelaram supérfluos. Ordena-se numa lista, pelo custo (ou pelo nível de carregamento), os circuitos adicionados. Tenta-se retirar cada circuito, um a um, verificando se sua retirada não produz sobrecarga. Se o circuito foi retirado, então deve-se percorrer novamente a lista até que não seja mais possível retirar nenhum circuito adicionado.

2.2.2 Métodos de Otimização Clássica

Na década de 80, iniciou-se uma nova fase na tentativa de resolver a formulação (2.1) de maneira ótima e a principal ferramenta matemática encontrada foram as técnicas de decomposição matemática. O objetivo principal era o de encontrar a solução ótima do problema de planejamento da transmissão usando o modelo DC, isto é, resolver a formulação (2.1), o que significa resolver um problema PNLIM.

Nesta perspectiva, a metodologia mais usada foi a *técnica de decomposição de Benders* a qual explora a decomposição natural do problema de planejamento da expansão de sistemas de transmissão em duas partes, ou seja:

- **Um subproblema de investimento** em que se escolhe um plano de expansão candidato e, calculam-se os custos de investimento associados ao mesmo.

- **Um subproblema de operação** onde é testado o plano de expansão candidato em termos do adequado atendimento da carga.

A otimização global é atingida através de uma resolução iterativa das resoluções separadas dos subproblemas de operação e investimento.

Os primeiros pesquisadores a empregar os esquemas de decomposição matemática foram os do CEPEL com os trabalhos apresentados em [10, 62]; nestes, usam a decomposição de Benders para resolver o problema de planejamento usando o modelo de Transportes e o modelo DC, separadamente.

Em um trabalho posterior [34], apresentado pelo mesmo grupo de pesquisadores, foram apresentados os resultados teóricos de uma pesquisa importante, usando a decomposição de Benders generalizada para problemas de planejamento. Nesta pesquisa, é realizada uma análise teórica sobre os problemas de convexidade e sobre as características dos cortes de Benders.

Retomando a pesquisa anterior realizada em [34], o grupo da Unicamp propõe em [27] o uso de um esquema de decomposição hierarquizado composto por três fases: A primeira fase resolve o problema do planejamento da transmissão usando decomposição de Benders no qual o subproblema de operação é resolvido com o modelo de transportes e relaxando as restrições de integralidade das variáveis de investimento. Na segunda fase a rede é modelada com o modelo híbrido. Finalmente na última fase considera o modelo DC e assim o subproblema de operação é resolvido usando um algoritmo de **PL** e o subproblema de investimento é resolvido usando um algoritmo de enumeração implícita [11].

Posteriormente aparecem outros trabalhos tal como o da referência [65] no qual emprega um esquema de decomposição hierarquizado composto de duas fases; e o da referência [64], o qual propõe um método heurístico que aproveita a decomposição natural do problema em subproblemas de operação e investimento. O subproblema de investimento é resolvido empregando um algoritmo heurístico.

Com estas metodologias de decomposição foram obtidas as soluções globais em sistemas de pequeno e médio porte, como é o caso do sistema Garver de 6 nós e 15 linhas e o sistema Sul brasileiro de 46 nós e 79 linhas. Já para sistemas de grande-porte, como é o caso do sistema Norte-Nordeste de 87 nós e 183 linhas a metodologia de decomposição não obteve convergência devido ao número incalculável de alternativas que se apresentam na última fase do processo onde se tem que resolver um problema de programação inteira.

Já que estas metodologias de decomposição foram incapazes de resolver sistemas de grande porte como o sistema Norte-Nordeste, foram iniciadas novas pesquisas relacionadas com os métodos de otimização combinatoriais, cujas características fundamentais são as de resolver sistemas de grande porte, chegar a soluções próximas ao ótimo global e obter soluções em tempos de computação razoáveis.

2.2.3 Métodos de Otimização Combinatorial

Resolver um problema de otimização combinatorial consiste em determinar a "melhor" ou a solução "ótima" entre um grande número de soluções alternativas. Assume-se para isto que a qualidade de uma solução é quantificável e que pode ser comparada com alguma outra solução; além disso, o conjunto de soluções é finito.

Nas últimas décadas uma grande variedade de problemas surgiram em diversas áreas, tais como: ciências matemáticas, ciências da computação, engenharia, etc. Entre todos os problemas de otimização combinatorial, o problema do caixeiro viajante é provavelmente o mais conhecido. Neste problema o caixeiro inicia o percurso em uma cidade, visita cada uma das cidades preescritas na lista e retorna à cidade de partida, de tal maneira que o comprimento da rota seja mínimo. A importância deste problema está no aspecto que combina as características típicas de grandes problemas de otimização combinatorial, contendo os elementos matemáticos complicantes.

Uma conseqüência direta da propriedade de "NP-complete" é que a solução ótima não pode ser obtida em razoáveis tempos de computação. Grandes problemas "NP-complete" podem ser resolvidos com algoritmos apropriados; estes, por sua vez, podem ser selecionados entre duas opções: uma que obtém o ótimo global com a desvantagem de requerer tempos de computação impraticáveis, e a outra que obtém soluções sub-ótimas com menor tempo de computação.

A primeira opção, constitui a classe de algoritmos de otimização. Exemplos bem conhecidos são métodos de enumeração usando planos de corte (*"branch and bound"*, entre outros) ou *técnicas de programação dinâmica.* A segunda opção, constitui a classe de algoritmos de aproximação, também freqüentemente chamados algoritmos heurísticos, tais como: *algoritmos de busca local* e *algoritmos randômicos.* A divisão entre estas duas classes de algoritmos não é muito restrita. Alguns tipos de algoritmos podem ser empregados com ambos propósitos; por exemplo, regras de limitação heurística de "branch and bound" podem ser facilmente transferidas a um algoritmo aproximado.

Além disso, distinguem-se em ambas classes: *algoritmos gerais* e *algoritmos especializados.* Os primeiros, aplicáveis a uma grande variedade de problemas e portanto podem ser chamados problemas independentes. Os segundos, empregam informação específica e sua aplicação é por conseguinte limitada a um conjunto restrito de problemas. Nos algoritmos especializados, em cada tipo de problema de otimização combinatorial um novo algoritmo é construído. Algoritmos tais como: "simulated annealing", genéticos e busca tabu, são algoritmos gerais de alta qualidade, apesar de que em implementações práticas estes algoritmos se comportam como algoritmos de aproximação.

Os algoritmos acima estão sendo empregados na solução de sistemas de grande porte; nestes, reportam-se resultados muitos satisfatórios na maioria das aplicações apresentadas. O **SA** faz uso de conceitos descritos na Mecânica Estatística. No processo físico o material

é esquentado a uma temperatura elevada para logo ser resfriado de maneira lenta de tal forma que a estrutura do material ao final do processo fique perfeita. O algoritmo genético, é uma técnica de busca baseada na mecânica da seleção natural das espécies. Busca tabu, nasce de regras geradas na inteligência artificial; neste, se conta com um esquema de busca local para explorar o espaço de soluções além do ótimo local.

Quando um método exato tem capacidade de resolver um problema de otimização então o mesmo deverá ser empregado; com isto, garante-se a obtenção da solução ótima. O grande incoveniente é o requerimento de um grande esforço computacional. No caso de não existir uma metodologia exata, como mencionado anteriormente, recorre-se aos métodos heurísticos os quais acham soluções que estão perto do ótimo global em tempos de computação relativamente baixos, dependendo da qualidade da resposta que se deseje obter.

Neste trabalho foram analisadas as metodologias de otimização combinatorial mencionadas anteriormente e aplicadas na resolução do problema da expansão das redes de transmissão a longo prazo.

Capítulo 3

PLANEJAMENTO DE SISTEMAS DE TRANSMISSÃO USANDO "SIMULATED ANNEALING"

3.1 INTRODUÇÃO

Neste capítulo se estuda o planejamento *estático* de sistemas de transmissão empregando para sua resolução o algoritmo de "Simulated Annealing" tanto na versão seqüencial como na versão paralela. Na versão seqüencial a ênfase principal é o desenvolvimento e avaliação experimental de diferentes programas de resfriamento e a calibração empírica dos parâmetros dos mesmos. Na versão paralela é feita uma análise com os diferentes algoritmos paralelos do **SA**, ou seja: *divisão*, *"cluster"* e *híbrido*. A escolha do algoritmo é feita após estudar sistemas pequenos, médios e de grande porte; onde também é feita uma avaliação experimental do programa de resfriamento e a calibração empírica dos parâmetros de controle.

Comparados os resultados com os obtidos em investigações anteriores, os quais usaram métodos de solução aproximados para resolver problemas de planejamento de sistemas de transmissão de grande porte e que foram apresentados em [35, 12] e métodos de otimização, geralmente usando técnicas de decomposição matemática apresentados em [10, 11, 27] pode-se concluir que os algoritmos de **SA** aqui estudados, tanto na versão seqüencial como paralela, apresentam excelentes resultados.

Este capítulo é organizado como segue. Inicialmente é apresentada um descrição teórica do algoritmo **SA**, depois é discutido brevemente as características da metodologia de **SA** e as características específicas na sua aplicação no problema de planejamento. Neste último, se inclui:

- A representação do problema,
- O mecanismo de transição,
- O programa de resfriamento, e
- O algoritmo **SA** implementado.

A seguir, são apresentados testes com sistemas pequenos e de médio porte. Também são apresentados resultados obtidos com um sistema de grande porte, o sistema Norte-Nordeste brasileiro. Na seqüência, é apresentado o algoritmo **SA** paralelo; este inclui tanto as metodologias básicas de paralelização como resultados obtidos com sistemas pequenos, médios e de grande porte. Finalmente são mencionadas as conclusões mais relevantes.

3.2 "SIMULATED ANNEALING": TEORIA

3.2.1 Algoritmo de Metrópolis

A metodologia "simulated annealing" foi definida no início da década de 80, como uma nova ferramenta a ser empregada na solução de grandes problemas combinatoriais. Surgiu do campo da termodinâmica como conseqüência da comparação de problemas formulados neste campo com os do campo da pesquisa operacional; é uma metodologia simples e de grande potencialidade para ser aplicada a uma grande variedade de problemas.

A idéia original que deu lugar a esta metaheurística é chamada *algoritmo de Metrópolis*, o que por sua vez está baseado no *método de Monte-Carlo*, com o qual se estudam as propriedades de equilíbrio na análise do comportamento microscópico dos corpos.

O algoritmo de Metrópolis gera uma seqüência de estados de um sólido, ou seja: dado um sólido em um estado i e com energia E_i, gera-se o estado seguinte j mediante a aplicação de um mecanismo que transforma para o estado seguinte através de um pequeno distúrbio. A energia do próximo estado é E_j; se a diferença de energia E_j - E_i é menor ou igual a zero, o estado j é aceito. Se a diferença de energia é maior que zero, o estado j é aceito com certa probabilidade, a qual é dada como:

$$e^{\{\frac{E_i - E_j}{k_b T}\}}$$

Em que T denota a temperatura e k_b uma constante física conhecida como *constante de Boltzmann*. A regra de aceitação descrita é chamada *critério de Metrópolis* e o algoritmo como *algoritmo de Metrópolis*.

Se a diminuicão da temperatura é feita de maneira paulatina, o sólido poderá alcançar o estado de equilíbrio em cada nível de temperatura. No algoritmo de Metrópolis isto se consegue após gerar um grande número de transições em um dado nível de temperatura.

Para cada valor de temperatura T, o sólido acha um equilíbrio térmico, caracterizado pela probabilidade de estar em um estado i com energia E_i dado pela distribuição de Boltzmann:

$$P_T\{X = i\} = \frac{1}{Z(T)} e^{(\frac{-E_i}{K_b T})}$$

Em que:

X denota uma variável estocástica do estado atual do sólido e Z(T) é definida assim:

$Z(T) = \sum_j e^{(\frac{-E_j}{K_b T})}$ é um fator de normalização, conhecido como função partição;

k_b é a constante de Boltzmann; e

$e^{(\frac{-E_i}{K_b T})}$ é conhecido como fator de Boltzmann.

3.2.2 Algoritmo "Simulated Annealing" (SA)

Neste algoritmo aplica-se uma ação combinada do mecanismo de geração de alternativas e do critério de aceitação. T_k denota o valor do parâmetro de controle (temperatura) e N_k o número de alternativas geradas na *k-ésima* iteração do algoritmo.

Inicialmente quando T é grande, grandes deteriorações da função objetivo são aceitas, quando T decresce, somente pequenas deteriorações são aceitas e finalmente quando T tende a zero, nenhuma deterioração é aceita. Esta característica faz com que o algoritmo **SA** seja diferente em relação aos algoritmos de ótimos locais.

A partir do estado i com custo *f(i)* gera-se o estado j com custo *f(j)*. O critério de aceitação determina se este novo estado é aceito; para isto calcula-se a seguinte probabilidade:

$$P_T\{aceita_j\} = \begin{cases} 1 & se \leftrightarrow f(j) \leq f(i) \\ e^{(\frac{f(i)-f(j)}{T})} & se \leftrightarrow f(j) > f(i) \end{cases}$$

A Figura 3.1 a seguir, apresenta a descrição do algoritmo *"annealing"*.

Algoritmo "Simulated Annealing";

inicia

inicializar (T_0, N_0);

$k := 0$;

configuração inicial S_i

repetir o procedimento

do $L := 1$ **to** N_k

gere $(S_j \ de \ S_i)$;

se $f(j) \leq f(i)$ **faça** $S_i := S_j$;

caso contrário

if $\exp\left(\frac{f(i)-f(j)}{T_k}\right) > \text{random}[0,1]$ **faça** $S_i := S_j$;

end do;

$k := k + 1$;

Cálculo do comprimento de (N_k);

Cálculo do parâmetro de controle (T_k);

Critério de parada

termina;

Fonte: *[Aarts & Korst (1989)]*

Figura 3.1: Descrição do algoritmo "annealing"

3.2.3 Analogia Física e Apresentação da Metaheurística

Fazendo uma analogia entre os parâmetros que intervem em uma simulação termodinâmica e os que aparecem em métodos de otimização local se tem:

TERMODINÂMICA	OTIMIZAÇÃO
Configuração	Solução factível
Configuração fundamental	Solução ótima
Energia da configuração	Custo da solução
Temperatura	Parâmetro de ajuste

A temperatura não corresponde a um significado real no campo da otimização; pelo contrário, é um parâmetro que se ajusta durante o processo. Em otimização um movimento será aceito se seu custo diminui; ou caso contrário, se sua probabilidade de aceitação é maior que um número aleatório, este tipo de estratégia permite sair de ótimos locais. Na medida em que evolui o processo, diminui a temperatura, e por sua vez a probabilidade de aceitar uma solução pior.

A estratégia seguida em **SA** é partir de uma temperatura alta, permitindo aceitar as soluções piores; posteriormente, se diminui a temperatura diminuindo a possibilidade de aceitar as soluções piores. Inicia-se de uma temperatura inicial T_o e uma velocidade de resfriamento; a temperatura T_{k+1} calcula-se a partir de T_k e a velocidade de resfriamento após ser feito $N(T_k)$ iterações na temperatura T_k.

3.2.4 Programa de Resfriamento

Não existem regras para selecionar o melhor programa de resfriamento, isto dependerá do tipo de aplicação e a forma como os parâmetros estão associados à representação do problema.

Temperatura inicial

Uma característica que deve cumprir toda heurística de busca é a de não ser dependente da solução inicial de partida. Isto, o **SA** faz partindo de uma temperatura inicial alta.

A temperatura inicial deve ser calculada de tal maneira que só será aceito um número determinado de configurações propostas; destas, as que tiverem pior valor da função objetivo não poderão ultrapassar uma porcentagem de piora em relação a função objetivo que a originou. Isto, de tal forma que a configuração não se afaste para regiões pouco

atrativas, o qual implicaria mais tempo de computação, ao tentar retornar às regiões mais atrativas.

O valor da temperatura inicial T_0 é obtido de forma construtiva, simulando as transições feitas no início do processo para a primeira cadeia de Markov. O tamanho da cadeia de Markov inicial (N_0) é estimado em função do número de linhas do sistema estudado, ou seja, um valor múltiplo de (nl), assim : $N_0 = k_{fator}.nl$, onde *kfator* assume valores inteiros de 1,2,3,... A escolha deste fator dependerá do sistema estudado, por exemplo, o sistema Garver poderá assumir valores de 1 ou 2; o sistema Sul-Sudeste de 2, 3 ou 4 e o sistema Norte-Nordeste de 3, 4, 5 ou 6.

A cadeia de Markov e a temperatura tem uma relação inversa; portanto, ao ser empregados comprimentos de cadeias mais longos, a temperatura inicial será mais baixa.

No algoritmo usado para obter a temperatura inicial, emprega-se a seguinte fórmula para o cálculo da relação de aceitações:

$$X = \frac{m_1 + m_2.e^{-\frac{\Delta F^+}{T}}}{m_1 + m_2} \tag{3.1}$$

Em que:

X é a taxa de aceitação;

$F(j)$ é o valor da função objetivo na alternativa j;

m_1 é o número de transições propostas de i a j, para as quais $F(j) \leq F(i)$;

m_2 é o número de transições propostas, para as quais $F(j) > F(i)$;

ΔF^+ é o incremento médio no custo das m_2 transições , calculado como:

$$\Delta F^+ = \frac{\sum_{i=1}^{m_2} \Delta F_i}{m_2} \tag{3.2}$$

De (3.1) obtém-se a expressão (3.3), que fornece a temperatura inicial:

$$T_0 = \frac{\Delta F^+}{ln(\frac{m_2}{m_2.X - m_1(1-X)})} \tag{3.3}$$

Algoritmo Para o Cálculo da Temperatura Inicial (T_0):

1. Inicializar $T_0 = 0$;
2. Executar a cadeia m_0;
3. Tentar nova alternativa.

   Se $|F(i) - F(j)| \leq 0$, então $m_1 = m_1 + 1$ e aplique a fórmula:

$$T_0 = \frac{\Delta F^+}{ln(\frac{m_2}{m_2.X - m_1(1-X)})}$$

   Caso contrário, $m_2 = m_2 + 1$ e aplique a fórmula para o cálculo de T_0.

4. Se $m_1 + m_2 = m_0$, terminou a cadeia e o valor de T_0 (calculado na última transição) é assumido como a temperatura inicial do processo. Caso contrário, volte ao passo 3.

Além da anterior, na literatura especializada emprega-se outras formas para o cálculo da temperatura inicial, tal como a proposta em [67] que sugere usar a seguinte relação;

$$T_0 = \frac{\mu}{-ln(\Phi)} F(x^\circ)$$

Em que Φ é uma probabilidade (%/100) de que uma solução seja aceita com um determinado μ (%/100), pior que a solução inicial. Assim, por exemplo, se é aceito um $\Phi = 13\%$, e seja uma solução que tenha um $\mu = 1\%$ pior que a solução inicial, então temos que:

$$T_0 = \frac{0,01}{-ln(0,13)} F(x^\circ) = 0,0049 F(x^\circ)$$

Taxa de resfriamento

O valor da cadeia N_k deveria ser suficientemente grande para que o sistema possa alcançar seu estado de equilíbrio para essa temperatura T_k, isto significa um grande número de iterações; como isto não é possível, ao calcular os parâmetros tem que haver um equilíbrio entre a taxa de resfriamento β e o comprimento da cadeia N_k.

Na literatura se recomendam taxas de resfriamento no intervalo [0,8 ; 0,99], que correspondem a um resfriamento lento. Com o fim de que se permita uma exploração mais intensa nas temperaturas baixas, poderá considerar-se um N_k crescente quando decrescer T_k. Esta regra não é geral e cada aplicação poderia ter um programa de resfriamento mais específico. Diferentes autores sugerem o uso de "aquecimento" quando a busca segundo

o resfriamento atual estiver infrutífera, assim o processo poderá sair de ótimos locais e visitar novas regiões.

Além do critério anterior para determinar a taxa de diminuição da temperatura, existem outros um pouco mais elaborados que avaliam o desempenho do processo no nível de temperatura T_k para determinar o valor de T_{k+1}, entre os quais:

$$T_{k+1} = \frac{T_k}{[1 + \frac{ln(1+\delta)T_k}{3\sigma(T_k)}]} \tag{3.4}$$

em que $\sigma(T_k)$ é o desvio padrão da função de custo das configurações geradas na temperatura T_k e $\delta \in [0,01; 0,20]$, e

$$T_{k+1} = \frac{T_k}{exp(\frac{\lambda T_k}{\sigma(T_k)})} \tag{3.5}$$

em que $\lambda \leq 1,0$.

Temperatura final

A temperatura correspondente ao "sistema frio" deveria ser aquela para a qual $T_f = 0$. Não obstante, antes de chegar a este valor a probabilidade $e^{\{\frac{-\delta}{T}\}}$ de que se aceite uma solução pior é quase nula, portanto $T_f > 0$.

Os critérios de parada estão baseados no argumento de que a execução do processo deve ser interrompida se a melhoria esperada na função objetivo, no caso de se continuar executando o algoritmo, é pequena. Existem várias maneiras simples para definir o critério de parada, assim:

- Fixar um número determinado de valores de T_k para os quais será processado o algoritmo, e esses valores podem variar de 6 a 50 níveis.
- [illegible]ar a busca quando, para um nível de temperatura T_k, não se obtenha uma solução [illegible]ue melhore a melhor solução encontrada até este instante.
- Quando no último nível de temperatura não se tenha encontrado um número mínimo de aceitações.

Além dos anteriores existem critérios de parada mais elaborados como o desenvolvido por [20] que sugere o seguinte critério de parada:

- Depois de realizar todas as tentativas para um determinado nível de temperatura T_k, compare a diferença entre os valores máximo e mínimo das funções objetivo entre as transições aceitas neste nível de temperatura; se as diferenças forem muito próximas já não existe necessidade de usar **SA**, portanto $T_{k+1} = 0$. O processo de otimização termina com uma busca local usando um algoritmo de melhoramento iterativo simples e rápido.

  Assim, sejam:

  ΔC_{max} a máxima variação no objetivo de uma transição aceita.

  C_{max} o valor máximo do objetivo entre as transições aceitas.

  C_{min} o valor mínimo do objetivo entre as transições aceitas.

  Então, o critério de parada do algoritmo de **SA** é satisfeito quando, num determinado nível de temperatura T_k, é verdadeira a seguinte relação:

$$\Delta C_{max} = C_{max} - C_{min}$$

Número de tentativas N_k

O valor de N_k, o número de tentativas de novas transições, e a regra de mudança do valor da temperatura T_k para T_{k+1} são analisados usando o conceito de quase-equilíbrio e a especificação desses dois critérios estão intimamente relacionados. O valor de N_k deve permitir que o processo possa atingir o estado de quase-equilíbrio para o nível de temperatura T_k e a diminuição de T_k para T_{k+1} não deve ser grande a fim de permitir restaurar o novo estado de quase-equilíbrio para o novo valor de temperatura. Assim, o quase-equilíbrio do processo pode ser garantido com elevados valores de N_k e grandes diminuições em T_k ou com pequenos valores de N_k e também pequenas diminuições em T_k.

Uma maneira simples de selecionar o comprimento da cadeia é fazê-lo de acordo com o tamanho do problema. Formas mais elaboradas para escolher o tamanho são baseadas no conceito de que para cada nível de temperatura T_k ao menos um número mínimo de tentativas sejam aceitas; neste procedimento se tem o problema que quando T_k tende a 0 a probabilidade de aceitação decresce, porque $N_k \longrightarrow \infty$.

Entretanto, a maioria de algoritmos escolhem o valor de N_k em função do tamanho do problema ou do número de variáveis e esse valor pode ser mantido constante ou pode variar com a diminuição da temperatura. Assim por exemplo, dois critérios empregados para determinar N_k, ambos em função do número de variáveis N_o (número de linhas ou caminhos candidatos), são os seguintes :

$$N_{k+1} = \eta N_o$$

em que: N_k *fixo*: $\eta \geq 1,0$; e

$$N_{k+1} = \rho N_k$$

em que: N_k *variável*: $\rho \geq 1,0$

3.2.5 Convergência

Teoricamente tem-se demonstrado que o algoritmo "simulated annealing" é convergente no sentido que se poderá ter probabilidade tão perto de 1 como se queira; o algoritmo encontra a solução ótima do problema independentemente da solução inicial de que inicie. Um aspecto distinto é o tempo requerido para obter a convergência (não se pode esquecer que a exploração exaustiva também garante a solução ótima).

3.3 *SA* SEQÜENCIAL APLICADO AO PROBLEMA DE PLANEJAMENTO

SA é uma técnica de otimização combinatorial que foi desenvolvida analisando o processo de "annealing" em sólidos. "Annealing" é um dos processos de construção de cristais perfeitos. No processo de "annealing" o material sólido é esquentado a temperaturas elevadas onde esse material assume uma estrutura atômica altamente desordenada. Depois o material é resfriado lentamente, mantendo o quase-equilíbrio térmico, até atingir o estado de mínima energia onde o material assume a estrutura de um cristal perfeito. Este processo de "annealing", desenvolvido pelos físicos para a construção de cristais perfeitos, foi usado para desenvolver um algoritmo com capacidade de resolver problemas do campo da pesquisa operacional. A grande vantagem destes métodos combinatoriais é que eles tem capacidade de resolver problemas complexos e de grande porte, sem as restrições existentes nos métodos de otimização clássicos como exigências de convexidade, diferenciabilidade, etc.

SA foi usado com sucesso em muitas áreas da pesquisa operacional e em algumas aplicações em engenharia elétrica. Tem a característica de que apresenta soluções ótimas ou quase-ótimas, mas a um esforço computacional elevado. O *programa de resfriamento* controla esta qualidade da solução e o esforço computacional. Teoricamente, **SA** converge à solução ótima de maneira assintótica com probabilidade igual a 1.

Um problema de otimização combinatorial que pode ser resolvido usando **SA** pode ser formulado como um par (G, v) onde G representa um conjunto finito (embora muito grande) de configurações e v é o custo associado a cada configuração. Assim, resolver um

problema combinatorial deste tipo consiste em encontrar a configuração de custo mínimo entre todas as configurações possíveis. Assim, **SA** parte de uma configuração inicial e depois gera um conjunto de configurações candidatas controladas pelo programa de resfriamento. Estas configurações candidatas são aceitas se apresentam um custo menor que a configuração corrente, mas elas também podem ser aceitas se o custo for maior que a configuração corrente de acordo com um critério probabilístico. Esta última característica, aceitar configurações com custo mais elevado, permite ao **SA** sair de eventuais ótimos locais. Este processo é controlado pelo programa de resfriamento.

A aplicação de um algoritmo de **SA** para resolver um problema de otimização combinatorial precisa da especificação de três aspectos significativamente distintos e independentes:

1. uma adequada representação do problema,
2. um mecanismo de transição, e
3. um programa de resfriamento ("cooling schedule").

Para cada tipo de problema, deve-se definir claramente cada um destes três aspectos e geralmente existem várias maneiras de se definir ou caracterizar cada um destes itens, o que leva a algoritmos de **SA** substancialmente diferentes. A seguir são discutidos, separadamente, cada um destes aspectos do algoritmo de **SA** para o problema de planejamento de sistemas de transmissão.

3.3.1 A Representação do Problema

O problema de planejamento de sistemas de transmissão usando o modelo DC assume a seguinte forma:

$$
\begin{aligned}
\min \quad v = \quad & \sum_{ij} c_{ij} n_{ij} + \sum_{i} \alpha_i r_i \qquad (3.6)\\
s.a. \quad & B(x^k + \gamma^o)\theta + g + r = d \\
& (x_{ij}^k + \gamma_{ij}^o)\,|\theta_i - \theta_j| \leq (x_{ij}^k + \gamma_{ij}^o)\overline{\phi}_{ij} \\
& 0 \leq g \leq \overline{g} \\
& 0 \leq r \leq d \\
& n_{ij} \ \ inteiro
\end{aligned}
$$

A representação do problema deve caracterizar corretamente o espaço de configurações possíveis e uma forma adequada de determinar a função de custo. Na formulação (3.6),

ambos aspectos estão claramente definidos: a factibilidade de uma configuração é determinada pelas restrições e a função de custo está constituída por dois termos diferentes. A primeira parte corresponde aos custos das linhas adicionadas e a segunda parte corresponde a uma penalidade devido ao corte de carga. Na verdade, a formulação (3.6) já é uma modificação do problema de planejamento que na sua formulação original somente leva em conta os custos das linhas adicionadas na função objetivo. Entretanto, nessa formulação original seria muito difícil e pouco eficiente implementar um algoritmo de **SA** pois essa formulação apresenta três grandes limitações:

1. precisa de uma configuração inicial factível para iniciar o processo, o que não é fácil encontrar em sistemas de grande porte,
2. a partir da configuração corrente, deve-se passar a outra configuração factível, isto é, sem corte de carga, o que também é difícil de encontrar,
3. esse mecanismo de transição, passar de uma configuração sem corte de carga a outra configuração sem corte de carga, limitaria a capacidade do algoritmo de **SA** de sair de um ótimo local. Assim, representar a função objetivo com penalidade devido aos cortes de carga ou sem ela, representam duas alternativas diferentes de implementar algoritmos de **SA** para o problema de planejamento.

No presente trabalho, usa-se a formulação (3.6) com a função objetivo com o termo correspondente a penalidade devido ao corte de carga. O motivo da escolha é que esta formulação não apresenta os problemas anteriormente mencionados, a mesma apresenta uma grande flexibilidade para passar de uma tentativa de configuração para outra. Nesta formulação, qualquer configuração candidata ou proposta de adição de linhas, n_{ij}^k, é "factível" para a formulação (3.6), isto é, as configurações que apresentam e não apresentam corte de carga. Assim, o espaço das configurações factíveis foi acrescido para facilitar o processo de transição. Por outro lado, o processo deve priorizar as configurações e regiões factíveis do problema original, isto é, das regiões sem corte de carga. Este objetivo é atingido de duas maneiras complementares:

1. implicitamente, pois as configurações com corte de carga são fortemente penalizadas produzindo uma função de custo elevada o que levará à rejeição de configurações deste tipo; quanto mais elevado o corte de carga de uma configuração, mais elevada será a chance de ser rejeitada. Já configurações com pequeno corte de carga serão mais facilmente aceitas, e
2. usando limitantes inferiores e superiores para a parte correspondente aos custos das linhas de uma configuração, informação que sempre está disponível para cada sistema e ainda pode ser melhorada usando informação do processo de convergência do próprio problema. A Figura 3.2 mostra de maneira esquemática os argumentos apresentados e que justificam a vantagem de implementar a formulação (3.6) para o problema de planejamento de sistemas de transmissão.

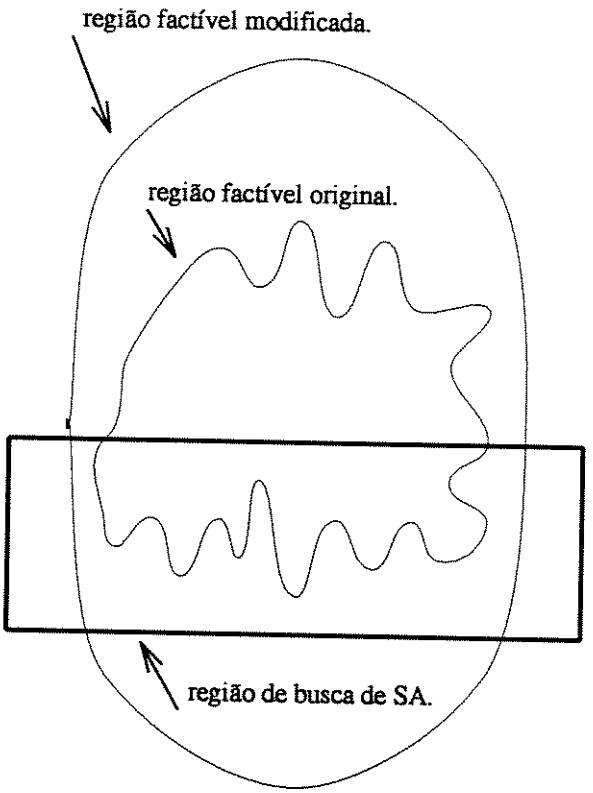


Figura 3.2: Espaço de Busca em *SA*

3.3.2 Mecanismo de Transição

O mecanismo de transição é o processo que consiste em transformar a configuração corrente em uma configuração subseqüente. Este processo é realizado em três etapas:

1. geração de uma configuração candidata a partir da configuração corrente, o que é implementado definindo uma *estrutura de vizinhança* para o problema,
2. determinação da variação da função de custo para a configuração candidata,
3. decidir se a configuração candidata é aceita, transformando-a em configuração corrente.

A seguir são apresentados detalhes destes aspectos para o problema de planejamento.

Estrutura de vizinhança

A estrutura de vizinhança é o mecanismo que permite gerar uma configuração candidata n_{ij}^{k+1} a partir da configuração corrente, n_{ij}^{k}.

Um mecanismo de geração simples, que chamar-se-á *mecanismo básico de transição* , pode ser implementado baseado na idéia apresentada em [24]. Assim, com as idéias antes mencionadas, podem ser caracterizados três tipos simples de obtenção de uma configuração candidata a partir da configuração corrente:

1. adicionar um circuito em um caminho candidato,
2. trocar circuitos, adicionando um circuito em um caminho candidato e retirando outro circuito em outro caminho candidato, e
3. retirar um circuito em um caminho candidato.

Estes mecanismos simples para gerar as novas alternativas na seqüência *adição-troca-retirada* foram implementados da seguinte maneira:

1. Escolhe-se aleatóriamente a linha i como sendo candidata a *adição.* A configuração candidata é obtida adicionando-se uma linha na posição i da configuração G_k. Se esta configuração é aceita pelos critérios de **SA** então foi obtida a configuração G_{k+1} e deve-se repetir o ítem 1. Caso contrário, tenta-se a troca de linhas.
2. Escolhe-se aleatoriamente a linha j entre aquelas linhas que já estão adicionadas na configuração atual. A *troca* consiste em adicionar uma linha na posição i e retirar uma linha da posição j na configuração G_k, para obter a configuração candidata. Se esta configuração é aceita pelos critérios de **SA** então foi obtida a configuração G_{k+1} e deve-se voltar ao ítem 1. Caso contrário, tenta-se a retirada de linha no ítem seguinte.
3. A linha j escolhida no ítem anterior é considerada candidata a *retirada.* Assim, a configuração candidata é obtida retirando-se uma linha na posição j da configuração G_k. Se esta configuração é aceita pelos critérios de **SA** então foi obtida a configuração G_{k+1} e deve-se voltar ao ítem 1. Caso contrário, fracassou as três tentativas *(adição-traoca-retirada)* e volta-se ao ítem 1.

No esquema anterior, cada vez que o processo volta ao ítem 1, o contador de iterações para o nível de temperatura T_k deve ser incrementado em 1, até atingir o limite de N_k; depois o processo deve passar à outro nível de temperatura. Entretanto, pode-se verificar

que para cada iteração controlada por N_k são realizadas até três tentativas. Assim, no esquema anterior, é evidente que para determinado nível de temperatura a quantidade de tentativas para a obtenção de novas configurações está dentro do intervalo $[N_k, 3 * N_{k+1}]$.

A seqüência de operações é *adição-troca-retirada*, isto é, primeiro tenta-se a adição de uma linha, depois a troca e finalmente a retirada de uma linha. No problema de planejamento o processo se inicia geralmente com um corte de carga muito grande o que, intuitivamente, levaria à conclusão de priorizar a adição de linhas; entretanto, uma vez obtida uma configuração sem corte de carga é evidente que a prioridade seria a de retirar linhas. Baseados nos argumentos anteriores, a seqüência de *adição-troca-retirada* deve ser invertida em determinados momentos do processo a fim de melhorar o desempenho do algoritmo.

O mecanismo básico de transição deve ser implementado de maneira eficiente com seqüência trocada: *adição-troca-retirada* quando a configuração corrente apresenta corte de carga maior que uma tolerância especificada w_o, onde deve ser priorizada a adição de linhas; ou *retirada-troca-adição* quando a configuração corrente apresenta um corte de carga menor que uma tolerância especificada w_o, e portanto deve ser priorizada a retirada de linhas.

A primeira seqüência é usada mais freqüentemente nas fases iniciais do processo onde o corte de carga é elevado e a segunda seqüência nas fases finais onde praticamente todas as configurações com corte de carga são rejeitadas. Este mecanismo básico de transição apresenta a vantagem de sua simplicidade e a possibilidade de implementar um eficiente algoritmo de PL para calcular rapidamente o corte de carga da configuração candidata a partir da solução da configuração corrente, como será mostrado mais adiante; entretanto, apresenta limitações para sistemas de grande porte com muitas barras não conexas.

No diagrama de fluxo apresentado na Figura 3.3, pode-se observar a maneira como os mecanismos de transição, *adição-troca-retirada*, foram implementados na metodologia **SA**.

Um mecanismo de geração de configurações mais elaborado deve levar em conta a topologia da rede elétrica, assim o processo de adição e retirada de linhas em novos caminhos deve garantir a conectividade do sistema, evitando que sejam adicionadas linhas ineficientes ou retiradas de linhas em novos caminhos que deixem outras linhas sem possibilidade de transportar energia elétrica. Em sistema grandes e complexos, como o Norte-Nordeste brasileiro, às vezes é necessária a adição de 3 a 6 linhas sucessivas para garantir a conectividade do sistema. Para garantir sempre a conectividade do sistema elétrico foi elaborado um *mecanismo de geração modificado* que mantém sempre uma conectividade adequada no sistema eliminando adições ou retiradas ineficientes. Este mecanismo de geração modificado realiza, às vezes, a adição, troca ou retirada de um conjunto de linhas, de caminhos novos completos, o que leva à caracterização de uma estrutura de vizinhança mais sofisticada.

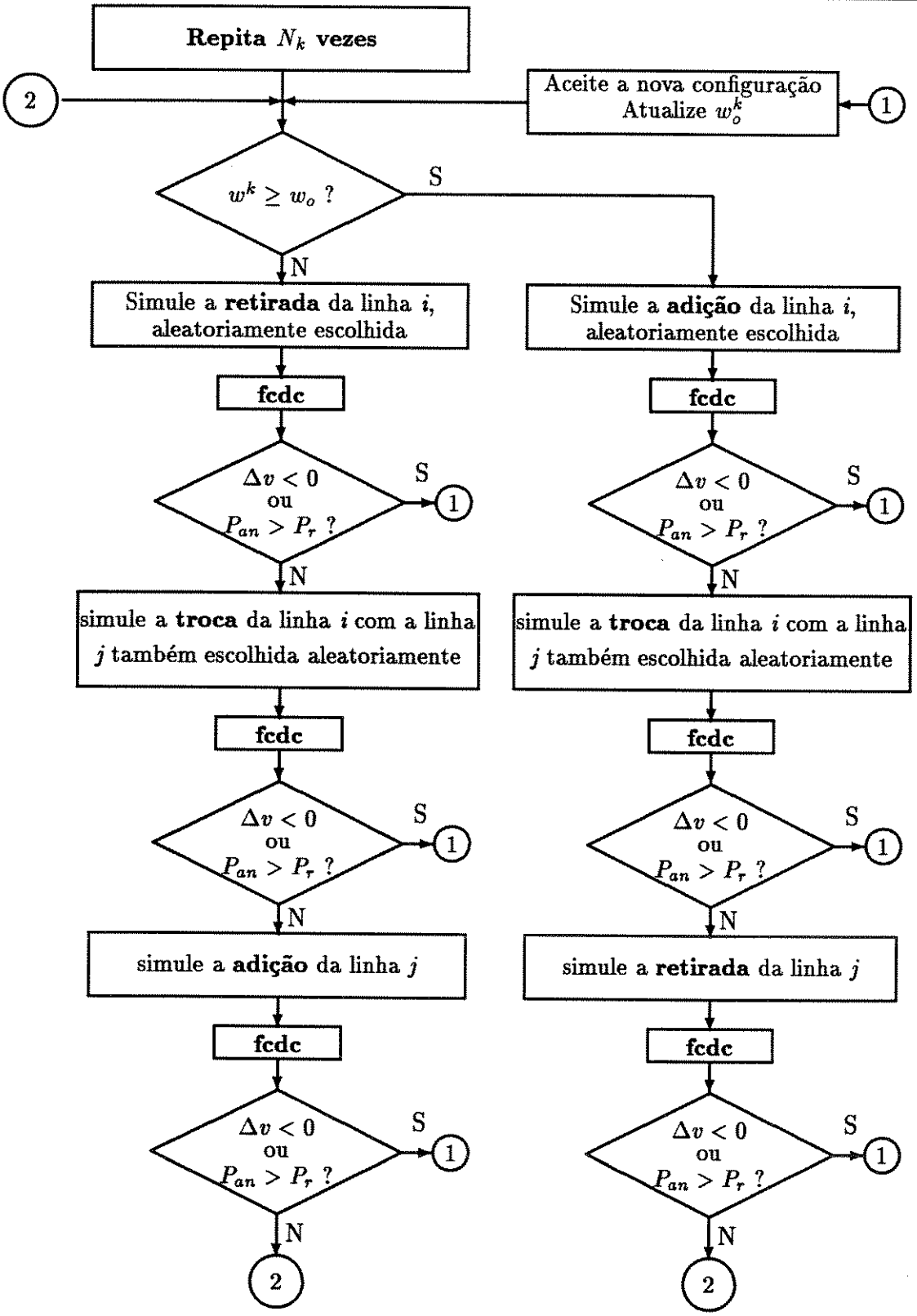


Figura 3.3: Diagrama de Fluxo do Mecanismo de Transição.

Mecanismo para geração de caminhos

Este mecanismo garante a conectividade do sistema identificando as linhas que formam um caminho completo, tratando-se de adição ou retirada. No primeiro caso, adição de linha, verifica-se se a mesma está fazendo trabalho útil. Em caso afirmativo, adiciona-se essa linha à configuração atual; caso contrário, identificam-se as linhas vizinhas, tal que ao serem conectadas efetuem um trabalho útil na rede, sem que fique alguma linha pendurada. No segundo caso, retirada de linha, verifica-se se ao ser retirada, não está deixando linhas vizinhas penduradas, ou seja, linhas que não estariam fazendo trabalho útil na rede. Neste caso, identifica-se quais linhas teriam que sair formando um caminho de saída; caso contrário, simplesmente retira-se a linha identificada.

A seguir, explica-se a maneira como trabalha o procedimento que gera os caminhos para evitar linhas isoladas, durante o processo de *adição-troca-retirada* ou *retirada-troca-adição.*

A Figura 3.4 será usada para justificar a elaboração de um mecanismo de transição mais eficiente. Nesta figura, as barras 16, 26, 27, 32, 40 e 43 fazem parte do sistema elétrico para o ano base de planejamento (a figura não mostra o sistema elétrico do ano base, ela aparece representada apenas pelos traços espessos), isto é, estas barras estão conexas com as outras barras do sistema. Por outro lado, as barras 28 e 31 são barras de geração que não estão conectados ao sistema base e, finalmente as barras 29, 30 e 41 são barras de ligação também inicialmente isoladas do caso base. Portanto, existem barras conexas (ao sistema base) e barras isoladas.

Nas condições especificadas para a Figura 3.4, observam-se os seguintes acontecimentos no provável processo de adição-troca-retirada de linhas:

- Em um determinado estágio do processo, imagine que é escolhida a adição (entrada) da linha 63. Se não existiam outras linhas adicionadas, a adição da linha 63 não contribui em nada para aliviar o corte de carga e ela ficaria literalmente "pendurada". Assim, uma chamada do PL mostraria um corte de carga exatamente igual à configuração anterior.

- Do item anterior, fica evidente que no caso de linhas que estão unidas às barras isoladas, a adição de uma linha deve levar à adição de outras que permitam completar um caminho até ligar estas linhas ao sistema elétrico pois, caso contrário, as adições não fazem sentido. Assim, por exemplo, se a linha 63 foi escolhida para entrar, então, de forma aleatória, deve ser escolhido o caminho 63-75-65 ou o caminho 63-75-64 para ser adicionado ao sistema. Portanto, nas condições mostradas na Figura 3.4, a escolha da linha 63 deve levar à adição das linhas 63-75-65, por exemplo. Fica evidente que, neste caso, a definição da estrutura de vizinhança é diferente do mecanismo simples.

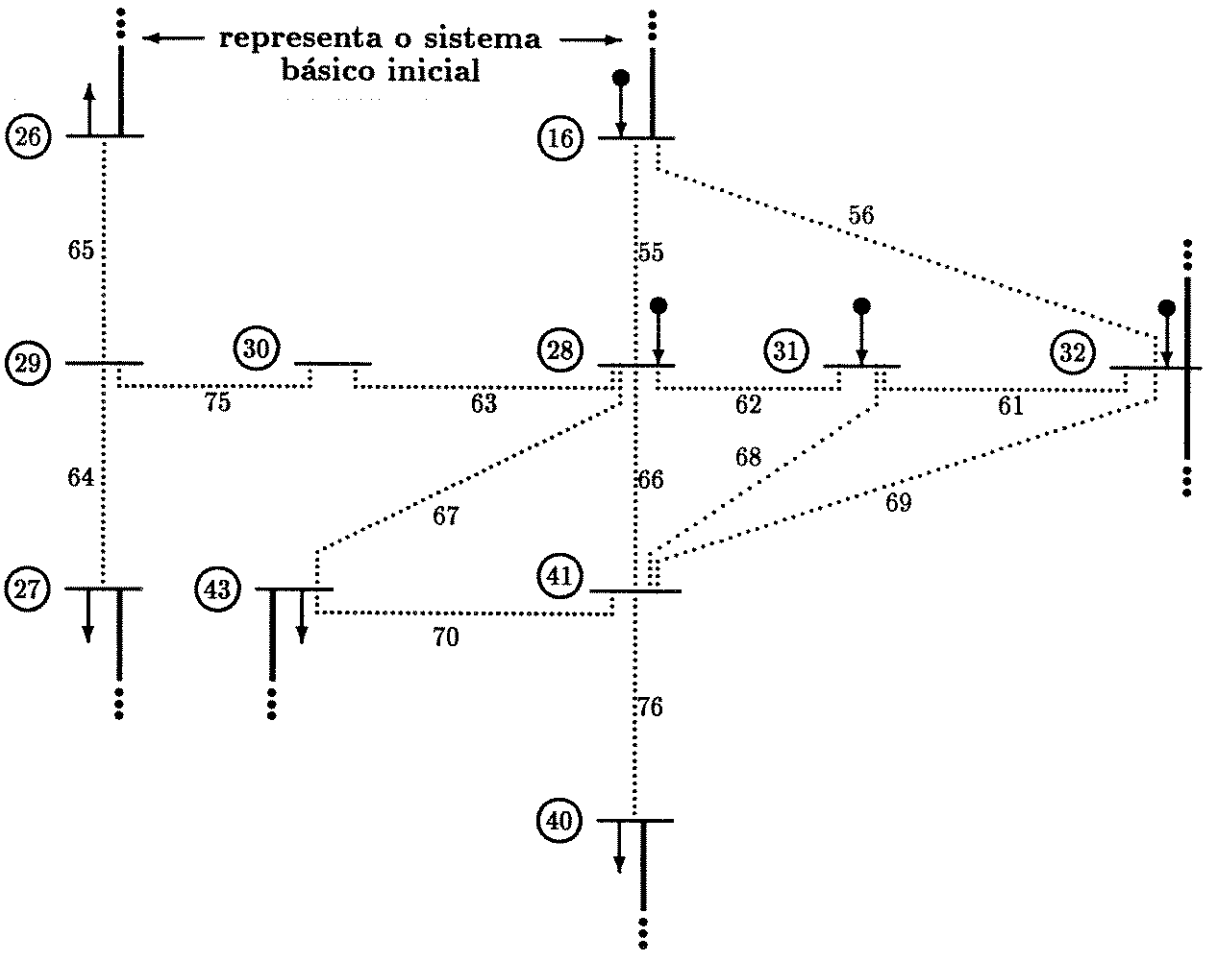


Figura 3.4: Identificação de caminhos

- Agora supor, novamente na Figura 3.4, que já existe um caminho adicionado; por exemplo, existe o caminho 63-75-65. Se nestas condições é escolhida de novo, aleatoriamente, a linha 63 para entrar; então, somente essa linha seria adicionada, pois esse caminho já está temporariamente conectado ao sistema. Na verdade, tentando generalizar, se já existem linhas no caminho 63-75-65, então a escolha aleatória para adição de uma das linhas 63, 75, 65 e 64 levaria à entrada somente daquela linha.

- A saída de linhas deve obedecer ao mesmo critério básico. Por exemplo, imagine que já foram adicionadas linhas no caminho 63-75-65. Nestas condições a linha 63 é escolhida para sair. Uma estratégia eficiente tomaria a seguinte decisão: se existe mais de uma linha em 63 então deve-se retirar somente uma linha em 63; caso contrário, isto é, se existe somente uma linha em 63 então deve-se retirar a linha em 63 e todas as linhas em 75 e 65.

- Finalmente, considere-se o seguinte exemplo. Imagine-se que já foram adicionadas linhas no caminho 63-75-65-64. Nestas condições, se a linha 65 é escolhida para retirada então deve-se retirar somente uma linha em 65 (caso parecido aconteceria com a linha 64). Por outro lado, se a linha 63 é escolhida para sair, a estratégia seria a seguinte: se existe mais de uma linha em 63 deve-se retirar somente uma linha em 63; caso contrário, se existe somente uma linha em 63, deve-se retirar essa linha em 63 e todas as linhas em 75.

- A escolha de outros caminhos para análise, na Figura 3.4, leva também a outras conclusões interessantes.

Em todos os exemplos mostrados em relação a Figura 3.4, a estratégia básica consiste em *evitar* que, em algum momento do processo de adição-troca-retirada, existam linhas adicionadas ao sistema elétrico mas que não estejam ligadas (conexas) ao sistema básico inicial. Em outras palavras, se em algum momento existe adição de linhas que ligam as barras isoladas, essa adição ou adições devem permitir a ligação das linhas adicionadas ao sistema elétrico. Sem estas considerações o mecanismo de transições para um sistema real ficaria muito complicado e seria muito difícil sair das soluções ótimas locais.

Baseadas nas considerações anteriores, pode-se definir uma *nova estratégia de vizinhança* mais eficiente, conforme a seguir.

A partir de uma configuração G_k pode-se obter uma configuração candidata através do processo de *adição-troca-retirada* de linhas com a condição adicional de que esta configuração candidata não deve ter nenhuma linha adicionada ao sistema elétrico e que esteja desligada da configuração básica inicial. Em outras palavras, sob nenhuma circunstância devem existir linhas isoladas ou não conexas ao sistema elétrico inicial. Esta configuração candidata pode, depois, ser aceita como sendo a nova configuração G_{k+1} de acordo com os critérios de aceitação de *"Simulated Annealing"*.

A mudança de estratégia em relação ao mecanismo de transições é relativamente simples do ponto de vista de sua formulação e além disso permite uma evolução mais eficiente do processo de convergência; mas, do ponto de vista de programação é relativamente complicada.

Avaliação da função objetivo

Dada a configuração corrente, somente a primeira parcela da função objetivo na formulação (3.6) pode ser determinada facilmente. A determinação da segunda parcela implica na resolução do seguinte problema de programação linear (PL):

$$\min \quad w = \sum_i \alpha_i r_i \tag{3.7}$$

$$
\begin{aligned}
s.a. \quad & B(x^k + \gamma^\circ)\theta + g + r = d \\
& (x_{ij}^k + \gamma_{ij}^\circ)\ |\theta_i - \theta_j| \leq (x_{ij}^k + \gamma_{ij}^\circ)\overline{\phi}_{ij} \\
& 0 \leq g \leq \overline{g} \\
& 0 \leq r \leq d
\end{aligned}
$$

Nesta formulação são alocados, de forma fictícia, geradores artificiais r_i que fornecem a potência necessária nessa barra quando as linhas que chegam à mesma não podem transportar a potência necessária, assim a somatória de todos os r_i, $w = \sum_i \alpha_i r_i$ é conhecido como corte de carga, isto é, a quantidade de carga que o sistema não pode fornecer.

Assim, conhecida a configuração corrente n_{ij}^k, a formulação (3.6) que é um problema PNLIM se transforma em um PL mostrado em (3.7). Portanto, a avaliação da função objetivo para cada configuração candidata implica na resolução de (3.7) usando um algoritmo de PL.

Esta característica do problema de planejamento, avaliar cada configuração candidata através de um PL, eleva consideravelmente o esforço computacional de uma iteração do algoritmo de **SA** para o problema de planejamento de sistemas de transmissão, quando comparado com outros problemas típicos de pesquisa operacional como o problema do caixeiro viajante onde a variação da função objetivo, da configuração corrente para a configuração candidata, é realizada simplesmente através de 4 somas algébricas. Esta característica do problema de planejamento, de precisar de um PL para analisar cada configuração candidata, também aparece em outros problemas de engenharia elétrica [66] e não depende da forma de representação da função de custo no problema de planejamento.

Critério de aceitação

O critério de aceitação de uma configuração candidata é determinado pelo critério de aceitação de Metrópolis $P(a)$, mostrado a seguir:

$$
Pa = \begin{cases} 1 & \text{se } \Delta v < 0 \\ exp(-\frac{\Delta v}{T}) & \text{se } \Delta v \geq 0 \end{cases} \tag{3.8}
$$

em que T é o valor do parâmetro temperatura e Δv é a diferença da função de custo entre a configuração candidata e a configuração corrente.

3.3.3 O Programa de Resfriamento

O programa de resfriamento é a estratégia geral de controle do processo **SA** desde o início do mesmo até atingir a convergência e está caracterizado pela escolha ou determinação dos quatro seguintes parâmetros:

- Valor inicial do parâmetro de controle T_o, chamado de temperatura inicial.
- Número de tentativas de transição N_k a cada nível de temperatura T_k.
- Taxa de diminuição de temperatura, $T_{k+1} = g(T_k)T_k$, em que $g(T_k)$ é uma função de T_k.
- Temperatura final T_f ou critério de parada.

Uma vez definidos esses parâmetros fica caracterizado um programa de resfriamento. Assim, pode-se implementar muitos programas de resfriamento dependendo do tipo de escolha ou a forma usada para obter cada um desses quatro parâmetros. A eficiência de um algoritmo de **SA**, isto é, a qualidade da configuração final, assim como o esforço computacional para atingir a convergência, depende da escolha adequada dos parâmetros de controle no programa de resfriamento. A seguir são apresentados e discutidos os critérios de escolha ou determinação desses parâmetros.

Temperatura inicial, T_o

Na descrição teórica do algoritmo **SA** foram apresentadas duas fórmulas para calcular a temperatura inicial T_o; os valores de temperatura calculados com os dois procedimentos não apresentam diferenças substanciais quando são analisadas para determinado comprimento da cadeia de Markov; apesar disso, se observa que a primeira contém uma maior solidez matemática permitindo cálculos da temperatura para diferentes comprimentos e maior coerência nos resultados quando foram testadas com redes elétricas de diferentes tamanhos. A seguir é apresentada a fórmula selecionada [19]:

$$T_o = \frac{\overline{\Delta V^+}}{ln(\frac{m_2}{m_2 X_o - m_1(1-X_o)})} \tag{3.9}$$

em que X_o é escolhido e os outros parâmetros são determinados através de um teste experimental de m_o tentativas; assim, m_1 e m_2 correspondem ao número de transições com funções de custo decrescentes e crescentes respectivamente e $\overline{\Delta V^+}$ é o valor médio das diferenças de custo Δv de todas as m_2 transições com aumento na função de custo. Logicamente, $m_o = m_1 + m_2$. No presente trabalho foi escolhido um $X_o = 0,85$ o que significa que na temperatura inicial são aceitas em torno de 85% das configurações candidatas.

Escolha de N_k

A escolha de N_k, número de tentativas a cada nível de temperatura, deve garantir o quase-equilíbrio em cada nível, assim a escolha deste parâmetro está ligada a determinação da taxa de diminuição de temperatura. A maioria dos algoritmos escolhem o valor de

N_k em função do tamanho do problema ou do número de variáveis e esse valor pode ser mantido constante ou pode variar com a diminuição da temperatura. No presente trabalho foi implementada uma maneira crescente para o cálculo do número de tentativas em função do número de variáveis N_o, que é o número de linhas ou caminhos candidatos a adição:

- *N_k variável*: $\rho \geq 1,0$

$$N_{k+1} = \rho N_k \tag{3.10}$$

Determinação de T_{k+1}

Existem muitos critérios para determinar a taxa de diminuição da temperatura, todos eles baseados no critério básico de manter o quase-equilíbrio do processo. Assim existem critérios simples, como manter uma taxa de diminuição fixa e outros um pouco mais elaborados, que avaliam o desempenho do processo no nível de temperatura T_k para determinar o valor de T_{k+1}. Assim, neste último caso a taxa de diminuição é variável e depende da evolução do processo de otimização. No presente trabalho foram analisadas três formas diferentes de determinar o valor de T_{k+1}:

- *Taxa constante*: $\beta \in [0,50; 0,99]$

$$T_{k+1} = \beta T_k \tag{3.11}$$

- *Taxa variável*: $\delta \in [0,01; 0,20]$

$$T_{k+1} = \frac{T_k}{[1 + \frac{ln(1+\delta)T_k}{3\sigma(T_k)}]} \tag{3.12}$$

em que $\sigma(T_k)$ é o desvio padrão da função de custo das configurações geradas na temperatura T_k.

- *Taxa variável*: $\lambda \leq 1,0$

$$T_{k+1} = \frac{T_k}{exp(\frac{\lambda T_k}{\sigma(T_k)})} \tag{3.13}$$

Observação: Usando as relações (3.12) e (3.13), obtém-se desempenhos similares a (3.11) em relação a qualidade das configurações encontradas, mas o número de iterações necessário para convergência cresce de maneira proibitiva. A influência dos parâmetros δ e λ assim como de $\sigma(T_k)$ é pequena e produzem taxas de diminuição de temperaturas variáveis, mas elevadas, variando entre 0,94 e 0,99. Portanto, resultados preliminares indicam que é mais prático usar a relação (3.11) com uma adequada calibração dos parâmetros.

Critério de parada: T_f

Existem vários critérios de parada desde os mais simples até os mais elaborados. Os mais simples sugeridos pelos pesquisadores são:

- Fixar no início do processo um número determinado de níveis de temperatura tradicionalmente entre 6 e 50.
- Usar a taxa de melhoria da função objetivo como critério de parada; assim se a função objetivo ou a incumbente não melhora para vários níveis consecutivos de temperatura então o processo atinge a convergência.
- Fixar um número mínimo de aceitações durante o último nível de temperatura; se no último nível não forem aceitas um número maior ao mínimo estabelecido, então o processo atinge a convergência.

Embora existam outros critérios mais elaborados, na presente pesquisa foram usados o segundo e terceiro critério simples descritos acima, baseado em testes preliminares que justificam que os critérios de parada não influenciam muito na qualidade da solução encontrada. Assim, nos testes realizados no presente trabalho, o processo pára se a função objetivo não melhora entre 2 a 4 níveis consecutivos de temperatura, ou se durante o último nível de temperatura não foram aceitos entre 1 a 6 novas propostas. Nos dois critérios de parada a seleção do valor depende dos sistemas testados.

3.3.4 Algoritmo de "Simulated Annealing" Implementado

Foi implementado o seguinte algoritmo que é válido para todos os programas de resfriamento; a Figura 3.5 mostra de maneira geral o algoritmo de **SA**.

1. Determinar a temperatura inicial: T_o.
2. Escolher ou calcular os parâmetros de controle ($\beta, \rho, etc.$) usados nas relações (3.10)-(3.13), assim como também o parâmetro μ, usado para definir o número de iterações na temperatura inicial, obtida através da relação $N_o = \mu.N_t$, onde N_t é o número de variáveis do sistema, isto é, o número de linhas candidatas. Escolha também w_o que define o corte de carga para fazer a seqüência *adição-troca-retirada* de linhas ou vice-versa . Valores típicos para w_o estão na faixa de 0,3 % a 1,0 % da demanda total do sistema e ε_w é o mínimo corte de carga permitido para que a configuração seja considerada factível. Em termos de planejamento de sistemas de transmissão foram usados valores de ε_w da ordem de 0,1 % a 0,3 % da demanda total do sistema.

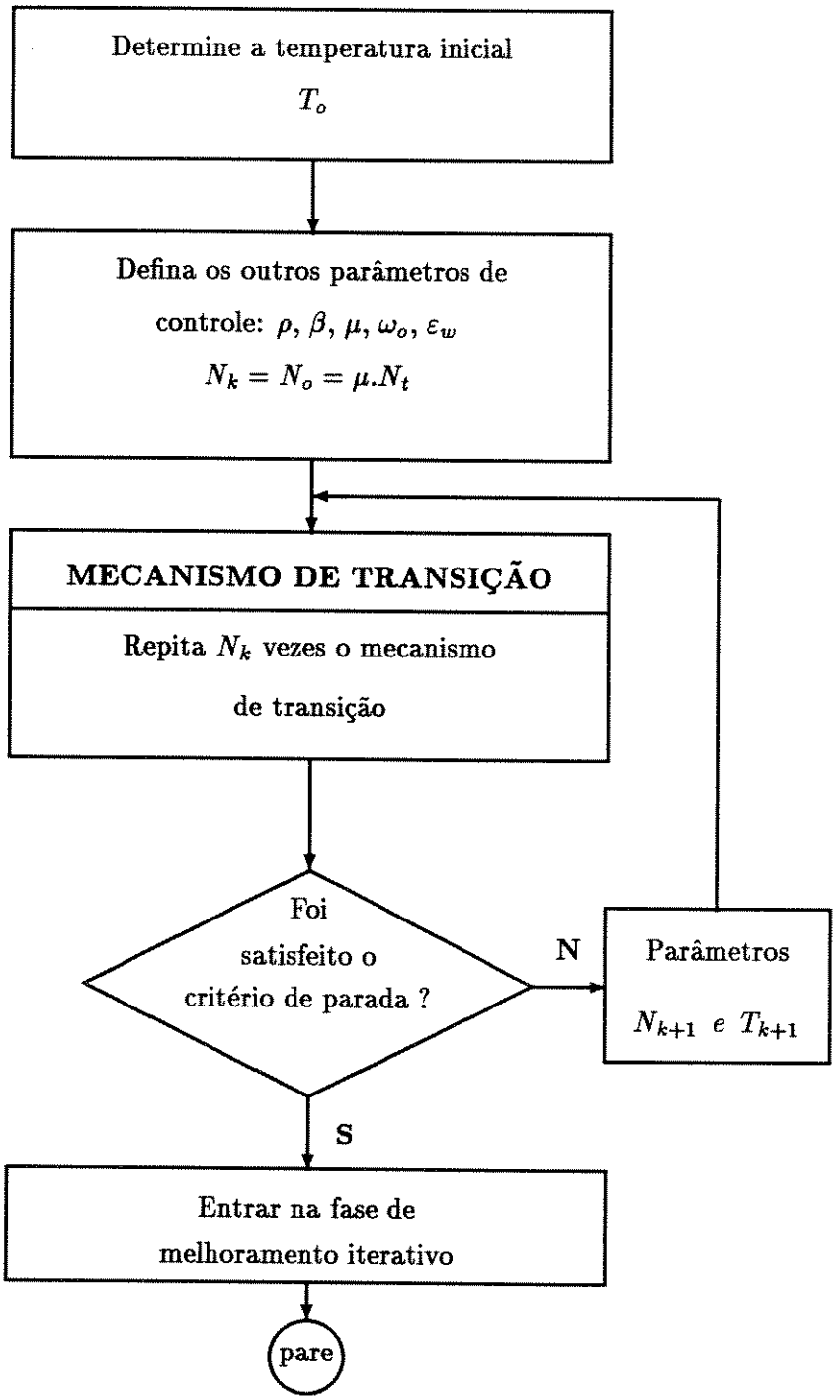


Figura 3.5: Diagrama de Fluxo do Algoritmo de *SA*

3. Incrementar o contador de iterações n_{con}. Se $n_{con} \geq N_k$ ir ao passo 7; caso contrário, ir ao passo 4.

4. Se o corte de carga corrente w_o^k for menor que w_o, ir ao passo 5; caso contrário, ir ao passo 6.

5. a) Obter uma configuração candidata simulando a *retirada* de uma linha i aleatoriamente escolhida; com a configuração candidata, assim obtida, processar um PL e testar sua aceitação usando (3.8). Se for aceita, atualize a configuração corrente e as outras grandezas do processo (w_o^k, função de custo corrente, melhor configuração encontrada (incumbente), etc.) e volte ao passo 3; caso contrário, ir a 5(b).

   b) Obter uma configuração candidata simulando a *troca* da linha i com uma linha j aleatoriamente escolhida; com a configuração candidata, assim obtida, processar um PL e testar sua aceitação usando (3.8). Se for aceita, atualize a configuração corrente e as outras grandezas do processo e volte ao passo 3; caso contrário, ir a 5(c).

   c) Obter uma configuração candidata simulando a *adição* da linha j aleatoriamente escolhida; com a configuração candidata, assim obtida, processar um PL e testar sua aceitação usando (3.8). Se for aceita, atualize a configuração corrente e as outras grandezas do processo. Em qualquer caso, volte ao passo 3.

6. Idem ao passo 5 com a seqüência trocada:

   a) Simular a *adição* de uma linha. Se a configuração for aceita, volte ao passo 3; caso contrário, ir a 6(b).

   b) Simular a *troca* de linhas. Se a configuração for aceita, volte ao passo 3; caso contrário, ir a 6(c).

   c) Simular a *retirada* de uma linha. Em qualquer caso, volte ao passo 3.

7. Testar o critério de parada: se o critério for satisfeito, ir ao passo 9; caso contrário, ir ao passo 8.

8. Atualizar os parâmetros de controle: N_{k+1} e T_{k+1} e voltar ao passo 3.

9. O processo de **SA** foi concluído, entrar na fase de melhoramento iterativo (busca local) e parar todo o processo.

O melhoramento iterativo (busca local) implementado é muito rápido, pois nesta fase somente são testadas aquelas configurações candidatas que produzem redução do investimento, portanto, somente são consideradas para avaliação os seguintes casos:

- **Retirada de linha:** Neste caso a tentativa de retirar uma linha que leva a uma diminuição do investimento e cada avaliação implica processar um PL. Mas como geralmente as linhas com adição são pequenas, esta parte leva a um esforço computacional muito pequeno.

- **Troca de linhas:** Neste caso, deve-se simular a retirada de uma linha que tem adição na configuração atual e a adição de cada uma das linhas candidatas. Esta parte também implica pouco esforço computacional pois somente são testados através do PL aquelas trocas que implicam numa diminuição em investimento, pois as outras combinações, isto é, aquelas que incrementam o investimento, são automaticamente rejeitadas.

Na fase de melhoramento iterativo não é avaliada a possibilidade de adição de linhas pois implicaria em um incremento do investimento. São aceitas todas as configurações que diminuem o investimento e apresentam um corte de carga menor que a tolerância ε_w especificada. Nesta fase, se é aceita alguma configuração, então o processo deve ser reinicializado. Também nesta fase, a função objetivo considerada é somente o custo das linhas, isto é, $v' = \sum_{(i,j)} c_{ij}\eta_{ij}$.

3.4 TESTES EM SISTEMAS PEQUENOS E MÉDIOS E CALIBRAÇÃO DE PARÂMETROS

Baseado no algoritmo anterior e usando as diferentes relações apresentadas para determinar diferentes programas de resfriamento, foram realizados testes exaustivos em dois sistemas de pequeno e médio porte cujas soluções ótimas são conhecidas e que são muito usados pelos especialistas para testar o desempenho de novos algoritmos: sistema de 06 barras de Garver e sistema Sul brasileiro para 1990. Em ambos casos foram obtidas as soluções ótimas conhecidas, mas o esforço computacional varia significativamente entre os diferentes programas de resfriamento e nestes mesmos programas também existem desempenhos diferentes dependendo do valor dos parâmetros de controle escolhidos. As soluções ótimas destes sistemas são mostradas a seguir.

Além do anterior, o algoritmo foi avaliado usando dois tipos de inicialização; na primeira, se parte de uma configuração inicial gerada de maneira aleatória, e na segunda, de uma configuração inicial gerada com a metodologia de Garver modificada.

3.4.1 Sistema de 06 Barras de Garver com Redespacho

Neste caso é permitido o redespacho da geração, isto é, a geração nas barras de geração pode variar de zero até o valor máximo; portanto, os níveis de geração são determinados pelo processo de otimização. Os dados deste sistema de 6 barras e 15 ramos candidatos aparecem no Apêndice A. A topologia inicial é mostrada na Figura 3.6; na mesma e em traços grossos se observa a rede inicial. Este sistema somente é avaliado para o caso em que a configuração inicial é gerada aleatoriamente, já que com o algoritmo inicializador baseado na metodologia de Garver modificada se obtém a solução ótima.

Os parâmetros obtidos foram os seguintes : $\rho = 1,2$, $\beta = 0,7$, $\alpha = 10$, $\mu = 2$, $w_o = 0$ e T_o na faixa de [150 - 400]. Neste processo a convergência é obtida entre 800 a 1.000 iterações (chamadas de PL).

Foram realizadas 4 adições de linhas para $n_{35} = 1$ e $n_{46} = 3$, com um investimento de 110. Esta configuração ótima já era conhecida. Na Figura 3.7 é mostrada a solução ótima para o sistema Garver com redespacho.

3.4.2 Sistema de 06 Barras de Garver sem Redespacho

Neste caso, não é permitido o redespacho da geração, isto é, a soma das gerações nas barras de geração é igual a demanda total do sistema; portanto, nas configurações factíveis, a soma da geração é igual a soma de demandas. Os dados deste sistema de 6 barras e 15 ramos candidatos aparecem no Apêndice A. Os outros dados são os mesmos do caso anterior. Este sistema somente é avaliado para o caso em que a configuração inicial é gerada aleatoriamente, já que com o algoritmo inicializador baseado na metodologia de Garver modificada se obtém a solução ótima.

Os parâmetros obtidos foram os seguintes: $\rho = 1,2$, $\beta = 0,7$, $\alpha = 10$, $\mu = 2$, $w_o = 0$ e T_o na faixa de [200 - 400]. Neste, o processo obtém convergência entre 1.000 a 1.300 iterações (chamadas de PL).

Foram realizadas 7 adições de linhas, ou seja, $n_{26} = 4$, $n_{35} = 1$ e $n_{46} = 2$, com um investimento de 200. Esta configuração ótima já era conhecida. Na Figura 3.8 é mostrada a solução ótima para o sistema Garver sem redespacho.

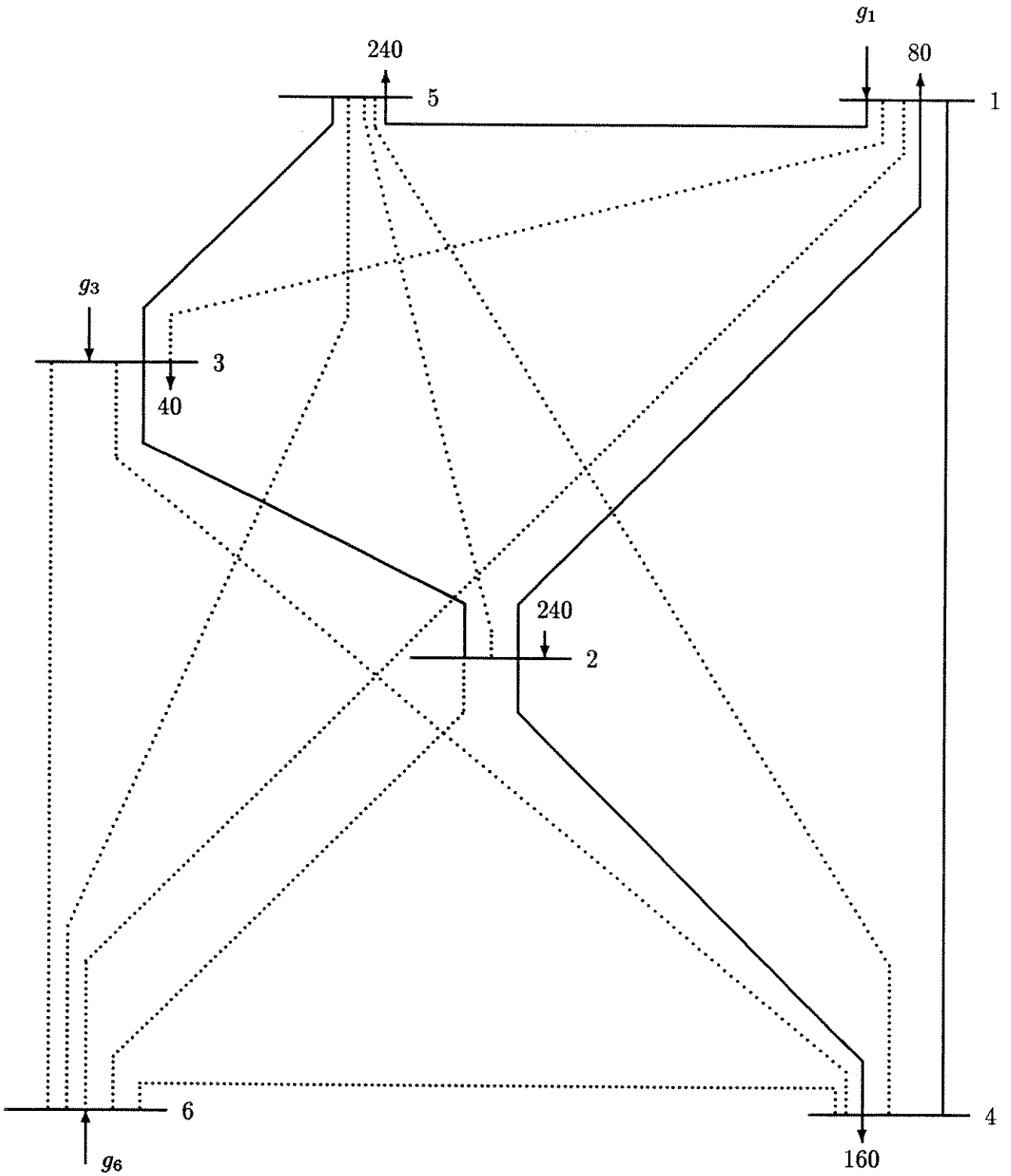


Figura 3.6: Sistema de 06 Barras - Configuração Inicial.

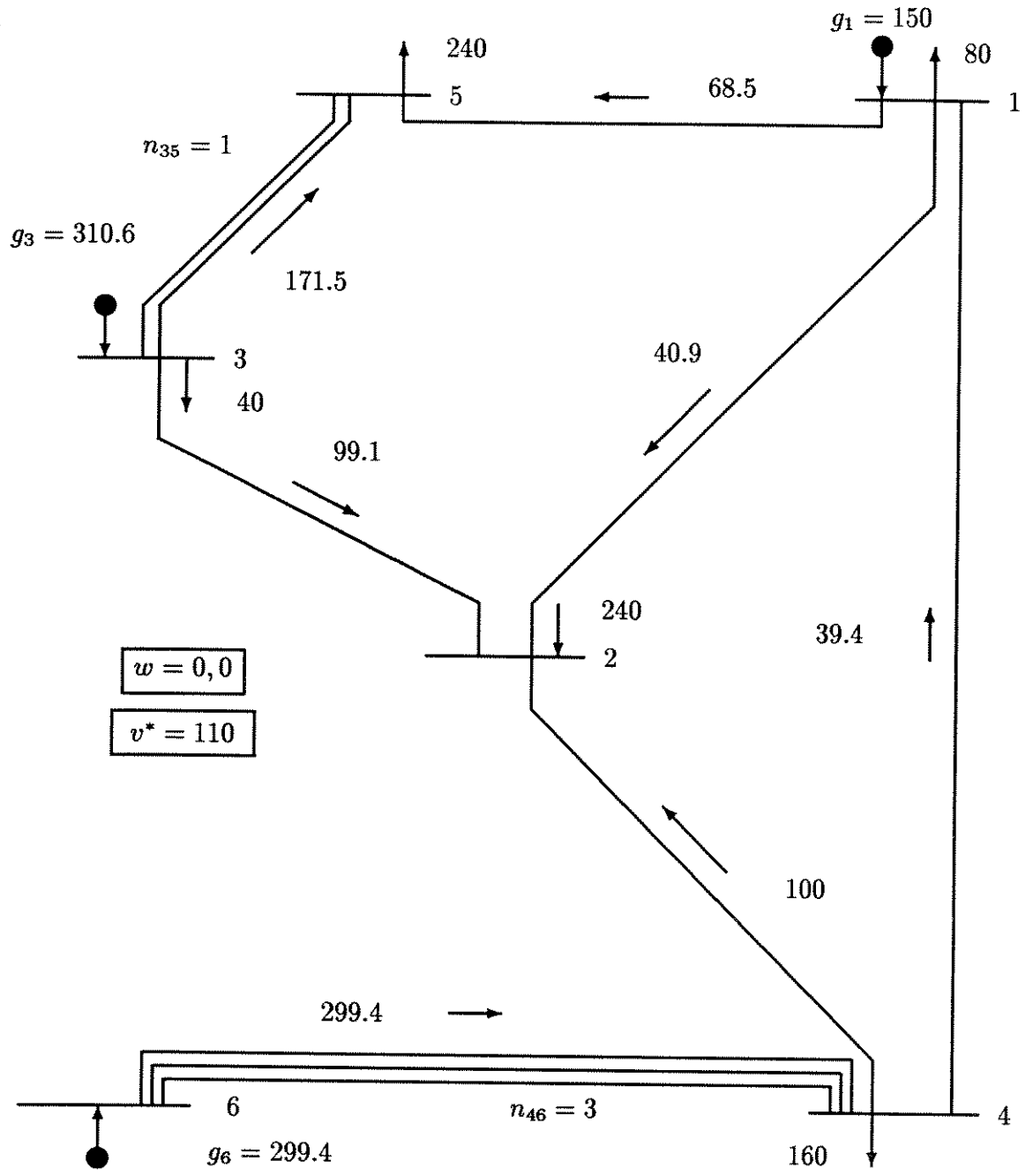


Figura 3.7 - Sistema de 06 Barras - Configuração Ótima com Redespacho

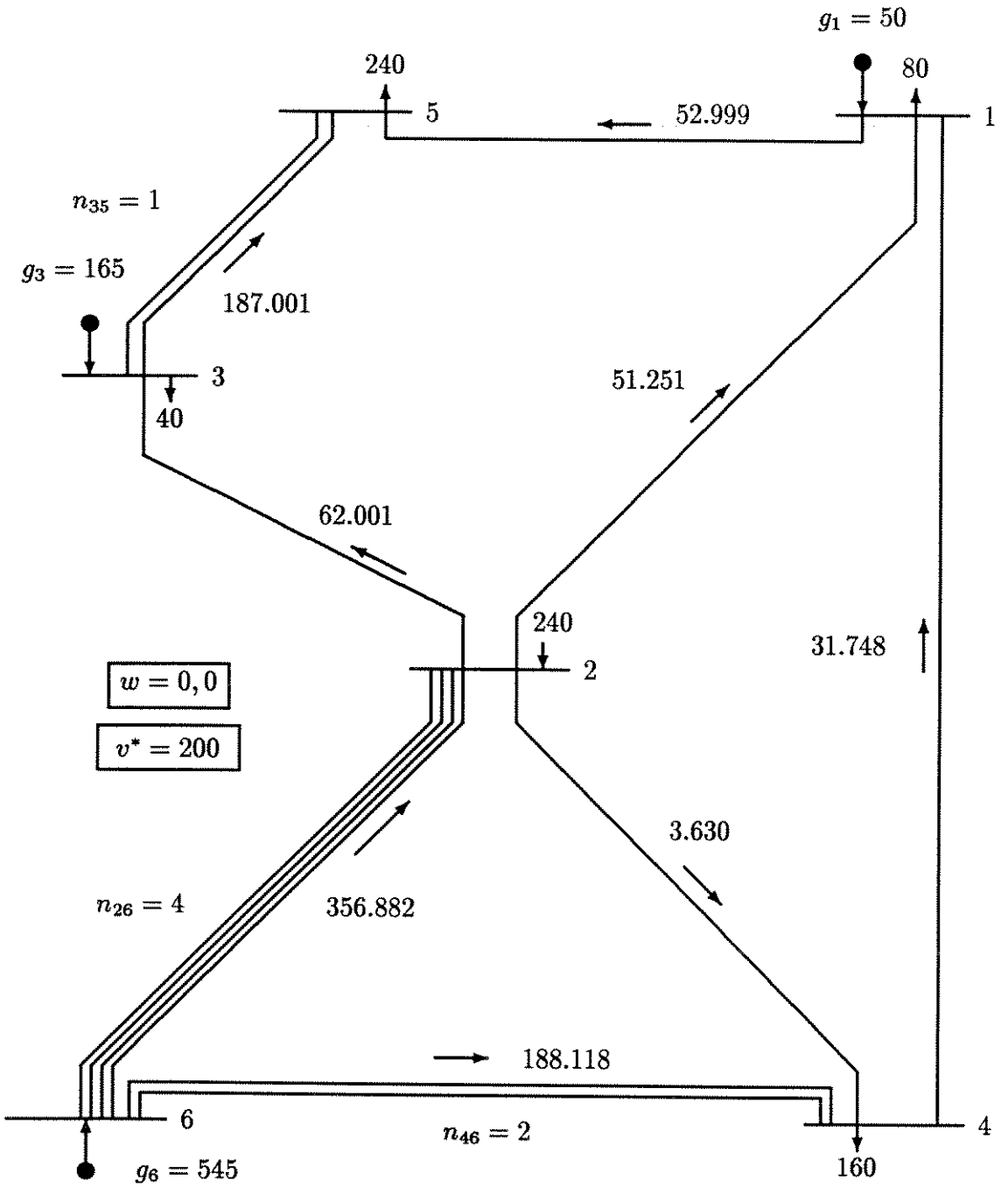


Figura 3.8: Sistema de 06 Barras - Configuração Ótima Sem Redespacho.

3.4.3 Sistema Sul Brasileiro com Redespacho

Neste caso, foi testado o sistema Sul brasileiro para 1990 tomando 1980 como ano de referência. Como foi mencionado anteriormente, aqui é permitido o redespacho da geração. Os dados deste sistema de 46 barras e 79 ramos candidatos aparecem no Apêndice A. A topologia inicial é mostrada na Figura 3.9; na mesma, e em traços grossos, se observa a rede inicial.

Este sistema foi avaliado tanto para o caso no qual a configuração inicial é gerada de maneira aleatória, como também para o caso em que a configuração inicial é gerada baseado-se na metodologia de Garver modificada. Apesar de que, com este último, se obtém uma solução inicial muito perto da ótima; quando é aplicado o algoritmo **SA**, esta configuração muda drásticamente devido ao fato de que para altas temperaturas o algoritmo **SA** apresenta uma alta diversidade. Isto é, nos primeiros níveis de temperatura uma grande maioria das linhas propostas são aceitas. Esta é uma característica típica do **SA** na qual o método apresenta um comportamento semelhante, independente da configuração inicial da qual se inicia o processo.

Os parâmetros obtidos foram os seguintes: $\rho = 1,2$, $\beta = 0,7$, $\alpha = 1.000$, $\mu = 3$, $w_o = 0$ e T_o na faixa de [15.000 - 25.000], para o caso no qual a configuração inicial é gerada baseado-se na metodologia de Garver modificada; e T_o na faixa de [40.000 - 50.000], no caso em que a configuração inicial é gerada de maneira aleatória. No primeiro caso, o processo converge entre 4.000 a 5.000 iterações (chamadas de PL), no segundo, o processo converge entre 7.000 a 8.000 iterações.

Na solução final o processo levou a um investimento de US$ 70.289.000,00 que corresponde a adição das seguintes 8 linhas: $n_{13-20} = 1$, $n_{20-23} = 1$, $n_{46-06} = 1$, $n_{20-21} = 2$, $n_{42-43} = 1$ e $n_{05-06} = 2$, isto é, foram adicionados somente 3 linhas e 5 transformadores. Esta configuração ótima também já era conhecida. Na Figura 3.10, é mostrada a solução ótima para o sistema Sul brasileiro com redespacho.

Uma característica importante do algoritmo de **SA** é o fato de que, além de encontrar supostamente a configuração ótima, durante a evolução do processo este algoritmo obtém muitas outras soluções ótimas locais ou quase-ótimas. Um fato adicional relevante é que estas soluções quase-ótimas existem em grandes quantidades sendo muitas delas tão boas quanto a melhor das soluções encontradas. Esta particularidade fornece ao algoritmo de **SA** um atrativo adicional, pois em contraste com outros algoritmos que somente fornecem uma solução (e nem sempre a ótima), o algoritmo de **SA** fornece a melhor solução (freqüentemente a ótima) e um número grande de outras excelentes configurações que eventualmente podem ter melhor desempenho em fases posteriores de análise do sistema elétrico. Assim, do ponto de vista do planejador, esta particularidade torna o algoritmo de **SA** muito atrativo.

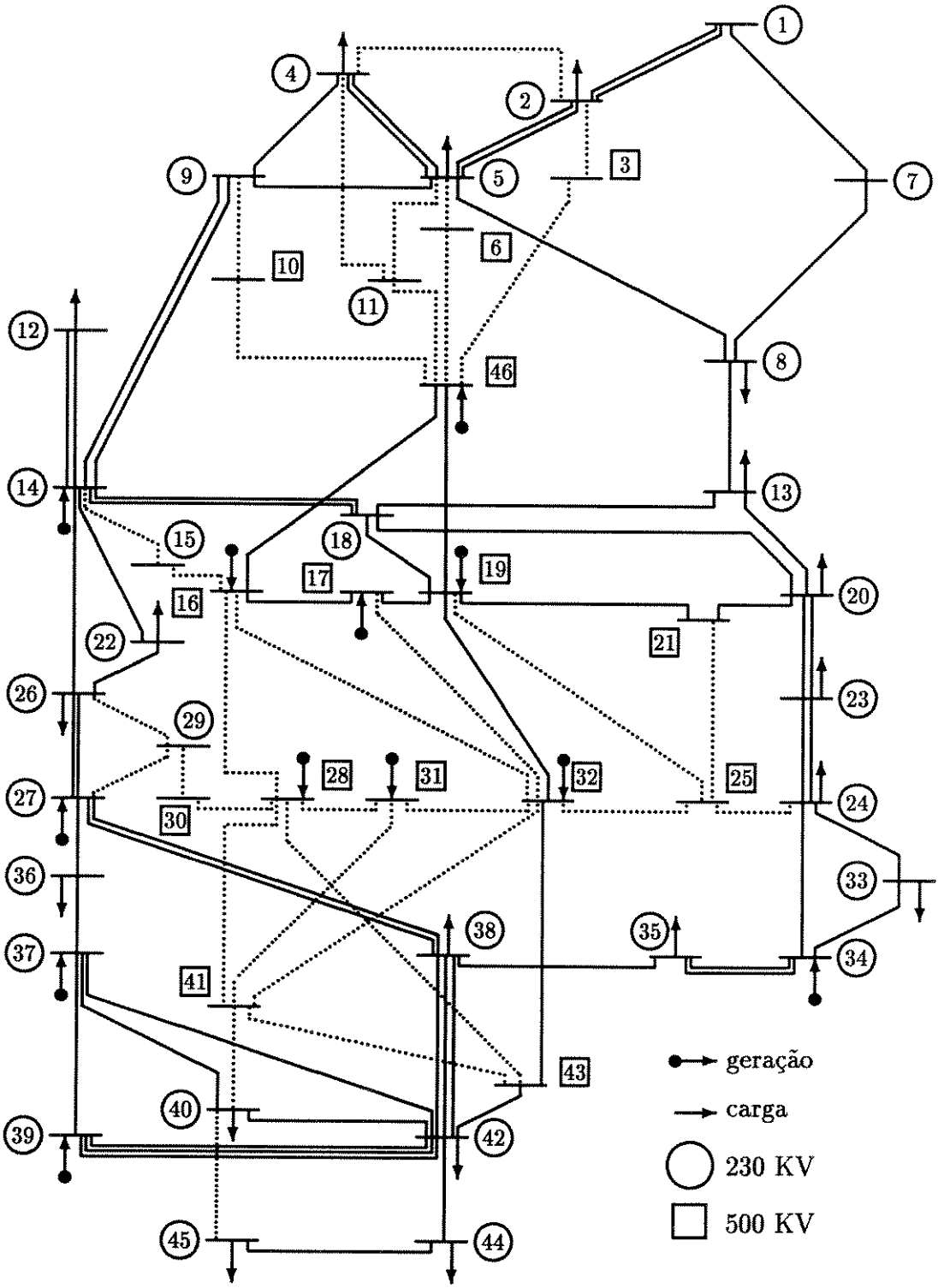


Figura 3.9: Sistema Sul brasileiro - Rede Inicial.

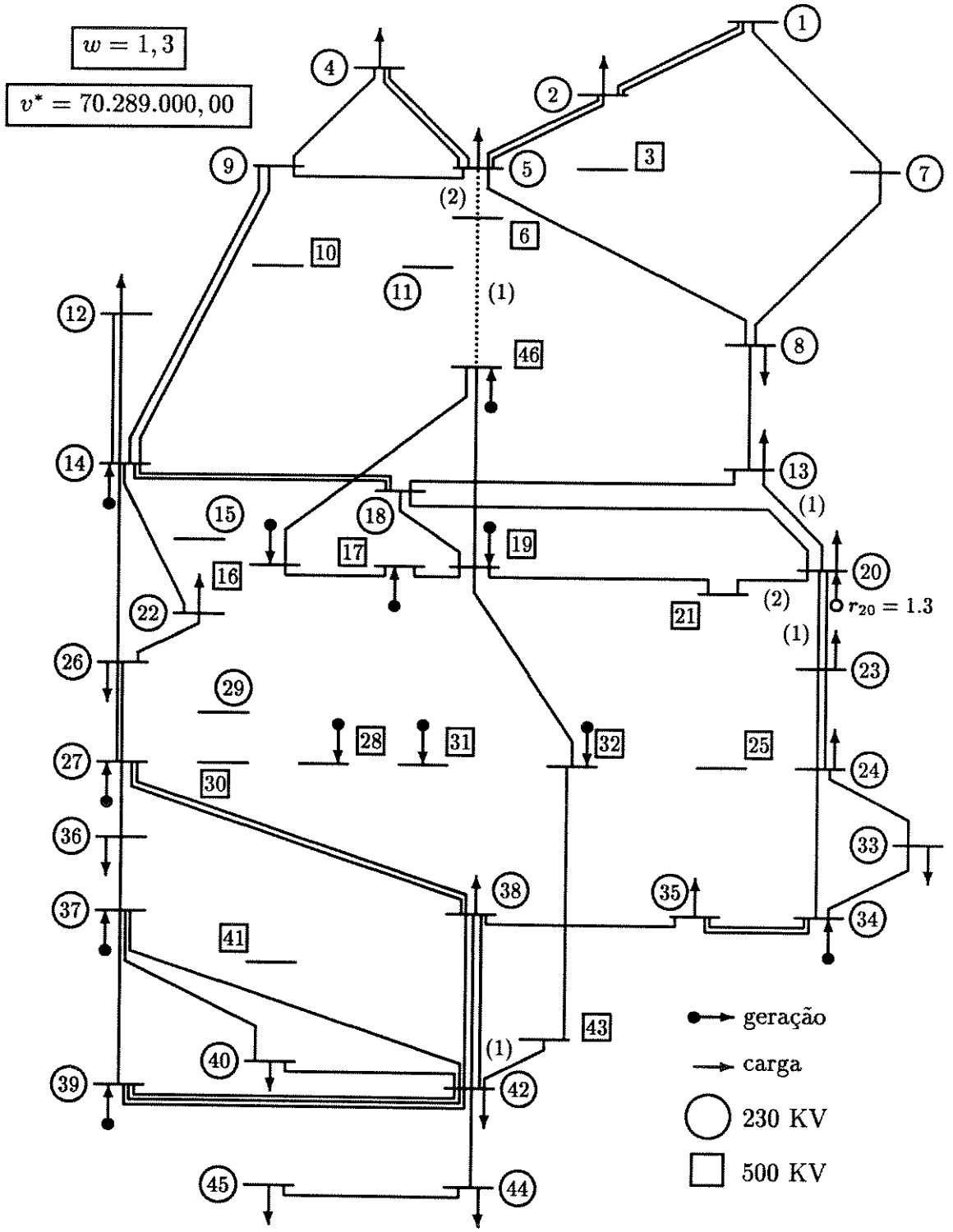


Figura 3.10: Sistema Sul brasileiro - Configuração Ótima Com Redespacho.

3.4.4 Sistema Sul Brasileiro sem Redespacho

Neste caso a soma das gerações é igual a soma das demandas; esta condição será alcançada pelas configurações factíveis do ponto de vista do problema real, uma vez que o uso de geradores fictícios sempre levarão à factibilidade do modelo; caso contrário, as configurações serão infactíveis. O valor da infactibilidade é chamado corte de carga e corresponderá a soma das gerações fictícias na rede, que na verdade correspondem à quantidade de demanda não abastecida por falta de capacidade na transmissão.

Este sistema foi avaliado tanto para o caso no qual a configuração inicial é gerada de maneira aleatória, como também para o caso em que a mesma é gerada baseada na metodologia de Garver modificada. Como no caso anterior, o algoritmo **SA** apresenta um comportamento semelhante independente da configuração inicial. Quando o algoritmo inicia de uma configuração gerada com Garver modificado, geralmente a temperatura T_o deverá ser mais baixa que para o caso no qual parte de uma configuração aleatória; isto com a finalidade de que o algoritmo faça proveito da configuração inicial, evitando desta maneira diversificações bruscas que levem o processo às regiões muito distantes das regiões atrativas do ponto de vista de custo. A metodologia Garver modificada mostrou grande potencialidade ao identificar nas configurações iniciais grande parte dos ramos da configuração ótima. Para aproveitar a vantagem anterior, as configurações iniciais geradas com Garver iniciam o processo com uma temperatura mais baixa em relação ao caso das configurações geradas de maneira aleatória.

Os parâmetros obtidos foram os seguintes: $\rho = 1,2$, $\beta = 0,7$, $\alpha = 1.000$, $\mu = 3$, $w_o = 0$ e T_o na faixa de [25.000 - 30.000], para o caso no qual a configuração inicial é gerada baseado-se na metodologia de Garver modificada; e T_o na faixa de [50.000 - 70.000], no caso em que a configuração inicial é gerada de maneira aleatória. No primeiro caso, o processo converge entre 6.000 a 7.000 iterações (chamadas de PL); no segundo, o processo converge entre 10.000 a 11.000 iterações.

Na solução final o processo levou ao investimento de US$ 154.420.000,00, que corresponde a adição das seguintes 16 linhas: $n_{20-21} = 1$, $n_{42-43} = 2$, $n_{46-06} = 1$, $n_{19-25} = 1$, $n_{31-32} = 1$, $n_{28-30} = 1$, $n_{26-29} = 3$, $n_{24-25} = 2$, $n_{29-30} = 2$ e $n_{05-06} = 2$. Esta configuração ótima também já era conhecida. Na Figura 3.11, é mostrada a solução ótima para o sistema Sul brasileiro sem redespacho.

3.5 TESTES EM UM SISTEMA DE GRANDE PORTE

Para realizar estes testes foi utilizado o sistema Norte-Nordeste brasileiro, que constitui em um sistema de grande porte com muitas barras isoladas da configuração inicial, o

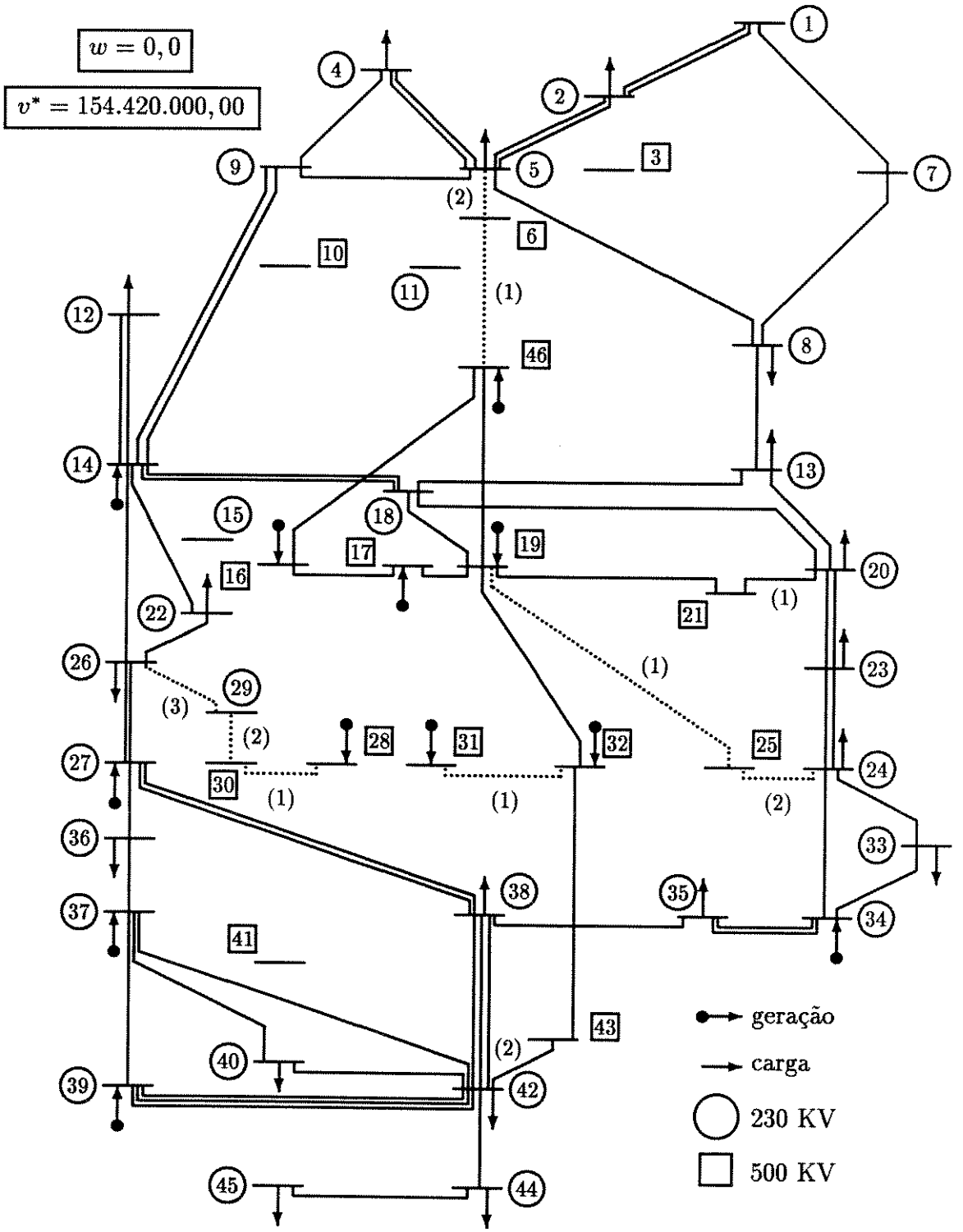


Figura 3.11: Sistema Sul brasileiro - Configuração Ótima Sem Redespacho.

que torna o processo de resolução muito difícil. Da literatura tem-se que este sistema não foi resolvido usando técnicas de otimização exatas, foram usados apenas métodos aproximados. Portanto, ainda não se conhece a solução ótima para o mesmo. A configuração inicial deste sistema é mostrada na Figura 3.12 onde pode-se observar o elevado nível de ilhamento, isto é, muitas barras estão isoladas na configuração básica inicial.

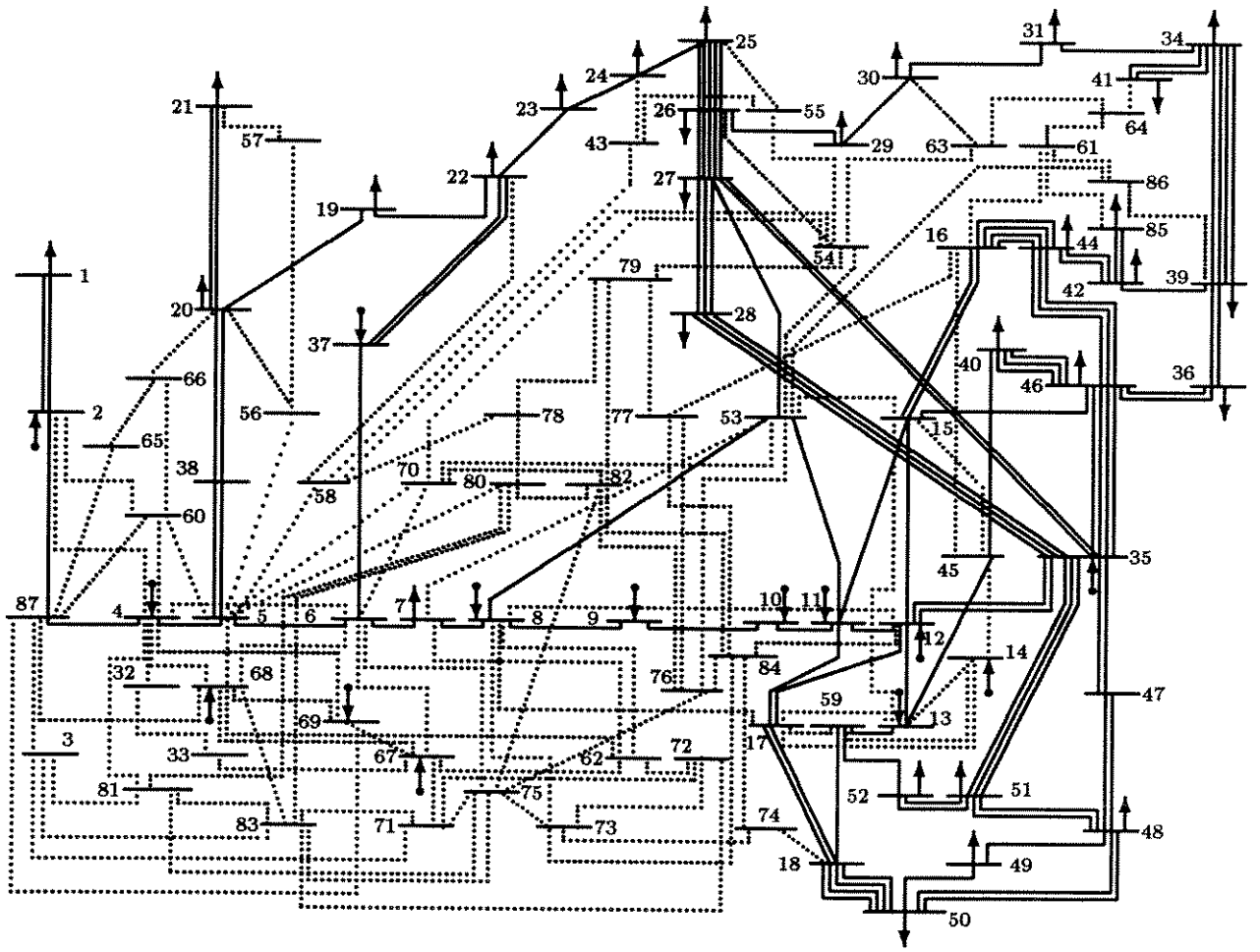


Figura 3.12: Sistema Norte-Nordeste Brasileiro

3.5.1 Sistema Norte-Nordeste Brasileiro - Plano 2008

Foi testado o sistema Norte-Nordeste brasileiro para o ano 2008 tomando 1980 como ano de referência. A Figura 3.12 mostra este sistema, o qual tem uma demanda total de 29.754 MW, os dados deste sistema de 89 barras e 183 linhas candidatas aparecem no Apêndice A.

A rede Norte-Nordeste é um sistema de difícil solução, sendo o número de possíveis soluções da ordem de $(11+1)^{183}$, para o caso em que são possíveis até 11 adições por caminho candidato. Quando o processo de **SA** é inicializado com uma configuração gerada de maneira aleatória, geralmente fica preso em regiões onde só existem configurações de má qualidade. Esta metodologia possui, nos primeiros níveis de temperatura, um alto grau de diversidade o qual leva à entrada e saída de linhas com muita facilidade e após vários níveis de temperatura geralmente o processo está localizado em regiões de ótimo local nem sempre de boa qualidade. Após muitos níveis de temperatura, o processo pode ficar preso em regiões pouco atrativas e nesse momento não possui temperatura suficiente e ficaria preso em ótimos locais pouco atrativos. Por isto, os processos são inicializados com configurações geradas com base na metodologia de Garver modificada.

Os parâmetros obtidos foram os seguintes: $\rho = 1,2$, $\beta = 0,7$, $\alpha = 1.000$, $\mu = 5$, $w_o = 300$ e T_o na faixa de [50.000 - 80.000]. O processo converge entre 250.000 a 300.000 iterações (chamadas de PL).

Diferentes soluções foram obtidas; por exemplo, uma solução com um custo de US$ 2.664.109.000,00 e um corte de carga de 17 MW. A melhor solução sem corte de carga apresentou uma configuração com um custo de US$ 2.676.704.000,00. As linhas requeridas correspondentes a essa solução são:

$n_{01-02} = 1$, $n_{02-04} = 1$, $n_{04-05} = 5$, $n_{04-06} = 3$, $n_{04-68} = 1$, $n_{05-56} = 1$, $n_{05-58} = 5$, $n_{06-07} = 2$, $n_{07-08} = 2$, $n_{08-17} = 2$, $n_{12-15} = 1$, $n_{12-17} = 1$, $n_{13-14} = 1$, $n_{13-15} = 4$, $n_{14-59} = 1$, $n_{15-16} = 3$, $n_{15-45} = 1$, $n_{15-46} = 1$, $n_{16-44} = 6$, $n_{17-18} = 2$, $n_{18-50} = 11$, $n_{18-74} = 3$, $n_{21-57} = 2$, $n_{22-23} = 1$, $n_{22-58} = 2$, $n_{24-43} = 1$, $n_{25-55} = 3$, $n_{26-54} = 3$, $n_{30-31} = 4$, $n_{30-63} = 3$, $n_{31-34} = 2$, $n_{35-51} = 1$, $n_{36-46} = 2$, $n_{40-45} = 2$, $n_{41-64} = 2$, $n_{42-44} = 2$, $n_{42-85} = 1$, $n_{43-55} = 2$, $n_{43-58} = 2$, $n_{48-49} = 1$, $n_{49-50} = 4$, $n_{52-59} = 1$, $n_{53-86} = 1$, $n_{54-58} = 2$, $n_{54-63} = 1$, $n_{56-57} = 1$, $n_{61-64} = 1$, $n_{61-85} = 2$, $n_{61-86} = 1$, $n_{67-68} = 1$, $n_{67-69} = 1$, $n_{67-71} = 3$, $n_{71-72} = 1$, $n_{72-73} = 1$, $n_{73-74} = 1$.

A metodologia ainda pode encontrar uma grande quantidade de soluções ótimas locais com valores muito próximos da melhor configuração encontrada.

Neste sistema é importante comentar o fato de que o número de linhas adicionadas é relativamente elevado: foram adicionadas 116 linhas em 55 caminhos, o que realmente tornaria muito difícil resolver este sistema por métodos de otimização exatos. Além disso, este sistema apresenta características muito especiais como a necessidade de adicionar 11 linhas entre as barras 18 e 50 para resolver o problema de carga para a cidade de Salvador.

A melhor solução encontrada apresenta um investimento de 174 milhões de dólares menor que a melhor solução conhecida para este sistema (2.850 milhões de dólares) e que foi obtida pelo método aproximado conforme apresentado em [36].

3.6 *SA* PARALELO APLICADO AO PROBLEMA DO PLANEJAMENTO

O esforço computacional requerido pelo algoritmo seqüencial "Simulated Annealing" depende em grande parte da natureza e tamanho do problema de otimização apresentado; assim, para o caso do planejamento de redes de transmissão de grande porte, este esforço de cálculo pode levar dias até conseguir a convergência. Existem vários fatores que podem melhorar o algoritmo no que se refere ao tempo de computação e qualidade da resposta; os mais importantes são:

- Projeto do algoritmo seqüencial,
- Aceleração do "hardware",
- Projeto de algoritmos paralelos.

Um bom projeto do algoritmo depende basicamente de uma seleção adequada da estrutura de vizinhança e dos parâmetros de controle. No que se refere ao processamento paralelo, oferece grande oportunidade de atender a convergência em um tempo computacional muito menor e com uma melhor qualidade da resposta, tratando-se de simulação de sistemas de grande porte.

No algoritmo **SA** o maior consumo de tempo está na etapa de geração de tentativas para a formação de uma cadeia de Markov. Nos algoritmos paralelos se toma vantagem desta característica atribuindo a cada processador uma parcela destas tentativas. As tentativas são particionadas em partes iguais, de acordo ao número de processadores.

Uma forma de acelerar o processo de convergência de um algoritmo de **SA** é implementar uma versão paralela do algoritmo. Implementações paralelas de **SA** foram usadas em muitas aplicações de pesquisa operacional, obtendo-se diferentes tipos de desempenho. As implementações com maiores sucessos foram as implementações paralelas especializadas ("tailored") que são possíveis especialmente em problemas que permitem partição de tarefas como alocação de VLSI e problemas de roteamento ("routing"). Para problemas mais complexos e que não permitem uma partição, somente são possíveis as implementações paralelas de caráter geral.

A parcela do algoritmo de **SA** que consome maior esforço computacional é a geração das N_k configurações candidatas para o nível de temperatura T_k. A avaliação de cada uma das N_k configurações candidatas implica em fazer os seguintes testes:

1. A partir da configuração corrente, selecionar uma configuração candidata usando a estrutura de vizinhança.

2. Calcular a diferença da função objetivo da configuração candidata com a configuração corrente.

3. Decidir se a configuração candidata deve ser aceita ou rejeitada.

4. Substituir a configuração corrente pela configuração candidata que foi aceita.

A implementação eficiente de um algoritmo de **SA** paralelo deve ser realizada na seqüência de passos mostrada anteriormente. Além disso, pode-se fazer as seguintes observações:

1. Os três primeiros passos são praticamente independentes e podem ser implementados em paralelo para diferentes configurações candidatas sem alterar a natureza seqüencial do algoritmo. Por outro lado, executar o passo 4 implica num processo estritamente seqüencial pois não podem ser aceitas novas configurações em paralelo;

2. O número de vezes em que são executados os passos 1-3 praticamente não varia em todo o processo para os diferentes níveis de temperatura; entretanto, a freqüência de execução do passo 4 varia muito durante o processo, sendo freqüentemente executada para valores elevados de temperatura e quase não é executada para valores pequenos de temperatura;

3. O esforço computacional necessário para implementar cada um dos quatros passos varia muito de um tipo de problema para outro. Assim, por exemplo, no problema do caixeiro viajante o esforço computacional para executar os passos 2 e 3 são parecidos pois é muito simples calcular a diferença de custos entre a configuração corrente e a configuração candidata, com um esforço computacional da mesma ordem que calcular a função exponencial que define o critério de aceitação do passo 3;

4. No problema de planejamento de sistemas de transmissão a longo prazo, o esforço computacional requerido pelo passo 2 é muito grande quando comparado com os outros passos. Executar o passo 2 implica resolver um problema de programação linear (PL) de grande porte para determinar o valor da função objetivo da configuração candidata. Assim, o tempo gasto nos outros passos são totalmente irrelevantes comparados com o tempo gasto no passo 2 no problema de planejamento;

5. A qualidade da solução final obtida é parecida a obtida por um algoritmo seqüencial, desde que o número de processadores não seja muito grande. Quando o número de processadores aumenta muito, o subconjunto de configurações analisadas por cada processador pode ser muito pequeno, não permitindo que o processo possa atingir o estado de quase-equilíbrio.

Baseadas nas observações anteriores e nas particularidades próprias do problema de planejamento de sistemas de transmissão, foram implementados algoritmos de **SA** paralelos de caráter geral conhecidos na literatura especializada de **SA**: *algoritmo paralelo da divisão* e *algoritmo paralelo de agrupamento* ("clustering").

3.6.1 Metodologias Básicas de Paralelização

Algoritmo da Divisão:

No *algoritmo da divisão* [19], para cada nível de temperatura a tarefa de busca de novas configurações é dividida entre todos os processadores. Se N_k é o número de tentativas de busca de novas configurações realizadas na temperatura T_k e np o número de processadores, então, nessa temperatura, cada processador realiza N_k/np tentativas, sendo que cada processador gera sua própria seqüência de configurações. O algoritmo da divisão permite duas estratégias diferentes. Na *primeira estratégia*, não existe comunicação entre processadores; isto é, cada processador realiza um conjunto de N_k/np tentativas e passa ao seguinte nível de temperatura com sua configuração corrente sem trocar informação (as configurações) com os outros processadores e, no final do processo, das np configurações (uma por processador) é escolhida a melhor configuração como sendo a configuração final. Na *segunda estratégia* existe troca de informação toda vez que existe mudança de temperatura onde é escolhida a melhor configuração das np configurações disponíveis (uma por processador). Assim, esta melhor configuração é usada como configuração inicial por todos os processadores no nível de temperatura seguinte.

O algoritmo da divisão apresenta pequena comunicação e portanto a eficiência pode ser elevada. Quando o número de processadores aumenta, o número de tentativas que cada processador realiza em cada nível de temperatura pode tornar-se pequeno; isto pode levar a um comportamento diferente quando comparado com um algoritmo seqüencial. Assim, no processamento paralelo, pode-se aumentar o número de tentativas N_k a cada nível de temperatura e/ou diminuir a taxa de diminuição de temperatura, incrementando o parâmetro β, a fim de que cada processador possa atingir o quase-equilíbrio do ponto de vista de **SA**. Em ambas estratégias, pode-se iniciar o processo com uma única configuração inicial ou com configurações diferentes.

A Figura 3.13 mostra a estrutura de operação do algoritmo da divisão para o caso de uma implementação paralela com 8 processadores.

Algoritmo de Agrupamento ("Clustering"):

No *algoritmo de agrupamento paralelo puro*[19], todos os processadores que formam o "cluster" avaliam cooperativamente as N_k configurações candidatas. Neste caso todos os

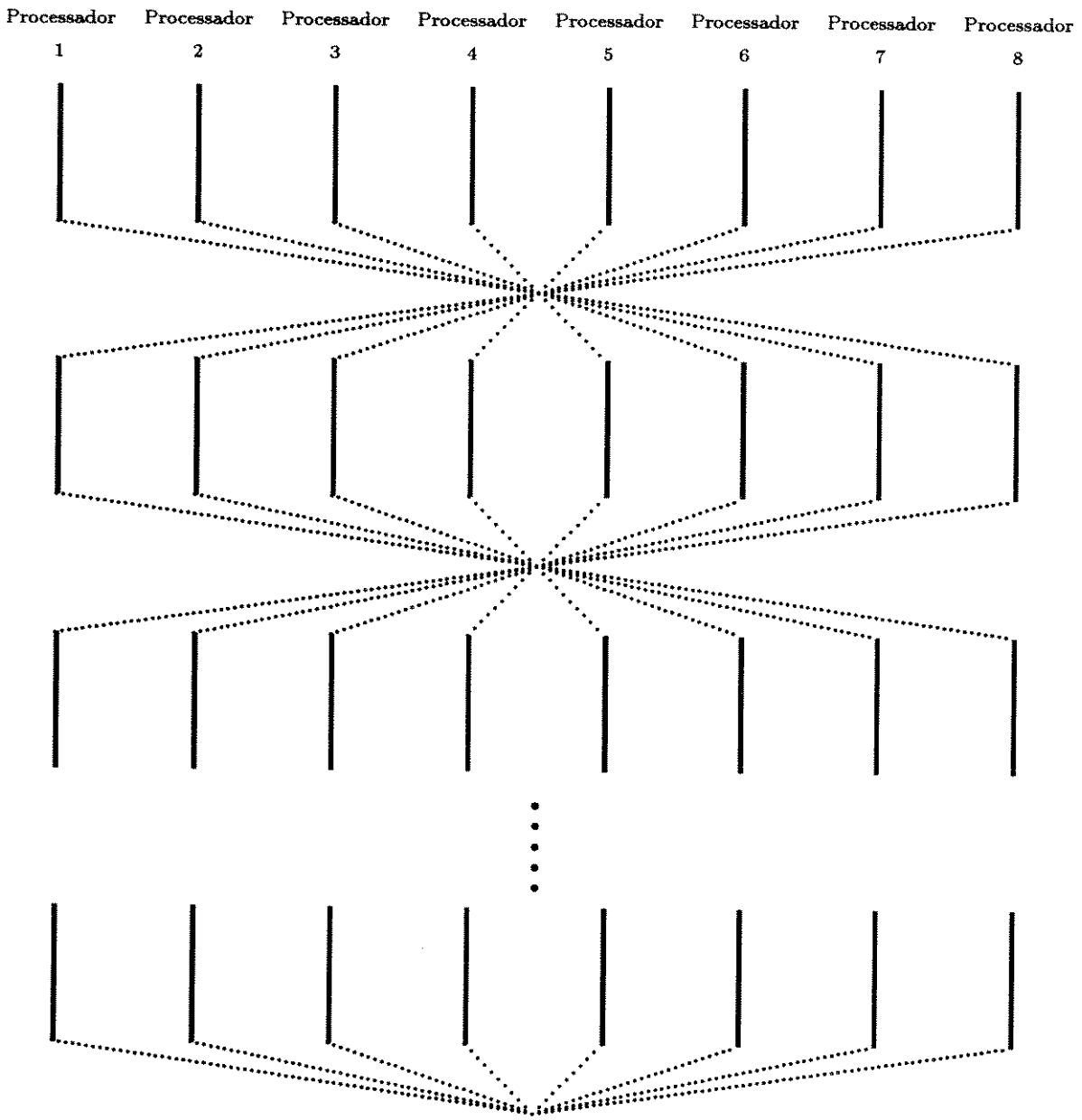


Figura 3.13: Estrutura de operação do Algoritmo da Divisão

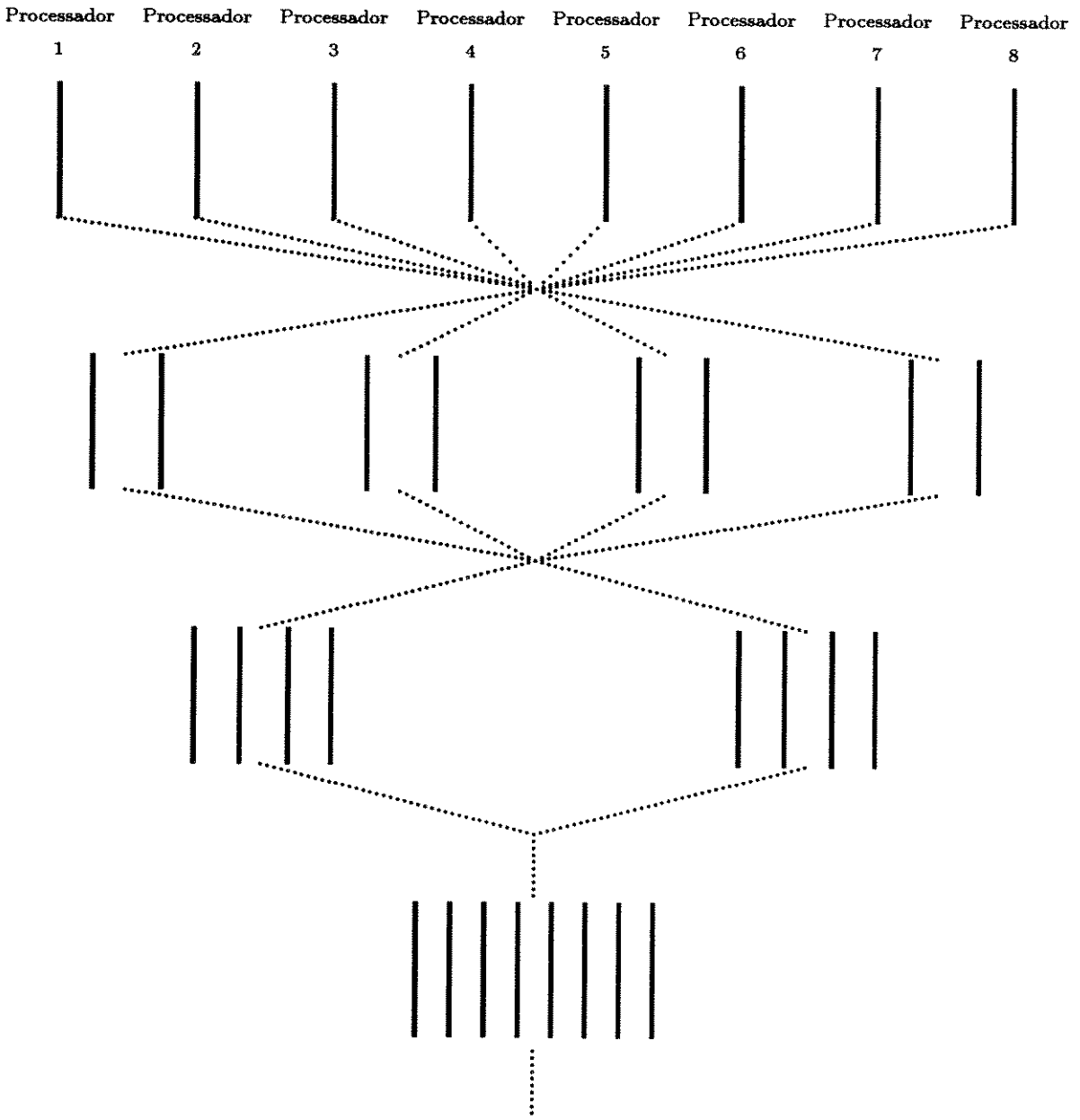


Figura 3.14: Estrutura de operação do Algoritmo "clustering" híbrido

processadores tem a mesma configuração corrente e, cada vez que um processador aceita uma nova configuração este fato deve ser comunicado aos outros processadores que devem interromper seu trabalho e atualizar sua configuração com a nova configuração. Assim, durante o processo existe somente uma configuração corrente que é usada por todos os processadores e quando aparece uma nova configuração obtida por algum processador, todos os outros processadores assumem esta nova configuração. Esta implementação apresenta eficiência pequena para valores elevados de temperatura, pois nessas condições muitas configurações são aceitas, o que levaria a que, permanentemente, alguns processadores interrompessem seu trabalho corrente para atualizar sua configuração, determinada por outro processador. Entretanto, para valores baixos de temperatura, onde a grande maioria de configurações são rejeitadas, a eficiência é elevada pois praticamente não existe "interferência" entre os processadores. Esta característica levou à implementação de um algoritmo de agrupamento híbrido que aproveita as vantagens do algoritmo de agrupamento puro e contorna suas desvantagens.

No *algoritmo de agrupamento híbrido* o processo é iniciado de maneira similar ao algoritmo da divisão. No final do processo de cada nível de temperatura é quantificado um critério de eficiência para determinar se os processadores podem ser agrupados. Este critério de eficiência está relacionado com o número de configurações candidatas aceitas em cada nível de temperatura e deve ter um valor decrescente com a diminuição da mesma. Após o primeiro agrupamento, cada grupo de dois processadores trabalham cooperativamente avaliando um subconjunto de configurações candidatas com a mesma configuração corrente e, logicamente, toda vez que um processador aceita uma nova configuração o outro processador interrompe seu trabalho para atualizar sua configuração com aquela que foi aceita pelo primeiro processador. O processo continua, eventualmente agrupando os processadores dois-a-dois, quatro-a-quatro, oito-a-oito, etc., toda vez que se muda o nível de temperatura e o critério de eficiência for aceito. Nas fases finais do processo todos os processadores trabalham cooperativamente como no algoritmo de agrupamento puro, com todos os processadores trabalhando com a mesma configuração corrente e todos eles participando da geração das N_k configurações candidatas.

A Figura 3.14 mostra a estrutura de operação do algoritmo paralelo "cluster" híbrido para o caso de uma implementação paralela com 8 processadores.

3.6.2 Descrição dos Algoritmos Paralelos Implementados

Algoritmo da Divisão

Todos os np processadores alocados executam o fluxo DC inicial; com isto, o nó "0", bem como os outros nós, terão chamado o software MINOS uma vez para resolver um PL e obtido uma solução de partida do sub-problema de operação. Os np processadores então

selecionam a temperatura inicial T_0, e o tamanho da cadeia de Markov inicial, $N_0 = \mu.nl$ (μ geralmente é igual a 2, 3 ou 4 para os sistemas Garver e Sul e 4, 5 ou 6 para o sistema Norte-Nordeste).

O tamanho da cadeia de Markov inicial, N_0, corresponde na prática ao número de tentativas de investimento que devem ser feitas no nível T_0 de temperatura. O mesmo vale para N_k e T_k e, enquanto a temperatura é decrementada para níveis cada vez mais baixos, a quantidade de tentativas é acrescida de cifras da ordem de 20% ou mais.

Nos algoritmos paralelos, a cadeia N_k é dividida entre os np processadores; assim, as sub-cadeias terão o tamanho N_k/np. Isto significa que o esforço de trabalhar sobre as N_k tentativas é repartido entre os np nós.

Assim que o nó "0" completa as iterações de sua sub-cadeia, ele armazena em variáveis temporárias a configuração aceita mais recente ($FO = kobje$); em seguida, informa aos demais processadores que sua sub-cadeia foi concluída. É, então, chegada a hora de cada um dos *np-1* processadores comunicar ao nó "0" a configuração aceita mais atual.

O nó "0" ao receber a configuração enviada por um dado nó, compara a sua solução ($FO = kobje$) com a que recebe. Caso a configuração que chega seja melhor, ela será trocada pela do nó "0" ($kobje^{velho} \leftarrow kobje^{novo}$). Se não, a configuração que chega será descartada. Este procedimento de seleção da configuração de melhor **FO** prossegue até que todos os *np-1* nós tenham comunicado suas configurações ao nó "0". Ao final ter-se-á a solução incumbente para esse nível de temperatura e para essa cadeia.

Para efeito de critério de parada, as configurações aceitas pelo nó "0" em cada nível de temperatura são contadas ($ncont1$). O mesmo é feito em relação as configurações que chegam e que sejam idênticas a do nó "0" ($ncont2$).

Em seguida, continuando o processamento, o nó "0" envia a solução incumbente a todos os *np-1* processadores. Todos os nós partirão trabalhando sobre essa solução incumbente num novo nível de temperatura e numa nova cadeia. Neste ponto, duas versões do algoritmo da divisão podem ser derivadas dependendo da rapidez desejada para se atingir o "congelamento":

1. **versão 1** - que, ao selecionar a incumbente, só muda T_k e N_k após completar np ciclos de processamento;
2. **versão 2** - que muda a temperatura a cada ciclo e a cadeia é mudada a cada np ciclos, ou seja, assim que se obtém a incumbente. (Esta versão trabalha com decaimentos/elevações mais freqüentes de T_k). Um ciclo completo corresponde ao cálculo de uma sub-cadeia e a conseqüente escolha e envio da incumbente.

O processamento terminará se for satisfeito um dos seguintes critérios de parada:

1. O número de iterações excedeu o limite pré-especificado.

2. O "congelamento" foi atingido. Este é um critério (de processamento paralelo) composto. O "congelamento" é alcançado se: **(a)** nenhuma configuração foi aceita pelo nó "0" num dado ciclo de processamento; ou **(b)** as configurações de todos os np nós são idênticas num dado ciclo.

Algoritmo da Divisão Versão 1:

Possui a mesma filosofia descrita anteriormente. Consiste em mudar o nível de temperatura e o tamanho da cadeia somente depois que cada processador tenha efetuado np sub-cadeias (isto é, np ciclos). Nesta versão, maneja-se um fator de decremento da temperatura de 0,75 ($\beta = 0,75$) e um incremento da cadeia de 1,2 ($\rho = 1,2$).

Algoritmo da Divisão Versão 2:

Possui a mesma filosofia descrita anteriormente. Entretanto, consiste em mudar o nível de temperatura a cada sub-cadeia que é completada. O tamanho da sub-cadeia é alterado somente a cada np ciclos. Nesta versão, maneja-se um fator de decremento da temperatura estimado a partir da relação $(0,75)^{\frac{1}{np}}$ ($\beta \geq (0,75)^{\frac{1}{np}}$) e um incremento da cadeia de $(1,2)^{\frac{1}{np}}$ ($\rho = (1,2)^{\frac{1}{np}}$).

Algoritmo do Agrupamento ("Clustering" - Híbrido)

O algoritmo "cluster" trabalha da seguinte maneira:

O processo começa trabalhando como o algoritmo da divisão mencionado anteriormente; após efetuar vários níveis de temperatura e cumprido o critério de eficiência começa o processo de agrupamento, reunindo em grupos de dois. Cada um dos processadores trabalha uma subcadeia calculada como $N_k/nsize$ ($nsize$ = tamanho do "cluster"). Durante o cálculo em cada "cluster" se compartilha a informação; dessa forma, a cada transição encontrada, o processador que a achou envia tal solução aos demais processadores do seu "cluster". Na verdade, por razões de coordenação e comunicação, todas as informações são centralizadas no nó "0".

Terminada a subcadeia, todos os "clusters" compartilham a informação tomando como incumbente a de melhor função objetivo; além disso é alterado o nível de temperatura.

Como modificar o tamanho do "cluster"?

Terminada a subcadeia, se calcula a porcentagem de transições aceitas no "cluster" que contém o nó zero da seguinte maneira:

- Porcentagem de transições aceitas = Número de configurações aceitas/Número total de propostas.
- Se a porcentagem é menor que $100/nsize$, o tamanho do "cluster" é duplicado: $nsize^{novo} = 2.nsize^{velho}$; caso contrário, continua com o mesmo tamanho. O tamanho do "cluster" é modificado até que *nsize* seja igual ao número de processadores disponíveis no processo.

Quando termina o processo?

O processo termina quando a porcentagem de transições aceitas for menor que um valor estipulado; por exemplo, quando o número de propostas aceitas for menor ou igual a 1,5% do total.

3.6.3 Testes em Sistemas Pequenos, Médios e de Grande Porte

As provas preliminares foram feitas sobre o sistema Garver e Sul brasileiro; em ambos casos se empregou temperaturas T_o calculadas com a equação (3.9). O número de tentativas iniciais $\mu.N_l$ foram estimadas entre $2.N_l$ e $6.N_l$, e um incremento, ao finalizar cada nível de temperatura, de 20%. O fator de decremento da temperatura se estimou entre 0,6 a 0,8.

Baseados na informação anterior, foram feitas diferentes análises; destas se conclui que, a partir de determinado valor de cadeia e decremento da temperatura, os valores ótimos dos sistemas estudados se obtém com maior freqüência. Dos resultados anteriores, se optou por selecionar uma cadeia inicial de (3 a 4) *nl* para o sistema Garver, de (4 a 5) *nl* para o sistema Sul brasileiro e de (5 a 6) *nl* para o sistema Norte Nordeste, e um fator de decremento de temperatura ao finalizar cada ciclo de 0,75.

Sistema de 06 Barras de Garver com Redespacho

Os dados deste sistema de 6 barras e 15 linhas candidatas aparecem no Apêndice A. A topologia inicial é mostrada na Figura 3.6. A configuração inicial é gerada de maneira aleatória. Os parâmetros empregados neste caso foram os seguintes: $\rho = 1,2$; $\beta = 0,75$; $\alpha = 10$; $\mu = 4$; $T_o = 460$; $w_o = 0$. No algoritmo da divisão a temperatura é mudada após efetuar *np* sub-cadeias, isto é, uma cadeia completa.

Na solução obtida foram realizados 4 adições de linhas: $n_{35} = 1$ e $n_{46} = 3$, com um investimento de 110.

- **Resultados obtidos com o sistema Garver empregando a técnica da divisão**

A Tabela 3.1 apresenta as estatísticas obtidas com a técnica da divisão para o sistema Garver com redespacho; baseados nestas, se determina o valor da moda da amostra.

Tabela 3.1: **Estatísticas do sistema Garver**

2 processadores		4 processadores	
Nº de iter.	Tempo(min)	Nº de iter.	Tempo(min)
2.169	16,6	1.568	12,6
1.464	11,6	1.178	9,7
2.171	16,6	1.262	10,2
2.634	20,0	1.458	11,8
2.171	16,6	1.566	12,7
2.173	16,8	1.019	8,4

A Tabela 3.2 apresenta o número de iterações e tempo gasto pela técnica da divisão para diferentes números de processadores.

Tabela 3.2: Técnica da divisão - Sistema Garver

Nº de proces.	Nº de iter.	Tempo (min)
1	5.253	41,3
2	2.171	16,6
4	1.262	10,2
8	971	7,6
16	583	4,6

A qualidade do processamento paralelo é medida em termos do "speedup". Este fator indica o quão rápido é um processo em termos de tempo quando são empregadas várias máquinas para executar uma operação de maneira conjunta, quando comparado com o melhor algoritmo serial.

O "speedup" é calculado da seguinte maneira :

- "speedup" = Tempo de execução do melhor algoritmo serial/Tempo de execução paralelo
- eficiência = "speedup"/número de processadores

Na Tabela 3.3 são apresentados valores de: tempo, "speedup" e eficiência, quando são empregados diferentes número de processadores.

Tabela 3.3: "Speedup" e eficiências do sistema Garver - técnica da divisão

Nº de Nós	Tempo total [min.]	*"Speedup"*	Eficiência
1	41,3	1,0	100%
2	16,6	2,5	125%
4	10,2	4,0	101%
8	7,6	5,4	68%
16	4,6	9,0	56%

- **Resultados obtidos com o sistema Garver empregando a técnica híbrida**

Na Tabela 3.4 apresenta-se o número de iterações e o tempo gasto, bem como os valores de "speedup" e eficiências para o sistema Garver com redespacho, quando o mesmo foi estudado empregando a técnica híbrida para diferentes números de processadores.

Tabela 3.4: "Speedup" e eficiências do sistema Garver - técnica híbrida

Nº de proces.	Nº de iter.	Tempo (min)	*"Speedup"*	Eficiência
1	5.253	41,3	1,0	100%
2	2.736	23,7	1,7	87%
4	1.506	14,4	2,9	72%
8	1.222	12,5	3,3	41%
16	616	8,1	5,1	32%

Sistema Sul Brasileiro sem Redespacho

Os dados deste sistema de 46 barras e 79 linhas candidatas aparecem no Apêndice A. A topologia inicial é mostrada na Figura 3.9.

A configuração inicial é gerada de maneira aleatória. Os parâmetros empregados neste caso foram os seguintes : $\rho = 1,2$; $\beta = 0,75$; $\alpha = 1.000$; $\mu = 4$; $T_o = 100.000$; $w_o = 0$. No algoritmo da divisão a temperatura é mudada após efetuar np sub-cadeias, isto é uma cadeia completa.

Na solução final o processo levou a um investimento de US$ 154.420.000,00 que corresponde a adição das seguintes 16 linhas: $n_{20-21} = 1$, $n_{42-43} = 2$, $n_{46-06} = 1$, $n_{19-25} = 1$, $n_{31-32} = 1$, $n_{28-30} = 1$, $n_{26-29} = 3$, $n_{24-25} = 2$, $n_{29-30} = 2$ e $n_{05-06} = 2$. Na Figura 3.11 é apresentada a configuração ótima.

- **Resultados obtidos com o sistema Sul empregando a técnica da divisão**

Na Tabela 3.5, apresenta-se as estatísticas obtidas com o algoritmo da divisão para o sistema Sul sem redespacho; baseado nestas se determina o valor da moda amostral.

Tabela 3.5: Estatísticas do sistema Sul sem redespacho

2 processadores		4 processadores		8 processadores	
Nº de iter.	Tempo(min)	Nº de iter.	Tempo(min)	Nº de iter.	Tempo(min)
7.669	335	4.249	189	8.164	361
9.355	405	7.622	330	9.192	410
9.365	410	8.208	354	9.498	414
11.382	487	9.391	402	10.366	452
11.392	500	9.392	402	10.669	474
11.397	503	11.511	497	10.358	452
13.821	602	12.344	544		
16.740	725	15.058	660		
16.742	728				

Na Tabela 3.6 apresenta-se os valores com o número de iterações e tempo gasto pela técnica da divisão, quando foi empregado diferente número de processadores.

Tabela 3.6: Técnica da divisão - sistema Sul sem redespacho

Nº de proces.	Nº de iter.	Tempo (min)
1	40.514	1.790
2	11.387	503
4	9.396	402
8	8.164	361
16	6.445	285

Na Tabela 3.7 são apresentados valores de: tempo, "speedup" e eficiência, quando são empregados diferente número de processadores.

Tabela 3.7: "Speedup" e eficiências do sistema Sul - técnica da divisão

Nº de Nós	Tempo total [min.]	*"Speedup"*	Eficiência
1	1.790	1,0	100%
2	503	3,6	178%
4	402	4,5	112%
8	361	5,0	62%
16	285	6,3	39%

- **Resultados obtidos com o sistema Sul sem redespacho empregando a técnica híbrida**

Na Tabela 3.8 apresenta-se os valores com o número de iterações e tempo gasto pela técnica híbrida quando foi empregado diferente número de processadores.

Tabela 3.8: Técnica híbrida - sistema Sul sem redespacho

Nº de proces.	Nº de iter.	Tempo (min)
1	40.514	1.790
2	14.394	672
4	9.374	446
8	6.955	323
16	5.645	285

Na Tabela 3.9 são apresentados valores de: tempo, "speedup" e eficiência quando são empregados diferente número de processadores.

Tabela 3.9: "Speedup" e eficiências do sistema Sul - técnica híbrida

Nº de Nós	Tempo total [min.]	*"Speedup"*	Eficiência
1	1.790	1,0	100%
2	672	2,7	133%
4	446	4,0	100%
8	323	5,5	69%
16	285	6,3	39%

Comentários a respeito dos resultados obtidos com os sistemas Garver e Sul Brasileiro

Tal como se menciona na teoria, tanto o algoritmo da divisão quanto o de "cluster" não alteram as propriedades básicas de convergência de "simulated annealing".

Com base na teoria, conclui-se que o algoritmo "cluster" não é eficiente para temperaturas altas, porém a medida que a temperatura diminui este algoritmo ganha em efetividade; por isso se implementou a técnica híbrida, a qual inicia o processo com o algoritmo da divisão e após vários ciclos de temperatura troca pelo algoritmo "cluster" que finaliza o processo.

Alguns resultados mostraram que a temperatura de chaveamento no algoritmo híbrido, para passar do algoritmo da divisão ao algoritmo "cluster", está em um intervalo de (15 - 22)% da temperatura inicial. Quando o chaveamento se efetua a temperaturas superiores, o algoritmo híbrido pode levar a soluções locais de qualidades não muito boas.

Efetuadas diferentes simulações, observou-se que a probabilidade de obter as soluções globais se fez mais alta ao aumentar o tamanho da cadeia de Markov, assim os sistemas Garver e Sul brasileiro foram estudados com longitudes de cadeias N_k entre $[2*nl\ ;\ 4*nl]$; bem como o valor de $\beta = 0,70$ passou a ser de $\beta = 0,75$ para os dois sistemas.

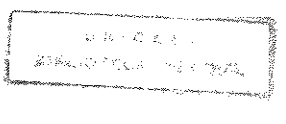
As subcadeias de Markov são calculadas como N_k/np, assim, na medida em que se aumenta o número de processadores as subcadeias ficam mais compridas; este fato, afeta o estado de quase-equilíbrio ao qual deverá chegar o processo em cada nível de temperatura. Por isto, deverá limitar-se o número de processadores ou aumentar a longitude da cadeia N_k. O aumento do tamanho da cadeia tem a desvantagem de incrementar o número de iterações fazendo o processo menos eficiente.

Tudo indica que a eficiência decresce se são usados um número de processadores muito grande, assim Garver poderia ser analisado com boa eficiência até com quatro processadores e o sistema Sul brasileiro até com oito processadores.

Para a análise de sistemas de grande porte recomenda-se gerar configurações iniciais, empregando-se configurações obtidas com o algoritmo **SA** seqüencial ou com os algoritmos aproximados, tais como: metodologia modificada de Garver, metodologia do mínimo esforço e metodologia de mínimo corte de carga.

Análises preliminares com o sistema Norte-Nordeste indicam que quando se tem mais processadores aumenta-se a chance de obter melhores soluções. Esta afirmação está baseada no estudo efetuado com os diferentes números de processadores.

Efetuadas as comparações entre os algoritmos da divisão e híbrido se conclui que o primeiro apresenta um melhor desempenho, dado pelas seguintes razões:



- É o algoritmo que apresenta as melhores facilidades de implementação,
- Na literatura em [19], o mesmo é recomendado como o melhor para implementação de uso intensivo de máquinas paralelas,
- Requer menos memória no "buffer" em relação ao algoritmo "cluster", isto indica que apresenta menos problemas para sua execução, tal como foi comprovado; bem como diminui a probabilidade de incorrer em erros devido ao grande tráfego de dados na rede,
- Comparado em relação ao algoritmo híbrido apresenta resultados semelhantes, chegando a respostas semelhantes, em tempos muito próximos um do outro.
- No algoritmo híbrido o grande esforço computacional é efetuado pelo algoritmo da divisão, fornecendo na seqüência ao "cluster" uma estrutura da rede muito definida.

Sistema de Grande Porte Norte-Nordeste Brasileiro - Plano 2008

O sistema Norte-Nordeste está constituído por 89 barras e 183 linhas; os dados da rede podem ser observados no Anexo A. A rede inicial é apresentada na Figura 3.12, onde as linhas grossas representam o sistema existente.

Baseados nas simulações com os sistemas Garver e Sul brasileiro e após efetuar análises preliminares com a rede Norte-Nordeste, foram selecionados os seguintes parâmetros de controle: $\beta = 0{,}75$, $\rho = 1{,}2$, $\mu = 6$, temperatura inicial T_o em um intervalo de [50.000 - 60.000].

Como soluções iniciais foram selecionadas configurações obtidas com o algoritmo **SA** seqüencial e que apresentavam valores de custo inferiores a US$ 2.750.000.000,00.

Para selecionar o algoritmo de paralelização que seria empregado na análise do sistema Norte-Nordeste, foram analisados os algoritmos da divisão, "cluster" e híbrido. Baseados nos resultados obtidos com os sistemas Garver e Sul e nos resultados preliminares obtidos com o sistema Norte-Nordeste foi selecionado o algoritmo da divisão. Em ítem anterior são expostas algumas razões que justificam a seleção do algoritmo da divisão.

A melhor solução obtida com um custo de investimento de US$ 2.630.290.000,00 e sem apresentar corte de carga foi a seguinte:

$n_{01-02} = 1$, $n_{02-04} = 1$, $n_{04-05} = 4$, $n_{04-06} = 4$, $n_{04-68} = 1$, $n_{05-56} = 1$, $n_{05-58} = 4$, $n_{06-07} = 2$, $n_{07-08} = 2$, $n_{08-17} = 2$, $n_{12-17} = 1$, $n_{13-15} = 4$, $n_{14-59} = 1$, $n_{15-16} = 3$, $n_{15-46} = 1$, $n_{16-44} = 6$, $n_{17-18} = 3$, $n_{18-50} = 11$, $n_{18-74} = 3$, $n_{21-57} = 2$, $n_{22-37} = 1$, $n_{22-58} = 1$, $n_{24-43} = 1$, $n_{25-55} = 3$, $n_{26-54} = 2$, $n_{30-31} = 3$, $n_{30-63} = 3$, $n_{31-34} = 2$, $n_{35-46} = 1$, $n_{35-51} = 1$, $n_{36-46} = 2$, $n_{40-45} = 1$, $n_{40-46} = 2$, $n_{41-64} = 2$, $n_{42-44} = 4$,

$n_{42-85} = 2$, $n_{43-55} = 2$, $n_{43-58} = 2$, $n_{48-49} = 1$, $n_{49-50} = 4$, $n_{52-59} = 1$, $n_{53-86} = 1$, $n_{54-58} = 2$, $n_{54-63} = 1$, $n_{56-57} = 1$, $n_{61-64} = 1$, $n_{61-85} = 2$, $n_{61-86} = 1$, $n_{67-68} = 1$, $n_{67-69} = 1$, $n_{67-71} = 3$, $n_{71-72} = 1$, $n_{72-73} = 1$, $n_{73-74} = 1$.

Os sistemas de grande porte se caracterizam por apresentar uma grande quantidade de ótimos locais, neste caso foram obtidas um grande número de soluções com valores de investimento muito próximos da melhor solução encontrada. Também foram obtidas muitas soluções com pequenos cortes de carga; assim, por exemplo, obteve-se uma solução com um valor de investimento ligeiramente inferior: US$ 2.627.265.000,00 com 6 MW de corte de carga. Quando o algoritmo foi relaxado nos primeiros 100 MW, isto é, cortes de carga inferiores a 100 MW são aceitas como factíveis, foram obtidas soluções com menor custo de investimento e que apresentam cortes de carga perto dos 100 MW; ou seja, a melhor solução encontrada apresenta um custo de investimento de US$ 2.600.745.000,00 e um corte de carga de 98 MW.

Na Tabela 3.10, apresenta-se estatísticas obtidas para o sistema Norte-Nordeste com um só nó.

Tabela 3.10: Estatísticas do sistema Nordeste com um só nó

1 processador	
Nº de iter.	Tempo(horas)
113.735	249
166.130	363
241.580	528

Na Tabela 3.11, apresenta-se os valores com o número de iterações e tempo gasto pela técnica da divisão quando foi empregado diferente número de processadores.

Tabela 3.11: Técnica da divisão - sistema Nordeste - sem corte de carga

Nº de proces.	Nº de iter.	Tempo (horas)	Invest. (10^3US$)
1	201.580	441,0	2.668.242,00
2	134.522	252,0	2.666.373,00
4	50.520	94,7	2.680.581,00
8	26.932	60,5	2.657.091,00
16	14.045	30,0	2.636.419,00
32	9.597	18,0	2.632.825,00

Na Tabela 3.12, são apresentados valores de: tempo, "speedup" e eficiência, quando são empregados diferente número de processadores.

Tabela 3.12: "Speedup" e eficiências do sistema Norte-Nordeste - técnica da divisão

Nº de proces.	Tempo total [horas.]	*"Speedup"*	Eficiência
1	441,0	1,00	100%
2	252,0	1,75	87%
4	94,7	4,65	116%
8	60,5	7,29	91%
16	30,0	14,7	91%
32	18,0	24,5	76%

Na Tabela 3.13, é apresentado a evolução do processo do algoritmo da divisão paralelo e no qual foram empregados 32 processadores; neste se pode observar o processo nos diferentes níveis de temperatura.

Tabela 3.13: Técnica da divisão - sistema Nordeste - 32 processadores

Temp	Nº de iter.	Horas	Corte de Carga (MW)	Invest. (10^3US$)	f.o. (10^3US$)
40.000	911	1,9	60,0	2.623.020,00	2.683.020,00
40.000	946	2,0	75,0	2.625.796,00	2.700.796,00
16.875	4.838	10,1	6,0	2.626.825,00	2.632.825,00
16.875	5.014	10,5	0,0	2.630.290,00	2.630.290,00
12.656	7.068	14,8	0,0	2.631.500,00	2.631.500,00
12.656	7.202	15,1	6,0	2.627.265,00	2.633.265,00
Congelo	7.700	16,1	0,0	2.630.290,00	2.630.290,00

3.7 CONCLUSÕES

O **SA** é uma técnica de otimização baseada em conceitos da Mecânica Estatística e com um grande número de aplicações em problemas de engenharia elétrica. Esta nova técnica foi validada através de comparações com métodos aproximados e técnicas de otimização baseados em decomposição de Benders. Para sua validação se empregam três sistemas; o primeiro é um sistema pequeno de 6 nós e 15 linhas chamado sistema de Garver, o segundo é um sistema de médio porte, rede Sul brasileira, de 46 nós e 79 linhas, e o terceiro é um sistema de grande porte, rede Norte-Nordeste, de 87 nós e 183 linhas. O algoritmo de **SA** proposto, tanto em sua versão seqüencial como paralela, encontra as soluções ótimas dos problemas pequenos e médios; para o caso de sistemas de grande porte, como a rede Norte-Nordeste onde a solução ótima não é conhecida, chega-se a uma solução com um custo 7% e 8% menores nas versões seqüêncial e paralela respectivamente, quando comparados com a melhor solução conhecida.

O problema do planejamento de sistemas de transmissão é um típico problema combinatorial onde o número de possíveis configurações a ser analisadas cresce exponencialmente com o tamanho da rede. Este é um tipo de problema de programação inteira onde frações de linha não são permitidas. Em estudos prévios, apresentados em [35, 12], se empregam métodos aproximados para resolver o problema da expansão da transmissão para sistemas de grande porte; são métodos computacionalmente rápidos, porém apresentam qualidades de resposta um pouco inferiores às apresentadas pelo algoritmo **SA**. Também em estudos apresentados em [11, 27], onde se aplicam métodos matemáticos baseados em decomposição de Benders e enumeração implícita obtém-se soluções ótimas para problemas pequenos e médios.

Foi desenvolvido um algoritmo de **SA** seqüencial com diferentes programas de resfriamento. Cada programa de resfriamento teve seus parâmetros de controle otimizados de forma empírica. O algoritmo encontra facilmente as configurações ótimas de sistemas de pequeno e médio porte. Entretanto, para sistemas de grande porte o esforço computacional necessário é relativamente elevado, mas em termos da qualidade das configurações obtidas, o algoritmo de **SA** encontra configurações melhores que os métodos aproximados, especialmente em sistemas de grande porte.

A técnica **SA** paralelo foi suficientemente testada; neste caso foram implementadas as versões dos algoritmos da divisão e híbrido. Os resultados obtidos pelas duas versões para sistemas pequenos e médios, foram satisfatórios. Já para sistemas de grande porte foi selecionado o algoritmo da divisão por questão de facilidade na implementação e de melhor comportamento no processamento, já que a comunicação neste só é feita ao terminar uma cadeia de Markov. No outro método, a comunicação é feita cada vez que uma configuração é aceita; isto congestiona o sistema de comunicação paralelo tornando difícil a manipulação dos dados. Este problema cresce na medida em que se analisam sistemas maiores, ao considerar mais processadores e maior quantidade de informação.

No processamento paralelo se obtém reduções significativas em tempo de computação e melhoramento da qualidade da resposta comparado com a versão seqüencial, quando são analisados sistemas de grande porte.

A paralelização de **SA** é feita de maneira que não afete a estrutura seqüencial do algoritmo **SA**. Neste, são estudados os diferentes algoritmos paralelos, variando os parâmetros de controle, temperatura inicial T_o, coeficiente de variação no nível da temperatura β, comprimento da cadeia de Markov N_k, coeficiente de variação no tamanho da cadeia ρ e para diferente número de processadores np.

Os parâmetros de controle são selecionados tal que seja aumentada a chance de obter o ótimo global. Para cada nível de temperatura, a cadeia de Markov de comprimento N_k é dividida entre os np processadores, correspondendo a cada processador analisar N_k/np configurações.

Os resultados apresentados anteriormente correspondem a testes realizados com uma versão inicial do algoritmo **SA**. Logicamente é possível melhorar o desempenho de um algoritmo combinatorial calibrando adequadamente seus parâmetros de controle e introduzindo melhoras adicionais nas partes básicas do mesmo, assim como nas partes especificas do problema de planejamento; por exemplo, melhorando o mecanismo de definição de vizinhança e a geração eficiente de novos caminhos.

Posteriormente foram realizados testes com um algoritmo **SA** melhorado, produzindo configurações de melhor qualidade. Assim, por exemplo, usando **SA** seqüencial, foi encontrada uma configuração para o sistema Norte-Nordeste com um investimento igual a 2.625,663 milhões de dólares, sem corte de carga.

Capítulo 4

PLANEJAMENTO DE SISTEMAS DE TRANSMISSÃO USANDO ALGORITMOS GENÉTICOS (AG)

4.1 INTRODUÇÃO

Neste capítulo é apresentada uma nova metodologia para o planejamento de sistemas de transmissão baseada nas idéias fundamentais dos algoritmos genéticos (**AG**) e algumas adaptações adicionais para incorporar as características específicas do problema de planejamento. A metodologia desenvolvida apresenta soluções (configurações) de qualidade ligeiramente melhores que as obtidas com algoritmos de **SA** e com uma redução adicional do esforço computacional.

O algoritmo genético contorna adequadamente os problemas de ótimos locais pois trabalha com uma população (conjunto de configurações), o que permite visitar regiões muito distantes na procura de caminhos promissores. Simula o princípio de seleção natural que acontece na natureza e que leva à sobrevivência dos indivíduos melhores dotados na luta com outros indivíduos por recursos escassos. Assim, o algoritmo genético parte de uma população inicial, realiza um processo de seleção das melhores configurações ("strings") com direito a participar na geração dos novos descendentes ("allsprings"), faz o cruzamento ("crossover") entre estas melhores configurações e finalmente um processo de mutação determina os elementos da nova geração.

O algoritmo genético simula o processo de seleção natural para resolver um problema de otimização combinatorial. O mecanismo de busca do algoritmo genético é de natureza probabilística, mas adequadamente direcionada. Os elementos ou configurações da população inicial geralmente são obtidos de forma aleatória. Assim, cada elemento ou gene

de uma configuração ("strings") é obtida aleatoriamente. Também, cada "strings" tem uma função objetivo que é usada no processo de seleção para determinar o número de descendentes a que tem direito essa configuração. Portanto, o número de descendentes que de fato é designado a cada configuração é estatisticamente proporcional a qualidade da correspondente função objetivo ("fitness") e finalmente as operações de *"crossover"* e *mutação* terminam gerando a nova população. Cada parte deste processo de obtenção de uma nova geração (seleção, "crossover" e mutação) é controlada de maneira probabilística. Um algoritmo genético é completamente definido depois de escolher: a população inicial, os parâmetros de controle (tamanho da população, taxa de "crossover" e taxa de mutação) e o critério de parada.

Com relação a combinação dos **AG** com outras heurísticas, muitos pesquisadores têm optado por adotar metaheurísticas como **SA** e **Busca Tabu** com um método baseado no uso de uma população e usando processos de recombinação como os dos **AG** [54].

Este capítulo é organizado como segue. Inicialmente é apresentada uma descrição teórica dos **AG**, depois é discutido os princípios básicos dos **AG** aplicados ao problema de planejamento de sistemas de transmissão. Neste último, se inclui:

- A representação e codificação do problema de planejamento,
- O mecanismo geracional: seleção - "crossover" - mutação,
- Geração da população inicial, e
- Parâmetros de controle e critérios de parada.

Em seguida são apresentadas as implementações especializadas realizadas nos **AG**. Nestas, se inclui:

- Taxa de mutação variável,
- Taxa de mutação controlada por **SA**,
- Seleção limitada,
- Implementação dos *"building blocks"* através dos caminhos novos,
- Formas especializadas de geração da população inicial, e
- Parâmetro α variável.

A seguir, é descrito o algoritmo implementado e apresentados testes com sistemas de pequeno e médio porte. Também são apresentados resultados obtidos com um sistema de grande porte, o sistema Norte-Nordeste brasileiro. Finalmente são mencionadas as conclusões mais relevantes.

4.2 O ALGORITMO GENÉTICO: TEORIA

O algoritmo genético é uma metodologia usada para resolver problemas de otimização combinatorial que apresentou um grande sucesso na última década na resolução de problemas de grande porte em áreas muito diversas. O algoritmo genético foi inicialmente formulado por Holland [3] baseado no princípio de seleção natural que acontece na natureza e que fornece maiores chances de sobrevivência aos indivíduos melhores dotados. Matematicamente, pode-se considerar o algoritmo genético como uma técnica de otimização combinatorial que tem uma alta probabilidade de encontrar a solução ótima global de problemas grandes e complexos e que apresentam muitas soluções ótimas locais.

O algoritmo genético está baseado no princípio de seleção natural. Na natureza os indivíduos melhores dotados tem maiores chance de sobrevivência e a capacidade de adaptação a um meio mutante é fundamental na sobrevivência de indivíduos e espécies. Foi baseada nesta lei natural que as espécies sobreviveram e evoluíram na Terra. As características específicas de um indivíduo determina sua capacidade de sobrevivência e, em última instância, essa capacidade específica é determinada pelo conteúdo génetico do indivíduo; isto é, pela unidade elementar chamada *gene* na biologia. Assim, nas mudanças do material genético das espécies acontece a evolução das mesmas. A seleção natural leva à sobrevivência dos indivíduos melhores dotados e no processo de reprodução estes indivíduos transmitem aos descendentes os melhores genes; por outro lado, os indivíduos menos dotados morrem no processo de competição por espaço, alimentos, etc. Assim, o princípio de seleção natural permite que somente os indivíduos melhores dotados gerem descendentes. Foi assim que aconteceu a evolução das espécies. A evolução acontece quando os cromossomos de dois indivíduos bem dotados se recombinam durante a reprodução. Neste processo acontece uma troca de material genético que pode levar à geração de um indivíduo muito bem dotado ao receber o melhor material genético dos pais. Este processo repetido leva à evolução, isto é, uma seleção com a sobrevivência dos indivíduos melhores dotados e a correspondente reprodução (com troca de material genético) entre os mesmos. Adicionalmente, no processo de evolução, existe um fenômeno chamado *mutação*, que é uma mudança do código genético dos indivíduos como conseqüência de sua interação com o meio ambiente.

No algoritmo genético o processo de otimização se inicia com a seleção de uma população, isto é, um conjunto de soluções (configurações) candidatas. Cada configuração é qualificada pelo valor da função objetivo que apresenta. Geralmente a codificação de uma configuração é realizada em sistema binário. Todos os elementos da população são classificados pela qualidade de sua correspondente função objetivo. Assim, cada elemento recebe uma determinada probabilidade para gerar os elementos (configurações) na geração seguinte. Os elementos melhores qualificados ("higher fitness") neste processo têm, no sentido probabilístico, maior probabilidade de participar na geração dos elementos da nova população. Essa nova geração é obtida com as operações de cruzamento ("cros-

sover") e mutação. Dessa forma, as melhores configurações são cruzadas para gerar os elementos da nova população e, nesse processo, o cruzamento leva a uma transferência de material genético entre as melhores configurações. Neste processo, eventualmente devem aparecer elementos da nova geração (filhos) que recebem o melhor material genético das configurações melhor qualificadas, produzindo novas configurações de excelente qualidade. A mutação termina a formação da nova população. Na natureza, a mutação produz a regeneração da perda de material genético. No algoritmo genético, a mutação produz uma modificação esporádica e aleatória de algum gene de um elemento (configuração) da população. Resumindo, pode-se dizer que os **AG** são uma técnica de otimização combinatorial que inicia o processo de otimização com uma população inicial, faz um ordenamento seletivo dos elementos da população avaliando a qualidade da função objetivo e gera uma nova população num processo de três etapas: na primeira é realizada a seleção (onde à cada configuração é designada um número determinado de descendentes), na segunda etapa é realizado o "crossover" entre os elementos melhores qualificados e com direito a gerar descendentes; e na etapa final, é realizada uma mutação aleatória para obter a nova população.

O algoritmo genético em sua forma mais simples é composto pelos seguintes componentes:

- População inicial,
- Mecanismo de codificação do problema,
- Parâmetros de controle,
- Avaliação da qualidade da função objetivo,
- Seleção e operadores genéticos do ciclo geracional.

4.2.1 População Inicial

A população inicial pode ser gerada de várias maneiras. Uma delas é gerar populações iniciais de maneira aleatória; uma outra, poderá ser a geração das populações iniciais usando algoritmos aproximados e cujas respostas se obtém em tempos curtos, com qualidades de respostas relativamente boas.

Fazer evoluir uma população gerada inicialmente de maneira aleatória, até chegar a ter configurações perto da solução ótima, é um bom teste para conhecer como está funcionando a implementação dos **AG**, já que teoricamente as características essenciais e críticas da solução final devem ser conseqüência do processo evolutivo e não das características dos métodos usados para gerar a população inicial, como é o caso de sistemas de pequeno e médio porte. Já em sistemas de grande porte, o processo necessita de um bom programa inicializador pelas razões que mais adiante serão expostas.

4.2.2 Mecanismo de Codificação do Problema

Em seus trabalhos, Holland representa os cromossomos (configurações) por "strings" binários, isto é, listas de zeros e uns. Este tipo de codificação tem sido empregado em diferentes aplicações; inclusive em muitas aplicações onde não é óbvio seu emprego. Apesar disto, existem uma grande variedade de aplicações onde essa representação não é efetiva; assim, alguns pesquisadores têm começado a explorar outras formas de representação; em geral aquelas que tem relação com aplicações práticas dos **AG**.

Alguns tipos de problemas são mais facilmente representáveis em codificação binária que outros e, portanto melhor condicionados para serem resolvidos por um algoritmo genético. Existem outros cujas variáveis são binárias, assim não existe nada a modificar ou adequar. Também existem problemas onde as variáveis de decisão são inteiras com limitante superior. Neste caso, deve-se fazer uma representação binária destas variáveis, o que aumenta o tamanho do problema. O problema mais complicado de resolver em termos de representação das variáveis acontece quando as variáveis de decisão são contínuas. Neste caso, deve-se fazer a representação binária de uma variável contínua variando em um intervalo.

No problema de planejamento de sistemas de transmissão as variáveis de decisão são as linhas adicionadas em cada caminho candidato e são portanto variáveis inteiras e, no caso mais geral, cada variável de decisão pode ser representada por 4 dígitos binários que equivale a adicionar mais de 8 linhas em cada caminho candidato. Assim, no problema de planejamento de sistemas de transmissão o problema de representação é resolvido de uma maneira relativamente simples. Entretanto, para sistemas de grande porte como o Norte-Nordeste brasileiro, esta representação levaria a representar cada elemento (configuração) da população por um vetor de 732 elementos binários o que tornaria computacionalmente cansativa a implementação do "crossover" e a preparação dos vetores para o cálculo da função objetivo. Mais adiante são apresentadas idéias alternativas de representação fundamentadas em três idéias básicas: representação binária, representação por bits independentes e representação decimal.

4.2.3 Parâmetros de Controle

As características básicas, assim como a eficiência de um algoritmo genético, são determinadas pela definição dos *parâmetros de controle*, pela forma de escolha da população inicial e pelo critério de parada adotado. Na formulação básica do algoritmo, os parâmetros de controle são: o tamanho da população, a taxa de "crossover" e a taxa de mutação. Portanto, os parâmetros de controle juntamente com o mecanismo de geração da população inicial e o critério de parada definem praticamente a qualidade de um algoritmo genético.

Valores típicos usados na literatura especializada para os parâmetros de controle são [2]:

- tamanho de população entre 30 e 200.
- taxa de "crossover" entre 0,5 e 1,0. e
- taxa de mutação entre 0,001 e 0,050.

4.2.4 Avaliação do Nível de "Fitness"

A função empregada para atribuir os valores "fitness" aos membros da população deverá possuir boa sensibilidade, tal que consiga fazer uma boa discriminação destes últimos. Assim por exemplo, os resultados da avaliação de uma função objetivo pode atribuir um valor de 1,09 a um indivíduo muito bom; por outro lado, pode dar um valor de 0,96 a um indivíduo muito ruim. Se estes valores forem empregados para selecionar os indivíduos na fase de cruzamento, seria necessário muito tempo de computação para que os descendentes do primeiro influam mais que os do pior indivíduo.

A normalização se faz trocando a referência dos valores das funções, isto é, subtraindo um valor preestabelecido de cada valor das funções. Desta maneira os indivíduos bons participarão em maior porcentagem na formação da nova geração.

Esta normalização deverá privilegiar as melhores configurações, porém sem descartar o material genético presente na população, já que se a normalização estiver influenciada fortemente pelo melhor indivíduo, a busca ficará em uma região do espaço de configurações perto daquela que influenciou na escolha. Caso contrário, se não existir influência das melhores soluções, pode-se ter uma busca demasiada lenta.

Assim, como no exemplo anterior, para um indivíduo bom se atribui um valor de 1,09 e para um indivíduo ruim um valor de 0,96.

O critério sem normalização é:

	$F(x_1)$	$F(x_2)$	Δ
Valores atribuídos aos indivíduos	1,09	0,96	0,13
Participação na nova população	53%	47%	13%

O critério com normalização se obtém subtraindo 0,80 de cada valor, ou seja:

	$F(x_1)$	$F(x_2)$	Δ
Valores atribuídos aos indivíduos	0,29	0,16	0,13
Participação na nova população	64%	36%	81%

Pode-se observar que o critério com normalização produz uma melhor selectividade, contornando em parte o problema de valores de $F(x)$ muito próximos. Existem ainda outras formas de normalização que poderiam ser implementadas para contornar este problema.

4.2.5 Seleção e Operadores Genéticos do Ciclo Geracional

O mecanismo de *seleção* tenta modelar o mecanismo de sobrevivência dos indivíduos melhores dotados na natureza. Assim, no algoritmo genético, uma vez definida a população, determina-se a função objetivo ou seu equivalente "fitness" de cada elemento ou configuração da população e aquelas com funções objetivos de melhor qualidade recebem uma maior ponderação para participar na geração dos elementos da nova população.

A mais simples implementação de seleção é o chamado *esquema de seleção proporcional.* Neste esquema cada configuração tem direito a gerar um número de descendentes proporcional à qualidade da função objetivo. Assim, as configurações com maiores valores da função objetivo participarão na geração de muitos descendentes e aquelas de menor valor da função objetivo praticamente não participarão na geração de novas configurações.

O esquema de seleção proporcional é implementado usando o chamado *esquema de seleção da roleta.* No esquema da roleta cada configuração ("string") é alocada num setor circular da roleta, proporcional ao valor da correspondente função objetivo. Um processo aleatório determina um número de descendentes a que tem direito cada configuração. Entretanto, é possível substituir o esquema da roleta por mecanismos mais eficientes e rápidos que designe a cada configuração um número de descendentes proporcional a porcentagem determinada para cada configuração e fazendo um ajuste no final do processo para determinar um número de descendentes igual ao tamanho da população corrente. Uma vez terminado o processo de seleção e portanto conhecido o número de descendentes a que tem direito cada configuração da população corrente, deve-se implementar o cruzamento ou "crossover".

Na formulação original do algoritmo genético o *"crossover"* é a operação fundamental e crucial e à mutação é designada uma função secundária. Uma vez definida a seleção, escolhe-se um par de configurações que tem direito a gerar descendentes e este par é

submetido a um cruzamento para gerar dois novos descendentes. A escolha das duas configurações para "crossover" pode ser feita de forma aleatória ou de forma sistemática, iniciando o cruzamento com aquelas configurações que tem direito a um maior número de descendentes. Existem quatro tipos de "crossover": simples ("single-point"), com dois pontos ("two-point"), multiponto ("multi-point") e uniforme.

No "crossover" simples, escolhe-se aleatoriamente um ponto de cruzamento; assim, o material genético além desse ponto é trocado. Seja L o tamanho ou número de elementos de uma configuração; então, escolhe-se aleatoriamente um número entre 1 e L-1. Seja L_p o número aleatório selecionado, então as duas configurações escolhidas para "crossover" trocam a parte correspondente às primeiras L_p posições. Essas duas novas configurações, assim geradas, são as candidatas a configurações na nova população. No "crossover" de dois pontos, são selecionados de maneira aleatória dois pontos de cruzamento, então é trocado o material genético contido entre estes dois pontos. No processo de generalização do "crossover" pode-se implementar o chamado "crossover" multiponto e uniforme. Neste último caso, *cada elemento* (bit) é selecionado de maneira aleatória; começando pelo primeiro, é eleito aleatoriamente o primeiro pai que contribui com o primeiro bit ao primeiro filho. Na seqüência, o segundo filho recebe o bit do segundo pai. Este processo continua até que todos os bits sejam atribuídos. Cada tipo de "crossover" apresenta vantagens e desvantagens.

Em relação a operação de "crossover", os algoritmos genéticos definem a chamada *taxa de "crossover"* (ρ_c). Um algoritmo genético geral não procede ao cruzamento de todas as configurações escolhidas. A taxa de "crossover" determina probabilisticamente a porcentagem de configurações que serão submetidas efetivamente a cruzamento neste algoritmo. Assim uma parcela das configurações da população corrente passa integralmente a constituir a lista de novas configurações candidatas para a nova população sem "crossover". Portanto, uma vez escolhida as duas configurações candidatas, gera-se um número aleatório entre 0 e 1. Se o número aleatório gerado for maior que ρ_c, a taxa de "crossover", então as duas configurações passam diretamente para a lista de configurações candidatas sem "crossover" e, caso contrário, procede-se ao "crossover". Tipicamente os valores de ρ_c variam entre 0,9 e 1,0.

A operação de *mutação* termina com o processo de obtenção da nova geração da população. O processo de mutação implica modificar de 0 para 1 ou vice-versa um elemento (gene) de uma configuração. Este processo é controlado por um parâmetro denominado *taxa de mutação* (ρ_m). No algoritmo genético simples, a mutação é considerada como um operador secundário comparado com o "crossover" e tem a finalidade de restaurar a perda de material genético que pode acontecer na população. Assim, por exemplo, se na posição k todas as configurações ("strings") tem valor 0 na representação binária, e na configuração ótima o valor correto nessa posição é 1, então o "crossover" não pode regenerar um 1 nessa posição; entretanto a operação de mutação pode resolver essa dificuldade.

4.3 O ALGORITMO GENÉTICO BÁSICO APLICADO AO PROBLEMA DE PLANEJAMENTO

Muitos problemas de otimização combinatorial que são encontrados na prática são resolvidos usando métodos aproximados devido a grande dificuldade em encontrar sua solução ótima. Os métodos combinatoriais, como é o caso dos **AG**, foram projetados para lidar com este tipo de problema. Os algoritmos genéticos foram usados com muito sucesso para resolver problemas complexos em áreas muito diversas e também em várias aplicações de engenharia elétrica [4, 14]. Matematicamente, pode-se considerar o algoritmo genético como uma técnica de otimização combinatorial que tem uma alta probabilidade de encontrar a solução ótima de problemas grandes, complexos e que apresentam muitas soluções ótimas locais.

Este é o caso do planejamento da expansão das redes de transmissão de grande porte, que devido às restrições de integralidade (isto é, as linhas só assumem valores inteiros) se tornam um problema combinatorial, e é de difícil resolução, devido ao grande número de soluções ótimas locais possíveis. Tal é o caso da rede Norte-Nordeste brasileira, com 89 nós e 183 linhas, que chega a ter até $(11+1)^{183}$ possíveis soluções, isto é, supondo um máximo de 11 linhas candidatas por ramo.

No algoritmo genético aparece um conjunto de componentes que define o tipo e a qualidade do algoritmo. Estes componentes são divididos em quatro partes a fim de facilitar a apresentação, ou seja:

1. A representação e codificação do problema de planejamento,
2. O mecanismo geracional: seleção - "crossover" - mutação,
3. Geração da população inicial e
4. Parâmetros de controle e o critério de parada.

4.3.1 A Representação e Codificação do Problema

Representação

A função objetivo em (2.1) já teve uma transformação inicial, pois além da parcela correspondente ao custo das linhas adicionadas, foi incluída outra parcela correspondente

a uma penalidade correspondente ao corte de carga que produz uma determinada configuração. Assim, no algoritmo genético, uma configuração ("string") da população está constituída unicamente pelas variáveis de investimento n_{ij} e o efeito das outras variáveis contínuas de operação são consideradas implicitamente em conjunto como corte de carga no sistema. Portanto, do ponto de vista do algoritmo genético, as variáveis que formam uma configuração são inteiras, isto é, o número de linhas adicionadas em cada caminho candidato.

O valor da função objetivo determina a qualidade da configuração ou elemento de uma população. Assim, as configurações com valores da função objetivo ou seu equivalente "fitness" de melhor qualidade participam com maior freqüência na geração dos elementos da nova população ou geração. Na *formulação tradicional* dos algoritmos genéticos o problema a otimizar é formulado na forma de maximização. A *transformação* de um problema de minimização em um problema equivalente de maximização para usar adequadamente um algoritmo genético não é muito trivial pois ela pode comprometer a seletividade do processo de seleção. No problema de planejamento, a transformação do problema de minimização em um problema de maximização foi realizada de maneira parecida ao realizado em [5], assim:

$$min\ v \iff max\ [K - v]$$

em que K é uma constante que pode ser fixa ou variável durante o processo de otimização, e sua escolha é feita baseando-se nos valores das funções objetivo das configurações iniciais ou, melhor ainda, pode assumir o valor correspondente àquela função objetivo de maior valor em cada geração.

No caso em que o algoritmo inicie o processo com configurações obtidas por outros métodos, K pode ser fixo e assumir um valor em torno de 20% maior do valor da função objetivo de uma configuração típica usada na geração da população inicial. Quando a população inicial é gerada aleatoriamente K assume um valor variável em cada geração e assume um valor 5% maior que a pior configuração. Todas estas implementações visam manter a seletividade dos **AG**.

Outra forma alternativa, usado por exemplo em [14], que fornece resultados muito parecidos é fazer a seguinte transformação:

$$min\ v \iff max\ [1/v]$$

No presente trabalho foram implementadas ambas alternativas.

Codificação do Problema

No Algoritmo genético tradicional as variáveis do problema são codificadas em *sistema binário* pois desta forma permite uma melhor implementação das operações de "crossover" e mutação. No problema de planejamento, as variáveis que formam uma configuração são inteiras e, portanto, a codificação mais óbvia consistiria em representar a variável inteira em sistema binário. Entretanto, analisando o problema específico de planejamento, pode-se verificar que existem três formas de codificar uma configuração:

1. Codificação em sistema binário,
2. Codificação em bits independentes e
3. Manter a codificação decimal.

A Figura 4.1 ajuda a analisar estes 3 tipos de codificação.

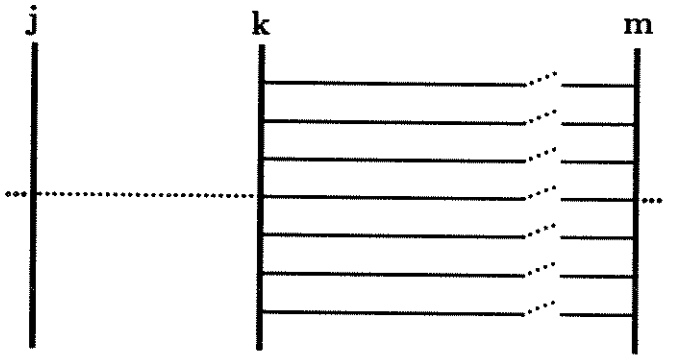


Figura 4.1: O problema de codificação

Na Figura 4.1, existe a possibilidade de adicionar até 7 linhas entre as barras k e m. Supõem-se, a seguir, que na configuração corrente foram adicionadas 3 linhas. A Figura 4.2 mostra a codificação da configuração somente para o caminho $k - m$ onde foram realizadas 3 adições. As três formas de codificação apresentam vantagens e desvantagens quando são considerados os aspectos mais importantes do processo.

Um aspecto importante é o *espaço necessário* para o armazenamento da configuração. No exemplo, a codificação tradicional em *sistema binário* usa 3 casas binárias, a representação por *bits independentes* usa 7 casas binárias e a *representação decimal* usa uma casa decimal. Um problema de grande porte, como o Norte-Nordeste brasileiro, tipicamente

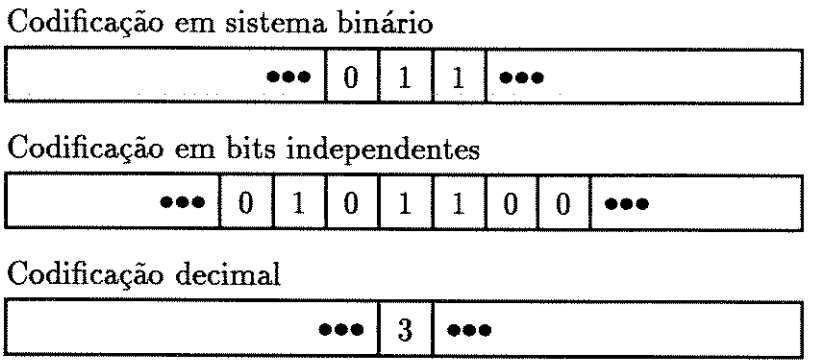


Figura 4.2: Diferentes formas de codificação

permite adicionar mais de 7 linhas por caminho candidato, existem em torno de 190 caminhos candidatos e uma população típica tem entre 160 e 200 configurações. Assim, entre as três alternativas existe uma diferença enorme em requisitos de memória necessárias para o armazenamento das configurações, mostrando a grande vantagem da codificação decimal. Adicionalmente, as duas primeiras formas de codificação requerem esforço computacional maior, pois cada configuração deve ser transformada em seu equivalente decimal para usar a subrotina de PL que determina o corte de carga correspondente a cada configuração.

Outro aspecto importante é que a codificação binária apresenta um problema conhecido como *penhasco de Hamming* ("Hamming cliffs") que significa que existe uma grande distância de Hamming para dois inteiros consecutivos. Por exemplo os números inteiros 3 e 4 são representados em sistema binário como 011 e 100. Assim um algoritmo genético deve transformar, usando "crossover" e mutação, todos os seus dígitos para passar de 3 para 4 linhas, as quais formam configurações vizinhas. Por outro lado, a codificação usando bits independentes e a codificação decimal não apresenta este problema.

Outro aspecto mais importante ainda, relacionado com a codificação escolhida, acontece na implementação do "crossover" e da mutação. A Figura 4.3 ilustra este aspecto que é crucial na implementação de um algoritmo genético eficiente. A Figura 4.3 apresenta, em codificação binária tradicional, duas configurações do caminho candidato $k-m$ da Figura 4.1. Assim, mostram-se duas configurações antes do "crossover", depois do "crossover" e depois de fazer a mutação. Antes do "crossover" as configurações indicam a alocação de 3 e 4 linhas respectivamente. Após o "crossover", as novas configurações indicam 7 e 0 linhas adicionadas respectivamente; portanto, *no ponto de cruzamento* não acontece uma troca de informação adequada (trocar 3 linhas com 4 linhas no caminho $k-m$). O que realmente acontece no ponto de cruzamento é uma mutação violenta (de 3 para 7 linhas e de 4 para 0 linhas). Igualmente, no caso da mutação, houve uma mutação na posição central da segunda configuração mas isto implica passar de 0 a 3 linhas indicando também uma mutação brusca. Este problema não acontece com as outras duas formas de codificação.

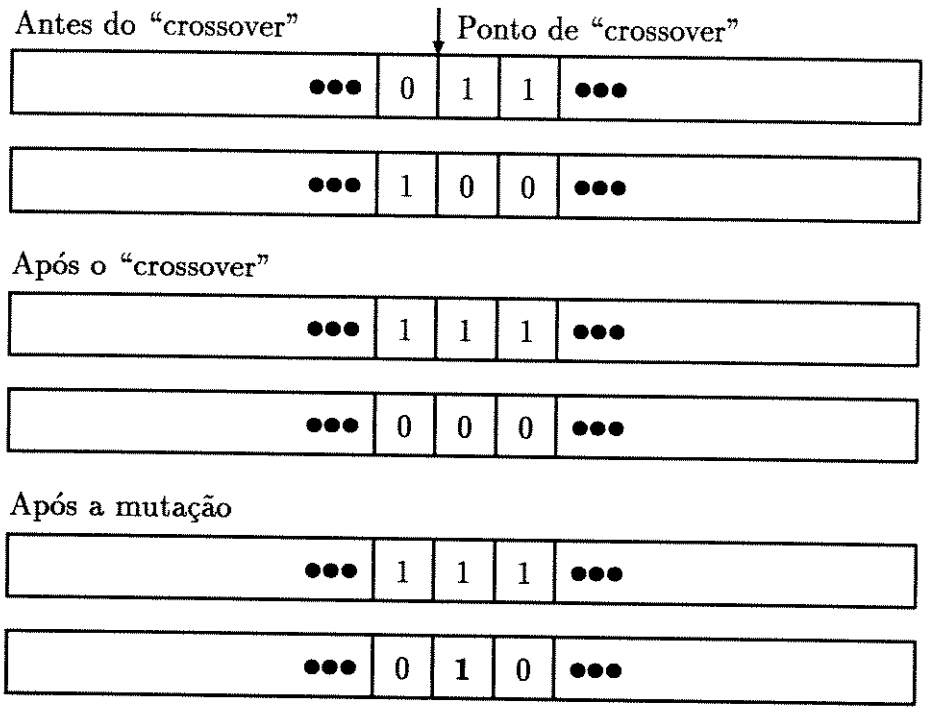


Figura 4.3: "Crossover" e mutação

Um aspecto final em relação a codificação é que na codificação binária e por bits independentes, as operações de "crossover" e mutação são implementadas, do ponto de vista computacional, de uma maneira trivial, enquanto que na representação decimal, deve-se redefinir a operação de mutação. Também é possível realizar uma codificação binária de forma implícita, isto é, manter uma codificação decimal mas realizar as operações de "crossover" e mutação simulando a codificação binária.

As observações levam a pensar que uma codificação decimal com um aumento da taxa de mutação seria equivalente e até mais eficiente que uma codificação binária em termos de qualidade de soluções encontradas e com um menor esforço computacional. Mantendo uma codificação decimal, o "crossover" seria até mais eficiente pois no ponto de "crossover" realmente aconteceria transferência adequada de informação (no exemplo da figura 4.3, 3 linhas passam para a configuração 1 e 4 linhas passam para a configuração 2) e a mutação simplesmente consistiria em aumentar ou diminuir em uma unidade o número de linhas para um caminho candidato. Esta codificação ainda facilita o trabalho computacional na hora de testar as configurações através de um PL. Pode-se ainda incorporar mutações maiores e assim é possível definir mutações simples, duplas, etc. Estas modificações foram incorporadas e testadas no algoritmo desenvolvido juntamente com a codificação binária implícita.

4.3.2 O Mecanismo Geracional: Seleção - "Crossover" - Mutação

A partir da população corrente é determinada uma nova população através de um processo geracional que implica realizar sucessivamente as operações de seleção, "crossover" e mutação.

Nos **AG** o *mecanismo de seleção* é a primeira etapa no processo de obtenção de uma nova população a partir da população corrente. No processo de seleção, determina-se o número de descendentes ("offspring") a que tem direito cada configuração ("string") da população corrente. Nas etapas seguintes são implementados o "crossover" e a mutação para terminar de gerar os elementos (configurações) da nova população.

A Figura 4.4 mostra as operações de "crossover" e mutação para duas configurações de uma rede simples e usando a codificação binária. Nesta, observa-se em primeira instância o procedimento de "crossover" entre configurações, para isto seleciona-se o ponto de cruzamento que neste caso corresponde ao quarto bit; então, o bloco identificado entre a posição seguinte e o último bit será trocado entre as duas configurações, neste caso a troca de blocos entre os dois cromossomos pais só inclui a linha 2-3. No passo seguinte observa-se o procedimento de mutação, o qual é determinado pela taxa de mutação; neste caso em cada configuração se efetuará uma mutação. Na primeira configuração foi selecionada a terceira posição que corresponde a linha 1-3, adicionando duas novas linhas; na segunda configuração foi selecionada a segunda posição, que corresponde à linha 1-2, retirando uma linha.

4.3.3 Geração da População Inicial

No caso do problema de planejamento de redes de transmissão gera-se a população inicial de duas maneiras. A primeira corresponde a uma forma aleatória na qual se controla o número de posições diferentes de zero, assim como também só é gerado um número aleatório de linhas para ser armazenadas nessas posições que é igual ou inferior ao máximo número de linhas. Tipicamente o número de posições diferentes de zero na configuração final está na faixa de [15 - 20]% do total de posições; assim, para a geração aleatória, só são ocupadas entre [30 - 40]% das posições. Este tipo de inicialização aleatória será empregada em sistemas pequenos (sistema Garver) e de médio porte (sistema Sul brasileiro). Com estes sistemas se chega, com alguma freqüência, à solução ótima, inicializando com configurações geradas de maneira aleatória.

Já para sistemas de grande porte, como é o caso da rede Norte-Nordeste, mostra-se pouco efetivo, devido ao grande número de regiões no espaço de soluções que contém esta rede. Iniciando o processo com configurações geradas de maneira aleatória, o mesmo pode

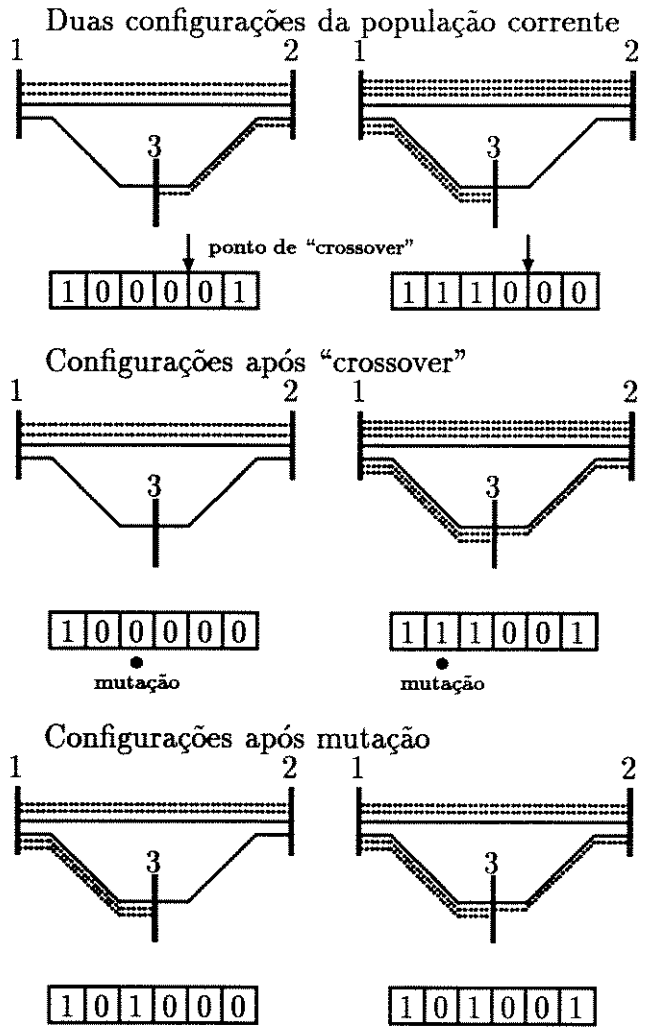


Figura 4.4: "Crossover" e mutação no sistema binário

ficar preso em alguma das regiões, apresentando grande dificuldade para passar às regiões de melhor qualidade. Isto se deve ao fato de que, iniciando o processo e até conseguir chegar às regiões mais atrativas, teria que enfrentar mudanças bruscas entre regiões, o qual torna o problema de difícil resolução.

Para algumas aplicações, especialmente sistemas de grande porte, pode ser conveniente inicializar usando métodos aproximados. Dentro das técnicas aproximadas mais conhecidas para gerar configurações iniciais no estudo do planejamento das redes de transmissão estão os métodos:

- do *mínimo esforço* [12],

4.4.1 Taxa de Mutação Variável

A mutação permite encontrar novas alternativas ou regiões promissoras ao alterar o estado em uma posição da configuração. Este aspecto é mais importante ainda nas fases finais do processo quando a maioria das configurações apresentam funções objetivo muito parecidas, onde uma taxa de mutação relativamente elevada pode produzir em muitas configurações modificações significativas dando mais dinamismo ao processo. Na presente pesquisa foram usadas taxas de mutação relativamente elevadas e variáveis.

Os algoritmos genéticos básicos usam taxas de mutação pequenas dando um papel secundário a este operador. Entretanto, novas pesquisas realizadas em problemas reais verificam a importância de se usar uma taxa de mutação mais alta e, ainda mais, variável; como é o caso deste trabalho. Por exemplo, para uma configuração dada é gerado um número aleatório entre 0 e 5, assim esse número aleatório define o número de pontos para mutação. Logo, para cada ponto de mutação escolhido também aleatoriamente, implementa-se uma mutação simples ou dupla na representação decimal, isto é, adiciona-se ou retira-se uma ou duas linhas na configuração do sistema. Esta última decisão (adição ou retirada) permite ainda duas variantes:

1. Implementação simples;
2. Avaliação do corte de carga da configuração candidata, com adição de linhas em caso de existir corte de carga e retirada de linhas caso contrário.

Logicamente esta segunda alternativa requer uma chamada à subrotina de PL.

Todas as implementações em relação a mutação antes mencionadas, tem dois objetivos claramente diferenciados:

1. Produzir mudanças bruscas na configuração a fim de romper a homogeneidade das configurações da população, o que é um objetivo muito mais importante do previsto, e
2. Recuperar a perda de material genético, isto é, fazer pequenas alterações nas configurações a fim de produzir uma configuração que não é possível obter por "crossover".

Em [14] foram implementados com muito sucesso *operadores especiais* ("additional set of operators") para melhorar o desempenho de um algoritmo genético onde os mesmos realizam uma tarefa de mutação elevada e variável. Estas idéias também foram implementadas neste trabalho.

4.4.2 Taxa de Mutação Controlada por *SA*

As idéias sobre a taxa de mutação variável não fazem parte do algoritmo genético básico e, na verdade, elas são idéias de **SA**. Uma mutação introduz informação nova, criando e destruindo "building blocks" nas configurações. Assim, pode-se usar a seguinte relação para determinar o critério de mutação [14]:

$$P(\Delta v) = exp(-\Delta v/T)$$

em que Δv é a variação da função objetivo e T é o parâmetro temperatura de **SA** que pode ser determinado e atualizado com os mesmos critérios usados em [6].

Com este critério, a mutação é implementada de uma forma aleatória: dada uma configuração candidata a mutação, então determina-se Δv, simulando a introdução da mutação e usando uma chamada ao PL. Se Δv é menor que zero, então aceita-se a mutação, caso contrário, a mutação é aceita de forma probabilística quando $Pr(\Delta v) > P(0,1)$; isto é, uma mutação de piora na função objetivo é aceita se $Pr(.)$ é maior que um número aleatoriamente gerado no intervalo [0,1]. Este mecanismo leva a uma maior diversidade do processo, especialmente nas fases iniciais. Esta estratégia foi implementada, neste trabalho, observando-se uma ligeira melhoria no algoritmo implementado. Esta estratégia foi usada com sucesso em [5, 14].

4.4.3 Seleção Limitada

No algoritmo genético básico, cada configuração tem direito a gerar um número de descendentes determinado pela qualidade da função objetivo correspondente. Assim, o número de descendentes de uma configuração não está limitado. Neste trabalho foi observado que esta implementação pode causar convergência prematura do processo. Nas primeiras gerações, geralmente as funções objetivo são de pobre qualidade e a presença de uma ou algumas configurações com função objetivo de boa qualidade pode eliminar o efeito das outras configurações, produzindo a convergência prematura num ótimo local. Este fenômeno é muito conhecido e existem várias formas de controle deste problema [3]. Todas estas formas de controle visam limitar o número de descendentes de cada configuração. Assim, uma forma muito simples de controlar este problema é limitar o número máximo de descendentes para cada configuração independentemente da qualidade de sua função objetivo. No algoritmo implementado, para os sistemas de grande porte, foi limitado o número máximo de descendentes de uma configuração entre 2 a 3% da população total. Em sistemas de pequeno e médio porte praticamente não aparece este fenômeno de convergência prematura.

4.4.4 Implementação dos "Building Blocks" Através dos Caminhos Novos

A grande capacidade do algoritmo genético em encontrar configurações ótimas ou quase-ótimas é explicada pela sistemática formação dos chamados *blocos construtivos ("building blocks")* nas configurações da população. Para explicar a formação de "building blocks", precisa-se da definição de alguns conceitos básicos dos **AG**.

Um *"schema"* representa um subconjunto de todas as possíveis "strings" que têm os mesmos bits em determinadas posições. Por exemplo, o "schema" $**000$ representa um subconjunto de todos os números binários de cinco dígitos que tem três posições fixas, assim este "schema" representa os seguintes quatro "strings" :{00000, 01000, 10000, 11000}. Relacionado com um "schema", define-se o *valor médio da função objetivo do "schema"* ("average fitness of the schema") como sendo o valor médio da função objetivo de todos os "strings" ou elementos do "schema". Adicionalmente, define-se a *distância de definição* ("defining lenght") de um "schema" como a distância entre duas posições fixas mais externas do "schema". Assim, por exemplo, o "schema" $**011*$ tem "defining lenght" igual a 2 e o "schema" $1*0**1$ tem "defining lenght" igual a 5.

Define-se então um *"building block"* como aquele "schema" que tem um valor objetivo de excelente qualidade ("high fitness values") e pequena distância de definição ("defining lenght"). Portanto, um "building block" é um excelente "schema" no sentido de que possui armazenada uma informação de excelente qualidade pois tem uma função objetivo de alta qualidade e, além disso, apresenta uma "defining lenght" pequena, o que significa que essa informação de qualidade *"está concentrada"*. A pequena "defining lenght" vai diminuir a chance de que um "building block" seja destruído pela operação de "crossover". Por exemplo, o "schema" $1*0**1$ pode ser mais facilmente destruído pela operação de "crossover" que o "schema" $**011*$ que tem a informação mais "concentrada".

A *hipótese de "building blocks"* supõe que, os operadores de "crossover" e mutação, fazem a geração, promoção e justaposição de "building blocks" para formar os "strings" ótimos. O "crossover" tende a conservar a informação genética presente nos "building blocks" das configurações que devem ser cruzadas e a mutação promove a geração de novos "building blocks". Este mecanismo de formação de "building blocks" é a *essência fundamental* do funcionamento dos **AG**.

Na presente pesquisa, além de implementar todas as operações básicas de geração de "building blocks" através do "crossover" e da mutação, que fazem parte de um algoritmo genético básico, foram realizadas algumas implementações adicionais usando a "essência fundamental" da formação e conservação dos "building blocks". Esta implementação foi realizada para uma parte do sistema elétrico, aquela que corresponde aos novos caminhos de adição e às barras isoladas da configuração base. A Figura 4.5, que mostra uma pequena parcela de um sistema elétrico, ilustra este caso particular de barras isoladas.

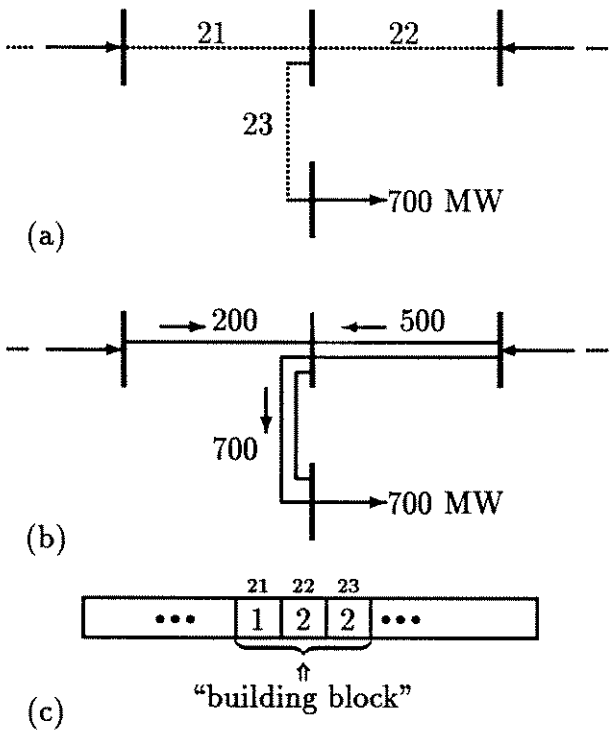


Figura 4.5: "Building blocks" e barras isoladas

Na Figura 4.5 **(a)** aparece uma carga de 700 MW, dois pontos prováveis de fornecimento de potência e três caminhos candidatos. Em **(b)** aparece a combinação de adições que resolve da maneira mais adequada o fornecimento da carga de 700 MW, e em **(c)** são apresentadas estas adições em codificação decimal. Nestas condições são importantes as seguintes observações:

1. Para passar da Figura 4.5(a) para 4.5(b) na verdade foi gerada uma espécie de "building block",

2. A combinação de adições nos três caminhos é que constitui uma informação de excelente qualidade e não a informação de adições isoladas,

3. Tanto "crossover" como mutação podem destruir o "building block" da Figura 4.5(b),

4. Na figura 4.5(a), na implementação de adições, não faz sentido adicionar linhas ao caminho 23 sem fazer adições simultaneamente ao caminho 21 e/ou 22, pois uma adição isolada em 23 é completamente inútil, e

5. Na codificação de uma configuração, normalmente os caminhos 21, 22 e 23 não ocupam casas consecutivas na representação da configuração e no sistema elétrico

existem muitos subsistemas como os mostradas na figura 4.5. Assim, com os caminhos 21, 22 e 23 ocupando casas distantes na configuração, os mesmos podem ser facilmente destruídos pelo operador crossover .

Levando em conta as observações anteriores, na presente pesquisa foi realizada uma implementação computacional que contorna adequadamente os problemas antes mencionados e aproveita ao máximo a idéia dos blocos construtivos. Assim, foi realizada uma implementação computacional que identifica "building blocks" para a parte do sistema correspondente aos novos caminhos. Um "building block" está constituído por linhas adicionadas em novos caminhos e que são consecutivas ou adjacentes. Desta forma, aparecem "building blocks" em zonas específicas geralmente em torno de grandes barras de carga ou geração. Nesta estratégia, uma vez definida as configurações da população inicial, o primeiro passo é a identificação do número e os elementos dos "building blocks" correspondentes aos novos caminhos. Apontadores adequados identificam os elementos de um "building block".

A operação de "crossover" não destrói estes tipos especiais de "building blocks" pois todos os elementos dos mesmos entram numa única nova configuração. Esta estratégia também elimina a possibilidade de que o "crossover" forneça configurações com adições inadequadas (linhas que não transmitem potência porque não fecham um caminho). Assim, estes "building blocks" aparecem ou desaparecem com a operação de mutação ou com a eliminação das configurações onde elas estão presentes. É interessante observar que com esta estratégia, estes "building blocks" apresentam "defining lenght" ideal pois, por definição, estes "building blocks" não são destruídos por "crossover", mesmo que seus elementos não sejam codificados em casas vizinhas ou consecutivas.

A operação de mutação pode aumentar ou diminuir o número destes "building blocks" presentes nas configurações do sistema, assim como o número de elementos de um "building block" específico. Por exemplo, na Figura 4.5(b), uma mutação em 21 e duas mutações sucessivas em 23 levaria à eliminação desse "building block". Por outro lado, uma mutação em 21 simplesmente levaria a um "building block" com somente 2 elementos. De maneira similar, a mutação pode criar um novo "building block". Esta implementação dá um papel mais importante à mutação, sendo que deve ser usada uma taxa de mutação maior que as tradicionalmente empregadas.

Mutação pode produzir configurações indesejáveis. Observe na Figura 4.5(b) que, se a mutação elimina a adição no caminho 21 e depois elimina as adições no caminho 23, então as adições no caminho 22 são irrelevantes ou indesejáveis. Portanto, na implementação da mutação, deve-se implementar uma estratégia que evite configurações indesejáveis. Desta forma, o algoritmo *contorna um problema*, a adição inadequada de linhas em novos caminhos, e *aproveita* uma característica específica do problema de planejamento formando "building blocks" com linhas vizinhas, aproveitando todas as vantagens que apresentam os "building block" gerados pelo algoritmo genético básico. Embora seja possível ge-

neralizar esta idéia de implementar "building blocks" para o sistema elétrico completo, considerando também os caminhos correspondentes a reforços em linhas já existentes, a implementação computacional cresce em complexidade e se torna difícil identificar os blocos independentes. Assim, estes outros caminhos funcionam como "building blocks" de um único elemento e eles formam "building blocks" tradicionais com as operações de "crossover" e mutação convencionais. Testes realizados em sistemas altamente ilhados mostram uma melhoria significativa dos **AG** com as implementações mencionadas.

4.4.5 Formas Especializadas de Geração da População Inicial

Existem várias maneiras de determinar a população inicial para o algoritmo genético. Neste trabalho foi usado um simples, porém eficiente, procedimento de inicialização baseado no algoritmo de Garver [9]. Neste, a rede de transmissão é modelada pelo modelo de transportes (a lei das tensões de Kirchhoff não é representada no modelo, assim o processo fica mais rápido porém a solução é relaxada para o modelo DC). O algoritmo de Garver foi modificado de forma que configurações radicalmente diferentes sejam geradas; com isto se introduz diversidade no material genético que é produzido na etapa inicial do processo de otimização. O algoritmo usado para gerar a população inicial é resumido como segue:

1. Determinar um plano de expansão ótimo usando variáveis de decisão contínuas,
2. Adicionar a parte inteira de todas as linhas/transformadores adicionadas no passo 1 e formar a nova configuração base,
3. Usando a configuração anterior, preparar uma lista de NL linhas/transformadores com investimento fracionário e que nesta configuração transportem os maiores fluxos de potência. (Com cada elemento da lista será obtida uma configuração).
4. Para cada elemento da lista, fazer o que segue:

   (a) Adicionar um circuito (linha/transformador) que identifica a lista para a configuração base e assim formar a configuração,

   (b) Usando um algoritmo de PL testar a factibilidade da configuração corrente. Se não existe necessidade de adição de novas linhas/transformadores, então foi encontrada uma configuração que deve ser incorporada à população inicial. Voltar para (a) para analisar outra configuração da lista. Caso contrário, ir a (c),

   (c) Obter uma nova configuração adicionando à configuração corrente a linha/transformador que transporta maior fluxo de potência, e voltar a (b).

4.4.6 Parâmetro α Variável

O parâmetro α penaliza a função objetivo com corte de carga tornando as configurações propostas pouco atrativas no processo de seleção. O parâmetro α deve assumir um valor relativamente elevado para garantir que o processo pare com uma configuração sem corte de carga. Mas, nas fases iniciais do processo, um α grande produz uma prematura eliminação de configurações com corte de carga e que eventualmente poderiam levar o processo para regiões promissoras. Assim, é desejável que nas fases iniciais do processo as configurações com corte de carga razoáveis tenham chance de sobrevivência por algumas gerações. Isto pode ser obtido com um α variável, iniciando-se com um α menor e incrementando-se seu valor após cada geração. Testes preliminares mostraram uma melhoria no desempenho do algoritmo com esta implementação. Em [14] foi implementada esta estratégia com resultados excelentes.

4.5 ALGORITMO IMPLEMENTADO

Levando em conta os critérios anteriores foi implementado o seguinte algoritmo genético:

- **Passo 1**: Definir o tamanho da população T_p e determinar a população inicial de forma aleatória, usando configurações conhecidas ou usando o algoritmo de Garver modificado. Definir também k_{max}, o critério de parada que pára o processo se, durante k_{max} gerações, a incumbente não varia, e k_{stop} que é o número máximo de gerações permitidas.

- **Passo 2**: Para cada configuração ("strings") ou elemento da população, determinar o valor da função objetivo ("fitness") correspondente, usando o algoritmo de PL.

  Atualizar, se possível, a incumbente. k_1 conta o número de gerações em que a incumbente permanece sem variar, e k_2 conta o número de gerações.

- **Passo 3**: Se $k_1 > k_{max}$ pare, pois o processo convergiu. Se $k_2 > k_{stop}$ pare, pois foi excedido o número máximo de gerações permitidas. Caso contrário, vá ao passo 4.

- **Passo 4**: Usando o mecanismo de seleção proporcional ou o mecanismo de seleção com limitação de número de descendentes, determinar o número de descendentes ("offsprings") que corresponde a cada "strings".

- **Passo 5**: Realizar "crossover" de maneira sistemática entre cada par de configurações que devem participar na formação da nova geração, até gerar um número de descendentes igual ao tamanho da população. A taxa de "crossover" ρ_c controla estatisticamente o número de configurações que serão cruzadas.

- **Passo 6**: Implementar a mutação de maneira sistemática. Cada configuração candidata realiza até n_m mutações (simples ou duplas) onde n_m é proporcional à taxa de mutação ρ_m. Este processo também é controlado estatisticamente pela taxa de mutação ρ_m. Voltar ao passo 2.

4.6 TESTES EM SISTEMAS PEQUENOS E MÉDIOS E CALIBRAÇÃO DE PARÂMETROS

Para encontrar os valores mais adequados dos parâmetros de controle e o critério de parada, foram realizados testes exaustivos para otimizá-los empiricamente. Para isto, foram usados dois sistemas cujas soluções ótimas são conhecidas: sistema de 06 barras de Garver e sistema Sul brasileiro de 46 barras. Depois foram usados estes resultados para testar um sistema de grande porte cuja solução ótima ainda não é conhecida: sistema Norte-Nordeste brasileiro.

Com relação a população inicial, como foi mencionado anteriormente, esta poderá ser gerada de duas formas diferentes, ou seja:

- Aleatoriamente controlada, e
- Pela Metodologia de Garver modificada

No caso do sistema Garver só é empregada a primeira alternativa, uma vez que a metodologia de Garver modificada encontra a solução ótima global deste sistema, tanto no caso com redespacho como sem redespacho.

No caso do sistema Sul brasileiro, em condições de redespacho e sem redespacho, foram feitas simulações tanto com populações geradas aleatoriamente como com populações geradas pela metodologia de Garver.

Já com o sistema Norte-Nordeste brasileiro, só foi analisado o caso quando a população inicial é gerada com a metodologia de Garver. Provas preliminares mostraram que quando a população inicial é gerada de maneira aleatória, o processo geralmente fica preso em regiões de ótimos locais pouco atrativas e para sair para outras regiões mais atrativas seria necessário um elevado esforço computacional, já que no início do processo este sistema contém muitos nós isolados. Então, as mudanças para passar de uma região a outra estariam comprometendo blocos de até oito elementos (linhas em série). A formação destes blocos é uma tarefa complicada já que este sistema contém um número muito grande de possíveis blocos para unir dois pontos da configuração base e, cuja enumeração resultaria impraticável.

Quando o processo é inicializado com a metodologia de Garver modificada, é identificada uma grande parte dos blocos que conectam os nós isolados; além disso, estes blocos geralmente contém informação de alta qualidade e são formados por linhas de grande capacidade de transmissão.

Da taxa de mutação foram analisados dois tipos:

- Taxa de mutação variável.
- Taxa de mutação controlada por **SA**.

No caso dos sistemas Garver e Sul brasileiro só foi implementada uma taxa simples de mutação variável; com esta, o algoritmo apresenta resultados satisfatórios.

Já com o sistema Norte-Nordeste foi necessário fazer algumas modificações no processo de mutação, com relação à empregada em sistemas de pequeno e médio porte. Assim, foi implementada uma taxa de mutação variável avaliando o corte de carga da configuração candidata. Nesta, adiciona-se, se existe corte e retira-se, caso contrário. Implementou-se também uma taxa de mutação controlada por **SA** na qual, além de avaliar o corte de carga da configuração candidata, tal como foi descrito anteriormente, introduz algumas idéias de **SA**.

Após efetuar algumas análises preliminares com o sistema Norte-Nordeste, foi selecionada a mutação controlada por **SA**; conseguindo-se assim obter configurações de melhor qualidade e maior rapidez no processo.

Com relação ao "crossover" foram analisados três tipos: *"crossover" multiponto binário*, *"crossover" multiponto por elemento* e *"crossover" duplo*. Já que o "crossover" multiponto apresenta um alto grau de diversidade, isto é, salta facilmente de uma região a outra, sua aplicação é recomendada apenas em sistemas pequenos e médios; ao ser empregado em sistemas grandes, e devido ao alto grau de diversidade que apresenta, não é possível que o processo possa executar uma busca exaustiva em cada uma das regiões visitadas. Assim, o sistema Garver e Sul brasileiro apresentam resultados satisfatórios quando é analisado com quaisquer dos três tipos de "crossover". Já no caso do sistema Norte-Nordeste, os melhores resultados são obtidos quando se emprega o "crossover" duplo.

4.6.1 Sistema de 06 Barras de Garver sem Redespacho

Os dados deste sistema aparecem no Apêndice A, e na Figura 3.6 é apresentada a configuração contendo todas as linhas candidatas.

Foram usados os seguintes critérios ou parâmetros: população inicial gerada aleatoriamente, $\alpha = 10$, $\rho_c = 0,9$, taxa de mutação variável a partir de $\rho_m = [0,04; 0,08]$ por bit

e o processo pára se a incumbente não melhora em 8 gerações consecutivas.

Com os critérios antes mencionados o processo converge facilmente para a solução ótima em grande parte dos testes realizados, para tamanhos de população variando de 20 a 40 configurações. Os melhores resultados, em termos de processamento, foram obtidos com tamanhos de população variando entre 24 e 34 configurações, onde o processo converge entre 25 e 35 gerações; o que implica que o algoritmo faz entre 700 a 1.000 chamadas ao algoritmo de PL. Na Tabela 4.1, é apresentado um quadro que contém dez casos, os quais foram analisados com configurações obtidas aleatoriamente, em que observa-se o número de gerações requeridas para chegar à solução ótima pela primeira vez e o número de gerações para que o processo finalize.

Adicionalmente foi observado que a população inicial gerada aleatoriamente produz uma ligeira variação no número de gerações para atingir a convergência e, em casos excepcionais, o algoritmo convergiu na solução ótima com apenas 80 chamadas ao algoritmo de PL.

Na Figura 3.8 é apresentada a configuração ótima e cujo custo é de $v = 200$ com as seguintes linhas adicionadas: $n_{26} = 4$, $n_{35} = 1$ e $n_{46} = 2$.

Tabela 4.1: Resultados do sistema Garver, com e sem resdepacho

Sistema Garver - Casos com e sem redespacho				
Inicialização aleatória				
	f.o. = 110		f.o. = 200	
Caso Nº	Nº Ger. 1ª vez	Nº Ger. final	Nº Ger. 1ª vez	Nº Ger. final
1	6	14	14	22
2	8	16	15	23
3	9	17	16	24
4	13	21	18	26
5	16	24	20	28
6	17	25	23	31
7	18	26	26	34
8	22	30	31	39
9	25	33	32	40
10	32	40	34	42

Na Tabela 4.2 se apresentam dez testes do sistema Garver sem redespacho, nesta observa-se duas situações com diferente tamanho de população: na primeira, a análise é feita com uma população de 20 configurações, na segunda, é feita um análise com uma população de 30 configurações e, pode-se verificar a qualidade do desempenho devido ao tamanho da população.

Tabela 4.2: Resultados para o sistema Garver sem redespacho, analisado para duas populações

Sistema Garver sem redespacho				
Inicialização aleatória				
	População de 20 Conf.		População de 30 Conf.	
Caso Nº	Nº Ger.	Valor da f.o.	Nº Ger.	Valor da f.o.
1	16	200	11	200
2	40	200	22	200
3	53	231	37	200
4	36	200	50	200
5	32	272	57	231
6	25	200	37	200
7	61	200	34	200
8	25	260	52	200
9	48	200	22	200
10	29	272	37	231

4.6.2 Sistema de 06 Barras de Garver com Redespacho

Os dados deste sistema são apresentados no Apêndice A. Neste, se empregaram os mesmos parâmetros que com o caso sem redespacho e com tamanhos de população variando de 16 a 40 configurações. Os melhores resultados, em termos de processamento, foram obtidos com tamanhos de população variando entre 16 e 26 configurações. O algoritmo converge usando, em média, de 3 a 5 gerações a menos com relação ao caso sem redespacho (em termos de PL se fazem entre 500 - 750 chamadas). Na Tabela 4.1, pode-se observar um quadro contendo dez casos, os quais foram inicializados com diferentes configurações obtidas aleatoriamente; neste observa-se o número de gerações requeridas para chegar a solução ótima pela primeira vez, assim como o número de gerações para que o processo finalize.

Na Figura 3.7 é apresentada a configuração ótima, cujo custo é de $v = 110$, com as seguintes linhas adicionadas: $n_{35} = 1$ e $n_{46} = 3$.

Na Tabela 4.3 apresentam-se dez testes do sistema Garver com redespacho, e na qual foram estudadas duas situações com diferente tamanho de população: na primeira, é feita uma análise com uma população de 16 configurações, na segunda, é feita um análise com uma população de 26 configurações.

Tabela 4.3: Resultados para o sistema Garver com redespacho, analisado para duas populações

Sistema Garver com redespacho				
Inicialização aleatória				
	População de 16 Conf.		População de 26 Conf.	
Caso Nº	Nº Ger.	Valor da f.o.	Nº Ger.	Valor da f.o.
1	30	110	17	110
2	26	130	25	110
3	33	110	17	130
4	40	130	26	141
5	44	110	33	130
6	36	130	12	110
7	40	110	17	110
8	39	110	25	110
9	25	110	17	110
10	32	130	13	110

4.6.3 Sistema Sul Brasileiro sem Redespacho

Os dados deste sistema aparecem no Apêndice A, e na Figura 3.9 é apresentada a configuração contendo todas as linhas candidatas.

Foram usados os seguintes critérios ou parâmetros: $\rho_c = 0,9$, $\alpha = [400-1.000]$, taxa de mutação variável a partir de $\rho_m = [0,02; 0,04]$ por bit, e o processo pára se a incumbente não melhora em 30 gerações consecutivas. Com relação a população inicial, esta foi gerada de duas maneiras, ou seja: aleatoriamente controlada e através da metodologia de Garver modificada.

População inicial gerada aleatoriamente

Com os critérios antes mencionados o processo converge facilmente na solução ótima na grande maioria dos testes. Os melhores resultados em termos de esforço computacional foram obtidos com populações de 50 a 60 configurações. O algoritmo converge na maioria dos casos, entre 140 e 180 gerações, o que implica que o algoritmo faz entre 8.000 a 9.000 chamadas ao algoritmo de PL. Foram observados testes que encontram a solução ótima em apenas 38 gerações com apenas 1.800 chamadas ao algoritmo de PL. Na Tabela 4.4 é apresentado um quadro contendo vinte casos, os quais foram inicializados com diferentes configurações obtidas aleatoriamente. Neste caso de inicialização aleatória foi usada uma população de 60 configurações.

A configuração ótima deste sistema é apresentada na Figura 3.11. Sua solução ótima é de $US\$154.420.000,00$ e as seguintes linhas adicionadas : $n_{20-21} = 1$, $n_{42-43} = 2$, $n_{46-06} = 1$, $n_{19-25} = 1$, $n_{31-32} = 1$, $n_{28-30} = 1$, $n_{26-29} = 3$, $n_{24-25} = 2$, $n_{29-30} = 2$ e $n_{05-06} = 2$.

Tabela 4.4: Resultados do sistema Sul brasileiro sem resdepacho

Sistema Sul brasileiro - caso sem redespacho						
	Inic. aleatória			Inic. Garver modificado		
Caso Nº	Nº Ger. 1ª vez	Nº Ger. final	Custo (10^3 US$)	Nº Ger. 1ª vez	Nº Ger. final	Custo 10^3 (US$)
1	42	72	154.420,00	6	36	154.420,00
2	54	84	154.420,00	10	40	154.420,00
3	55	85	154.420,00	11	41	161.188,00
4	64	94	154.420,00	15	45	154.420,00
5	67	97	154.420,00	18	48	154.420,00
6	70	100	154.420,00	19	49	154.420,00
7	80	110	154.420,00	20	50	154.420,00
8	84	114	154.420,00	22	52	154.420,00
9	90	120	154.420,00	24	54	154.420,00
10	93	123	164.880,00	26	56	154.420,00
11	107	137	154.420,00	27	57	154.420,00
12	115	145	168.095,00	28	58	154.420,00
13	117	147	154.420,00	32	62	154.420,00
14	125	155	170.273,00	35	65	178.603,00
15	129	159	154.420,00	41	71	169.796,00
16	130	160	164.752,00	48	78	154.420,00
17	136	166	154.420,00	56	86	175.733,00
18	140	170	157.198,00	65	95	154.420,00
19	143	173	154.420,00	68	98	175.733,00
20	160	190	154.420,00	75	105	154.420,00

População inicial gerada com a metodologia Garver modificada

Como no caso anterior, o processo também converge facilmente para a solução ótima na grande maioria dos testes. Os melhores resultados em termos de esforço computacional foram obtidos com populações de 40 a 50 configurações. O algoritmo converge na maioria dos casos, entre 80 e 100 gerações, o que implica que o algoritmo faz entre 3.500 a 4.500 chamadas ao algoritmo de PL. Foram observados testes que encontram a solução ótima em apenas 6 gerações com apenas 240 chamadas ao algoritmo de PL. Na Tabela 4.4 é apresentado um quadro que contém a informação de vinte casos; estes foram ana-

lisados com diferentes combinações das configurações geradas com a metodologia Garver modificada. Neste caso foi usada uma população de 40 configurações.

Observação:

O número de chamadas à subrotina de PL (da ordem de 9.000), que determina o esforço computacional do algoritmo, poderia parecer um número relativamente alto. Na verdade o número de chamadas é muito pequeno, e cada chamada equivale a avaliar uma configuração candidata. Neste sistema é permitido realizar até 4 adições para cada um dos 79 caminhos candidatos, portanto, o número de configurações possíveis apresentado é igual a $(4+1)^{79} \cong 2^{183}!$. Pode-se verificar que o algoritmo genético analisa somente um número muito reduzido de configurações do espaço das configurações.

4.6.4 Sistema Sul Brasileiro com Redespacho

Neste caso são usados os mesmos parâmetros que o caso sem redespacho. O sistema foi avaliado com os dois tipos de população inicial; no primeiro, a população inicial é gerada de maneira aleatória, no segundo, usa-se o método Garver modificado.

No primero caso, população inicial gerada aleatoriamente, os melhores resultados em termos de esforço computacional foram obtidos com populações de 50 a 60 configurações. O algoritmo converge na maioria dos casos, entre 120 e 160 gerações, o que implica que o algoritmo faz entre 7.000 a 8.000 chamadas ao algoritmo de PL. Na Tabela 4.5, é apresentado um quadro contendo os resultados de vinte casos, analisados com configurações obtidas aleatoriamente, usando diferentes números aleatórios iniciais.

No segundo caso, população inicial gerada com a metodologia Garver modificada, os melhores resultados em termos de esforço computacional foram obtidos com populações de 40 a 50 configurações. O algoritmo converge, na maioria dos casos, entre 70 e 90 gerações o que implica que o algoritmo faz entre 3.200 a 4.200 chamadas ao algoritmo de PL.

Os resultados apresentados na Tabela 4.5 foram obtidos empregando uma população de 50 configurações, no caso de inicialização aleatória, e uma população de 40 configurações, no caso de inicialização com a metodologia Garver modificada.

Na Figura 3.10, é apresentada a solução ótima, com um custo de US$ 70.289.000,00, que corresponde à adição das seguintes 8 linhas: $n_{13-20} = 1$, $n_{20-23} = 1$, $n_{46-06} = 1$, $n_{20-21} = 2$, $n_{42-43} = 1$ e $n_{05-06} = 2$.

Tabela 4.5: Resultados do sistema Sul brasileiro com resdepacho

Sistema Sul brasileiro - caso com redespacho						
	Inic. aleatória			Inic. Garver modificado		
Caso Nº	Nº Ger. 1ª vez	Nº Ger. final	Custo (10^3 US$)	Nº Ger. 1ª vez	Nº Ger. final	Custo 10^3 (US$)
1	58	88	70.289,00	5	35	70.289,00
2	60	90	70.289,00	5	35	70.289,00
3	63	93	102.821,00	8	38	70.289,00
4	65	95	70.289,00	9	39	70.289,00
5	66	96	70.289,00	12	42	70.289,00
6	70	100	90.748,00	14	44	70.289,00
7	73	103	70.289,00	18	48	70.289,00
8	75	105	89.586,00	18	48	70.289,00
9	84	114	70.289,00	21	51	70.289,00
10	85	115	104.407,00	24	54	70.289,00
11	89	119	70.289,00	27	57	70.289,00
12	91	121	70.289,00	32	62	70.289,00
13	91	121	70.289,00	38	68	70.289,00
14	98	128	70.289,00	44	74	70.289,00
15	99	129	70.289,00	47	77	70.289,00
16	102	132	70.289,00	52	82	70.289,00
17	103	133	70.289,00	56	86	70.289,00
18	106	136	70.289,00	72	102	70.289,00
19	108	138	70.289,00	83	113	70.289,00
20	120	150	70.289,00	85	115	70.289,00

4.7 TESTES EM SISTEMAS DE GRANDE PORTE

4.7.1 Sistema Norte - Nordeste Brasileiro - Plano 2008

O terceiro sistema que será analisado é apresentado na Figura 3.12; representa a rede brasileira correspondente às regiões Norte e Nordeste e está constituído por 89 barras e 183 linhas candidatas. Os dados deste sistema são apresentados no Apêndice A. O sistema apresenta uma demanda de 29,754 MW e ainda não se conhece a solução ótima deste sistema.

Foram usados os seguintes critérios ou parâmetros: $\rho_c = 0,9$, $\alpha = 1.000$, taxa de

mutação variável a partir de $\rho_m = [0,01; 0,03]$, por bit. $K = [5.500.000 - 3.500.000]$, iniciando no valor mais alto e diminuindo até o valor mais baixo, na medida em que evolui o processo. Tamanho da população entre [160 - 200] configurações. Temperatura para o caso da taxa de mutação controlada por **SA** em uma faixa de [50.000 - 80.000], com um fator de diminuição a cada geração de 0,996. O processo pára se a incumbente não melhora em 80 gerações consecutivas.

Tabela 4.6: Desempenho do algoritmo com taxa de mutação variável

Nº da geração	Investimento 10^3 US$	Nº de PL
0	2.857.234,00	1
1	2.774.380,00	114
6	2.750.344,00	684
23	2.739.194,00	2.622
100	2.738.525,00	11.400
125	2.730.599,00	11.970
138	2.723.769,00	12.882
160	2.716.259,00	13.224
240	2.709.749,00	20.292
304	2.703.514,00	34.656
374	2.699.279,00	43.776
399	2.698.195,00	45.486
460	2.689.269,00	57.114
527	2.681.343,00	66.918
608	2.680.703,00	70.452
647	2.669.993,00	73.758
677	2.663.973,00	74.898
689	2.661.239,00	75.126
701	2.650.313,00	75.696
788	2.648.313,00	90.972

Melhor investimento = 2.643.313.000,00 , corte de carga = 5 Mw

Com relação à população inicial, a mesma é gerada usando a metodologia de Garver modificada e, além disso, foram obtidas algumas configurações empregando métodos aproximados. Baseado nestes últimos, foi elaborado um quadro com estatísticas apresentado mais adiante.

O algoritmo converge para este caso, entre 350 e 500 gerações, o que implica que o algoritmo faz entre 180.000 a 250.000 chamadas ao algoritmo de PL.

Neste sistema de grande porte, foram realizadas todas as implementações específicas anteriormente mencionadas e foram realizados diversos testes para recalibrar os principais

parâmetros.

Tabela 4.7: Desempenho do algoritmo com taxa de mutação controlada por *SA*

Nº da geração	Investimento 10^3 US$	Nº de PL
0	2.857.234,00	1
1	2.798.426,00	414
4	2.763.686,00	703
10	2.758.351,00	995
13	2.742.791,00	1.541
19	2.720.526,00	2.099
27	2.715.646,00	3.507
34	2.695.166,00	4.346
37	2.692.176,00	4.637
45	2.691.931,00	4.921
52	2.679.005,00	8.207
78	2.670.495,00	13.464
133	2.668.655,00	42.033
152	2.664.190,00	47.251
155	2.660.015,00	50.776
184	2.656.650,00	56.463
193	2.655.865,00	58.626
214	2.654.560,00	64.088
241	2.652.548,00	68.746
281	2.650.787,00	71.447
348	2.649.883,00	95.099
387	2.649.323,00	102.941
456	2.639.528,00	124.422

Melhor investimento = 2.638.528.000,00 , corte de carga = 1 Mw

Com a finalidade de estudar o desempenho do algoritmo com os dois tipos de mutação descritos anteriormente, taxa de mutação variável e taxa de mutação controlada por **SA**, foram analisados diversos casos. Nas Tabelas 4.6 e 4.7, são apresentados dois casos típicos nos quais mostra-se o desempenho do algoritmo ao ser utilizados estes dois tipos de mutação. Nestes dois casos empregou-se a mesma população inicial, a que foi gerada utilizando métodos aproximados. Observa-se destas duas tabelas um melhor desempenho da taxa de mutação controlada por **SA** com relação a taxa de mutação variável, neste sentido, empregou-se a primeira.

No caso da taxa de mutação variável emprega-se uma mutação dinâmica, onde a cada configuração pode-se efetuar até l mutações e cada uma destas consiste em adicionar ou retirar uma ou duas linhas ou caminhos. Este tipo de mutação é muito importante nas

fases finais do processo onde a maioria das configurações apresentam funções objetivo muito semelhantes, produzindo assim configurações diferentes, o que dá maior dinamismo ao processo.

Na taxa de mutação controlada por **SA**, são feitas menos mutações que o caso anterior; aqui cada mutação implica uma chamada ao PL. Nesta, aceitam-se mutações que melhorem o valor da função objetivo da configuração; caso contrário, aceitam-se com uma certa probabilidade. Nesta forma de mutação é obtida uma maior diversidade pois nas fases iniciais do processo, com temperatura elevada, são aceitas configurações piores com maior freqüência. Por outro lado, nas fases finais, praticamente todas as configurações piores são rejeitadas produzindo um nível de intensificação no processo.

Casos estudados

Análise foi feita empregando duas populações com diferentes características, geradas empregando métodos diferentes; ou seja, a primeira foi gerada utilizando métodos aproximados (cuja característica é a de apresentar um caminho de linhas de 3.200 MW), a segunda foi gerada utilizando a metodologia de Garver modificada (cuja característica é a de apresentar dois caminhos alternativos de linhas de 3.200 MW).

No primeiro caso, a melhor configuração obtida apresentou um investimento de US$ 2.628.773.000,00 e as seguintes adições de linhas:

$n_{01-02} = 1$, $n_{02-04} = 1$, $n_{04-05} = 4$, $n_{04-06} = 4$, $n_{04-68} = 1$, $n_{05-56} = 1$, $n_{05-58} = 4$, $n_{06-07} = 2$, $n_{07-08} = 2$, $n_{08-17} = 2$, $n_{12-15} = 2$, $n_{13-14} = 1$, $n_{13-15} = 3$, $n_{14-17} = 1$, $n_{14-59} = 1$, $n_{15-16} = 3$, $n_{15-46} = 2$, $n_{16-44} = 6$, $n_{17-18} = 2$, $n_{18-50} = 11$, $n_{18-74} = 3$, $n_{21-57} = 2$, $n_{22-37} = 1$, $n_{22-58} = 1$, $n_{24-43} = 1$, $n_{25-55} = 3$, $n_{26-54} = 2$, $n_{30-31} = 3$, $n_{30-63} = 3$, $n_{31-34} = 2$, $n_{35-51} = 1$, $n_{36-46} = 2$, $n_{40-45} = 1$, $n_{40-46} = 1$, $n_{41-64} = 2$, $n_{42-44} = 2$, $n_{43-55} = 2$, $n_{43-58} = 2$, $n_{48-49} = 1$, $n_{49-50} = 4$, $n_{52-59} = 1$, $n_{53-86} = 1$, $n_{54-58} = 2$, $n_{54-63} = 1$, $n_{56-57} = 1$, $n_{61-64} = 1$, $n_{61-85} = 2$, $n_{61-86} = 1$, $n_{67-68} = 1$, $n_{67-69} = 1$, $n_{67-71} = 3$, $n_{71-72} = 1$, $n_{72-73} = 1$, $n_{73-74} = 1$.

No segundo caso, a melhor configuração obtida apresentou um investimento de US$ 2.600.593.000,00 e com as seguintes adições de linhas:

$n_{01-02} = 1$, $n_{02-04} = 1$, $n_{04-05} = 4$, $n_{04-81} = 3$, $n_{05-56} = 1$, $n_{05-58} = 4$, $n_{06-75} = 1$, $n_{12-15} = 1$, $n_{13-15} = 4$, $n_{14-59} = 1$, $n_{15-16} = 3$, $n_{15-46} = 2$, $n_{16-44} = 6$, $n_{18-50} = 11$, $n_{18-74} = 5$, $n_{21-57} = 2$, $n_{22-58} = 2$, $n_{24-43} = 1$, $n_{25-55} = 4$, $n_{30-31} = 1$, $n_{30-63} = 2$, $n_{35-51} = 2$, $n_{36-46} = 2$, $n_{39-86} = 1$, $n_{40-45} = 1$, $n_{40-46} = 1$, $n_{41-64} = 2$, $n_{42-44} = 1$, $n_{43-55} = 2$, $n_{43-58} = 3$, $n_{48-49} = 1$, $n_{49-50} = 4$, $n_{52-59} = 1$, $n_{53-86} = 1$, $n_{54-58} = 1$, $n_{54-63} = 1$, $n_{56-57} = 1$, $n_{61-85} = 3$, $n_{61-86} = 1$, $n_{63-64} = 1$, $n_{67-68} = 1$, $n_{67-69} = 1$, $n_{67-71} = 3$, $n_{69-87} = 1$, $n_{71-72} = 1$, $n_{72-73} = 1$, $n_{73-74} = 2$, $n_{73-75} = 1$, $n_{75-81} = 1$.

Baseando-se na população de configurações gerada com os métodos aproximados e cuja característica principal é a de apresentar um caminho de linhas de 3.200 MW, foram analisados vinte casos, os quais foram inicializados com diferentes números aleatórios. A população inicial é formada por configurações com custos e valores de corte de carga compreendidos entre US$ 2.659.839.000,00 com corte de carga de 6 MW e US$ 2.832.637.000,00 com corte de carga de 114 MW. Na Tabela 4.8 é apresentado o quadro com as estatísticas resultantes.

Tabela 4.8: Estatísticas do sistema Norte-Nordeste

Caso Nº	Investimento 10^3 US$	Corte de carga (Mw)
1	2.621.774,00	30,0
2	2.628.723,00	0,0
3	2.628.723,00	0,0
4	2.631.289,00	5,0
5	2.631.554,00	18,0
6	2.637.619,00	6,0
7	2.638.077,00	16,0
8	2.639.078,00	10,0
9	2.639.332,00	10,0
10	2.643.313,00	5,0
11	2.662.248,00	7,0
12	2.663.975,00	8,0
13	2.663.975,00	8,0
14	2.669.527,00	2,0
15	2.669.527,00	2,0
16	2.673.578,00	10,0
17	2.676.460,00	2,0
18	2.681.795,00	7,0
19	2.684.244,00	10,0
20	2.746.931,00	0,0

Observação:

A melhor configuração encontrada produz uma redução de investimento da ordem de 250 milhões de dólares comparado com os obtidos por métodos aproximados. Quando comparado aos resultados obtidos com o algoritmo de **SA** seqüencial apresentado no Capítulo 3, o algoritmo genético produz uma redução de investimento da ordem de 76 milhões de dólares. Finalmente, quando o investimento obtido é comparado com o algoritmo de **SA** paralelo, o **AG** produz uma redução de investimento de 2 milhões para o caso em que é utilizada a população gerada com os métodos aproximados e cuja característica principal é a de apresentar um caminho de linhas de 3.200 MW (as mesmas que também foram empregadas na análise do algoritmo de **SA** paralelo); e uma redução de 30 milhões

de dólares quando é utilizada a população gerada com a metodologia Garver modificada. Além disso o número de chamadas ao algoritmo de PL é ligeiramente menor no caso do algoritmo genético.

4.8 CONCLUSÕES

Algoritmos genéticos é uma técnica de otimização baseada em processos encontrados na natureza e os quais têm sido aplicados em um grande número de problemas de engenharia elétrica.

Os algoritmos genéticos apresentam uma grande eficiência quando são comparados com outras técnicas de otimização. Os resultados obtidos com os sistemas de Garver e Sul brasileiro apresentam o **AG** como sendo mais efetivo que os de **SA**; além disto, apresenta melhor desempenho para aplicações com a rede Norte-Nordeste, para a qual não é conhecida a solução ótima. Nesta, foi obtida uma família de interessantes soluções com custos ao redor dos US $ 2.600.853.000,00; ou seja, esta solução tem um custo 8,8% menor que a melhor solução obtida com técnicas de otimização convencional, 2,84% menor que a melhor solução obtida com **SA** seqüencial e 1,13% menor que a melhor obtida com **SA** paralelo.

Dos métodos aproximados para gerar as configurações da população inicial, o que apresentou melhor desempenho foi a metodologia Garver modificada, por apresentar configurações de melhor qualidade especialmente na alocação adequada de linhas de grande capacidade de transmissão. Esta metodologia de Garver leva em conta somente a primeira lei de Kirchhoff; as configurações obtidas não adiciona linhas pequenas permitindo que esta parte da alocação seja adequadamente executada pelo algoritmo genético.

A população inicial é gerada aleatoriamente ou através de metodologias aproximadas; com relação a isto, pode-se concluir que, em sistemas pequenos e de médio porte, tanto com populações iniciais geradas aleatoriamente como com populações geradas com a metodologia de Garver modificada, chega-se à resposta ótima com uma taxa de freqüência relativamente alta. Quando, em sistemas de médio porte, emprega-se a inicialização baseada na metodologia de Garver modificada, consegue-se ganhos no número de iterações com relação a inicialização aleatória. Já em sistemas de grande porte, como é o caso da rede Norte - Nordeste, não é possível usar populações geradas aleatoriamente devido ao grande número de nós isolados contido neste sistema, sendo a formação dos caminhos ótimos que conectam estes nós uma tarefa muito complicada; neste caso, os melhores resultados foram obtidos com populações iniciais geradas com a metodologia Garver.

Capítulo 5

PLANEJAMENTO DE SISTEMAS DE TRANSMISSÃO USANDO BUSCA TABU (TS)

5.1 INTRODUÇÃO

Busca Tabu (Tabu Search - **TS**) é um procedimento metaheurístico utilizado para gerenciar um algoritmo heurístico de busca local evitando que o processo pare em um ótimo local. Assim **TS** realiza uma exploração através do espaço de configurações contornando adequadamente os ótimos locais.

TS foi inventado por Fred Glover [51] na década de 80 e na década de 90 está sendo intensamente usado para resolver problemas complexos em áreas muito variadas da pesquisa operacional. Entretanto, a aplicação de **TS** em problemas de sistemas de energia elétrica ainda se encontra na fase inicial.

TS está baseado em conceitos que pertencem ao domínio da inteligência artificial e segundo Glover: "neste se guia um procedimento de busca local para explorar o espaço de soluções além do *ótimo local*". Como na busca local, a busca tabu seleciona de modo agressivo o melhor dos movimentos possíveis em cada passo; mas ao contrário do que ocorre em uma busca local, busca tabu permite mover-se em sua vizinhança, mesmo que o movimento selecionado não seja tão bom como o atual. Deste modo, pode-se escapar dos ótimos locais e continuar a busca em outras regiões.

Para evitar que o processo regresse aos ótimos locais e entre em um ciclo repetitivo, a busca tabu classifica os movimentos mais recentes como "movimentos tabu"; estes proíbem que uma configuração seja revisitada.

Este método contém dois tipos de memória: *memórias de curto* e *longo prazo*. Com relação à de curto prazo, a mesma contém os eventos ocorridos mais recentemente; já na de longo prazo, se armazena dados de freqüência de determinados eventos.

Nas considerações de longo prazo, emprega-se a chamada memória baseada em freqüência, esta contém informação relacionada com o tempo em que certos atributos pertencem ou não às soluções visitadas. Esta informação é fundamental para definir as estratégias de diversificação, as quais permitem pular para regiões não visitadas anteriormente.

Com relação às estratégias de intensificação e diversificação, na primeira se estabelece uma busca detalhada ao redor de boas soluções; na segunda, é possível passar para outras regiões.

Quando um movimento tiver sido anteriormente classificado como tabu e após ser analisado, esse movimento produz uma função objetivo melhor que um valor referencial escolhido (que pode ser a incumbente ou outra boa solução previamente encontrada), então se aplica a denominada *regra de aspiração*; esta consiste em cancelar a proibição e aceitar o movimento.

As estratégias de intensificação e diversificação podem integrar-se usando a estratégia chamada de "path relinking", mediante a geração de novas soluções obtidas ao explorar as trajetórias que conectem as boas soluções.

Outra estratégia de busca muito importante é a *oscilação estratégica*, nesta os movimentos são guiados até um limite e depois, no processo, permite-se cruzar esse limite voltando e cruzando o limite novamente no sentido oposto. A repetição deste procedimento é oscilatória.

Como se pode observar, a busca tabu, ao contrário de **SA** e **AG**, baseia-se em movimentos determinísticos e não aleatórios. A busca tabu faz uso de estruturas especiais de memória e de estratégias de busca dinâmica.

Neste trabalho foram implementadas duas estratégias de busca tabu. A primeira inicia com uma única configuração e durante o processo se armazenam as melhores configurações. Nas etapas seguintes, estas configurações são analisadas e cada vez que uma configuração nova apresenta um valor melhor, com relação às armazenadas, esta substitui a de pior função objetivo. O processo termina quando todas as configurações armazenadas forem analisadas sem obter alguma melhor. Ainda nesta estratégia, no processo de diversificação, emprega-se informação de freqüência. Na segunda estratégia inicia-se com k configurações. Esta estratégia é semelhante a anterior, com a diferença de que agora se inicia com k configurações. Esta inicialização, com k configurações, é própria dos algoritmos genéticos. As duas estratégias de busca são baseadas nos conceitos de intensificação e diversificação.

Neste capítulo é desenvolvido um algoritmo de Busca Tabu para ser usado no problema de planejamento de sistemas de transmissão. Assim, inicialmente é apresentada uma

análise teórica sobre os fundamentos de **TS** que na verdade é um resumo de [55], depois são discutidos as características mais importantes das funções de **TS** que podem ser implementadas num algoritmo **TS** para o problema de planejamento. São apresentadas as principais características do problema de planejamento que podem ser incorporadas ao algoritmo **TS**. É descrito também o algoritmo **TS**. Na seqüência, são apresentados testes com sistemas de pequeno e médio porte cujas soluções ótimas são conhecidas; além destes, são apresentados testes com um sistema de grande porte. Finalmente são mencionadas as conclusões mais relevantes.

No Capítulo seguinte, apresenta-se versões híbridas deste algoritmo; isto é, foram incorporados ao algoritmo **TS** original funções próprias de outros algoritmos combinatoriais como **SA** e **AG**. Testes realizados mostram que os algoritmos híbridos resultantes apresentam um desempenho ligeiramente superior comparado com o algoritmo **TS** puro. Neste contexto, são apresentadas algumas argumentações teóricas que justificam o desempenho prático destes algoritmos, quando aplicados ao problema de planejamento da expansão de sistemas de transmissão.

5.2 BUSCA TABU: TEORIA

Busca Tabu foi desenvolvida criando-se um conjunto de princípios (funções) que, de forma integrada, permitem resolver um problema da maneira mais inteligente. Assim, **TS** teve sua origem em conceitos que originalmente faziam parte do campo da inteligência artificial. O princípio filosófico de Busca Tabu pode ser explicado com as próprias palavras de seu inventor F. Glover [55] que transcrevemos com pequenas modificações:

> **TS** *está baseado na premissa de que a resolução de um problema pode ser considerado inteligente se esse processo incorpora a* memória adaptativa *e a* exploração sensível. *O uso de* memória adaptativa *contrasta com as técnicas sem memória (como "Simulated Annealing" e o Algoritmo Genético) e com as técnicas de memória rígida (como as técnicas de inteligência artificial e de "branch and bound"). De igual maneira, a idéia de* exploração sensível *em* **TS** *está inspirada na suposição de que uma escolha ruim realizada por uma* estratégia *produz mais informação que uma boa escolha aleatória (numa crítica evidente, por exemplo, a* **SA** *que faz escolhas aleatórias). Assim, se a estratégia que guia um algoritmo que usa memória (como* **TS***) faz uma escolha ruim (passa a uma configuração de baixa qualidade) então, pode-se aproveitar essa informação (escolha ruim) para evitar voltar a visitar essa configuração (ruim) e, ainda melhor, para* modificar (melhorar) a própria estratégia *que guia o processo de busca para ter capacidade de encontrar ou escolher configurações de melhor qualidade .*

PRINCIPAIS CARACTERÍSTICAS DE BUSCA TABU

Memória Adaptativa

Seletividade (incluindo esquecimento estratégico)

Abstração e decomposição (usando memória explícita e por atributos)

Tempo:

- Recência de eventos
- Freqüência de eventos
- Diferenciação entre curto e longo prazo

Qualidade e impacto:

- Importância relativa de escolhas alternativas
- Impacto de mudanças de relações em estrutura ou restrições

Contexto

- Interdependência regional
- Interdependência estrutural
- Interdependência seqüencial

Exploração Sensível

Imposição estratégica de proibições e induções
(condições tabu e níveis de aspiração)

Enfoque concentrado em boas regiões e em boas características das soluções
(processos de intensificação)

Caracterização e exploração de novas regiões promissoras
(processo de diversificação)

Padrões de busca não monótonos
(oscilação estratégica)

Integração e geração de novas soluções
("path relinking")

Figura 5.1: Funções de *TS*

A Figura 5.1 mostra os elementos principais (funções) de **TS**. Portanto, as pesquisas em **TS** consistem em usar as funções indicadas na figura de maneira integrada e eficiente para resolver cada problema específico, levando em conta que cada uma das funções podem ser implementadas de várias formas diferentes dependendo das características do problema e do nível de sofisticação da implementação do algoritmo. Outra linha de pesquisa muito ativa consiste em criar outras funções para que sejam incorporadas como parte da estrutura de **TS** da Figura 5.1 e/ou encontrar variantes de implementação das funções mostradas. Um algoritmo **TS** para um problema específico pode usar algumas ou todas as funções mostradas na Figura 5.1 integradas das formas mais variadas. As características específicas de cada problema e a experiência do pesquisador deve determinar a forma e a qualidade do algoritmo implementado. É oportuno lembrar que **TS** nasceu somente com algumas das funções indicadas na Figura 5.1.

Apresentamos brevemente a essência da lógica de trabalho de Busca Tabu. **TS** resolve problemas genéricos do tipo:

$$\begin{aligned} & Min \quad f(x) \\ s.a. \quad & x \in X \end{aligned} \tag{5.1}$$

TS resolve (5.1) iniciando com um processo similar a qualquer algoritmo heurístico de busca local. Na busca local, dada uma configuração x (solução), define-se uma vizinhança de x como sendo o conjunto de todas as configurações $x' \in N(x)$ que podem ser obtidos pela aplicação de um mecanismo de transição a partir de x. A estrutura de vizinhança define as condições para que x' seja vizinho de x. Assim, por exemplo, no problema de planejamento de sistemas de transmissão se pode definir como vizinho de uma configuração x todas aquelas configurações que podem ser obtidas a partir a x pela adição de um circuito, a retirada de um circuito ou troca de dois circuitos (retirada de um circuito e adição de outro circuito) como foi implementado em **SA**. No algoritmo de busca local, a partir da configuração corrente, passa-se para a configuração vizinha que apresenta uma maior diminuição da função objetivo. Um procedimento repetitivo desta estratégia leva o algoritmo de busca local a parar no momento em que não existe nenhuma configuração vizinha que produz uma diminuição da função objetivo, o que significa que foi encontrado um ótimo local.

TS é diferente de um algoritmo de busca local em dois aspectos fundamentais:

1. A partir da configuração corrente, passa-se à melhor configuração vizinha ou à menos pior, o que implica que é permitida uma degradação da qualidade da função objetivo.

2. O conjunto de vizinhos de x não se caracteriza de maneira estática. Assim, **TS** define uma nova estrutura de vizinhança, $N^*(x)$ que varia *dinamicamente em estrutura e*

tamanho durante todo o processo de otimização. Esta estratégia permite a **TS** realizar uma busca eficiente e inteligente. A Figura 5.2 mostra uma relação entre $N(x)$ e $N^*(x)$. Os elementos de $N^*(x)$ são determinados de várias formas. A título de ilustração, pode-se indicar as seguintes formas:

- *Usando uma lista tabu que armazena atributos de configurações consideradas tabu (proibidas).* Neste caso $N^*(x) \subset N(x)$ pois alguns vizinhos definidos pela estrutura de vizinhança estão proibidos. Esta estratégia evita retornar a configurações já visitadas e evita a chamada *ciclagem.*
- *Usando estratégias para diminuir a vizinhança ou a lista de configurações candidatas.* Geralmente o número de configurações $x' \in N(x)$ pode ser muito grande e avaliar a função objetivo de todas essas configurações para encontrar aquela que apresenta melhor desempenho pode precisar de elevado esforço computacional. **TS** apresenta pelo menos quatro métodos diferentes para encontrar uma vizinhança de tamanho reduzido tal que $N^*(x) \subset N(x)$. Esta estratégia visa diminuir o esforço computacional necessário em cada transição.
- *Usando configurações de elite e "path relinking" para caracterizar e encontrar novas configurações candidatas.* Esta estratégia visa encontrar novas configurações de alta qualidade que dificilmente seriam encontradas a partir da definição de $N^*(x)$. É fácil verificar que neste caso a relação $N^*(x) \subset N(x)$ já não é mais verdadeira.
- Redefinir o conjunto $N(x)$ durante o processo de otimização.

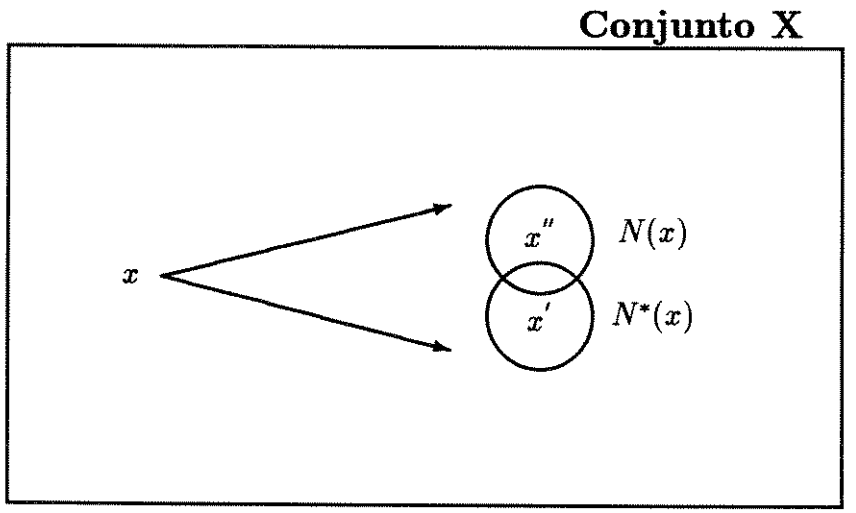


Figura 5.2: Transição em TS

Apresentamos a seguir as quatro principais estratégias de **TS**: *memória de curto prazo, memória de longo prazo, estratégica de oscilação* e *"path relinking"* .

5.2.1 Memória de Curto Prazo e seus Principais Elementos

Um algoritmo **TS** pode ser implementado simplesmente usando a estratégia de curto prazo; muitas das pesquisas iniciais foram implementadas desta forma. A memória de curto prazo usa basicamente a informação de atributos de configurações que foram modificados no passado recente. Esta informação é conhecida como *memória baseada em recência* ("recency") ou em fatos recentes. A idéia desta estratégia é considerar todos os atributos selecionados no passado recente como sendo proibidos; portanto, todas as configurações candidatas que possuem algum dos atributos proibidos (tabu-ativos) são excluídas na formação do conjunto $N^*(x) \subseteq N(x)$ de configurações vizinhas para avaliação.

A idéia básica da memória baseada em recência é evitar revisitar configurações já visitadas e aquelas configurações que compartilham os atributos tabu-ativos. Esta última parte pode representar um problema pois podem ser eliminados de $N^*(x)$ configurações muito atrativas. O critério de aspiração contorna em parte esta limitação. No problema de planejamento de sistemas de transmissão os atributos podem ser os circuitos que fazem parte de uma configuração. Assim, por exemplo, um *circuito que é adicionado* na configuração deve permanecer proibido de ser *retirado* nas próximas k configurações geradas, sendo k um número inteiro escolhido. De igual maneira, um circuito que é retirado da configuração corrente deve permanecer proibido de ser adicionado nas próximas k configurações. É evidente que nesta implementação devem existir duas listas de atributos proibidos.

A Figura 5.3 mostra uma iteração do algoritmo **TS** que usa memória de curto prazo sendo que uma iteração significa que, a partir da configuração corrente, deve-se avaliar todas as configurações candidatas $x' \in N^*(x)$ e escolher a melhor como sendo a nova configuração corrente.

Na parte A da Figura 5.3 são geradas todas as configurações candidatas $x' \in N(x)$. Na parte B é acionada a lista tabu. Podem ser criadas uma ou várias listas tabu dependendo do tipo do problema e do nível de sofisticação do algoritmo. A lista tabu pode ser operada de duas formas:

1. uma implementação contínua que não elimina automaticamente uma configuração candidata que possui atributos proibidos; assim configurações deste tipo são penalizadas na função objetivo (ou seu equivalente) em C para que se tornem pouco atrativas, e

2. uma implementação discreta em que toda configuração com atributos proibidos são eliminados automaticamente. Neste último caso, deve-se eliminar o bloco D e da saída de C voltar diretamente ao bloco A. O número k de iterações que um atributo permanece proibido é chamado de *status tabu* ("tabu tenure") e o tamanho de k pode

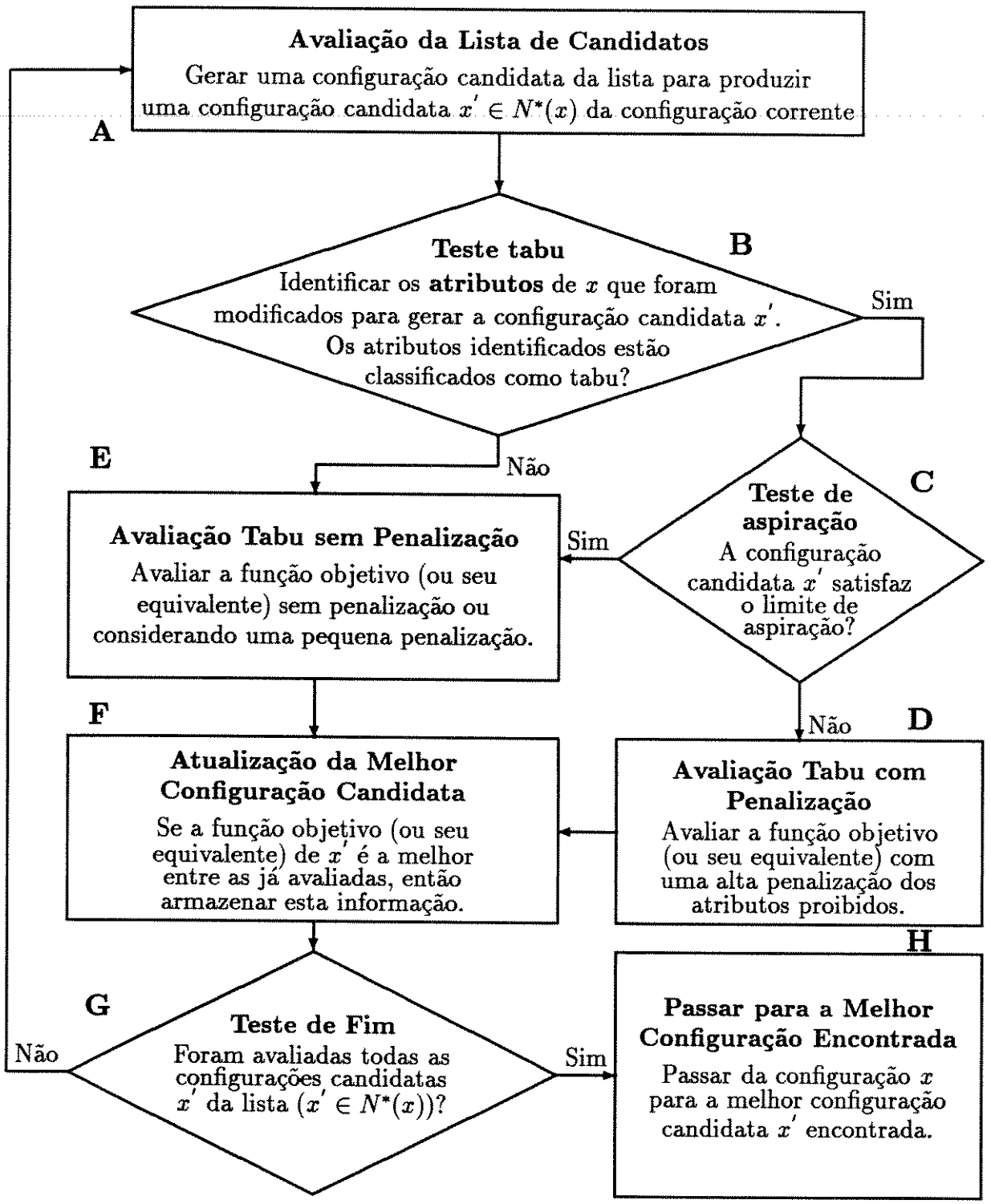


Figura 5.3: Execução de um movimento *TS* com memória de curto prazo

variar em cada lista e durante o processo produzindo algoritmos **TS** com diferentes tipos de desempenho.

No bloco C é implementado o chamado *critério de aspiração* cuja finalidade é fornecer maior flexibilidade ao processo. Neste caso uma configuração que possui atributo(s) proibido(s) pode ter a proibição descartada se essa configuração satisfaz um critério de aspiração. Na verdade aqui se tenta contornar a "rigidez" da lista tabu que armazena atributos proibidos. Uma configuração candidata de excelente qualidade pode ter um atributo proibido porque compartilha esse atributo com uma configuração já visitada no passado recente. Assim, a lista tabu ao proibir revisitar configurações visitadas no passado recente pode também proibir visitar configurações novas de excelente qualidade. Este fato é contornado pelo critério de aspiração que elimina a proibição de uma configuração candidata se ela produz uma função objetivo de boa qualidade.

Existem várias formas de definir o critério de aspiração. Obviamente, o critério mais rígido é definir o critério de aspiração da seguinte forma: se uma configuração candidata produz uma função objetivo melhor que a incumbente (melhor configuração já encontrada) então deve-se eliminar a proibição e armazenar essa configuração candidata no bloco F.

No bloco E da Figura 5.3, é realizada a avaliação da função objetivo (ou seu equivalente) para cada configuração candidata. No bloco F é armazenada a melhor configuração candidata encontrada; no bloco G é verificada se já todas as configurações candidatas foram analisadas e em H é realizada a transição em que a configuração corrente é substituída pela melhor configuração candidata encontrada.

Existe um último aspecto importante no algoritmo **TS** com memória de curto prazo: *a diminuição do tamanho de configurações candidatas que devem ser analisadas.* Para ilustrar este fato sejam:

$N(x)$ as configurações candidatas definidas pelo critério ou a estrutura de vizinhança,

$N^{**}(x)$ um subconjunto reduzido de $N(x)$ obtido usando algum critério de redução de configurações candidatas e

$N^{*}(x)$ os elementos de $N^{**}(x)$ que não estão proibidos pela lista tabu.

Para o problema de planejamento de sistemas de transmissão podemos definir, por exemplo, $N(x)$ como todas as configurações que podem ser obtidas a partir da configuração corrente da seguinte forma: adicionando um circuito, retirando um circuito ou trocando dois circuitos (adicionando um circuito e retirando outro circuito previamente adicionado). Para um sistema real esta definição de vizinho significa que $N(x)$ teria vários milhares de configurações candidatas e como a avaliação de cada configuração candidata precisa da resolução de um problema de PL, é evidente que esta definição de vizinho é impraticável

pois precisaria da resolução de vários milhares de PL para realizar um simples passo do algoritmo **TS**. Uma forma de contornar este problema seria redefinir a vizinhança.

TS contorna o problema anterior de uma forma mais eficiente encontrando um subconjunto reduzido $N^{**}(x)$ de $N(x)$, por exemplo da ordem de algumas dezenas no caso do problema de planejamento. **TS** possui quatro estratégias para realizar este processo de redução de configurações candidatas: *aspiração "plus", formação de uma lista reduzida de configurações candidatas de elite, filtros sucessivos* e *estratégia de mudança cotada.* Entretanto, na presente pesquisa foi implementada uma estratégia de redução de configurações candidatas ainda mais eficiente para o problema de planejamento de sistemas de transmissão e cujos detalhes serão apresentadas na próxima seção.

5.2.2 Memória de Longo Prazo e suas Estratégias Associadas

A incorporação de uma estratégia de longo prazo fornece ao algoritmo **TS** uma sofisticação adicional e um desempenho melhor na maioria das aplicações. Isto significa que além da memória de curto prazo (estratégia de todo algoritmo **TS** básico) é incorporada uma estratégia adicional chamada de memória de longo prazo.

Existem três aspectos fundamentais relacionados com a memória de longo prazo:

1. A memória baseada em freqüência,
2. A estratégia de intensificação, e
3. A estratégia de diversificação.

Memória Baseada em Freqüência

A *memória baseada em freqüência* consiste basicamente em armazenar a informação do número de vezes em que um atributo foi escolhido para gerar ou participar na formação das configurações durante o processo **TS**. Assim existem dois tipos de memória baseadas em freqüência: as *freqüências de transição* que armazena o número de vezes em que um atributo é retirado ou adicionado para formar novas configurações, e a *freqüência de residência ou permanência* que armazena a informação do número de vezes em que um atributo permanece nas novas configurações ou em todas as configurações geradas durante o processo **TS**. Esta informação baseada em freqüência pode ser utilizada para penalizar ou incentivar configurações formadas com determinados atributos.

No problema de planejamento de sistemas de transmissão é possível armazenar a informação do número de vezes em que um circuito é adicionado ou retirado na formação

das configurações no processo **TS** (freqüência de transição). De igual maneira, também é possível armazenar o número de vezes em que um circuito permanece adicionado em configurações consecutivas ou o número de configurações em que um circuito participa com adição (freqüência de residência).

Estratégia de Intensificação

A *estratégia de intensificação* consiste em mudar os critérios de seleção das novas configurações candidatas, isto é, mudar $N^{**}(x)$ (ou $N^{*}(x)$) para incentivar a formação de novas configurações aproveitando a informação acumulada no processo **TS**. Intensificação pode ser implementada de várias formas diferentes, tais como:

- Eliminando vizinhos ou incorporando novos vizinhos x' para a configuração corrente x, tipicamente considerando configurações de elite ou configurações com esses atributos como sendo "vizinhos" de x.
- Modificando a caracterização de configuração vizinha, isto é, redefinindo o conjunto $N(x)$.
- Retornando às regiões atrativas para realizar uma busca mais intensa na vizinhança das mesmas. A Figura 5.4 mostra uma estratégia para esta alternativa.
- Realizando intensificação por decomposição.

As duas primeiras alternativas dispensa comentários adicionais. Na intensificação retornando às regiões atrativas, mostrada na Figura 5.4, a parte mais importante é o bloco B identificado como *estratégia de seleção de configurações de elite*. Este bloco identifica a estratégia que deve ser implementada para encontrar um conjunto de configurações de elite em que cada uma delas identifica uma região promissora. Assim, a partir de cada uma das configurações de elite armazenadas na lista, deve-se implementar um novo processo de busca de excelentes configurações guiado por uma estratégia **TS** de memória de curto prazo como mostra o bloco D.

Existem três estratégias para encontrar as k configurações de elite que devem ser armazenadas na lista no bloco B que são mostradas a seguir:

- Introduzir alguma forma de diversificação para encontrar e armazenar configurações significativamente diferentes, apagar toda a memória de curto prazo e reinicializar o processo a partir da última configuração armazenada.

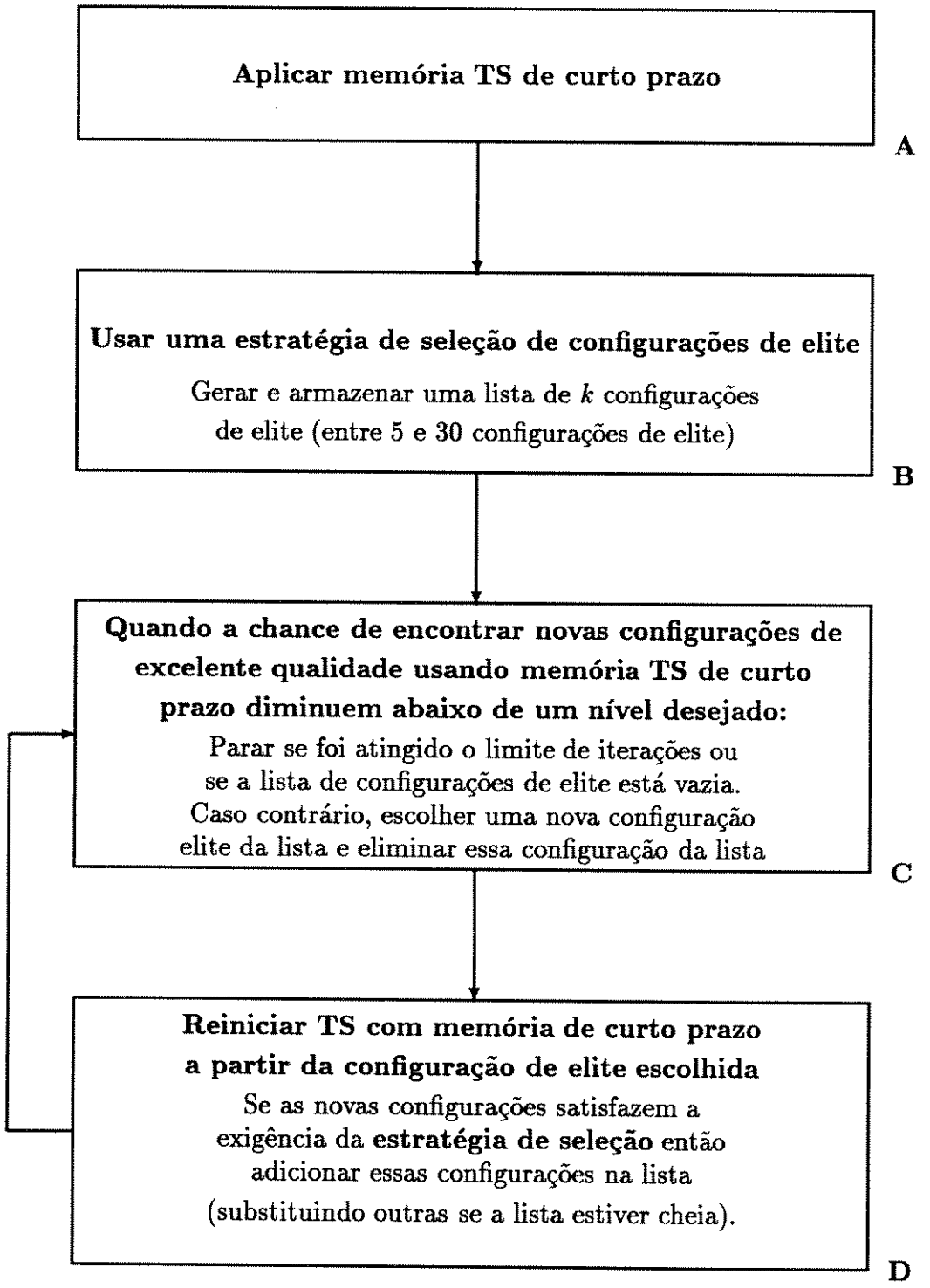


Figura 5.4: Implementação simples de intensificação em TS

- Formar e manter uma lista de tamanho especificado e adicionar uma nova configuração como último elemento da lista somente se ela é melhor que alguma configuração já armazenada. O último elemento da lista sempre é a primeira escolhida para reinicializar a busca. Neste caso é necessário armazenar a memória de curto prazo da última configuração armazenada para evitar voltar a visitar essa configuração.
- Uma terceira variante consiste em construir e manter em forma ordenada uma lista com as k melhores configurações encontradas durante um número determinado de iterações. Depois o processo é reiniciado, durante um número de iterações especificado, com a pior configuração e usando busca tabu probabilística para evitar retornar à mesma configuração. Durante este processo, qualquer configuração que seja melhor que a pior da lista deve ser incorporada na lista substituindo a pior configuração e colocada em forma ordenada na lista de acordo com o valor de sua função objetivo.

Outro tipo de intensificação é a técnica de *intensificação por decomposição* que consiste em impor novas restrições sobre atributos, estruturas ou partes do problema visando diminuir o espaço de configurações do problema. Assim, por exemplo, no problema de planejamento de sistemas de transmissão em algum momento do processo **TS** (por exemplo após terminar um **TS** de memória de curto prazo), pode-se usar intensificação por decomposição das seguintes formas:

- Fixando no valor zero o número de linhas candidatas que nunca foram adicionadas nas configurações visitadas (ou aquelas que foram raramente acionadas), isto é, eliminando um conjunto de circuitos como candidatos a adição (redução do número de candidatos).
- Incorporando para a configuração base aquelas linhas que aparecem em todas as configurações de elite. Neste caso, o número de linhas que deve ser incorporada em cada caminho candidato deve ser o menor que aparece entre as configurações de elite escolhidas para implementar a intensificação e, deve-se ainda permitir a adição de novas linhas nestes caminhos.
- Fixando estruturas completas de alguma configuração de elite. Por exemplo, se para eliminar uma grande demanda foi necessário adicionar um conjunto de linhas que estão formando uma espécie de bloco construtivo, então toda essa estrutura pode ser incorporada à configuração base.

As três propostas anteriores podem ser implementadas em conjunto ou separadamente na estratégia de intensificação por decomposição. Todas essas estratégias visam diminuir o número de alternativas para exploração e permitem focalizar a busca num número mais reduzido de alternativas. Assim, esta estratégia de intensificação por decomposição produz um efeito que pode ser chamado de implosão combinatorial em contraste com explosão combinatorial.

Estratégia de Diversificação

A estratégia de diversificação foi projetada para levar o processo de busca para regiões novas e atrativas. Esta estratégia é implementada mudando a definição de vizinhança ou de configurações candidatas $N(x)$, incorporando "vizinhos" constituídos por atributos que foram pouco usados. Nesta parte podem ser usadas as estruturas de memória usadas em intensificação ou em memória de curto prazo mas com critérios de duração modificados. A Figura 5.5 mostra uma estratégia simples de *diversificação* que mantém uma *memória baseada em freqüência* sobre todo o passado do processo.

Foram obtidas melhorias significativas com a aplicação de **TS** com memória de curto prazo da Figura 5.5. Os ótimos locais obtidos com esta técnica podem ser usados depois para uma seqüência de diversificação.

As estratégias de diversificação podem também usar uma forma de *memória de longo prazo baseado em recência* visando incrementar o estado tabu dos atributos das configurações visitadas. Uma versão simples desta metodologia é apresentada na Figura 5.6.

5.2.3 Oscilação Estratégica

Oscilação estratégica fornece um excelente mecanismo para realizar uma interação entre intensificação e diversificação a médio ou longo prazo. Esta técnica opera quando foi atingido um nível crítico em que o método TS normalmente pararia. Assim, antes de parar após atingido o nível crítico, pode-se mudar as regras de escolha de novas configurações candidatas (mudar $N(x)$) para passar fora do limite do nível crítico. Uma vez passado o nível crítico, muda-se novamente as regras de escolha de configurações candidatas tanto no afastamento como no retorno ao nível crítico. Um processo repetitivo desta estratégia produz um movimento oscilatório em torno do nível crítico.

O exemplo mais simples de oscilação estratégica acontece com o problema da mochila multidimensional. Neste caso as variáveis binárias são modificadas, por exemplo usando uma estratégia **TS** com memória de curto prazo, até chegar à fronteira da factibilidade. Nesse momento, pode-se entrar na região infactível, mudando a forma de avaliação da função objetivo ou seu equivalente na região infactível. Um processo de afastamento e retorno para a região de fronteira produz uma busca oscilatória que permitiria encontrar configurações factíveis de excelente qualidade.

No problema de planejamento de sistemas de transmissão, pode-se usar a estratégia de oscilação estratégica de uma maneira mais eficiente. Assim, por exemplo, pode-se iniciar o processo a partir de uma configuração infactível e usando uma estratégia **TS** com memória de curto prazo até encontrar uma configuração factível. Uma vez encontrada uma configuração factível, pode-se realizar um processo de intensificação aceitando

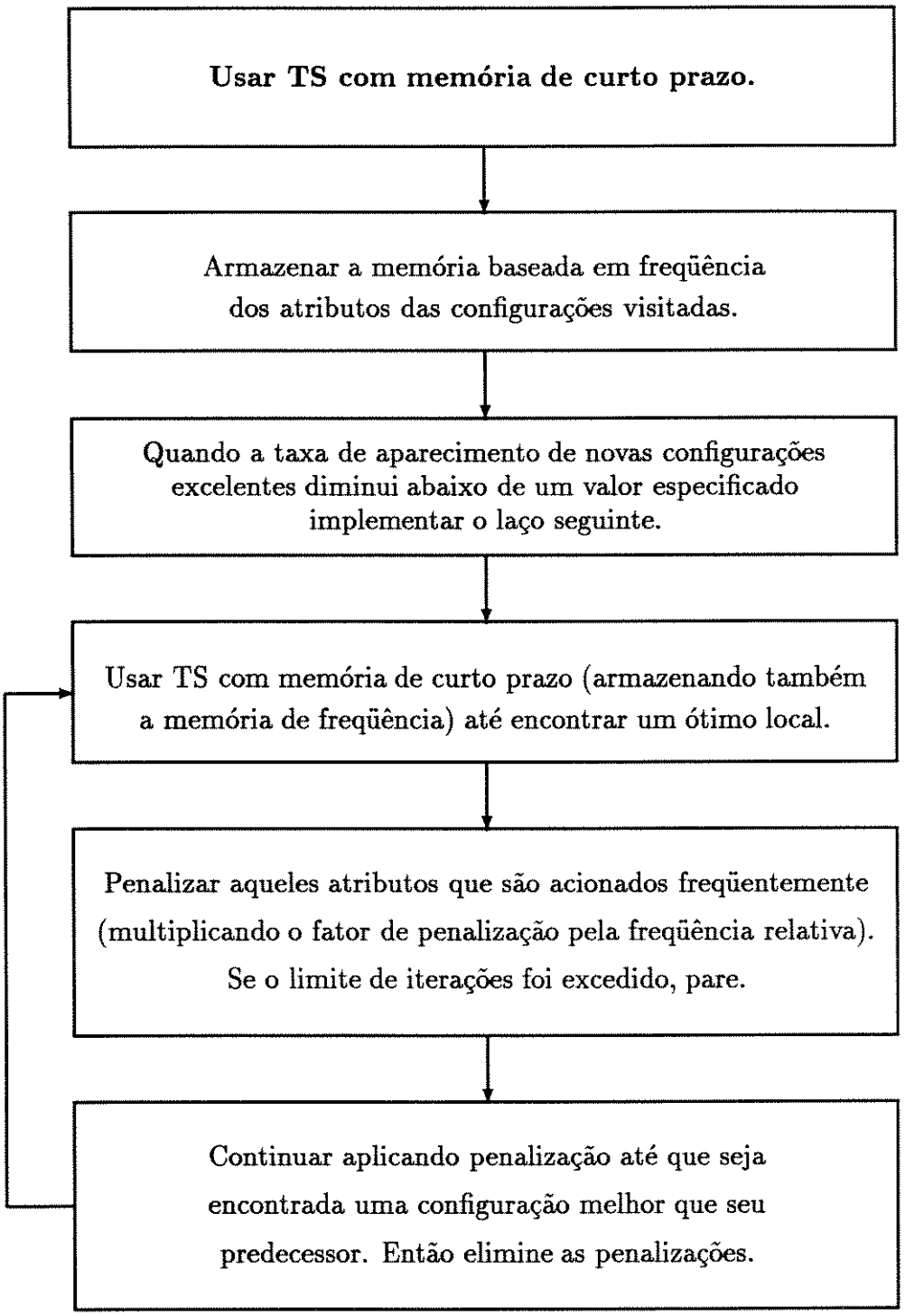


Figura 5.5: Implementação simples de diversificação em TS

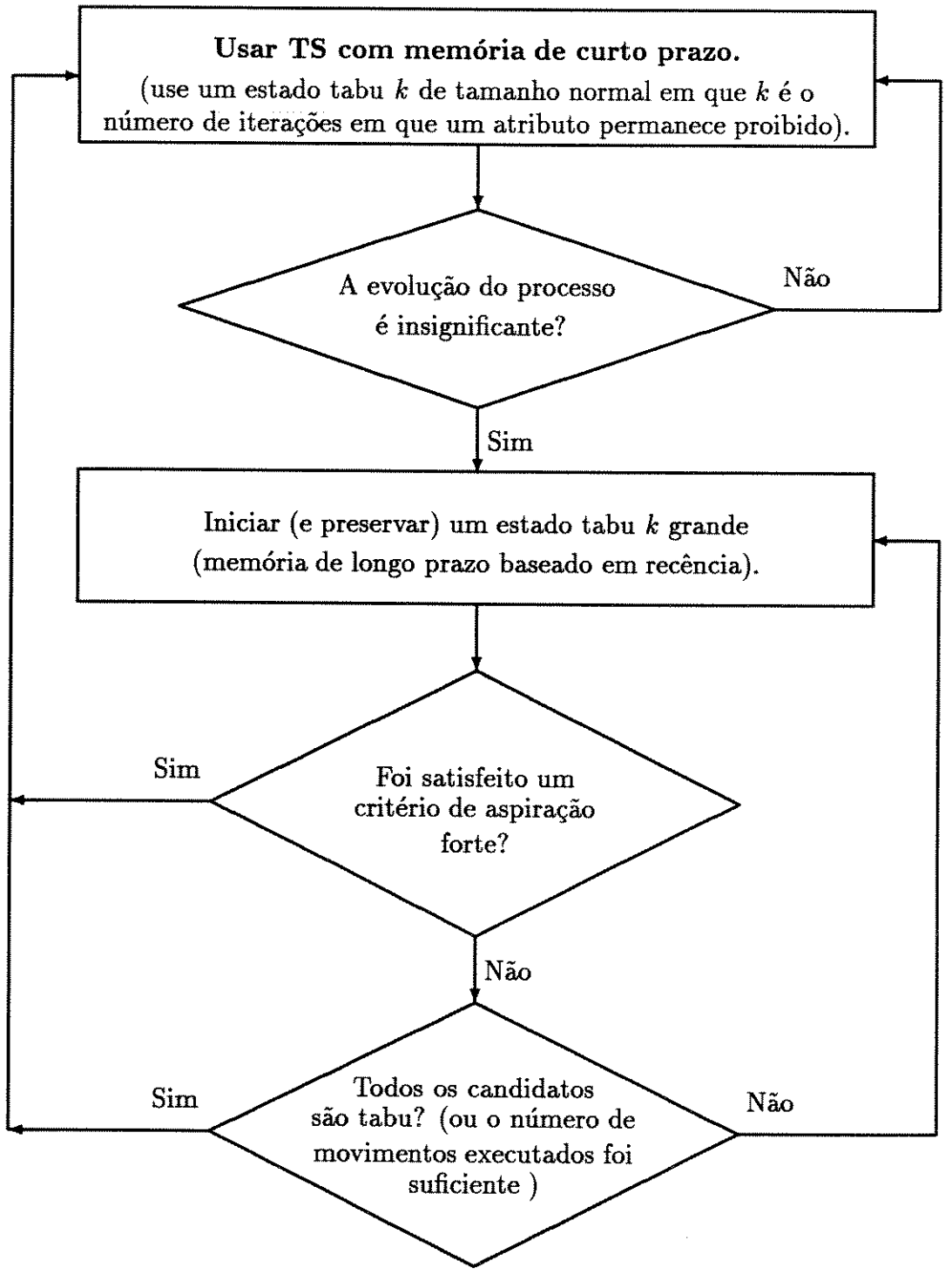


Figura 5.6: Diversificação usando memória de longo prazo baseado em recência.

simplesmente configurações factíveis que permitam encontrar o ótimo local correspondente a essa região. Uma vez terminada a intensificação, pode-se implementar uma rápida diversificação para entrar novamente na região infactível e reiniciar a estratégia **TS** de memória de curto prazo para voltar novamente a uma nova região factível e reiniciando a estratégia de busca oscilatória. Em cada parte do processo devem ser implementadas regras específicas para definir as configurações candidatas e evitar voltar a visitar regiões já visitadas. No presente trabalho foi implementada esta estratégia com um maior nível de sofisticação.

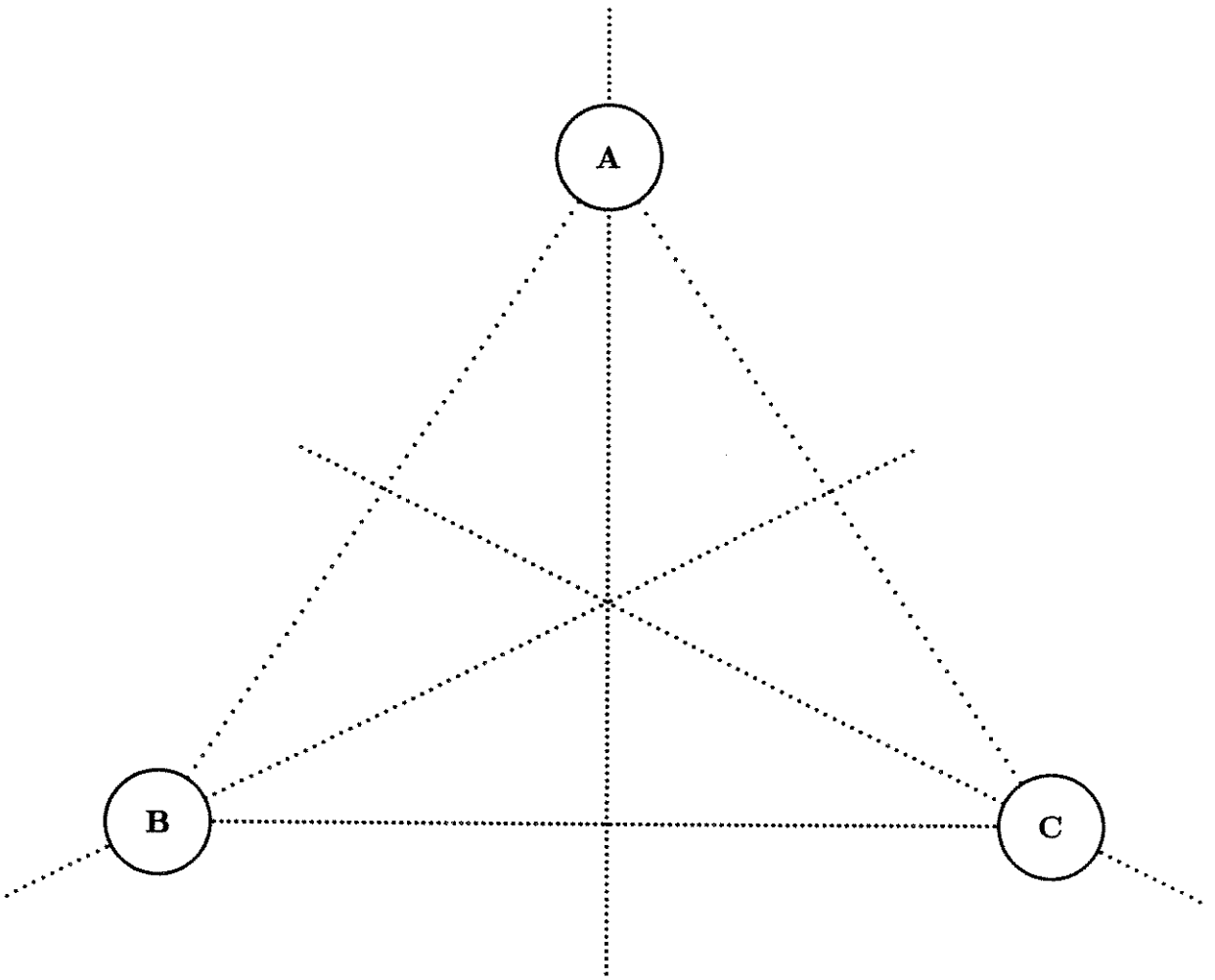
5.2.4 "Path Relinking"

"Path relinking" permite uma interessante integração entre intensificação e diversificação. Esta estratégia permite encontrar novas configurações usando as trajetórias que "unem" configurações de elite. Nesta estratégia é escolhida uma configuração de elite chamada de configuração base e uma ou várias configurações de elite chamadas de configurações guia. Assim, pode-se obter novas configurações a partir da configuração base e incorporando atributos presentes nas configurações guia. As configurações, assim obtidas, podem ser usadas para intensificação ou diversificação dependendo da quantidade de atributos incorporadas na configuração base.

A idéia de "path relinking" é obter configurações constituídas por atributos de alta qualidade, pertencentes a configurações de alta qualidade. Assim, na implementação de "path relinking" para encontrar uma configuração atrativa para continuar o processo, em cada passo é escolhido, de maneira preferencial, o atributo mais atrativo presente nas configurações guia e usando os critérios usuais de **TS**.

Na implementação de "path relinking", uma vez definida a configuração base (inicial) e as configurações guia de elite, deve-se continuar o processo **TS** na forma usual mas induzindo a escolha de atributos presentes nas configurações de elite. Este efeito é obtido tipicamente dando uma ponderação especial aos atributos presentes nas configurações de elite, "forçando" sua escolha em relação aos atributos que não se encontram presentes nas configurações guia. Neste contexto, implementar "path relinking" significa fazer uma mudança na caracterização da vizinhança $N(x)$ onde as configurações candidatas que possuem atributos presentes nas configurações guia recebem ponderação especial, por exemplo, incentivando sua função objetivo.

A função da configuração base e da configuração guia pode ser trocada nas configurações de elite e, portanto, as configurações de elite que participam em "path relinking" são denominadas simplesmente de *configurações referenciais*. A Figura 5.7 mostra uma forma de implementar "path relinking" onde são mostradas três configurações de elite e uma delas pode ser usada como configuração guia. As trajetórias que podem ser formadas pelas configurações geradas não são mostradas na figura, mas essas trajetórias geralmente



Intensificação: Gera trajetórias a partir de configurações similares.

Diversificação: Gera trajetórias a partir de configurações muito diferentes.

Aspiração: Gera desvios de trajetórias para vizinhos promissores.

Figura 5.7: "Path Relinking" em TS

devem ser mais complexas e com muitas inclinações.

"Path relinking" também pode ser usada para gerar novas configurações de uma maneira mais rápida. Assim, pode-se incorporar na configuração base e numa única operação todos os atributos considerados atrativos e presentes nas configurações guia. Dependendo da quantidade de atributos incorporados, a configuração assim obtida pode ser usada para intensificação ou diversificação. Esta estratégia foi implementada no problema de planejamento de sistemas de transmissão. Outra estratégia é produzir uma configuração para realizar intensificação usando configurações de elite que possuem muitos atributos parecidos ou então produzir uma configuração para diversificação usando configurações de elite que possuem atributos significativamente diferentes.

5.2.5 Aspectos Importantes na Implementação Prática

Nesta Seção, comenta-se brevemente alguns detalhes importantes na implementação prática de um algoritmo **TS**.

A) Estruturas de Memória Baseada em Recência e em Freqüência

Estrutura de Memória Baseada em Recência:

Uma estrutura de memória baseada em recência é usada basicamente para identificar os atributos que se encontram ativos (proibidos) e, portanto, todas as configurações vizinhas ($x' \in N(x)$) que possuem estes atributos também devem estar proibidas e assim devem ser excluídas de $N^*(x)$. Denotemos por $S = \{1, 2, ..., s\}$ como sendo um conjunto de índices que identificam atributos de uma configuração. Para o problema de planejamento de sistemas de transmissão, supor que o índice $i \in S$ representa um caminho candidato (circuito) a adição no sistema elétrico. Supor que em determinado momento foi adicionada uma linha no caminho j, então deve-se proibir que essa linha seja retirada durante um número especificado de iterações. Uma forma simples de implementar a memória de curto prazo para esta parte do problema deve levar em conta os seguintes passos:

- Inicializar um vetor $tabu_start(i) = 0$ que é um vetor que deve permitir identificar o estado tabu de um atributo. Como foi adicionado um circuito, então deve-se proibir a retirada desse circuito.
- Supor que numa determinada iteração do processo, identificado pelo contador de iterações *iter_corrente*, o atributo $j \in S$ é acionado para adição (é adicionado um circuito no caminho j). Assim, deve-se proibir a *retirada* de um circuito no caminho j durante um número de iterações que identificamos como $dura_tabu(j)$.

- Fazer $tabu_start(j) = iter_corrente$
- Portanto, a retirada do circuito j permanece proibida sempre que a seguinte relação for verdadeira:

$$tabu_start(j) + dura_tabu(j) \geq iter_corrente$$

Como *iter_corrente* é atualizada a cada iteração então o estado tabu deve ser desativado após $dura_tabu(j)$ iterações.

Um problema complexo, como o problema de planejamento de sistemas de transmissão, deve implementar várias estruturas de memória de curto prazo proibindo diferentes tipos de atributos tais como: adição de circuitos, retirada de circuitos, adição e/ou retirada de dois circuitos simultaneamente, etc.

Um aspecto particularmente importante em **TS** é definir o valor de cada elemento do vetor *dura_tabu*(.), isto é, o número de iterações em que um atributo permanece proibido. Uma forma simples é especificar um valor fixo para todos os atributos e tipicamente é escolhido o tamanho de *dura_tabu* como sendo uma fração do tamanho de S, isto é, do número de circuitos candidatos no problema de planejamento de sistemas de transmissão. Entretanto, existem formas mais sofisticadas de implementação que levam em conta o histórico do processo e características específicas do problema ou dos atributos. Esta implementação dinâmica, conhecida como *estado tabu adaptativo*, usa valores diferentes de *dura_tabu*(.) para cada atributo e geralmente é composta de duas partes: uma parte fixa comum para todos os atributos e uma outra parte variável para cada atributo.

O tamanho de *estado_tabu*(.) não deve ser muito pequeno, pois nesse caso pode acontecer ciclagem e, não pode ser muito grande, pois nesse caso eliminaria a possibilidade de aceitar configurações vizinhas de boa qualidade. Embora este último caso seja contornado em parte pelo critério de aspiração, o tamanho de *estado_tabu*(.) deve ter um valor intermediário que evite ciclagem e permita aceitar boas configurações vizinhas.

Estrutura de Memória Baseada em Freqüência:

Neste caso existem dois tipos de memória: *memória de transição* e *memória de residência (permanência)*.

Para a *memória de transição* no problema de planejamento de sistemas de transmissão onde é possível adicionar vários circuitos para cada caminho candidato, pode-se implementar duas listas tabu: uma quando é realizada a retirada de circuito e outra quando é realizada a adição de circuitos. Uma implementação mais complexa pode ainda armazenar trocas de circuitos.

A implementação de *memória de residência* no problema de planejamento de sistemas de transmissão precisa de uma notação matricial mas sua implementação é relativamente simples.

B) Redução da Lista de Configurações Candidatas

Uma estratégia crucial em **TS** é a obtenção de uma lista reduzida de configurações candidatas, isto é a obtenção dos elementos de $N^{**}(x)$. **TS** não deve analisar todas as configurações vizinhas $x' \in N(x)$ pois normalmente esse número de configurações é muito grande. O caráter agressivo de **TS** exige que, a partir da configuração corrente, deve-se passar para o melhor vizinho $x' \in N(x)$, o que significa que para realizar uma simples transição deve-se avaliar a função objetivo de todas as configurações de $N(x)$. Como o número de elementos de $N(x)$ é muito grande, o esforço computacional para realizar uma simples transição seria muito grande.

TS contorna o problema anterior avaliando simplesmente um subconjunto reduzido $N^{**}(x) \subset N(x)$ e escolhendo a melhor configuração desse subconjunto reduzido. O grande problema é como encontrar de maneira eficiente os elementos de $N^{**}(x)$ levando em conta que esse subconjunto deve ter poucos elementos e de preferência as melhores configurações de $N(x)$. Assim, a melhor estratégia para caracterizar $N^{**}(x)$ é aquela que produz poucos elementos e não exclui as melhores configurações de $N(x)$. Na literatura de **TS** existem as seguintes estratégias para encontrar $N^{**}(x)$:

- Analisar um número especificado de configurações de $N(x)$ *aleatoriamente escolhidas.*
- *Aspiração Adicional (Aspiração "Plus"):*

  Nesta estratégia são analisadas as configurações que pertencem a $N(x)$ até que seja satisfeito algum critério de qualidade que pode ser, por exemplo, a redução da função objetivo em um determinado valor ou algum outro critério. Uma vez encontrada uma configuração que satisfaz esse critério de qualidade, avalia-se ainda um número determinado de configurações adicionais ("plus"). É selecionada a melhor configuração encontrada dentre as analisadas e para contornar alguns problemas evidentes é fixado um número mínimo e máximo de configurações que devem ser analisadas. Todos os critérios (critério de qualidade, valores do número mínimo e máximo de iterações a serem analisadas e o tamanho da aspiração "plus") podem ser modificados dinamicamente.
- *Lista Reduzida de Candidatos de Elite*

  Esta estratégia consiste em analisar todos ou uma grande maioria de movimentos (atributos) que produzem as configurações candidatas $x' \in N(x)$ e armazenar um conjunto reduzido destes movimentos em uma lista de elite, por exemplo, entre

15 a 50 elementos. Assim, durante um número determinado de iterações, a nova configuração é escolhida usando os atributos armazenados na lista de elite. Este processo é repetido até atingir um número determinado de iterações ou até que a eficiência desse processo de escolha diminua abaixo de um nível esperado. Depois, deve-se elaborar uma nova lista de elite analisando novamente um número grande de elementos de $N(x)$. Este processo implica reavaliar a função objetivo para cada atributo da lista de elite porque a inclusão de um atributo prévio pode produzir um reordenamento das prioridades dos atributos da lista de elite.

No problema de planejamento de sistemas de transmissão foi usada uma metodologia de encontrar configurações candidatas ainda mais eficiente. Assim, os elementos (atributos) que identificam as configurações vizinhas $x' \in N^{**}(x)$ é constituído por um grupo muito reduzido de atributos (circuitos) fornecidos pelos melhores algoritmos heurísticos construtivos usados no problema de planejamento de sistemas de transmissão.

5.3 BUSCA TABU APLICADO AO PROBLEMA DE PLANEJAMENTO

5.3.1 Apresentação do Problema

TS é um algoritmo que está sendo amplamente usado no campo da pesquisa operacional e existem muitas publicações de caráter geral, entre as quais podem ser citadas [54, 55].

Nesta seção, é apresentada a característica fundamental de funcionamento do algoritmo **TS** aplicado ao problema de planejamento da expansão de sistemas de transmissão. A Figura 5.8 mostra o módulo básico de funcionamento do algoritmo **TS**.

No módulo mostrado na Figura 5.8, o processo de otimização é iniciado com uma configuração inicial que pode ser obtida aleatoriamente ou, melhor ainda, pode ser obtida por uma modificação do algoritmo de Garver [9]. Esta configuração é submetida a um processo de **TS** com memória de curto prazo seguido de intensificação (**TS**/*intensificação*), cujo mecanismo de funcionamento é apresentado posteriormente. Após o processo de **TS**/intensificação, deve-se realizar um processo de diversificação visando levar o processo de busca às regiões atrativas e ainda não visitadas. Depois de realizada a diversificação, a configuração corrente está pronta para um novo processo **TS**/intensificação. Este processo pode ser repetido de maneira cíclica até que seja satisfeito um critério de parada.

O mecanismo de implementação de **TS**/intensificação e de diversificação, de maneira cíclica, constitui a base fundamental de funcionamento de um algoritmo **TS**. No presente

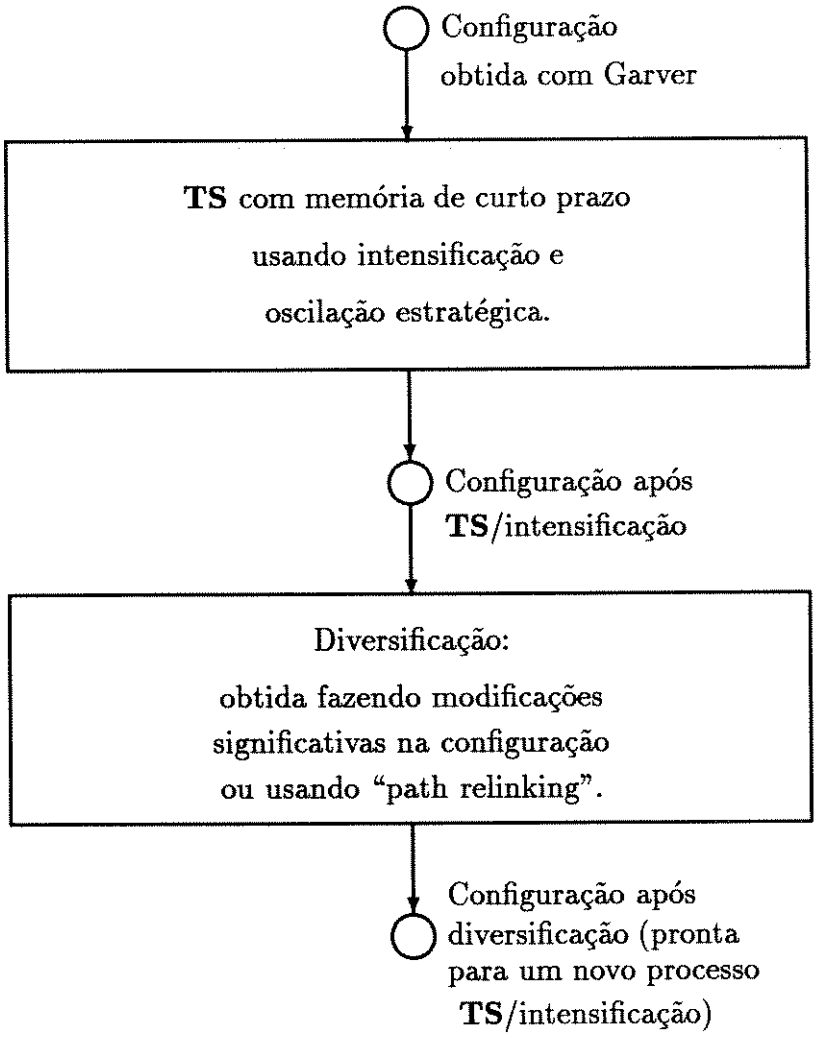


Figura 5.8: Mecanismo de "Tabu Search"

trabalho são implementados formas sofisticadas de intensificação e diversificação e, adicionalmente, é implementada uma estratégia mais geral, sistematizada em um algoritmo **TS**, que usa funções próprias dos algoritmos "simulated annealing" e genético. Também é analisado a influência do uso de configurações obtidas usando a metodologia modificada de Garver, como também configurações obtidas de maneira aleatória.

5.3.2 Solução do Problema por *TS*

Foram implementados dois tipos de algoritmos **TS** utilizando os mecanismos de **TS**/intensificação e diversificação apresentados na Figura 5.8. A diferença básica entre eles consiste que, no primeiro, o processo é inicializado com uma única configuração;

no segundo, o processo inicia com k configurações.

Na Figura 5.9 é apresentado o algoritmo **TS** com uma única configuração de partida. Neste a configuração inicial é submetida a um processo **TS**/intensificação. Durante este processo, as p melhores configurações obtidas são armazenadas. No passo seguinte, das configurações armazenadas é selecionada aquela de melhor função objetivo a qual é submetida a um processo de diversificação-**TS**/intensificação; implementado de maneira semelhante a da Figura 5.10. Durante o processo de **TS**/intensificação, cada vez que se obtém uma configuração com pequeno corte de carga, é comparada sua função objetivo com a das armazenadas; se é melhor, substitui-se a de pior função objetivo. O número de configurações substituídas é limitado a p. Caso se obtenha mais de p configurações com melhor função objetivo, se substitui alguma das p anteriormente armazenadas.

Das configurações armazenadas, seleciona-se as melhores para formar a *lista elite*, a qual será empregada no processo de diversificação. O processo termina quando todas as k configurações armazenadas tiverem indicação de que já foram estudadas.

A seguir, é apresentado com maiores detalhes o algoritmo para o caso de k configurações de partida.

Na Figura 5.10, no nível I, é mostrado um conjunto de k configurações iniciais para iniciar o algoritmo **TS**. Neste sentido, este algoritmo é diferente do algoritmo **TS** apresentado na Figura 5.9, que inicia o processo com uma única configuração inicial. Portanto, nesta parte do processo, o algoritmo apresentado usa um conjunto de configurações de maneira similar ao conceito de população inicial no algoritmo genético. As k configurações iniciais podem ser geradas aleatoriamente ou usando um algoritmo aproximado eficiente. Os resultados obtidos mostram que o uso de configurações iniciais obtidas usando métodos aproximados, tipicamente uma versão modificada do algoritmo de Garver, levam a uma melhoria significativa do algoritmo **TS** especialmente na diminuição do esforço computacional.

Todas as configurações iniciais, mostradas no nível I, são submetidas a um processo de **TS**/intensificação, cada configuração independentemente das outras. Assim, as configurações mostradas no nível II são as configurações iniciais após um processo de **TS**/intensificação. Esta fase independente é necessária a fim de melhorar a qualidade das configurações e encontrar k configurações com topologias significativamente diferentes.

A partir do nível II é realizado um processo de otimização integrado. Assim, antes de iniciar esta fase, deve-se constituir a *lista de elite* formada por um subconjunto das melhores configurações do nível II. No nível II, cada uma das configurações serão submetidas a um processo cíclico de diversificação e um processo **TS**/intensificação.

No nível II, o processo é iniciado pela configuração com melhor função objetivo. Na figura 5.10, supor que a configuração 3 tem a melhor função objetivo. Assim, a configuração

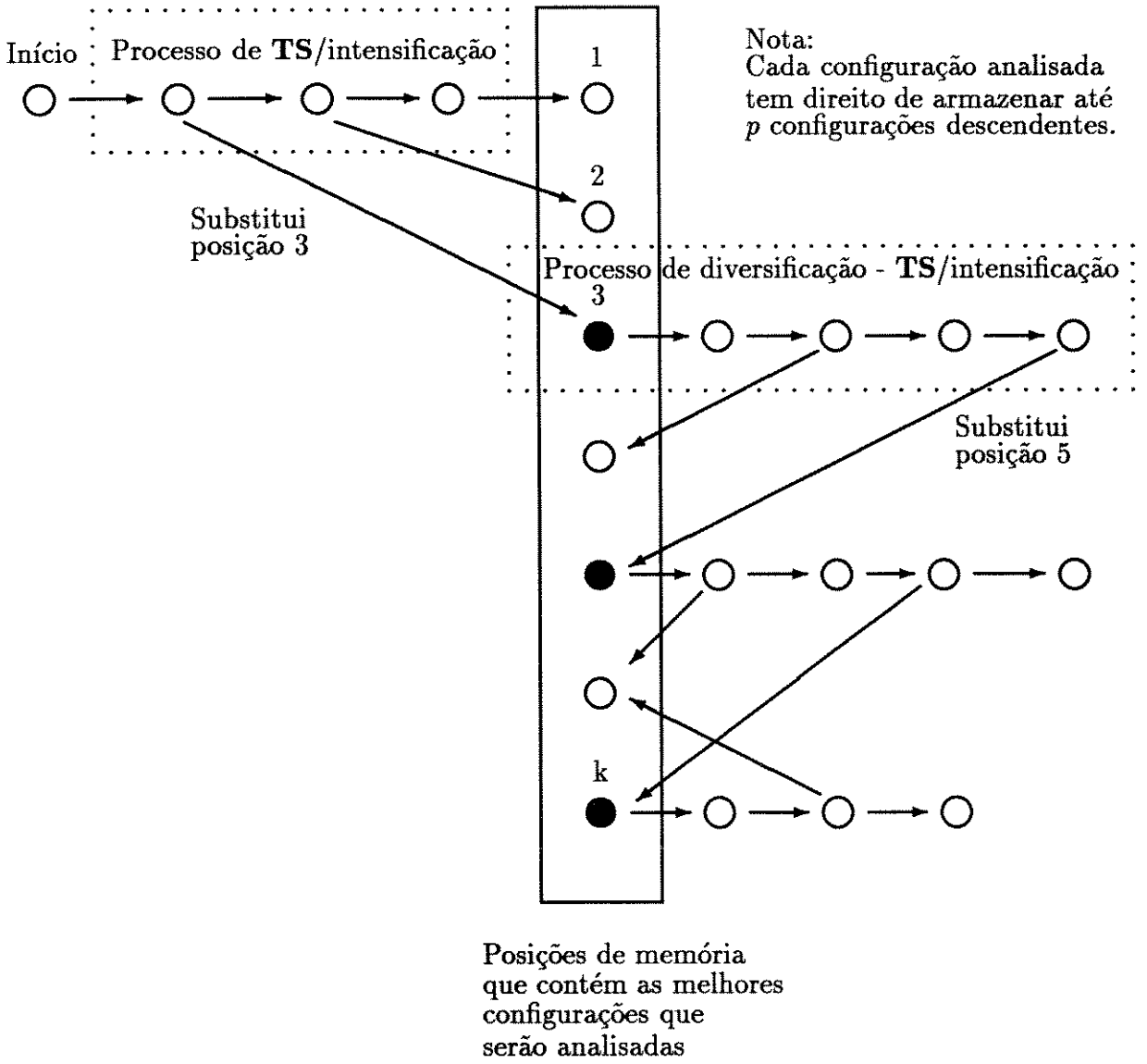


Figura 5.9: Algoritmo de Tabu Search com uma única configuração de partida

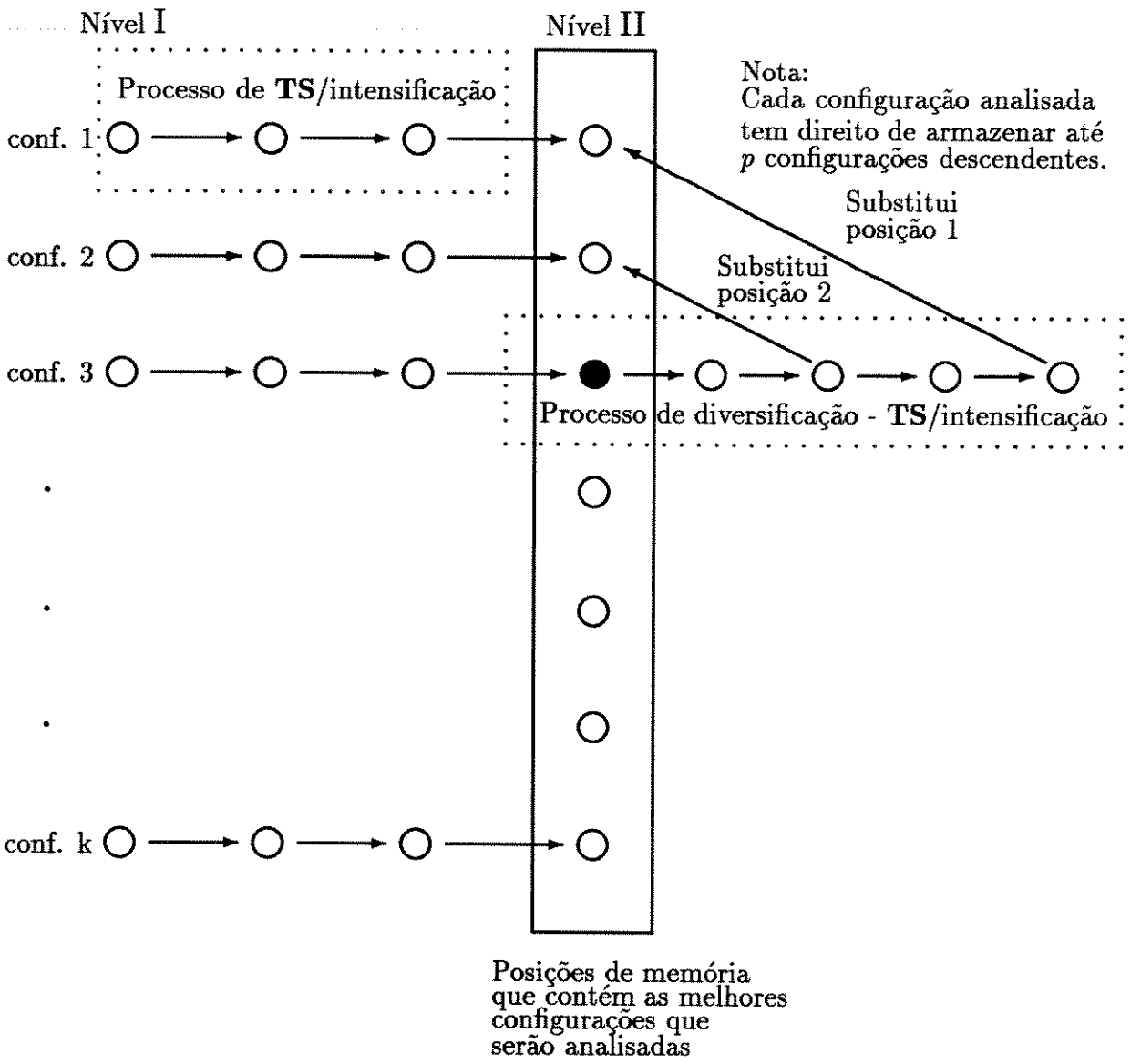


Figura 5.10: Algoritmo de Tabu Search com k configurações de partida

3 é submetida a um processo de diversificação seguido de um processo **TS**/intensificação. No processo de **TS**/intensificação, toda vez que é obtida uma configuração com pequeno corte de carga, deve-se comparar sua função objetivo com os objetivos correntes das outras $(k-1)$ configurações. Se a configuração encontrada é melhor que alguma das $(k-1)$ configurações da lista, então, deve-se substituir essa configuração pela configuração encontrada. Por exemplo, na figura 5.10, a diversificação-**TS**/intensificação da configuração 3 encontrou duas configurações atrativas que substituirão as configurações 1 e 2. Na verdade, cada vez que uma configuração é explorada através de diversificação-**TS**/intensificação, e aparece uma configuração com pequeno corte de carga, deve-se verificar a possibilidade de atualizar as configurações da lista (como explicado anteriormente), as configurações de elite que são usadas para "path relinking" e a incumbente do processo integral.

Toda vez que uma configuração é analisada, deve-se colocar a indicação de que a mesma já foi explorada. Esta informação é mantida usando um vetor auxiliar. Assim, na figura 5.10, antes de iniciar a exploração da configuração 3, deve-se indicar no vetor auxiliar a informação de que esta configuração já foi explorada bem como seu valor da função objetivo.

Na diversificação-**TS**/intensificação da configuração 3, eventualmente, podem aparecer configurações atrativas que podem substituir as outras configurações e inclusive a própria configuração 3 armazenada no vetor auxiliar. Este processo finaliza após a última configuração corrente da diversificação-**TS**/intensificação da configuração 3 ser analisada. Além disso, toda vez que uma configuração da lista é substituída por outra configuração de melhor qualidade, além de substituir a configuração, deve-se retirar a indicação de que a mesma já estava explorada.

O processo de diversificação-**TS**/intensificação continua, em cada passo, escolhendo a configuração da lista que tem uma melhor função objetivo e que não tem a indicação de que já foi analisada. O processo integral do algoritmo **TS** pára quando todas as configurações da lista tem a indicação de que já foram analisadas ou se tem esgotado um número máximo de chamadas ao algoritmo de PL.

5.3.3 O Problema da Estrutura de Vizinhança

TS é um procedimento que guia um algoritmo de busca local para procurar o espaço de soluções além da simples otimalidade local. Isto se consegue através de uma busca agressiva, selecionando o melhor dos movimentos em cada passo.

TS se baseia em procedimentos determinísticos em vez de procedimentos aleatórios, como no caso de **SA** e **AG**. **TS** faz uso de estratégias especiais de busca, empregando diferentes tipos de memória e selecionando o melhor dos movimentos a cada iteração. Nestas circustâncias, a estrutura de vizinhança deverá restringir o número de linhas can-

didatas, já que uma estrutura com muitos vizinhos é onerosa do ponto de vista do tempo de computação gasto para avaliar cada uma das alternativas, que neste caso seria de um PL para cada elemento da lista que é analisado. Assim, para melhorar este aspecto, usa-se então estratégias de listas de candidatas com a finalidade de restringir o número de soluções examinadas em uma dada iteração. As linhas candidatas deverão ser selecionadas de maneira cuidadosa; dessa forma, regras eficientes para gerar as listas de candidatas são críticas neste processo.

Baseando-se nestas características, foi implementada uma estrutura de vizinhança que permita identificar os novos vizinhos; este processo de seleção dos vizinhos é feito através de listas elites de linhas candidatas. Existem então dois tipos de listas: para o corte de carga da configuração candidata maior ou menor que um limite estabelecido. Assim, se a configuração analisada tem um corte de carga maior que o limite estabelecido, a lista será formada por linhas a serem adicionadas/trocadas, caso contrário, quando o corte de carga é inferior ao limite, a lista será formada por linhas candidatas a serem retiradas/trocadas; este procedimento para a formação destas listas está baseado em métodos aproximados [9, 12, 35], como será explicado mais adiante.

As listas elite de linhas candidatas são empregadas tanto na etapa de **TS**/intensificação como na etapa de diversificação, efetuando procedimentos de entrada/troca ou retirada/troca de linhas.

Além destas listas, existem alguns vizinhos que são selecionados de maneira aleatória tanto na etapa de **TS**/intensificação como na de diversificação. Ou seja, na etapa de **TS**/intensificação, se após aplicar os procedimentos de entrada/troca ou retirada/troca de linhas, com base nas listas elite, não se obtém uma configuração com melhor função objetivo, são selecionadas de maneira aleatória algumas linhas e se efetua um procedimento de troca. Na etapa de diversificação, além das diversificações feitas com base nas listas elite, são efetuadas algumas trocas de linhas selecionadas de maneira aleatória.

5.3.4 Características de Memória de Curto e Longo Prazo

Neste algoritmo são empregadas estes dois tipos de memórias: de curto e longo prazo. A memória de curto prazo mais comumente usada, leva em conta os atributos residentes nas soluções ou que tenham sido trocados em passado recente, bem como soluções elite visitadas durante a busca. Na memória de longo prazo são armazenadas informações baseadas na freqüência; estas operam introduzindo penalizações e incentivos, determinados pelo tempo em que um atributo tenha permanecido em uma solução. Cada tipo de memória está acompanhada de suas próprias estratégias especiais.

Memória de Curto Prazo

Neste tipo de memória serão armazenados os atributos das soluções, levando em conta os do passado recente (designados como *tabu-ativos*), sendo que as configurações que os contém se convertem em tabu; com isto, se impedirá que soluções recentemente visitadas sejam revisitadas.

Neste processo foram implementadas dois tipos de listas tabu, que armazenam os seguintes atributos:

- Linha que ao ser proposta para entrar e foi aceita,
- Linha que ao ser proposta para sair e foi aceita.

Nestas listas é armazenado então a duração em que um atributo (linha adicionada ou retirada) permanece tabu-ativo (medido em iterações). Esta duração é variável e permite atribuir diferentes valores a cada atributo. No cálculo da duração tabu, deve-se ter em conta de que a mesma não apresente os seguintes problemas:

- Ocorrência de ciclagem,
- Deterioração na qualidade da solução.

O primeiro caso pode apresentar-se devido ao fato de que a duração tabu seja relativamente curta; o segundo pode apresentar-se devido a uma duração tabu longa. A efetividade da duração tabu é empiricamente encontrada e depende do tamanho do problema. Na figura 5.11, apresenta-se os efeitos de selecionar uma duração tabu de maneira inadequada. A melhor escolha é um valor intermediário entre os dois extremos.

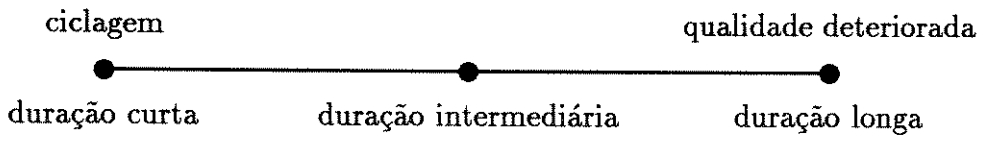


Figura 5.11: Duração tabu

Neste caso foi implementada uma duração variável ao longo do tempo, chamada de *duração dinâmica*, constituída por uma parcela fixa e outra variável; então, para o cálculo

da duração tabu, à parcela fixa se soma outra cujo valor é calculado como o produto de um valor fixo por um número aleatório selecionado no intervalo [0,1].

Neste problema também se emprega o *critério de aspiração*, o qual introduz um elemento importante de flexibilidade na busca tabu. Este processo consiste em admitir soluções de boa qualidade, mesmo que estas estejam classificadas como tabu. No desenvolvimento deste problema emprega-se um critério simples de aspiração que consiste em remover a classificação tabu quando o movimento produz uma melhor solução do que a obtida (incumbente) ou uma solução melhor que as obtidas durante um número especificado de iterações.

Outros elementos empregados são as listas de elementos candidatos, as quais dão um caráter agressivo ao algoritmo **TS** procurando o melhor movimento disponível que possa ser determinado com uma quantidade apropriada de esforço computacional. Estas listas são formadas usando métodos aproximados e critérios de planificação, como será explicado mais adiante.

Memória de Longo Prazo

Tipos de memórias baseadas em freqüência são fundamentais em considerações de longo prazo. Estas operam introduzindo penalidades e incentivos, e são determinadas pelo tempo em que um atributo permanece ativo nas soluções visitadas durante a busca. *Freqüências de transição*, mantém um registro da freqüência com que se trocam os atributos; *freqüência de residência*, mantém um registro (n_o. de iterações) em que os atributos permanecem nas configurações.

A informação de freqüência é empregada no processo de diversificação, e está projetada para conduzir a busca na exploração de novas regiões. Com a informação de freqüência se deseja modificar as regras de eleição para levar à solução atributos que não tenham sido usados freqüentemente. Durante o processo é contabilizado o número de vezes em que, ao adicionar ou retirar uma linha, melhora a função objetivo; assim como também o número de vezes em que uma linha reside nas configurações analisadas. Também é armazenada informação referente às trocas que melhoram e que não melhoram a função objetivo.

Neste trabalho foram selecionados diferentes tipos de freqüência dos atributos aplicáveis ao problema do planejamento da expansão de redes de transmissão, os quais estão contidos em vetores manejados de maneira dinâmica e seu valor é acrescido na medida em que evolui o processo; também foi levado em conta a informação referente a trocas, armazenadas em matrizes. As informações armazenadas em vetores e matrizes são as seguintes:

- *Freqüência por adição*, conta o número de vezes em que uma linha ao ser adicionada melhora o valor da função objetivo da configuração analisada.

- *Freqüência por retirada*, conta o número de vezes em que uma linha ao ser retirada melhora o valor da função objetivo da configuração analisada.
- *Freqüência por residência*, conta o número de vezes em que uma linha está presente na configuração analisada.
- *Freqüência por troca que melhora*, conta o número de vezes em que uma troca de linhas melhora o valor da função objetivo. Neste caso, emprega-se uma matriz para armazenar a informação.
- *Freqüência por troca que não melhora*, conta o número de vezes em que uma troca de linhas não melhora o valor da função objetivo.

A informação de freqüência poderá ser usada de duas maneiras: A primeira se refere a penalizar (ou incentivar) o valor da função objetivo analisada; a segunda refere-se a usá-la como índice para selecionar as linhas à adicionar, à retirar ou à trocar na configuração.

No primeiro caso, a informação de freqüência é multiplicada por um fator de diversidade, o qual traduz de unidades de freqüência à valores da função objetivo; o valor obtido é adicionado (ou subtraído) do valor da função objetivo da seguinte maneira:

$f.o._{nova} = f.o._{anterior} + fd1.$freqüência por adição - $fd2.$freqüência por residência

$f.o._{nova} = f.o._{anterior} + fd1.$freqüência por retirada - $fd2.$freqüência por residência

No segundo caso, os valores de freqüência são empregados para selecionar diretamente as linhas que são adicionadas, retiradas ou trocadas na configuração. Assim, são efetuadas trocas que historicamente têm estado presentes em configurações de boa qualidade, assim como também são efetuadas trocas que não tenham sido realizadas anteriormente. O processo também proíbe troca de linhas que historicamente não têm dado melhoria na função objetivo das configurações analisadas.

Para complementar o processo de diversificação é feita uma troca de linhas que são selecionadas de maneira aleatória.

Como parte das estratégias de longo prazo foram implementadas funções como: a de eliminar a memória de curto prazo da configuração anterior antes de reiniciar o processo. Isto é feito às soluções contidas nas listas para serem analisadas tal como se mostra nas Figuras 5.9 e 5.10.

Outra estratégia de longo prazo é manter uma lista de tamanho limitada, que adiciona ao final uma nova solução só se a mesma for melhor que quaisquer das anteriormente armazenadas.

No tipo de memória explícita, são registradas soluções completas e consiste de uma

elite de soluções visitadas durante a busca, as quais são armazenadas. O número de configurações elite armazenadas estará limitado pelos grandes requerimentos de memória.

Na aplicação do problema proposto, deve-se diferenciar dois tipos de configurações elite: o primeiro grupo formado pelas configurações elite candidatas, as quais são analisadas durante o processo de diversificação - **TS**/intensificação; o outro grupo é formado pelas melhores soluções encontradas durante o processo e que na verdade constitui um grupo reduzido de configurações "vizinhas" altamente atrativas e que poderão ser empregadas no processo de "path relinking".

5.3.5 Intensificação e Diversificação no Algoritmo Implementado

Estrategia de memória de curto prazo usando intensificação e oscilação estratégica

O processo **TS**/intensificação para o problema de planejamento de sistemas de transmissão é realizado em um processo que apresenta uma estratégia com três partes claramente diferenciadas, conforme esta ilustrado na Figura 5.12:

1. A parte A realiza adição/troca de circuitos quando a configuração corrente apresenta corte de carga, (região de configurações não factíveis)
2. A parte B realiza a retirada/troca de circuitos quando a configuração corrente não apresenta corte de carga (região de configurações factíveis onde é implementada a intensificação), e
3. A parte C realiza retirada de circuitos quando não existe uma configuração factível vizinha (sem corte de carga) melhor que a configuração corrente (também sem corte de carga) (retorno para a região infactível e reinício da oscilação estratégica).

A parte A da estratégia, usada quando a configuração corrente apresenta corte de carga, visa encontrar rapidamente uma configuração sem corte de carga através de uma sucessão de configurações com corte de carga cada vez menores usando a estratégia agressiva de **TS** de passar para a melhor configuração vizinha. O mais importante desta parte da estratégia de memória de curto prazo é a caracterização da estrutura de vizinhança para a configuração corrente. Assim, é definido um número reduzido de vizinhos formados por uma lista de adições/trocas de linhas. Esta lista de configurações candidatas de elite é obtida por:

Figura 5.12: Estratégias empregadas no algoritmo

(a) as melhores adições propostas pelos métodos aproximados: [9, 12, 35] (algoritmo de Garver baseado no modelo de transportes [9], método de mínimo esforço baseado no modelo DC [12] e método de mínimo corte de carga baseado em um PL especializado [35]);

(b) circuitos ou conjunto de circuitos vizinhos de barras onde existem grandes perdas de carga ou geração (corte de carga ou geração limitada porque não existem linhas suficientes saindo da barra de geração) e finalmente,

(c) uma lista pequena de adições e adições/trocas de circuitos aleatoriamente escolhidas.

Esta forma de caracterizar a vizinhança produz uma lista de configurações vizinhas de elite, altamente atrativas e de tamanho reduzido quando comparado com as técnicas tradicionais de caracterização de vizinhança, ou elaboração de listas de configurações candidatas. A formação da lista implica na utilização de três PL's (um para cada método aproximado); mas por outro lado, avaliar cada elemento da lista (configuração candidata) também precisa de um PL o que mostra a importância de formar uma pequena lista de vizinhos altamente atrativos.

Nesta parte A da estratégia é usada memória de curto prazo baseada em recência e o atributo proibido é a retirada ou adição de circuitos durante um determinado número de

iterações. Assim, existe duas listas de movimentos proibidos, uma para proibir a adição de linhas e outra para proibir a retirada de linhas. Portanto, nesta parte, o processo de transição consiste em passar para a melhor configuração vizinha a partir da configuração corrente. Também é usado um critério de aspiração que retira a proibição de uma adição e/ou troca se a configuração candidata apresenta uma função objetivo melhor que as obtidas nesta fase de intensificação.

A parte B da estratégia é implementada quando é encontrada uma configuração sem corte de carga. Nesta parte, deve-se implementar uma outra lista de configurações vizinhas onde basicamente devem ser realizadas pequenas alterações tentanto encontrar outras configurações sem corte de carga na vizinhança da configuração corrente; o que equivale a realizar uma intensificação visando encontrar o ótimo local da região. A lista de configurações vizinhas da configuração corrente neste caso é formada como segue:

(a) retirada de circuitos de baixo custo;

(b) retirada de alguns circuitos escolhidos aleatoriamente;

(c) troca de circuitos de baixo custo; e

(d) troca de alguns circuitos escolhidos aleatoriamente.

Se é encontrada uma configuração vizinha sem corte de carga melhor que a corrente, então é atualizada a configuração e se repete o processo; caso contrário, esta parte da estratégia termina. Nesta parte do processo não existe lista de atributos proibidos.

A parte C da estratégia consiste em remover alguns circuitos da configuração corrente sem corte de carga (tipicamente as mais baratas) excluindo os circuitos adicionados na parte B para evitar ciclagem. Com esta implementação o processo entra novamente na região infactível e, deve-se repetir o processo cíclico A-B-C de oscilação estratégica até esgotar um número determinado de iterações.

Na estratégia de intensificação mostrada anteriormente são realizadas três tarefas específicas: na parte A, a configuração corrente apresenta corte de carga e a prioridade é encontrar, o mais rápido possível, uma configuração sem corte de carga e passar para a parte B; onde é priorizada a busca de configurações sem corte de carga na região da configuração corrente (ótimo local). Finalmente, a parte C prioriza a saída do ótimo local levando o processo a uma região infactível vizinha para tentar retornar a outra região factível também vizinha.

Diversificação

A idéia da diversificação é levar o processo de busca a regiões não visitadas na tentativa de encontrar configurações promissoras. Cada configuração mostrada nas Figuras 5.9 e 5.10,

no nível II, é submetida a diversificação antes de iniciar o processo de memória de curto prazo. No algoritmo apresentado, a diversificação é realizada de duas maneiras diferentes:

1. Realizando algumas modificações na configuração através de adições, retiradas e/ou trocas de linhas, e
2. Fazendo "path relinking".

Uma primeira forma de realizar diversificação é realizando pequenas modificações na topologia da configuração através de adições, retiradas ou trocas de circuitos. Por exemplo, supor que a configuração 3 da Figura 5.9, no nível II, vai ser submetida a diversificação. Assim, a primeira coisa a verificar é se a configuração 3 apresenta corte de carga. Se apresentar corte de carga, deve-se realizar um processo de adição e troca de circuitos; caso contrário, deve-se realizar um processo de retirada e/ou troca de circuitos.

Quando a configuração apresenta corte de carga, os circuitos adicionados são escolhidos de uma lista de circuitos obtidos pelos métodos aproximados mencionados anteriormente. Esta lista é ordenada levando em conta a função objetivo que apresenta mais uma parcela adequadamente penalizada com a informação de memórias baseadas em freqüência do processo como um todo (conforme descrito anteriormente no item memórias de longo prazo). Durante o processo é armazenada a informação dos circuitos que ao serem adicionados produzem uma maior diminuição do corte de carga; assim, estes circuitos sofrem menor penalização na função objetivo.

São escolhidos para adição os m circuitos melhores classificados de acordo com a função objetivo penalizada. Adicionalmente são implementadas algumas trocas de circuitos escolhidas dentre aquelas trocas que produziram maior diminuição da função objetivo e que foram armazenadas durante todo o processo usando uma matriz de trocas. Também é feita uma troca entre duas linhas selecionadas de maneira aleatória, tirando da configuração uma linha e adicionando alguma outra; verifica-se então se esta troca tem um baixo índice de freqüência, caso contrário, se seleciona alguma outra troca.

Quando a configuração não apresenta corte de carga, é realizada uma retirada e troca de circuitos de maneira parecida ao caso anterior. Neste caso os circuitos retirados são escolhidos de uma lista ordenada formada pelos circuitos de menor custo e outros obtidos aleatoriamente. A função objetivo associada a cada elemento da lista é penalizada de acordo com o valor da freqüência, e ordenada.

Uma outra forma de realizar diversificação é usando "path relinking". Neste caso é realizado "path relinking" entre a configuração candidata à diversificação e uma das configurações de elite. A decisão de realizar uma das duas formas de diversificação é decidida de forma aleatória; assim a diversificação é realizada com "path relinking" com uma determinada probabilidade (tipicamente entre 0,1 e 0,2) e, caso contrário, é realizado o tipo de diversificação anterior.

5.4 IMPLEMENTAÇÕES ESPECIALIZADAS NO ALGORITMO DE BUSCA TABU

5.4.1 Implementação do "Path Relinking"

O "path relinking" proporciona uma integração muito útil das estratégias de intensificação e diversificação. Este enfoque se baseia em explorar trajetórias que conectam soluções elite para gerar novas soluções, iniciando desde uma destas soluções chamada solução inicial, e gerando uma trajetória na vizinhança que conduz até outras soluções, chamadas *soluções guia*. Assim, por exemplo, é possível gerar uma nova configuração ligando uma configuração da população e uma configuração elite.

No algoritmo de **TS** implementado, o procedimento "path relinking" está sendo empregado como uma das estratégias de diversificação; a outra estratégia é baseada em freqüência. A seleção entre estes dois tipos de diversificação se faz de maneira aleatória, ou seja: seleciona-se um número aleatório no intervalo [0;1]. Se este número é menor ou igual a 0,1 o procedimento selecionado é o de "path relinking", caso contrário, é selecionada a estratégia baseada em freqüência.

No procedimento diversificação-itensificação, por exemplo, suponha que a configuração 3 da Figura 5.9 é selecionada para ser analisada. Primeiro seleciona-se o tipo de diversificação, neste caso selecionou-se "path relinking"; então, já que a configuração 3 é escolhida como configuração inicial, tem que ser selecionada uma configuração guia. A configuração guia é selecionada entre algumas das soluções elite armazenadas durante o processo.

A forma mais simples de fazer "path relinking" é identificando o número de posições na configuração onde existem adições diferentes nas duas configurações submetidas a "path relinking". Portanto, assumindo uma das configurações como referência, copia-se uma parcela das posições diferentes que estão presentes na outra configuração e assim gera-se uma nova configuração, que na verdade é a configuração de referência com alguns de seus atributos modificados usando a outra configuração. Obtém-se assim uma nova configuração potencialmente atrativa.

O "path relinking" pode ser empregado em processos de intensificação ou diversificação, ou seja, se o número de posições diferentes presentes na configuração guia que serão copiadas na configuração referência é pequeno e de pouco impacto, então a configuração fica em uma região muito próxima de onde estava; nestas condições a configuração obtida pode ser usada em intensificação. No caso em que o número de posições copiadas da configuração guia para a configuração referência for grande, então foi obtida uma configuração de uma nova região ou seja a configuração assim obtida pode ser usada para diversificação.

5.4.2 Critério de Parada

Emprega-se três critérios de parada:

- Por máximo número de iterações (onde cada iteração corresponde a um PL),
- Por máximo número de diversificações,
- Quando as k configurações armazenadas na matriz de trabalho tiver sido analisadas, ou seja, as k posições do vetor associado às configurações tiverem sinal de revisão.

5.5 ALGORITMO DE BUSCA TABU IMPLEMENTADO

O algoritmo **TS** implementado funciona basicamente como é mostrado na Figura 5.12. Nesta figura é mostrada a evolução de uma configuração inicial. Assim, para melhorar o desempenho do processo, pode-se trabalhar com uma população de forma equivalente ao **AG**, escolhendo um conjunto de configurações iniciais, cada uma realizando a estratégia mostrada na Figura 5.12. Neste contexto, cada uma das configurações realiza o ciclo mostrado na figura até que um critério de parada seja satisfeito, tipicamente, até que sejam realizadas um número especificado de movimentos consecutivos com degradação da função objetivo.

Na Figura 5.13, apresenta-se o diagrama de fluxo da metodologia busca tabu usado no planejamento de redes de transmissão. Esta metodologia é constituída basicamente por dois grandes processos, chamados de diversificação e intensificação, que formam a base fundamental do algoritmo.

O algoritmo inicia com um processo de intensificação tal como é apresentado nas Figuras 5.9 e 5.10; neste se faz uma busca detalhada entre os vizinhos mais próximos com base em listas dos melhores candidatos e com uma estrutura de vizinhança que permite a entrada - troca de linhas ou saída - troca de linhas; após um número de PL's no qual não se obtém melhoria na solução, este processo termina. No processo de diversificação é selecionada a configuração de melhor função objetivo e sem análise prévia, nesta se efetua um processo de diversificação que permite saltar para regiões vizinhas e iniciar nestas um novo processo de intensificação.

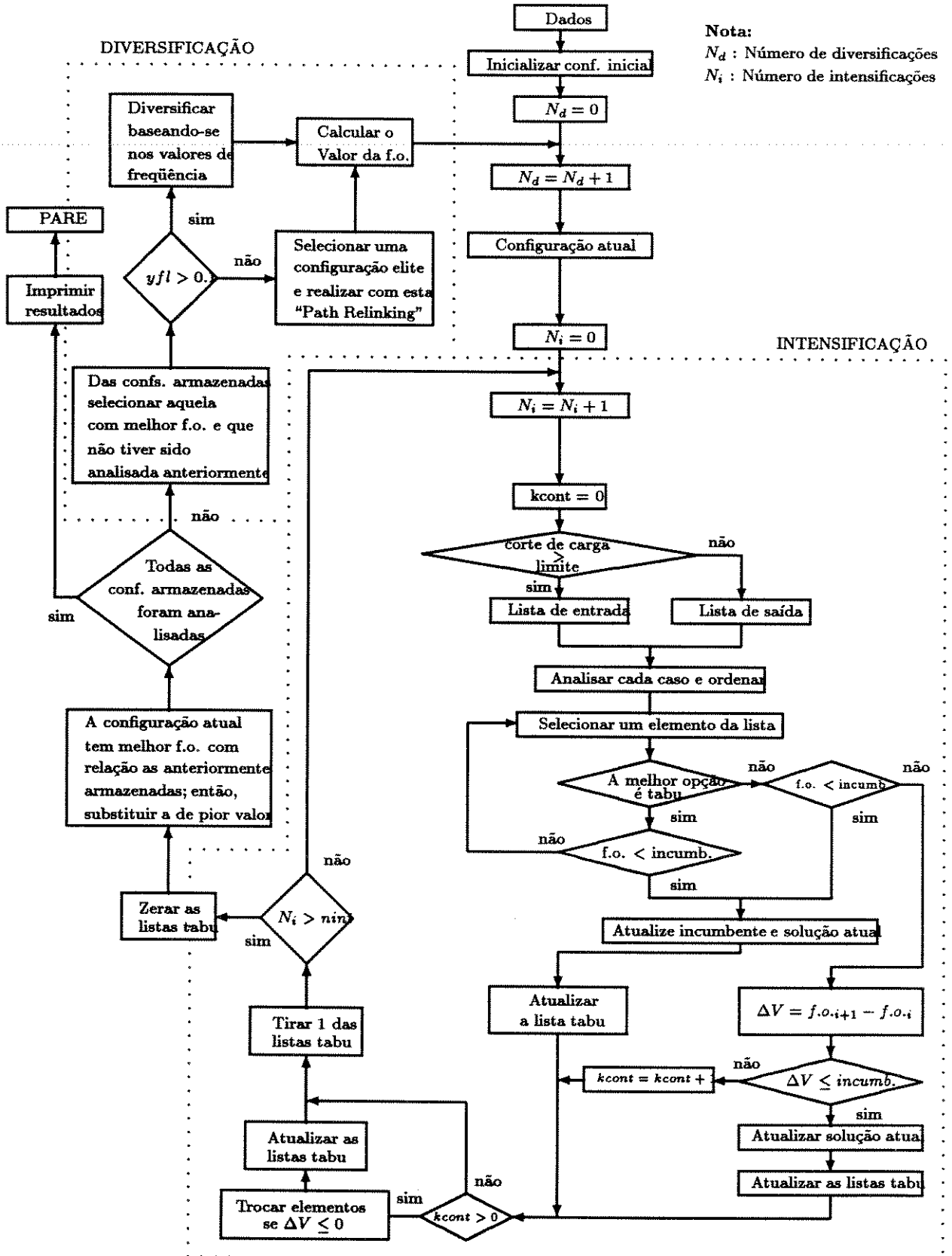


Figura 5.13: Diagrama de fluxo da metodologia Busca Tabu

5.6 TESTES EM SISTEMAS DE PEQUENO E DE MÉDIO PORTE

Os resultados apresentados neste capítulo permitem observar o desempenho do método busca tabu quando é usado na solução de sistemas de pequeno e médio porte. Além disso, permite estabelecer uma comparação do método busca tabu quando este trabalha com uma ou com várias configurações, assim como o impacto produzido por uma adequada determinação das configurações iniciais obtidas pelo algoritmo de Garver modificado. Os parâmetros dos algoritmos foram calibrados para permitir desempenhos com praticamente 100% de eficiência em encontrar as soluções ótimas dos sistemas de 06 barras e Sul brasileiro apresentados. Assim, a comparação deve ser feita em relação ao esforço computacional (PL's utilizadas) requerido para cada algoritmo .

5.6.1 Sistema de 06 Barras de Garver com Redespacho

Os dados deste sistema são apresentados no Apêndice A. Na Figura 3.6, observa-se a configuração inicial. Neste caso se apresentam os resultados de testes realizados com configurações iniciais obtidos aleatoriamente, pois quando é usado o algoritmo de Garver modificado, a configuração ótima é encontrada pelo mesmo, tornando desnecessário o restante do algoritmo. Na Figura 3.7 é apresentada a configuração ótima.

Na Tabela 5.1, o número que aparece entre parênteses representa o número de diversificações-intensificações realizadas (**ND**). *Nº PL final* é o número de chamadas ao programa de PL usado pelo algoritmo (parâmetro ou critério mais importante na comparação) e *Nº PL 1ª vez* é o número de chamadas ao programa de PL necessários para encontrar, pela primeira vez, a configuração ótima. A solução ótima do sistema analisado foi encontrada em todos os testes feitos.

Este algoritmo contém um grande número de parâmetros, entre os quais:

- Máximo número de vezes em que a solução não melhora,
- Máximo número de vezes em que as configurações são submetidas ao processo de diversificação,
- Número de iterações em que um elemento é declarado tabu ativo,
- Número de configurações elite armazenadas,
- Número de configurações armazenadas durante o processo de análise,
- Máximo número de trocas permitidas, escolhidas de maneira aleatória ou entre aquelas que historicamente melhoraram a função objetivo.

Tabela 5.1: Resultados do sistema Garver com redespacho, inicializado com uma e com k configurações

Busca Tabu - Sistema Garver - com redespacho, f.o. = 110				
Inicialização aleatória				
	(inicia com uma configuração conforme Figura 5.9) (nove posições de memória são empregadas)		(inicia com k configurações conforme figura 5.10) (quatro posições de memória são empregadas)	
Caso Nº	Nº PL 1ª vez(ND)	Nº PL final(ND)	Nº PL 1ª vez(ND)	Nº PL final(ND)
1	44(1)	522(11)	315(3)	450(7)
2	62(1)	554(12)	206(1)	483(6)
3	44(1)	496(11)	56(1)	496(7)
4	67(1)	527(6)	39(1)	427(6)
5	299(5)	505(11)	39(1)	415(6)
6	44(1)	449(13)	43(1)	515(6)
7	257(6)	552(12)	147(1)	491(8)
8	252(5)	515(12)	29(1)	427(7)
9	190(4)	624(13)	61(1)	413(7)
10	73(1)	537(12)	87(1)	427(6)
11	54(1)	573(11)	230(1)	464(8)
12	37(1)	574(12)	156(1)	392(7)
13	49(1)	557(12)	217(1)	527(8)
14	159(3)	648(13)	39(1)	498(7)
15	390(8)	574(12)	107(1)	412(6)
16	403(7)	531(12)	54(1)	423(6)
17	77(1)	629(12)	211(1)	512(7)
18	66(1)	572(12)	92(1)	414(6)
19	49(1)	543(11)	57(1)	499(6)
20	73(1)	494(12)	94(1)	445(7)
	Média de PL's 549		Média de PL's 457	

Além destes, tem-se os parâmetros para formar as configurações candidatas de elite, com os quais se atribui uma parcela de elementos candidatos identificados pelos algoritmos aproximados. A outra parcela de elementos é selecionada de maneira aleatória e através de linhas de mínimo custo.

Neste trabalho foram selecionados os parâmetros para chegar à solução ótima com uma alta freqüência. A variação destes parâmetros poderá ocasionar uma grande variação no número de PL para atender a convergência. Então, fica muito difícil estabelecer uma comparação adequada entre as diferentes versões dos algoritmos implementados.

5.6.2 Sistema de 06 Barras de Garver sem Redespacho

Como no caso de Garver com redespacho, apresenta-se aqui somente os resultados de testes realizados com configurações iniciais obtidos aleatoriamente (Tabelas 5.1 e 5.2), já que a solução ótima deste sistema é encontrada pela metodologia de Garver modificada, tornando desnecessário a aplicação do restante do processo. Na Figura 3.8 observa-se a configuração ótima. A solução ótima do sistema analisado foi encontrada em todos os testes feitos.

5.6.3 Sistema Sul Brasileiro com Redespacho

Os dados deste sistema são apresentados no Apêndice A. Na Figura 3.9, observa-se a rede inicial. Na Figura 3.10, pode-se observar a configuração ótima, com um custo de US$ 70.289.000,00.

Este sistema é analisado com base em dois tipos de inicialização; a primeira, quando se emprega uma inicialização aleatória, e a segunda, com inicialização usando a metodologia Garver modificada. Além disto, se estuda o desempenho do algoritmo iniciando com uma única configuração e quando é inicializado com k configurações. (Ver Tabelas 5.3 e 5.4.)

5.6.4 Sistema Sul Brasileiro sem Redespacho

Para a análise deste sistema foram utilizados os dois tipos de inicialização mencionados anteriormente; geração aleatória de configurações e geração com a metodologia Garver modificada.

Na Figura 3.11, pode-se observar a configuração ótima deste sistema, com um custo de US$ 154.420.000,00 que corresponde a adição de 16 linhas. As Tabelas 5.5 e 5.6 apresentam os resultados obtidos.

Tabela 5.2: Resultados do sistema Garver sem redespacho, inicializado com uma e com k configurações

Busca Tabu - Sistema Garver - sem redespacho, f.o. = 200				
Inicialização aleatória				
	(inicia com uma configuração conforme Figura 5.9) (onze posições de memória são empregadas)		(inicia com k configurações conforme figura 5.10) (cinco posições de memória são empregadas)	
Caso Nº	Nº PL 1ª vez(ND)	Nº PL final(ND)	Nº PL 1ª vez(ND)	Nº PL final(ND)
1	80(1)	562(15)	80(1)	596(7)
2	62(1)	707(15)	70(1)	710(8)
3	171(3)	649(14)	73(1)	633(7)
4	67(1)	608(14)	69(1)	587(7)
5	72(1)	748(16)	136(2)	586(7)
6	77(1)	698(16)	57(1)	604(7)
7	140(2)	638(14)	65(1)	641(9)
8	72(1)	737(14)	69(1)	606(7)
9	99(2)	600(15)	61(1)	642(7)
10	134(2)	648(15)	135(2)	605(7)
11	145(2)	640(13)	140(2)	616(7)
12	78(1)	580(13)	74(1)	667(7)
13	116(2)	613(14)	71(1)	698(9)
14	73(1)	668(17)	76(1)	652(7)
15	126(2)	632(14)	139(1)	610(9)
16	500(7)	831(17)	65(1)	603(9)
17	77(1)	774(15)	68(1)	642(7)
18	66(1)	572(12)	140(1)	660(10)
19	82(1)	610(16)	309(1)	594(7)
20	73(1)	592(13)	66(1)	737(10)
	Média de PL's 656		Média de PL's 634	

Tabela 5.3: Resultados do sistema Sul brasileiro com redespacho, iniciando com uma única configuração

Busca Tabu - Sistema Sul brasileiro - com redespacho (inicia com uma configuração conforme Figura 5.9)						
	Inic. aleatória 40 posições de memória são empregadas			Inic. Garver modificado 40 posições de memória são empregadas		
Caso Nº	Nº PL 1ª vez(ND)	Nº PL final(ND)	Custo (10^3US$)	Nº PL 1ª vez(ND)	Nº PL final(ND)	Custo (10^3US$)
1	4.166(37)	5.936(59)	70.289,00	278(4)	4.846(51)	70.289,00
2	828(8)	5.351(57)	70.289,00	880(8)	4.928(49)	70.289,00
3	940(9)	5.540(56)	70.289,00	3.477(36)	4.838(53)	70.289,00
4	692(5)	5.730(56)	70.289,00	90(1)	5.212(56)	70.289,00
5	2.653(23)	5.727(54)	70.289,00	1.074(9)	5.802(58)	70.289,00
6	902(7)	5.549(53)	70.289,00	543(5)	4.369(44)	70.289,00
7	496(5)	5.154(51)	70.289,00	363(3)	6.166(56)	70.289,00
8	2.197(20)	5.154(56)	70.289,00	1.006(9)	5.596(57)	70.289,00
9	670(6)	5.080(52)	70.289,00	557(6)	4.543(47)	70.289,00
10	498(5)	4.899(50)	70.289,00	2.260(23)	4.753(46)	70.289,00
11	4.189(35)	7.261(71)	70.289,00	913(7)	6.898(65)	70.289,00
12	1.226(10)	5.940(60)	70.289,00	897(9)	5.529(57)	70.289,00
13	1.239(11)	5.160(55)	70.289,00	1.078(7)	4.825(49)	70.289,00
14	465(3)	5.191(59)	70.289,00	102(1)	4.140(47)	70.289,00
15	1.632(14)	5.471(58)	70.289,00	696(8)	4.927(50)	70.289,00
16	1.219(13)	4.551(51)	70.289,00	340(3)	4.605(44)	70.289,00
17	694(5)	5.291(56)	70.289,00	982(10)	6.060(68)	70.289,00
18	688(7)	4.711(56)	70.289,00	712(8)	4.870(53)	70.289,00
19	302(2)	4.824(52)	70.289,00	81(1)	4.312(43)	70.289,00
20	300(2)	4.067(48)	70.289,00	237(2)	5.818(66)	70.289,00
	Média de PL's 5.329			Média de PL's 5.152		

Tabela 5.4: Resultados do sistema Sul brasileiro com redespacho, iniciando com k configurações

Busca Tabu - Sistema Sul brasileiro - com redespacho (inicia com k configurações conforme figura 5.10)						
	Inic. aleatória 18 posições de memória são empregadas			Inic. Garver modificado 10 posições de memória são empregadas		
Caso Nº	Nº PL 1ª vez(ND)	Nº PL final(ND)	Custo (10^3US$)	Nº PL 1ª vez(ND)	Nº PL final(ND)	Custo (10^3US$)
1	2.559(*)	5.479(29)	70.289,00	707(2)	1.739(13)	70.289,00
2	1.528(*)	5.122(23)	70.289,00	72(*)	1.703(15)	70.289,00
3	2.876(3)	4.657(26)	70.289,00	436(*)	2.017(17)	70.289,00
4	405(*)	4.785(23)	70.289,00	285(*)	1.667(14)	70.289,00
5	966(*)	4.608(22)	70.289,00	193(*)	1.839(14)	70.289,00
6	413(*)	4.456(32)	70.289,00	119(*)	1.707(13)	70.289,00
7	1.223(*)	5.110(26)	70.289,00	1.207(8)	1.787(14)	70.289,00
8	937(*)	4.825(25)	70.289,00	561(*)	2.498(23)	70.289,00
9	1.361(*)	5.290(27)	70.289,00	129(*)	2.012(16)	70.289,00
10	795(*)	5.077(24)	70.289,00	404(*)	1.751(14)	70.289,00
11	403(*)	4.969(23)	70.289,00	719(2)	1.650(12)	70.289,00
12	468(*)	4.860(24)	70.289,00	546(*)	1.913(17)	70.289,00
13	3.410(10)	5.281(32)	70.289,00	295(*)	1.735(14)	70.289,00
14	3.055(2)	4.679(21)	70.289,00	58(*)	1.733(13)	70.289,00
15	2.755(*)	4.957(25)	70.289,00	212(*)	1.853(13)	70.289,00
16	804(*)	5.106(25)	70.289,00	1.049(5)	1.965(17)	70.289,00
17	268(*)	4.748(25)	70.289,00	27(*)	1.720(15)	70.289,00
18	1.195(*)	5.075(25)	70.289,00	64(*)	1.593(13)	70.289,00
19	3.505(9)	4.781(26)	70.289,00	528(*)	2.558(21)	70.289,00
20	3.237(5)	4.796(22)	70.289,00	239(*)	1.547(13)	70.289,00
	Média de PL's 4.933			Média de PL's 1.849		

$*$ - resultado obtido durante o primeiro processo de otimização (nível I)

Tabela 5.5: Resultados do sistema Sul brasileiro sem redespacho, iniciando com uma única configuração

Busca Tabu - Sistema Sul brasileiro - sem redespacho (inicia com uma configuração conforme Figura 5.9)						
	Inic. aleatória 44 posições de memória são empregadas			Inic. Garver modificado 44 posições de memória são empregadas		
Caso Nº	Nº PL 1ª vez(ND)	Nº PL final(ND)	Custo (10^3US$)	Nº PL 1ª vez(ND)	Nº PL final(ND)	Custo (10^3US$)
1	2.018(15)	6.840(60)	154.420,00	5.585(57)	6.929(71)	154.420,00
2	796(6)	6.730(6)	154.420,00	2.375(21)	6.145(61)	154.420,00
3	225(2)	6.540(60)	154.420,00	5.170(53)	6.873(70)	154.420,00
4	170(1)	5.842(57)	154.420,00	3.082(27)	5.987(56)	154.420,00
5	1.013(6)	5.677(57)	161.588,00	5.333(57)	6.188(68)	154.420,00
6	313(2)	4.924(48)	154.420,00	3.727(42)	6.282(71)	154.420,00
7	174(1)	5.021(53)	154.420,00	3.825(39)	5.135(54)	154.420,00
8	4.982(48)	5.677(58)	164.752,00	5.592(56)	7.665(71)	154.420,00
9	4.551(40)	5.462(51)	154.420,00	6.253(59)	6.801(66)	154.420,00
10	3.891(35)	5.818(55)	154.420,00	1.678(14)	6.635(70)	154.420,00
11	245(1)	5.922(64)	154.420,00	1.552(14)	6.396(69)	154.420,00
12	3.221(26)	5.688(53)	154.420,00	488(4)	5.711(59)	154.420,00
13	397(2)	6.038(54)	154.420,00	198(3)	6.207(65)	154.420,00
14	6.797(61)	7.054(64)	154.420,00	256(3)	6.486(63)	154.420,00
15	2.350(22)	7.011(71)	154.420,00	3.614(35)	5.074(52)	154.420,00
16	4.435(48)	6.665(71)	154.420,00	617(4)	6.463(67)	154.420,00
17	6.302(57)	7.165(64)	159.863,00	1.034(10)	5.890(62)	154.420,00
18	1.478(11)	6.917(71)	154.420,00	4.196(47)	5.064(60)	154.420,00
19	1.468(12)	5.705(53)	154.420,00	282(2)	5.390(62)	154.420,00
20	4.449(32)	7.052(59)	154.420,00	1.340(12)	5.894(64)	154.420,00
	Média de PL's 6.187			Média de PL's 6.160		

Tabela 5.6: Resultados do sistema Sul brasileiro sem redespacho, iniciando com k configurações

Busca Tabu - Sistema Sul brasileiro - sem redespacho (inicia com k configurações conforme figura 5.10)						
	Inic. aleatória 18 posições de memória são empregadas			Inic. Garver modificado 12 posições de memória são empregadas		
Caso Nº	Nº PL 1ª vez(ND)	Nº PL final(ND)	Custo (10^3US$)	Nº PL 1ª vez(ND)	Nº PL final(ND)	Custo (10^3US$)
1	3.031(*)	6.313(24)	154.420,00	356(*)	3.061(19)	154.420,00
2	1.040(*)	6.422(29)	154.420,00	69(*)	2.477(16)	154.420,00
3	434(*)	5.877(26)	154.420,00	345(*)	2.344(16)	154.420,00
4	498(*)	6.166(27)	154.420,00	80(*)	4.556(34)	154.420,00
5	3.467(*)	6.603(31)	154.420,00	424(*)	2.439(18)	154.420,00
6	2.148(*)	5.837(23)	154.420,00	270(*)	3.213(19)	154.420,00
7	152(*)	5.641(23)	154.420,00	875(*)	3.286(23)	154.420,00
8	922(*)	5.878(22)	154.420,00	439(*)	2.366(16)	154.420,00
9	833(*)	6.352(28)	154.420,00	657(*)	3.493(25)	154.420,00
10	153(*)	5.548(25)	154.420,00	80(*)	2.343(14)	154.420,00
11	330(*)	5.492(23)	154.420,00	653(*)	2.662(19)	154.420,00
12	325(*)	5.534(21)	154.420,00	1.840(*)	2.330(16)	154.420,00
13	152(*)	5.199(21)	154.420,00	1.360(*)	3.294(22)	154.420,00
14	1.425(*)	6.015(25)	154.420,00	930(*)	3.108(24)	154.420,00
15	1.176(*)	6.185(29)	154.420,00	765(*)	2.548(15)	154.420,00
16	559(*)	6.397(26)	154.420,00	820(*)	2.425(17)	154.420,00
17	381(*)	5.207(22)	154.420,00	226(*)	2.674(18)	154.420,00
18	1.263(*)	6.044(26)	154.420,00	286(*)	2.252(18)	154.420,00
19	208(*)	6.208(28)	154.420,00	263(*)	2.193(16)	154.420,00
20	753(*)	6.119(24)	154.420,00	847(*)	2.140(14)	154.420,00
	Média de PL's 5.952			Média de PL's 2.760		

$*$ - resultado obtido durante o primeiro processo de otimização (nível I)

5.7 COMENTÁRIOS SOBRE OS RESULTADOS OBTIDOS, INICIANDO COM *UMA* E COM k CONFIGURAÇÕES

São analisados os sistemas Garver de 06 barras e o sistema Sul brasileiro de 46 barras; tanto o primeiro como o segundo são estudados com e sem redespacho. Na metodologia busca tabu, em sua forma original, inicia-se o procedimento com uma única configuração, e na medida em que encontra soluções de boa qualidade as mesmas são armazenadas. Aquelas que, em passos posteriores, servirão como ponto de partida para novas buscas.

Inicia-se com várias configurações, como acontece nos algoritmos genéticos, onde para cada configuração inicial, aplica-se o processo de intensificação, armazenando a melhor solução encontrada durante este processo, e assim, até completar as k configurações iniciais. Completado este primeiro ciclo, seleciona-se a melhor configuração continuando um procedimento similar ao empregado com uma só configuração.

A seleção de parâmetros ótimos é tarefa difícil. Neste trabalho, os mesmos foram escolhidos após efetuar algumas simulações preliminares, observando neste seu desempenho quanto ao número de PL's requeridos para atender a convergência e a qualidade da solução final.

Analisados os resultados tanto para o sistema Garver como para o sistema Sul brasileiro, observa-se que o algoritmo de melhor desempenho é aquele que emprega uma inicialização com k configurações; e apresenta os melhores resultados quando as configurações iniciais são geradas empregando a metodologia de Garver modificada.

Em busca tabu, sistemas de pequeno e médio porte poderão ser analisados simplesmente com adequados procedimentos de **TS**/intensificação, além disto, e para maior consistência, pode-se empregar procedimentos de diversificação. Já para sistemas de grande porte, deve-se usar procedimentos de diversifição mais sofisticados. Neste caso, observou-se como o sistema Garver é resolvido com o procedimento de **TS**/intensificação; o sistema Sul brasileiro poderia ser resolvido com o procedimento de **TS**/intensificação, porém com menor probabilidade de chegar à solução ótima. Quando se aplica o procedimento de diversificação a probabilidade de chegar à solução ótima melhora substancialmente. Com o sistema Norte-Nordeste se requerirá tanto um bom procedimento de **TS**/intensificação como de diversificação.

Em busca tabu, um bom procedimento de **TS**/intensificação se baseia em uma boa escolha da lista de vizinhos candidatos. Neste trabalho, a lista é feita com base em métodos aproximados, como foi mencionado anteriormente, com os quais se forma a chamada lista de elementos candidatos a entrar; assim, esta lista será formada por bons candidatos que darão maior solidez ao procedimento.

A idéia de inicializar o processo com k configurações, nasce dos algoritmos genéticos. Na literatura, diferentes autores mencionam que os melhores métodos de otimização combinatorial resultam da combinação de diferentes métodos, ou seja, **SA**, **AG**, **TS**, entre outros. No capítulo seguinte, a busca tabu será combinada com **SA** e **AG** para formar um algoritmo chamado *Híbrido*. Já que esta combinação apresenta melhores resultados com relação ao tabu puro, no capítulo posterior se continua com a busca tabu incorporando, em etapas, funções próprias dos outros algoritmos.

5.8 TESTES EM SISTEMAS DE GRANDE PORTE

Estes testes foram feitos baseados no sistema Norte-Nordeste brasileiro, o qual é apresentado na Figura 3.12 e cuja característica é a de apresentar um grande número de barras isoladas.

Para a análise deste sistema foi selecionado o algoritmo que inicia com k configurações, o qual pode ser observado na Figura 5.10. Este algoritmo apresentou melhor desempenho quando comparado com aquele que inicia com uma configuração e cujas análises foram feitos com o sistemas Garver e Sul brasileiro. Este algoritmo inicia com um processo de **TS**/intensificação em cada uma das k configurações. Concluído este processo, a configuração de melhor função objetivo é selecionada para iniciar um processo de diversificação - intensificação; quando todas as k configurações armazenadas forem analisadas, o processo termina.

5.8.1 Sistema Norte - Nordeste Brasileiro - Plano 2008

Este sistema esta constituído por 89 barras e 183 linhas candidatas e uma demanda total de 29.754 MW e cujos dados são apresentados no apêndice A.

As configurações iniciais foram obtidas usando a metodologia de Garver modificada. Como foi mencionado anteriormente, quando o processo inicia com configurações geradas de maneira aleatória, geralmente fica preso em um ótimo local de má qualidade. Quando inicia com configurações geradas com Garver modificada se obtém uma grande rapidez do processo, além disto se podem alcançar regiões que contém configurações de qualidade muito boa. Nesta análise foram empregadas 22 configurações iniciais. As melhores configurações analisadas foram armazenadas em 70 posições diferentes e as configurações elite em 6 posições diferentes. O processo termina quando todas as configurações armazenadas tiverem a indicação de que já foram analisadas e que não foi possível melhorar as soluções previamente armazenadas.

Entre as melhores configurações que foram encontradas pelo algoritmo **TS** estão: uma com um investimento de US$ 2.637.412.000,00 e cuja característica principal é a de conter um só caminho de 3.200 MW, neste caso o processo precisou de 183.556 PL's em 170 diversificações para atender a convergência; e outra com um investimento de US$ 2.598.726.000,00, cuja característica principal é a de ter dois caminhos de 3.200 MW, neste caso o processo precisou de 181.672 PL's em 150 diversificações para atender a convergência.

A configuração cujo investimento é de US$ 2.637.412.000,00 tem as seguintes linhas adicionadas.

$n_{01-02} = 1$, $n_{02-04} = 1$, $n_{04-05} = 6$, $n_{04-06} = 2$, $n_{05-56} = 1$, $n_{05-58} = 5$, $n_{06-07} = 1$, $n_{07-08} = 1$, $n_{08-17} = 2$, $n_{12-17} = 1$, $n_{13-15} = 4$, $n_{13-59} = 1$, $n_{14-59} = 1$, $n_{15-16} = 3$, $n_{15-46} = 1$, $n_{16-44} = 6$, $n_{17-18} = 2$, $n_{18-50} = 11$, $n_{18-74} = 3$, $n_{21-57} = 2$, $n_{22-58} = 2$, $n_{24-43} = 1$, $n_{25-55} = 3$, $n_{26-54} = 2$, $n_{30-31} = 2$, $n_{30-63} = 2$, $n_{35-46} = 1$, $n_{35-51} = 1$, $n_{36-46} = 1$, $n_{39-86} = 1$, $n_{40-45} = 1$, $n_{40-46} = 1$, $n_{41-64} = 3$, $n_{43-55} = 2$, $n_{43-58} = 2$, $n_{48-49} = 4$, $n_{48-50} = 1$, $n_{49-50} = 2$, $n_{52-59} = 1$, $n_{53-86} = 1$, $n_{54-58} = 3$, $n_{54-63} = 2$, $n_{56-57} = 1$, $n_{61-85} = 3$, $n_{61-86} = 1$, $n_{63-64} = 1$, $n_{67-68} = 1$, $n_{67-69} = 1$, $n_{67-71} = 3$, $n_{71-72} = 1$, $n_{72-73} = 1$, $n_{73-74} = 1$

A configuração cujo investimento é de US$ 2.598.726.000,00 tem as seguintes linhas adicionadas.

$n_{01-02} = 1$, $n_{02-04} = 1$, $n_{04-05} = 5$, $n_{04-81} = 3$, $n_{05-56} = 1$, $n_{05-58} = 4$, $n_{13-15} = 4$, $n_{14-59} = 1$, $n_{15-16} = 3$, $n_{15-46} = 1$, $n_{16-44} = 6$, $n_{18-50} = 11$, $n_{18-74} = 5$, $n_{21-57} = 2$, $n_{22-58} = 3$, $n_{24-43} = 1$, $n_{25-55} = 4$, $n_{30-31} = 1$, $n_{30-63} = 2$, $n_{35-46} = 1$, $n_{35-51} = 2$, $n_{36-46} = 2$, $n_{39-86} = 1$, $n_{40-45} = 1$, $n_{40-46} = 2$, $n_{41-64} = 2$, $n_{42-44} = 1$, $n_{43-55} = 2$, $n_{43-58} = 3$, $n_{48-49} = 1$, $n_{49-50} = 4$, $n_{52-59} = 1$, $n_{53-86} = 1$, $n_{54-58} = 1$, $n_{54-63} = 1$, $n_{56-57} = 1$, $n_{61-85} = 3$, $n_{61-86} = 1$, $n_{63-64} = 1$, $n_{67-68} = 1$, $n_{67-69} = 1$, $n_{67-71} = 3$, $n_{71-72} = 1$, $n_{72-73} = 1$, $n_{73-74} = 2$, $n_{73-75} = 1$, $n_{75-81} = 1$.

Quando foi empregado o conceito de intensificação por descomposição e inicializando o processo com as melhores soluções encontradas, foram obtidas soluções de melhor qualidade; a melhor solução correspondeu a um investimento de US$ 2.574.745.000,00, para a qual o processo precisou de 170.391 PL's em 135 diversificações.

A configuração cujo investimento é de US$ 2.574.745.000,00 tem as seguintes linhas adicionadas.

$n_{01-02} = 1$, $n_{02-04} = 1$, $n_{04-05} = 4$, $n_{04-81} = 3$, $n_{05-58} = 4$, $n_{12-15} = 1$, $n_{13-15} = 4$, $n_{14-45} = 1$, $n_{15-16} = 4$, $n_{15-46} = 1$, $n_{16-44} = 6$, $n_{16-61} = 2$, $n_{18-50} = 11$,$n_{18-74} = 6$, $n_{19-22} = 1$, $n_{20-21} = 3$, $n_{20-38} = 2$, $n_{22-37} = 1$, $n_{22-58} = 2$, $n_{24-43} = 1$, $n_{25-55} = 3$, $n_{26-54} = 1$, $n_{27-53} = 1$, $n_{30-31} = 2$, $n_{30-63} = 2$, $n_{35-51} = 2$, $n_{36-39} = 1$, $n_{36-46} = 3$, $n_{40-45} = 2$, $n_{41-64} = 2$, $n_{43-55} = 2$, $n_{43-58} = 2$, $n_{48-49} = 1$, $n_{49-50} = 4$, $n_{52-59} = 1$,

$n_{54-58} = 1$, $n_{54-63} = 1$, $n_{61-64} = 1$, $n_{61-85} = 3$, $n_{67-69} = 2$, $n_{67-71} = 3$, $n_{68-69} = 1$, $n_{69-87} = 1$, $n_{71-72} = 1$, $n_{72-73} = 1$, $n_{73-74} = 2$, $n_{73-75} = 1$, $n_{75-81} = 1$.

Na Tabela 5.7 apresentam-se estatísticas com soluções obtidas da análise do sistema Norte-Nordeste, para diferentes casos. Estes casos foram analisados com diferente número de configurações iniciais. Também foram alterados alguns parâmetros tais como: o número aleatório inicial, número de configurações elite armazenadas, valor do corte de carga permitido, duração tabu e tamanho da lista, entre outros.

Tabela 5.7: Resultados do sistema Norte-Nordeste brasileiro, algoritmo Tabu

Algoritmo Tabu - Sistema Norte-Nordeste - plano 2008				
	Inicialização Garver modificado			
Caso Nº	Valor da f.o.(10^3US$)	Inversão (10^3US$)	Corte de carga (MW)	Nº de PL's
1	2.581.772,00	2.550.218,00	31,6	119.337
2	2.586.559,00	2.545.598,00	41,0	161.649
3	2.593.253,00	2.585.781,00	7,5	143.288
4	2.594.517,00	2.592.517,00	2,1	151.672
5	2.594.656,00	2.591.656,00	3,2	130.492
6	2.596.216,00	2.591.216,00	4,5	151.672
7	2.598.726,00	2.598.726,00	0,0	181.672
8	2.599.236,00	2.590.236,00	8,8	158.198
9	2.599.451,00	2.595.451,00	4,3	141.560
10	2.600.028,00	2.593.028,00	6,9	192.226
11	2.600.593,00	2.597.593,00	3,3	171.214
12	2.601.111,00	2.596.111,00	5,2	146.340
13	2.601.707,00	2.594.707,00	7,3	143.272
14	2.604.028,00	2.593.028,00	11,5	178.846
15	2.604.538,00	2.600.538,00	3,6	178.830
16	2.605.726,00	2.598.726,00	0,0	128.445
17	2.608.988,00	2.599.988,00	9,1	182.952
18	2.621.493,00	2.581.493,00	40,0	97.721
19	2.637.412,00	2.637.412,00	0,0	183.556
20	2.643.936,00	2.643.936,00	0,0	171.250
21	2.649.118,00	2.639.118,00	9,9	196.067
22	2.672.086,00	2.633.086,00	0,0	95.493

5.9 CONCLUSÕES

A busca tabu que é uma técnica que surge de conceitos da inteligência artificial, é um procedimento de busca que explora o espaço de soluções além do ótimo local. A busca tabu permite mover-se para uma solução da vizinhança de cada configuração corrente mesmo que não seja tão boa como a corrente, escapando de ótimos locais e continuando a busca para soluções melhores. Esta técnica foi validada através de comparações com métodos aproximados, técnicas de otimização baseadas em decomposição de Benders, a metodologia de **SA** e dos **AG**. Para sua validação empregou-se três sistemas: de pequeno, médio e grande porte, os quais foram descritos anteriormente.

Os parâmetros de controle foram otimizados de forma empírica, utilizando para isto os resultados dos sistemas de pequeno e médio porte. O método **TS** mostrou-se eficiente quando foi aplicado a sistemas de pequeno e médio porte, pois encontrou a solução ótima em quase 100% dos casos que foram analisados. Já para sistemas de grande porte, chegou-se a soluções de parecida qualidade com relação às obtidas com **SA** paralelo, **AG** ou algoritmo híbrido.

No método **TS** é empregada a memória de longo prazo para armazenar informação de freqüência; com a mesma se efetua o procedimento de diversidade, penalizando ou incentivando configurações formadas com determinados atributos. Isto é, baseando-se no passado, se estabelecem regras que permitem encontrar configurações localizadas em regiões que sejam atrativas e que não foram analisadas. Esta mesma função desempenha o parâmetro temperatura no **SA** e o operador "crossover" nos **AG**.

Também é empregado o "path relinking"; neste caso utilizado como procedimento de diversificação. Dispondo-se então de duas maneiras para diversificar: a primeira está baseada em freqüência e a segunda em "path relinking".

A memória de curto prazo é utilizada para armazenar informação recente; com base na mesma se efetua o procedimento de **TS**/intensificação e a estratégia de oscilação, utilizando para isto as estratégias A - B - C.

Capítulo 6

PLANEJAMENTO DE SISTEMAS DE TRANSMISSÃO USANDO UM ALGORITMO HÍBRIDO

6.1 INTRODUÇÃO

Nos capítulos anteriores são analisados os algoritmos **SA**, **AG** e **TS**, quando aplicados para resolver o problema de planejamento de sistemas de transmissão a longo prazo, avaliando-os quanto as diferentes partes da estrutura de funcionamento. Fez-se então uma comparação das funções desenvolvidas em cada um dos algoritmos, observando-se que funções similares são executadas pelos mesmos. Por exemplo, com a função diversificação o processo pode saltar de uma região para outra alterando alguns dos elementos da configuração e dependendo do tipo de mudança realizada, pode-se chegar a configurações vizinhas ou distantes; a diversificação pode ser feita nos três algoritmos de modo diferente, ou seja: em **SA** se faz através do parâmetro temperatura, aceitando soluções piores com uma certa probabilidade. No caso do **TS**, se faz através da variável freqüência, somando (subtraindo) da função objetivo da configuração um valor, o qual é determinado em função da freqüência das linhas propostas nessa configuração. E no caso dos **AG** a diversificação é conseguida na operação de cruzamento das configurações. Assim como esta, são analisadas cada uma das estruturas que formam os três algoritmos.

Neste capítulo se estudará um algoritmo chamado **híbrido**, o qual será formado por diferentes funções extraídas do **SA**, **AG** e **TS**. Analisadas as funções e o processo que as executa em cada um deles, serão selecionandas as de melhor desempenho, e com as mesmas será formado o algoritmo híbrido usando **TS** como base.

Inicialmente é apresentada uma descrição teórica do algoritmo *híbrido* proposto, depois

é discutida a aplicação do algoritmo ao problema de planejamento de redes de transmissão, que inclui:

- Apresentação das funções (das estratégias) que formarão os algoritmos *híbrido*,
- A representação do problema,
- O problema da estrutura de vizinhança,
- Características de memória de curto e longo prazo, e
- Intensificação e diversificação.

A seguir, são apresentadas as implementações especializadas estudadas no algoritmo híbrido, que inclui:

- Mecanismo básico do algoritmo: estratégia A - B - C,
- Uso de uma população de configurações,
- Implementação de configurações elite,
- Implementação de "path relinking",
- Uso de estratégias inspiradas em **SA**,
- Aproveitamento das características do **SA**, **AG** e **TS** no problema do planejamento de redes de transmissão,
- Implementação de versões híbridas, e
- Critério de parada.

Também é apresentado o algoritmo híbrido implementado, assim como os resultados obtidos com sistemas pequenos, médios e de grande porte, nestes se inclui resultados dos algoritmos formados por **TS** e **SA**, **TS** e **AG** assim como **TS**, **SA** e **AG**. Finalmente são mencionadas as conclusões mais relevantes.

6.2 ALGORITMO HÍBRIDO - TEORIA

Nos últimos anos começaram a aparecer alguns algoritmos que aproveitam as características mas importantes dos algoritmos combinatoriais puros (**SA**,**TS**, **AG**, etc). Estes algoritmos chamados híbridos geralmente usam um dos algoritmos combinatoriais como

base e incorporam funções típicas dos outros algoritmos combinatoriais produzindo um algoritmo potencialmente mais competitivo para determinados tipos de problemas. Assim por exemplo alguns algoritmos híbridos desenvolvidos usando o algoritmo genético como base são chamados de algoritmos meméticos. No presente trabalho foram implementados algoritmos híbridos usando **TS** como base e incorporando as principais características de **SA** e **AG**.

Uma das principais características do algoritmo genético é usar um conjunto de configurações em contraste com **SA** e **TS**. Esta característica pode ser facilmente incorporada num algoritmo baseado em **TS**. Assim, pode-se gerar um conjunto de configurações usando por exemplo o algoritmo de Garver modificado e implementar uma estratégia **TS** com memória de curto prazo para cada uma das configurações da população. Na continuação, pode-se gerar uma nova geração da população usando cruzamento com os mesmos critérios usados no algoritmo genético e também pode ser usado "path relinking" para gerar parte dos elementos da nova geração.

Outra estratégia característica do algoritmo genético que pode ser implementado num algoritmo híbrido usando **TS** como base é o conceito de bloco construtivo. Para o problema de planejamento de sistemas de transmissão foi implementada uma estratégia de bloco construtivo que consiste de um conjunto de linhas correspondentes a novos caminhos que são adicionados, trocados e/ou eliminados em conjunto, pois esse bloco é altamente atrativo na forma de configuração de bloco e não na forma isolada. Esta estratégia, que foi implementada no algoritmo genético aproveitando as características específicas do problema de planejamento de sistemas de transmissão, pode ser integralmente incluída num algoritmo híbrido que usa **TS** como base.

A característica aleatória do algoritmo de "Simulated Annealing" também pode ser incorporada num algoritmo híbrido que usa **TS** como base. Assim por exemplo quando é analisado o conjunto reduzido de vizinhos candidatos pode acontecer que todos as configurações candidatas produzem uma degradação da função objetivo. Neste caso, um algoritmo **TS** padrão escolhe aquela configuração candidata que produz uma menor degradação da função objetivo. Uma estratégia alternativa neste caso é escolher uma configuração candidata usando o mecanismo de aceitação do algoritmo de "Simulated Annealing". Assim é aceita alguma das configurações com uma certa probabilidade permitindo que seja escolhida a nova configuração dentre várias candidatas. Uma estratégia parecida é usada no algoritmo **GRASP** ("Greedy Randomized Adaptive Search Procedure") [67] em que não necessariamente é escolhida a melhor configuração candidata.

Na literatura especializada aparecem outras estratégias para implementar algoritmos híbridos, mas as propostas anteriormente mencionadas foram consideradas como as mais atrativas para serem incorporadas num algoritmo híbrido usando **TS** como base para o problema de planejamento de sistemas de transmissão.

6.3 ALGORITMO HÍBRIDO APLICADO AO PROBLEMA DE PLANEJAMENTO

6.3.1 Apresentação das Estratégias que Formarão o Algoritmo Híbrido

SA inicia o processo de otimização a partir de uma configuração inicial e depois gera um conjunto de configurações candidatas controladas pelo programa de resfriamento. As configurações candidatas são aceitas se apresentam um custo menor que a configuração corrente; mas também podem ser aceitas se o custo for maior que a configuração corrente de acordo com um critério probabilístico. Esta última característica, aceitar configurações com custo mais elevado, permite a **SA** sair de eventuais ótimos locais. Este processo é controlado pelo programa de resfriamento. Assim a utilização de um algoritmo de **SA** implica especificar três aspectos distintos e independentes:

1. Uma adequada representação do problema,
2. Um mecanismo de transição e
3. Um programa de resfriamento.

Geralmente existem várias formas de especificar cada um destes ítens, o que leva a diferentes algoritmos de **SA**.

AG inicia o processo de otimização a partir de uma população inicial, isto é, de um conjunto de configurações que podem ser geradas aleatoriamente ou através de uma heurística construtiva. Um mecanismo geracional, definido pelo processo de seleção, "crossover" e mutação, permite gerar novas populações ou gerações cada vez melhores. O mecanismo de seleção direciona a busca de ótimos locais ao priorizar as configurações com funções objetivo de melhor qualidade. Por outro lado, no contexto do problema de planejamento de sistemas de transmissão, o operador de "crossover" permite diversificar a busca em regiões atrativas e o mecanismo de mutação permite gerar configurações vizinhas, dando ao algoritmo genético um nível de intensificação.

TS parte de uma configuração inicial (ou de um conjunto de configurações candidatas obtidas usando um algoritmo aproximado) e realiza um processo de busca mais agressiva que os métodos anteriores. Neste contexto, **TS** define uma estrutura de vizinhança para cada configuração corrente e passa para o *melhor vizinho*, isto é, para a configuração vizinha que produz uma maior diminuição da função objetivo ou para aquela que produz uma menor degradação da função objetivo. Uma estratégia inteligente permite analisar somente os vizinhos mais promissores e uma lista classifica um conjunto de configurações

(ou seus atributos) como sendo proibidos por um determinado número de iterações, a fim de evitar retornar a configurações já visitadas. Nesta estratégia básica **TS** usa a memória (histórico do processo) para direcionar a busca proibindo operações pouco atrativas. Adicionalmente existem operadores mais sofisticados como a intensificação e a diversificação que permite a **TS** realizar uma busca intensa nas regiões mais atrativas. A dimensão e a participação destes operadores podem ser variados dinamicamente durante o processo levando em conta o histórico do processo assim como as características específicas do problema.

A configuração inicial é geralmente infactível, isto é, apresenta corte de carga. Assim, o algoritmo **TS** realiza um processo cíclico com parcelas claramente diferenciadas:

1. Uma estratégia, chamada **A**; onde o processo, partindo de uma configuração infactível, encontra uma configuração factível.
2. Uma estratégia, chamada **B**; que consiste em encontrar configurações vizinhas factíveis da configuração corrente factível, e
3. Outra estratégia, chamada **C**, que consiste em sair da região factível e entrar na região infactível para tentar entrar em uma outra região factível usando novamente a estratégia **A**.

Este processo repetido permite encontrar muitos ótimos locais.

TS permite projetar estratégias diferentes em cada parte do processo dentro do esquema conhecido como *oscilação estratégica* onde existem três estratégias diferentes: uma para a região factível, outra para a zona de fronteira e a última para a região infactível. Nestas estratégias estão presentes os mecanismos de memória, intensificação e diversificação ponderados de maneira diferente.

6.3.2 A Representação do Problema

Representar adequadamente um problema significa que deve existir uma forma clara de identificar uma configuração e avaliar a sua função objetivo. No problema de planejamento da expansão de sistemas de transmissão uma configuração é uma proposta de adição de linhas e a função objetivo apresenta duas parcelas: uma correspondente aos custos das linhas adicionadas e outra correspondente a uma penalização pela existência de corte de carga correspondente à configuração analisada, tal como é apresentado na equação (2.1). Assim a representação do problema de planejamento é praticamente equivalente para os três algoritmos.

6.3.3 O Problema da Estrutura de Vizinhança

A estrutura de vizinhança é o mecanismo de transição que permite identificar todos os vizinhos da configuração corrente. Em **SA** uma configuração vizinha é toda configuração que pode ser obtida através de adição, troca ou retirada de uma linha ou conjunto de linhas no caso de novos caminhos. Neste contexto, o número de vizinhos é muito grande, mas em **SA** essa definição de vizinho não constitui problema pois **SA** analisa simplesmente um vizinho candidato e decide se aceita ou não aceita essa nova configuração; e nesse processo de decisão não é levada em conta a qualidade e outras características dos outros vizinhos. Em **AG** não existe a definição de vizinhança normal pois o mecanismo de transição que permite encontrar uma nova geração é realizado pelos operadores de seleção, "crossover" e mutação.

Em **TS** a vizinhança deve ser definida de uma forma mais sofisticada pelo caráter agressivo de **TS**, isto é, a partir da configuração corrente, **TS** passa para o *melhor vizinho* e para isto devem ser avaliados todos os vizinhos. Os algoritmos de **TS** normalmente definem um critério de vizinhança com muitos vizinhos, mas são implementadas estratégias dinâmicas para avaliar somente um número reduzido de vizinhos e escolher a melhor configuração dentre os analisados. No problema de planejamento de sistemas de transmissão foi implementado uma estrutura de vizinhança ainda mais sofisticada, como foi mostrado no capítulo anterior.

6.3.4 Características de Memória de Curto e Longo Prazo

A *memória* em **TS** é usada para redirecionar o processo de otimização e tem basicamente quatro componentes:

1. Recência ("recency"),
2. Freqüência,
3. Qualidade ("quality") e
4. Influência ("influence").

Recência e *freqüência* consiste basicamente em armazenar as configurações (ou seus atributos) dos últimos movimentos e do processo como um todo, respectivamente. Essa informação é usada para escolher da forma mais adequada as novas configurações. *Qualidade* se refere à capacidade de diferenciar o mérito das configurações encontradas. Assim, por exemplo, se poderia identificar adições de linhas que são comuns às excelentes configurações; e essas adições, ou conjunto de adições localizadas, podem ser usadas para

encontrar novas configurações assim como para implementar a estratégia de "path relinking". A *influência* considera o impacto de uma escolha durante o processo de busca. Assim, por exemplo, se a adição de uma linha produz uma grande diminuição do corte de carga então a retirada dessa linha deve ter pouco incentivo, isto é, deve permanecer proibida durante um maior número de iterações. Portanto, **TS** faz um uso extensivo e sistemático da estrutura de memória a fim de direcionar o processo de busca às regiões atrativas e evitar revisitar configurações já analisadas.

SA faz pouco uso de memória, isto é, do histórico do processo de otimização; ou seja, **SA** pode facilmente revisitar uma configuração já analisada e a busca está baseada num critério de mecânica estatística. Entretanto **SA**, assim como os outros métodos, pode armazenar a melhor configuração encontrada (incumbente), mas esta informação não redireciona o processo de busca. Uma implementação mais sofisticada de **SA**, que usa o histórico do processo, consiste em determinar o desvio padrão das funções objetivo das configurações candidatas durante cada nível de temperatura, e essa informação é usada para calcular a taxa de diminuição de temperatura e portanto o valor do próximo nível de temperatura. Neste caso, os novos valores de temperatura dependem do histórico do processo na temperatura anterior.

AG apresenta uma estratégia diferente através do mecanismo de seleção, "crossover" e mutação. Neste contexto a operação de seleção e a construção sistemática de "building blocks" nas configurações constituem uma espécie de memória implícita que direcionam o processo de busca. Adicionalmente, quando **AG** usa taxas de "crossover" menores que a unidade, uma parcela das configurações selecionadas não são submetidas a "crossover", constituindo um caso de memória explícita que mantém um pequeno subconjunto de configurações sem cruzamento a fim de preservar a qualidade das configurações nas próximas gerações.

6.3.5 Intensificação e Diversificação

O mecanismo de *intensificação* em **TS** consiste em analisar de uma forma mais sistemática a vizinhança de excelentes configurações encontradas durante o processo de otimização. Essas excelentes configurações armazenadas constituem, na verdade, configurações correspondentes a diferentes regiões onde deve existir um ótimo local. Assim, a intensificação tenta encontrar o ótimo local de cada região correspondente às excelentes configurações armazenadas. No processo de intensificação devem ser redefinidos os critérios de busca e no problema de planejamento esta fase corresponde à chamada estratégia **B** que visa encontrar a melhor configuração na vizinhança de uma configuração factível. Adicionalmente podem ser implementados outros processos de intensificação usando configurações de elite armazenadas e utilizando o "path relinking" para encontrar outras configurações excelentes.

Em **SA** a intensificação é um processo menos sistemático. O critério de aceitação de tipo estatístico de **SA** faz com que este tipo de algoritmo apresente grande diversificação nos primeiros níveis de temperatura, onde muitas configurações com incremento da função objetivo são aceitas. Por outro lado, nas fases finais do processo, **SA** realiza basicamente intensificação procurando o melhor ótimo local e, esporadicamente, pode diversificar passando a uma outra região vizinha.

Em **AG** a intensificação é realizada de forma não sistemática pelo operador de mutação, que realiza pequenas alterações numa configuração e, portanto, produz uma configuração vizinha ou muito próxima da configuração submetida a mutação. Também, nas fases finais de **AG** o "crossover" faz basicamente intensificação, pois os elementos da população estão constituídas por configurações muito parecidas.

A *diversificação* em **TS** é projetada para levar o processo de busca a regiões novas ou não visitadas e que pode ser implementada basicamente modificando as regras de escolha das configurações, usando atributos de configurações pouco utilizados. No problema de planejamento, a diversificação é implementada na estratégia **C** que visa entrar na região não factível (retirando linhas e proibindo sua entrada) a fim de levar o processo a uma outra região de ótimo local. Além disso é implementado diversificação usando "path relinking".

Em **SA** a forma aleatória de aceitação de uma configuração introduz diversificação ao processo de busca com maior intensidade nas fases iniciais do processo e com menor intensidade nas fases finais do processo, onde raramente são aceitas configurações com uma deteriorização da função objetivo.

Em **AG** a diversificação está presente fundamentalmente na forma do operador de "crossover" onde duas boas configurações trocam parte de seus elementos para gerar duas outras configurações que eventualmente podem corresponder a configurações de outras regiões não visitadas do processo. Adicionalmente, a diversificação está presente no caráter aleatório da escolha das configurações que devem ser cruzadas assim como dos pontos de "crossover".

6.4 IMPLEMENTAÇÕES ESPECIALIZADAS NO ALGORITMO HÍBRIDO

6.4.1 Mecanismo Básico do Algoritmo: Estratégia *A* - *B* - *C*

Na Figura 5.12 é mostrado o funcionamento do algoritmo **TS**; este inicia-se com uma configuração e se aplica a seqüência de estratégias **A** - **B** - **C**. Assim, para melhorar o desempenho do processo, pode-se trabalhar com uma população de forma equivalente aos

AG, escolhendo um conjunto de configurações iniciais, cada uma realizando a estratégia mostrada na Figura 5.12. Neste contexto, cada configuração realiza o ciclo mostrado na figura até que um critério de parada seja satisfeito; tipicamente, até que sejam realizadas um número especificado de movimentos consecutivos com degradação da função objetivo. Uma vez que cada configuração esgota o número de iterações, termina o processo de intensificação. Então, passa-se a selecionar uma nova configuração entre as previamente armazenadas e que não haja sido analisada previamente; analisadas todas as configurações, são submetidas a diversificação, o qual está baseado na operação "crossover".

A seguir são descritas as três estratégias empregadas:

Estratégia *A*

Nesta parte o algoritmo parte de uma configuração infactível obtida de um conjunto de configurações conforme especificado no capítulo 4. Assim, a estratégia é encontrar uma configuração factível, isto é, sem corte de carga.

O aspecto mais importante desta parte é a caracterização da vizinhança da configuração corrente. Assim, é considerado vizinho da configuração corrente, e armazenado numa lista, um conjunto de adições e trocas de linhas obtidas da seguinte forma:

1. As melhores propostas de adição obtidas de três métodos aproximados,
2. Propostas de adição de linhas nas barras onde os problemas de geração ou corte de carga são mais críticos para a configuração corrente, incluindo neste caso linhas consecutivas na formação de novos caminhos,
3. Um pequeno conjunto de adições de linhas escolhidas aleatoriamente, e
4. Um pequeno conjunto de trocas de linhas escolhidas aleatoriamente.

Neste contexto são avaliadas as funções objetivo de cada elemento da lista e passa-se para aquela configuração que produz uma maior diminuição da função objetivo. No caso em que todos os vizinhos produzem um incremento da função objetivo, escolhe-se entre as k configurações menos piores de acordo com um critério similar ao critério de aceitação de **SA**; isto é, às vezes é escolhida a menos pior das configurações (como sugere **TS**) e outras vezes se escolhe aleatoriamente entre as outras $k-1$ menos piores. Tipicamente são usados valores $k \leq 5$. Esta estratégia, no conjunto, visa manter a agressividade própria do algoritmo **TS** puro e dar um pequeno nível de diversidade presentes em **SA** e **AG**. Deve-se lembrar que a avaliação de cada vizinho candidato implica uma chamada à subrotina de PL.

Os três métodos aproximados usados para formar a lista de configurações candidatas são o algoritmo de Garver [9], o algoritmo de mínimo esforço [12] e o algoritmo de

mínimo corte de carga [35] amplamente usados em trabalhos de planejamento. Cada um destes algoritmos fornece um conjunto de alternativas atrativas de adição com um esforço computacional que implica uma chamada ao algoritmo de PL. Em todo o processo da estratégia **A** é mantida duas listas tabu com memória baseada em recência ("recency") que proíbe a entrada ou saída de linhas (que são atributos das configurações) por um número determinado de iterações. O número de iterações em que um atributo (linha(s)) deve permanecer proibido pode ser também controlado dinamicamente. Adicionalmente também é implementado um *critério de aspiração* que libera da condição de tabu um atributo (linha(s)) que produz uma configuração melhor que todas as encontradas durante a estratégia **A**.

Estratégia *B*

A estratégia **B** é fundamentalmente um processo de intensificação, isto é, encontrar as melhores configurações factíveis na vizinhança da configuração corrente factível. Neste caso não é possível obter informação dos métodos aproximados como na estratégia **A** e a lista de configurações vizinhas prioriza outros critérios. As configurações factíveis na vizinhança da configuração corrente podem ser encontradas basicamente com pequenas modificações da configuração corrente, especificamente através de trocas e/ou retiradas de linhas menos importantes e de pequeno impacto. Assim, a lista de configurações candidatas é formada por um conjunto de linhas formadas da seguinte forma:

1. Linhas candidatas a sair constituída por uma parcela das linhas mais baratas e outra parcela aleatoriamente escolhidas, e
2. Linhas candidatas a trocar, escolhidas com o mesmo critério anterior.

Cada vez que é encontrada uma nova configuração factível, é elaborada uma nova lista e a estratégia **B** termina quando, para uma determinada lista, não foi encontrada uma melhor configuração factível e, nesse caso, deve-se implementar a estratégia **C**. Na estratégia **B** não existe a lista proibida ("tabu tenure"), apenas são armazenadas as linhas que são adicionadas durante a estratégia.

Estratégia *C*

Esta parte é simples e rápida. Neste caso são retiradas uma ou algumas linhas de importância secundária (tipicamente as mais baratas) que não foram adicionadas na estratégia **B** e proibe-se sua(s) entrada(s) por um número relativamente elevado de iterações. Este mecanismo permite entrar na região infactível, que deve levar o processo a uma configuração factível implementando novamente a estratégia **A**. Esta estratégia visa evitar retornar à região factível já visitada e entrar numa outra região factível vizinha da anterior (ver Figura 5.12).

No problema de planejamento de sistemas de transmissão, a estratégia **A** solicita um maior esforço computacional e freqüentemente existe dificuldade em encontrar uma configuração factível. Por outro lado a estratégia **B** é mais rápida pois existem relativamente poucas configurações factíveis na vizinhança da configuração corrente factível; e a estratégia **C** é apenas um passo, mais adequadamente implementado.

O processo mostrado na Figura 5.12 termina uma vez satisfeito o critério de parada que consiste em parar quando, durante um número consecutivo de iterações, foram realizados movimentos de piora da função objetivo. Esta parte do processo tipicamente termina na estratégia **A** e, ao final do processo, deve-se armazenar a incumbente encontrada durante toda esta parte do processo.

6.4.2 Uso de Uma População de Configurações

O conjunto de configurações iniciais é obtido usando diversas variantes do algoritmo de Garver modificado, como apresentado no capítulo 4. Foi observado que o algoritmo de Garver, com pequenas modificações, produz um conjunto de configurações de excelente qualidade. Típicamente, no problema de planejamento, este algoritmo consegue identificar as linhas mais importantes que fazem parte da configuração ótima assim como os novos caminhos mais importantes ligando as barras isoladas da configuração base. Esta forma de encontrar configurações iniciais fornece aos quatro algoritmos uma eficiência inusitada quando comparados com configurações geradas aleatoriamente, como podem ser verificados nos resultados apresentados.

6.4.3 Implementação de Configurações Elite

Durante todo o processo do algoritmo podem ser armazenadas um conjunto das melhores configurações (tipicamente entre 2 e 6 configurações no problema de planejamento), com a finalidade de implementar processos adicionais de intensificação usando estas configurações de elite e também para fazer "path relinking", a fim de encontrar excelentes configurações pertencentes a outras regiões.

6.4.4 Implementação de "Path Relinking"

O "path relinking" é feito entre duas configurações, uma chamada de *referência* e outra chamada de *guia*, neste processo, partes da configuração guia são copiadas na configuração de referência. Este processo é controlado e depende do tipo de aplicação, intensificação, diversificação ou para reiniciar novamente todo o processo. Neste algoritmo o "path

relinking" será utilizado para diversificação. No capitulo 5 foi explicado o procedimento que se segue.

6.4.5 Uso de Estratégias Inspiradas em SA

O **SA** é empregado, na etapa de intensificação, nas estratégias A e B do processo de oscilação estratégica. No algoritmo **TS** emprega-se um procedimento agressivo no processo de intensificação, ou seja: sempre é aceito o melhor vizinho que pode ser de melhor ou pior qualidade com relação a solução corrente. No algoritmo **SA** se aceita uma solução se esta é de melhor qualidade e com uma certa probabilidade, caso contrário. Esta última característica foi então incorporada ao processo de oscilação nas estratégias A e B. Nos dois algoritmos é possível sair dos ótimos locais aceitando soluções de pior qualidade com relação a corrente, ou seja, em **TS** são aceitas configurações com função objetivo não muito boas. Em **SA** é possível controlar a aceitação de configurações péssimas através de uma adequada escolha do parâmetro temperatura; com isto se permite que o processo faça uma busca mais detalhada na região que está sendo estudada.

6.4.6 Aproveitamento das Características do SA, GA e TS no Problema do Planejamento de Redes de Transmissão

Os algoritmos combinatoriais estão sendo usados em problemas onde já existiam outros métodos aproximados e, além disso, cada um destes problemas apresentam características específicas que favorecem ou prejudicam a utilização dos algoritmos combinatoriais. No problema de planejamento de sistemas de transmissão podem ser analisados três aspectos relacionados com este assunto: configuração inicial, incorporação de características específicas do problema, e incorporação e aproveitamento de características importantes de outros algoritmos aproximados.

Os três algoritmos, **SA**, **AG** e **TS**, precisam de uma configuração ou um conjunto de configurações para iniciar o processo e essas configurações podem ser geradas aleatoriamente ou usando um outro método aproximado. Embora a análise teórica mostre que a configuração inicial não deve influir no desempenho do algoritmo, foi verificado experimentalmente, para o problema de planejamento de redes de transmissão, que uma boa configuração inicial influi tanto no esforço computacional assim como na qualidade da melhor configuração encontrada. No problema de planejamento, as configurações encontradas com o algoritmo de Garver [9], com ligeiras modificações, fornecem excelentes configurações iniciais para os três algoritmos combinatoriais.

Em relação às características específicas do problema, no problema de planejamento existe o caso peculiar da existência de barras isoladas na configuração base; assim,

freqüentemente, deve-se adicionar várias linhas simultaneamente através dos novos caminhos. Nos três algoritmos, este problema é contornado eficientemente pela implementação de um procedimento sistemático que faz a adição/retirada de linha através dos novos caminhos de uma maneira inteligente. Entretanto **AG** faz um aproveitamento mais eficiente desta característica modificando a estratégia de formação dos "building blocks".

A incorporação de características importantes de outros algoritmos aproximados é realizada de maneira mais eficiente em **TS**. A geração de boas configurações iniciais usando algoritmos aproximados extremamente rápidos é uma característica aplicável aos três algoritmos. Entretanto, **TS** permite um uso mais eficiente de outras características como, por exemplo, aproveitar aspectos importantes de algoritmos aproximados usados para resolver o problema em análise. Um problema que **TS** enfrenta, na formulação geral, é a redução do tamanho da vizinhança, pois pela lógica de trabalho agressiva de **TS**, uma vez definida a vizinhança, deve-se avaliar todos os vizinhos da configuração corrente a fim de passar para o melhor vizinho. Assim, em aplicações gerais, deve-se dispor de uma técnica para reduzir dinâmica e drasticamente o tamanho da vizinhança a um número reduzido de candidatos ("candidate list strategies"). No problema de planejamento, essa vizinhança reduzida pode ser implementada de uma forma eficiente. Assim, por exemplo, na estratégia **A** da Figura 5.12, quando a configuração corrente apresenta corte de carga, pode ser elaborada uma lista de linhas ou conjunto de linhas candidatas a adição constituída pelas melhores candidatas a adição fornecida pelos melhores métodos aproximados os quais podem ser obtidos com pequeno esforço computacional. A maioria dos métodos aproximados fornecem em cada estágio uma lista de candidatas à adição, e cujos indicadores estão baseados em sensibilidade; assim **TS** pode usar eficientemente estas informações dos métodos aproximados para gerar uma lista pequena e de qualidade que na verdade é uma nova forma de caracterizar a vizinhança da configuração corrente.

6.4.7 Implementação de Versões Híbridas

TS é o mais adequado para ser usado como base ou referencial ao implementar versões híbridas. Podem ser formulados algoritmos de **TS** que utilizem componentes de **SA** e **AG** e também de outros algoritmos aproximados amplamente utilizados para resolver o problema de planejamento de sistemas de transmissão a longo prazo. As seguintes versões híbridas foram implementadas: **TS - SA**, **TS - AG** e **TS - SA - AG**.

6.4.8 Critério de Parada.

Três critérios de parada são empregados, um para a estratégia individual mostrada na Figura 5.12 e os outros dois para o processo global. O primeiro, pára se durante um número determinado de iterações não obtém uma melhor solução; já o segundo e terceiro

são critérios de parada do programa, um deles emprega um critério similar ao empregado nos algoritmos genéticos; assim, o processo pára quando a solução incumbente não é melhorada em um dado número de ciclos, o outro critério pára o programa quando as k configurações armazenadas na matriz de trabalho tiverem sido analisadas, ou seja, as k posições do vetor associado as configurações tiverem sinal de revisão. O programa pára quando é satisfeito um dos critérios de parada.

6.5 ALGORITMO HÍBRIDO IMPLEMENTADO

O algoritmo *híbrido* implementado funciona basicamente como é mostrado na Figura 5.12. Ou seja, a versão híbrida utiliza fundamentalmente os conceitos básicos do **TS** e aproveita outros conceitos extraídos dos **AG** e **SA**, como são o emprego de uma população de configurações, do "crossover" (como elemento básico de diversidade) e do parâmetro temperatura, que permitem sair dos ótimos locais mais eficientemente no processo de intensificação.

Na Figura 6.1 é apresentado um diagrama de fluxo da metodologia híbrida, a qual está constituída basicamente pelos processos de intensificação e diversificação, que são as principais funções da metodologia desenvolvida.

Na metodologia híbrida, o processo de intensificação está presente nas estratégias **A** e **B**, onde se visa encontrar configurações vizinhas melhores.

Com relação ao processo de diversificação, o mesmo está presente nas estratégias **A** e **C**. Na primeira, através do **SA**, aceitando soluções piores com certa probabilidade; isto permite que o processo possa sair mais eficientemente de soluções locais. Na segunda, fazendo substituições obrigatórias, que visam levar o processo para uma região vizinha a partir da configuração corrente factível. Estes dois processos de diversidade são considerados simples, ou seja, de pouca intensidade.

Além da diversidade anterior, no presente algoritmo foi implementado uma diversificação brusca ou intensa que visa levar o processo às regiões distantes. Este objetivo é atingido usando um conjunto de configurações formando uma população, como no algoritmo genético, em lugar de uma simples configuração de **TS** básico. Assim, depois de que cada configuração da população realiza o processo da Figura 5.12, é armazenada sua respectiva incumbente, com as quais se forma uma nova população, e é efetuado um cruzamento direcionado com todas as incumbentes armazenadas. Este cruzamento direcionado é realizado de tal forma que evite a destruição dos novos caminhos já constituídos nas incumbentes submetidas a cruzamento. Uma vez gerada os elementos da nova população o processo pode ser reinicializado. Evidentemente, essa implementação de cruzamento direcionado produz uma intensa diversificação, permitindo visitar regiões distantes na tentativa de encontrar novas regiões promissoras.

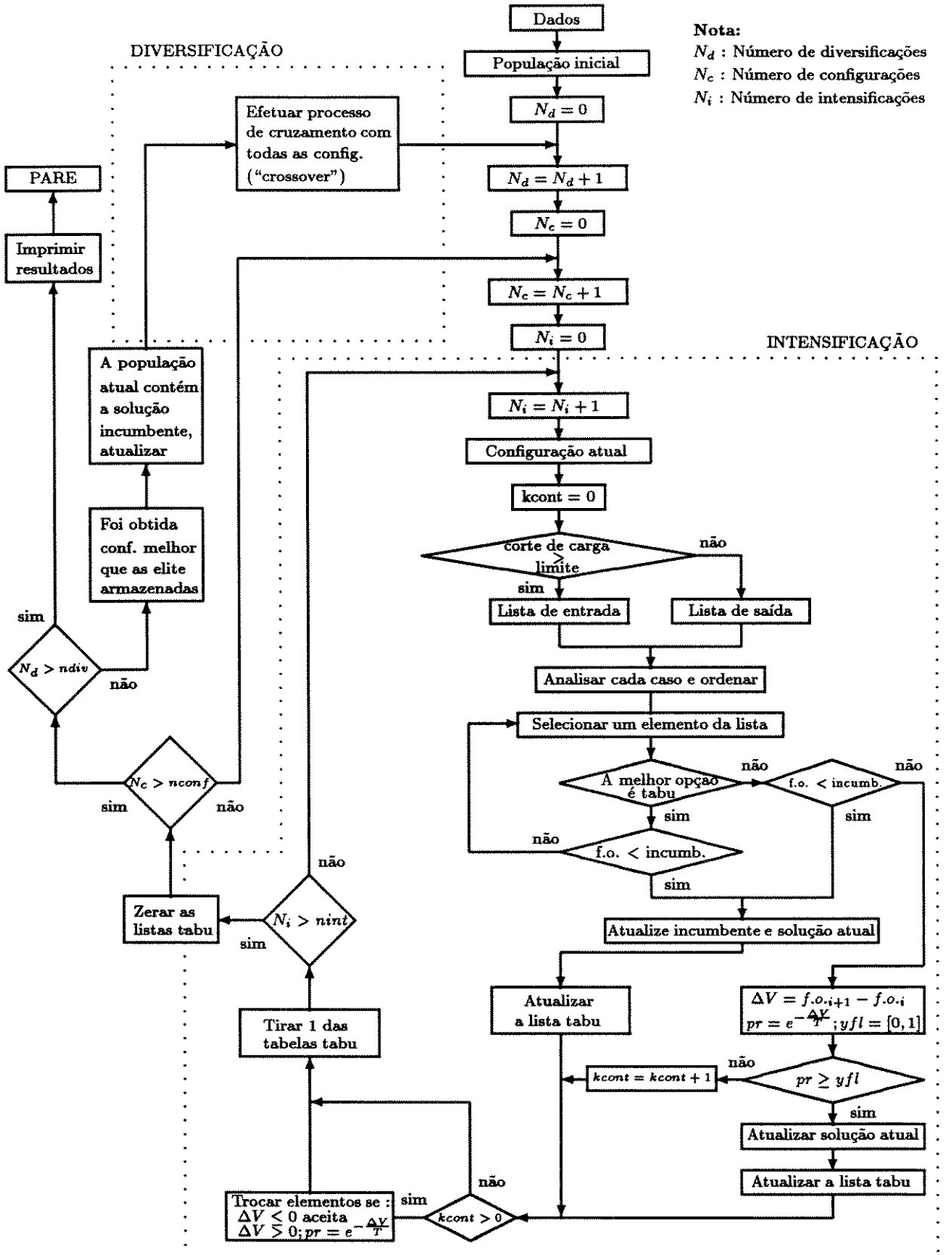


Figura 6.1: Diagrama de fluxo da metodologia híbrida

6.6 TESTES EM SISTEMAS DE PEQUENO E DE MÉDIO PORTE

Com os resultados apresentados se poderá observar o desempenho do método híbrido quando são empregados sistemas de porte pequeno e médio. Nestes testes se estabelecerá uma comparação das diferentes combinações na qual **TS** utiliza processos próprios do **SA** e **AG**. Assim, são apresentados testes com combinação de **TS - SA**, **TS - AG** e **TS - AG - SA**, formando estes os chamados *métodos híbridos*. O melhor dos resultados do método híbrido será comparado com o **SA**, **AG** e **TS** apresentados nos capítulos 3, 4 e 5 respectivamente.

Entre os parâmetros de controle, tem-se: custo de corte de carga, número máximo de intensificações, número máximo de diversificações, número de proibições tabu, valor fixo e valor variável do número de proibições, tamanho das listas para seleção de configurações candidatas, número de configurações que formam a população, número de configurações elite armazenadas, número de diversificações aleatórias.

Para encontrar os valores mais adequados dos parâmetros de controle foram realizados testes preliminares. Os parâmetros dos algoritmos foram calibrados para permitir desempenhos de praticamente 100% de eficiência em encontrar as soluções ótimas dos sistemas de 06 barras e sul brasileiro apresentados. Assim, a comparação deve ser feita em relação ao esforço computacional requerido por cada algoritmo (PL's utilizados).

6.6.1 Sistema de 06 Barras de Garver com Redespacho

São apresentados os resultados de testes realizados com configurações iniciais obtidas aleatoriamente. Foram analisados os três tipos de algoritmos híbridos descritos anteriormente.

Na Tabela 6.1 são apresentados os resultados de 20 testes para cada um dos algoritmos propostos, empregando para este estudo o sistema Garver com redespacho.

Na primeira versão do algoritmo híbrido, **TS** incorporou-se características do **SA**. Foram empregadas quatro posições de memória para armazenar as melhores configurações, que depois são analisadas tal como se apresenta na Figura 5.9. Nesta versão, o valor médio dos 20 casos analisados necessários para atender a convergência foi de 452 PL's.

Na segunda versão do algoritmo híbrido, **TS** é combinada com **AG**. Foram empregadas quatro posições de memória para armazenar as melhores soluções, tal como se apresenta na Figura 5.10. Nesta, o valor médio dos 20 casos analisados necessários para atender a convergência foi de 383 PL's.

Tabela 6.1: Resultados do sistema Garver com redespacho

Híbrido - Sistema Garver - caso com redespacho, f.o. = 110						
Testes dos algoritmos híbridos formados por **TS - SA ; TS - AG ; TS - SA - AG**						
	Inicialização aleatória					
	TS - SA		**TS - AG**		**TS - SA - AG**	
Caso Nº	Nº PL 1ª vez(ND)	Nº PL final(ND)	Nº PL 1ª vez(ND)	Nº PL final(ND)	Nº PL 1ª vez(ND)	Nº PL final(ND)
1	161(*)	440(7)	95(*)	306(3)	95(*)	367(3)
2	39(*)	423(7)	312(1)	496(3)	146(*)	330(3)
3	335(3)	466(7)	167(*)	334(3)	152(*)	410(3)
4	102(*)	457(6)	101(*)	344(3)	199(*)	461(3)
5	39(*)	458(6)	39(*)	412(3)	99(*)	374(3)
6	39(*)	463(7)	39(*)	403(3)	203(*)	379(3)
7	93(*)	432(7)	103(*)	411(3)	361(2)	398(3)
8	291(2)	420(6)	94(*)	325(3)	233(*)	419(3)
9	39(*)	358(6)	285(1)	414(3)	155(*)	408(3)
10	138(*)	405(7)	102(*)	390(3)	39(*)	390(3)
11	39(*)	458(7)	95(*)	373(3)	39(*)	461(3)
12	117(*)	536(7)	346(3)	387(3)	165(*)	372(3)
13	148(*)	399(6)	39(*)	311(3)	360(2)	452(3)
14	99(*)	453(7)	97(*)	392(3)	39(*)	364(3)
15	217(*)	468(6)	222(*)	375(3)	65(*)	462(3)
16	148(*)	463(6)	351(2)	419(3)	202(*)	403(3)
17	65(*)	398(7)	99(*)	377(3)	225(*)	395(3)
18	136(*)	408(7)	39(*)	361(3)	39(*)	356(3)
19	68(*)	398(6)	317(2)	437(3)	309(1)	457(3)
20	134(*)	417(7)	39(*)	384(3)	277(1)	398(3)

* - resultado obtido durante o primeiro processo de otimização (nível I)

Na terceira versão do algoritmo híbrido, **TS** é combinada com **AG** e **SA**. Foram empregadas quatro posições de memória para armazenar as melhores soluções, tal como se apresenta na Figura 5.10. Nesta, o valor médio dos 20 casos analisados necessários para atender a convergência foi de 403 PL's.

6.6.2 Sistema de 06 Barras de Garver sem Redespacho

Na Tabela 6.2 são apresentados os resultados de 20 testes para cada um dos algoritmos propostos, empregando para este estudo o sistema Garver sem redespacho.

Na primeira versão do algoritmo híbrido, **TS - SA**, foram empregadas cinco posições de memória para armazenar as melhores configurações, tal como se apresenta na Figura 5.9. Nesta, o valor médio dos 20 casos analisados necessários para atender a convergência foi de 652 PL's.

Na segunda versão do algoritmo híbrido, **TS - AG**, foram empregadas cinco posições de memória para armazenar as melhores soluções, tal como se apresenta na Figura 5.10. Nesta, o valor médio dos 20 casos analisados foi de 569 PL's.

Na terceira versão do algoritmo híbrido, **TS - AG - SA**. Foram empregadas cinco posições de memória para armazenar as melhores soluções, tal como se apresenta na Figura 5.10. Nesta, o valor médio dos 20 casos analisados foi de 628 PL's.

6.6.3 Sistema Sul Brasileiro com Redespacho

Para este sistema foram feitos testes tanto com configurações iniciais geradas de maneira aleatória como com configurações geradas com a metodologia de Garver modificada.

Nas Tabelas 6.3, 6.4 e 6.5, a seguir, apresentam-se os resultados do sistema Sul brasileiro com redespacho, usando diferentes metodologias combinadas, ou seja: **TS-SA**, **TS-AG** e **TS-SA-AG**.

6.6.4 Sistema Sul Brasileiro sem Redespacho

Nas Tabelas 6.6, 6.7 e 6.8, a seguir, apresentam-se os resultados do sistema Sul brasileiro sem redespacho, usando as diferentes metodologias híbridas.

Tabela 6.2: Resultados do sistema Garver sem redespacho

Híbrido - Sistema Garver - caso sem redespacho, f.o. = 200						
Testes dos algoritmos híbridos formados por **TS - SA ; TS - AG ; TS - SA - AG**						
	Inicialização aleatória					
	TS - SA		**TS - AG**		**TS - SA - AG**	
Caso Nº	Nº PL 1ª vez(ND)	Nº PL final(ND)	Nº PL 1ª vez(ND)	Nº PL final(ND)	Nº PL 1ª vez(ND)	Nº PL final(ND)
1	65(*)	637(7)	219(*)	621(3)	141(*)	608(3)
2	66(*)	712(7)	57(*)	588(3)	65(*)	712(3)
3	65(*)	634(8)	305(*)	562(3)	423(1)	673(3)
4	66(*)	598(7)	57(*)	545(3)	66(*)	570(3)
5	66(*)	661(9)	65(*)	577(3)	142(*)	717(3)
6	139(*)	601(7)	145(*)	557(3)	66(*)	645(3)
7	66(*)	686(8)	66(*)	657(3)	65(*)	612(3)
8	79(*)	610(7)	78(*)	600(3)	65(*)	551(3)
9	66(*)	667(9)	149(*)	586(3)	66(*)	599(3)
10	65(*)	661(7)	65(*)	537(3)	66(*)	627(3)
11	66(*)	612(7)	57(*)	565(3)	66(*)	652(3)
12	66(*)	646(9)	66(*)	578(3)	66(*)	621(3)
13	348(*)	682(7)	157(*)	533(3)	136(*)	621(3)
14	66(*)	653(7)	229(*)	606(3)	305(*)	643(3)
15	80(*)	669(8)	143(*)	519(3)	66(*)	665(3)
16	66(*)	717(7)	57(*)	554(3)	368(*)	556(3)
17	65(*)	671(9)	66(*)	545(3)	150(*)	633(3)
18	136(*)	611(7)	145(*)	534(3)	65(*)	626(3)
19	66(*)	671(8)	65(*)	533(3)	179(*)	625(3)
20	134(*)	632(7)	366(*)	574(3)	313(*)	611(3)

$*$ - resultado obtido durante o primeiro processo de otimização (nível I)

Tabela 6.3: Resultados do sistema Sul brasileiro com redespacho (*TS - SA*)

Híbrido - Sistema Sul brasileiro - caso com redespacho						
Testes do algoritmo híbrido formado por **TS - SA**						
	Inicialização aleatória			Inicialização Garver modificado		
Caso Nº	Nº PL 1ª vez	Nº PL final	Custo (US$ mil)	Nº PL 1ª vez	Nº PL final	Custo (US$ mil)
1	2.003(*)	5.241(27)	70.289,00	1.291(7)	2.446(22)	70.289,00
2	1.354(*)	6.195(34)	70.289,00	226(*)	2.469(23)	70.289,00
3	3.794(14)	5.391(34)	70.289,00	473(*)	2.137(14)	70.289,00
4	2.183(*)	4.926(22)	70.289,00	27(*)	1.883(14)	70.289,00
5	3.268(5)	4.700(24)	70.289,00	108(*)	2.034(14)	70.289,00
6	1.906(*)	4.907(22)	70.289,00	389(*)	2.578(17)	70.289,00
7	2.367(*)	4.816(21)	70.289,00	988(3)	2.059(13)	70.289,00
8	3.329(6)	5.438(31)	70.289,00	968(2)	2.125(14)	70.289,00
9	4.316(18)	5.880(36)	70.289,00	325(*)	1.815(16)	70.289,00
10	2.485(*)	4.653(22)	70.289,00	1.266(5)	2.351(18)	70.289,00
11	1.151(*)	5.414(32)	70.289,00	858(2)	2.014(14)	70.289,00
12	2.794(*)	5.180(27)	70.289,00	1.029(3)	2.006(12)	70.289,00
13	915(*)	4.515(21)	70.289,00	759(*)	2.130(17)	70.289,00
14	2.764(2)	4.513(22)	70.289,00	239(*)	2.195(17)	70.289,00
15	581(*)	4.733(22)	70.289,00	428(*)	2.013(18)	70.289,00
16	1.179(*)	4.919(25)	70.289,00	1.038(6)	1.830(17)	70.289,00
17	263(*)	4.570(24)	70.289,00	27(*)	1.645(13)	70.289,00
18	715(*)	4.760(24)	70.289,00	152(*)	1.639(12)	70.289,00
19	3.079(2)	4.593(22)	70.289,00	297(*)	1.781(15)	70.289,00
20	2.040(*)	4.988(25)	70.289,00	1.221(8)	1.959(19)	70.289,00

∗ - resultado obtido durante o primeiro processo de otimização (nível I)

OBS: - Configurações geradas de maneira aleatória, com dezoito posições de memória, obtendo-se um valor médio de 5.017 PL's
- Configurações geradas com metodologia Garver modificada, com dez posições, obtendo-se um valor médio de 2.056 PL's.
- Ambas, conforme Figura 5.9

Tabela 6.4: Resultados do sistema Sul brasileiro com redespacho (*TS - AG*)

Híbrido - Sistema Sul brasileiro - caso com redespacho						
Testes do algoritmo híbrido formado por **TS - AG**						
	Inicialização aleatória			Inicialização Garver modificado		
Caso Nº	Nº PL 1ª vez	Nº PL final	Custo (10^3US$)	Nº PL 1ª vez	Nº PL final	Custo (10^3US$)
1	538(*)	4.848(4)	70.289,00	637(1)	1.651(4)	70.289,00
2	3.056(2)	4.739(4)	70.289,00	236(*)	1.902(4)	70.289,00
3	234(*)	4.260(4)	70.289,00	156(*)	1.748(4)	70.289,00
4	3.202(3)	4.634(4)	70.289,00	341(1)	1.533(4)	70.289,00
5	2.641(2)	4.066(4)	70.289,00	23(*)	1.817(4)	70.289,00
6	2.718(2)	4.460(4)	70.289,00	39(*)	1.589(4)	70.289,00
7	2.661(2)	4.107(4)	70.289,00	109(*)	1.641(4)	70.289,00
8	2.474(2)	4.216(4)	70.289,00	71(*)	1.770(4)	70.289,00
9	430(*)	4.302(4)	70.289,00	243(*)	1.406(4)	70.289,00
10	2.761(2)	3.921(4)	70.289,00	267(*)	1.731(4)	70.289,00
11	673(*)	4.025(4)	70.289,00	296(*)	1.757(4)	70.289,00
12	2.383(*)	4.379(4)	70.289,00	330(*)	1.637(4)	70.289,00
13	2.455(*)	4.240(4)	70.289,00	274(*)	1.653(4)	70.289,00
14	2.253(*)	4.435(4)	70.289,00	219(*)	1.643(4)	70.289,00
15	2.357(1)	4.326(4)	70.289,00	624(1)	1.610(4)	70.289,00
16	1.767(1)	4.148(4)	70.289,00	331(*)	1.679(4)	70.289,00
17	1.573(*)	4.584(4)	70.289,00	580(1)	1.891(4)	70.289,00
18	3.206(3)	3.962(4)	70.289,00	150(*)	1.542(4)	70.289,00
19	637(*)	4.176(4)	70.289,00	317(*)	1.525(4)	70.289,00
20	1.731(*)	4.374(4)	70.289,00	223(*)	1.464(4)	70.289,00

∗ - resultado obtido durante o primeiro processo de otimização (nível I)

OBS: - Configurações geradas de maneira aleatória, com doze posições de memória, obtendo-se um valor médio de 4.310 PL's
- Configurações geradas com metodologia Garver modificada, com seis posições, obtendo-se um valor médio de 1.660 PL's.
- Ambas, conforme Figura 5.10

Tabela 6.5: Resultados do sistema Sul brasileiro com redespacho (*TS - SA - AG*)

Híbrido - Sistema Sul brasileiro - caso com redespacho						
Testes do algoritmo híbrido formado por **TS - SA - AG**						
	Inicialização aleatória			Inicialização Garver modificado		
Caso Nº	Nº PL 1ª vez	Nº PL final	Custo (10^3US$)	Nº PL 1ª vez	Nº PL final	Custo (10^3US$)
1	2.560(2)	4.203(4)	70.289,00	314(*)	1.629(4)	70.289,00
2	189(*)	3.665(4)	70.289,00	440(1)	1.624(4)	70.289,00
3	2.239(1)	4.230(4)	70.289,00	299(*)	1.575(4)	70.289,00
4	718(*)	4.374(4)	70.289,00	876(2)	1.449(4)	70.289,00
5	2.988(3)	3.776(4)	70.289,00	23(*)	1.557(4)	70.289,00
6	937(*)	4.260(4)	70.289,00	45(*)	1.651(4)	70.289,00
7	514(*)	3.381(4)	70.289,00	339(*)	1.537(4)	70.289,00
8	190(*)	3.787(4)	70.289,00	399(*)	1.565(4)	70.289,00
9	122(*)	3.847(4)	70.289,00	162(*)	1.570(4)	70.289,00
10	2.078(2)	3.559(4)	70.289,00	215(*)	1.722(4)	70.289,00
11	312(*)	3.376(4)	70.289,00	275(*)	1.573(4)	70.289,00
12	2.430(2)	3.769(4)	70.289,00	107(*)	1.699(4)	70.289,00
13	2.427(2)	3.634(4)	70.289,00	46(*)	1.866(4)	70.289,00
14	2.287(2)	3.503(4)	70.289,00	40(*)	1.448(4)	70.289,00
15	519(*)	3.881(4)	70.289,00	142(*)	1.969(4)	70.289,00
16	2.522(2)	3.648(4)	70.289,00	118(*)	1.776(4)	70.289,00
17	1.981(1)	4.016(4)	70.289,00	118(*)	1.703(4)	70.289,00
18	1.290(*)	3.803(4)	70.289,00	314(*)	1.449(4)	70.289,00
19	2.353(1)	4.020(4)	70.289,00	23(*)	1.438(4)	70.289,00
20	2.955(*)	3.745(4)	70.289,00	42(*)	1.649(4)	70.289,00

* - resultado obtido durante o primeiro processo de otimização (nível I)

OBS: - Configurações geradas de maneira aleatória, com doze posições de memória, obtendo-se um valor médio de 3.824 PL's
- Configurações geradas com metodologia Garver modificada, com seis posições, obtendo-se um valor médio de 1.622 PL's.
- Ambas, conforme Figura 5.10

Tabela 6.6: Resultados do sistema Sul brasileiro sem redespacho (TS - SA)

Híbrido - Sistema Sul brasileiro - caso sem redespacho						
Testes do algoritmo híbrido formado por **TS - SA**						
	Inicialização aleatória			Inicialização Garver modificado		
Caso Nº	Nº PL 1ª vez	Nº PL final	Custo (10^3US$)	Nº PL 1ª vez	Nº PL final	Custo ($10^3 US$)
1	1.864(*)	6.641(33)	154.420,00	1.298(3)	2.197(14)	154.420,00
2	980(*)	5.573(25)	154.420,00	910(2)	2.468(19)	154.420,00
3	235(*)	5.526(22)	154.420,00	1.867(9)	2.641(18)	154.420,00
4	149(*)	5.294(20)	154.420,00	531(*)	3.614(30)	154.420,00
5	587(*)	5.528(22)	154.420,00	952(2)	2.373(18)	154.420,00
6	152(*)	5.366(22)	154.420,00	282(*)	2.747(20)	154.420,00
7	3.203(*)	5.910(29)	154.420,00	2.071(13)	3.038(25)	154.420,00
8	152(*)	5.648(22)	154.420,00	144(*)	2.324(16)	154.420,00
9	1.023(*)	5.470(24)	154.420,00	623(*)	2.373(19)	154.420,00
10	1.101(*)	5.670(25)	154.420,00	322(*)	2.970(22)	154.420,00
11	1.094(*)	5.171(26)	154.420,00	371(*)	3.388(27)	154.420,00
12	1.010(*)	5.217(23)	154.420,00	1.256(5)	2.120(17)	154.420,00
13	516(*)	5.253(27)	154.420,00	432(*)	2.648(20)	154.420,00
14	2.222(*)	5.578(26)	154.420,00	204(*)	3.201(24)	154.420,00
15	149(*)	5.350(22)	154.420,00	957(*)	2.523(17)	154.420,00
16	4.912(16)	5.878(31)	154.420,00	175(*)	2.697(20)	154.420,00
17	936(*)	5.597(21)	154.420,00	860(*)	2.686(20)	154.420,00
18	735(*)	5.860(25)	154.420,00	1.132(3)	2.205(16)	154.420,00
19	3.955(5)	5.214(21)	154.420,00	750(*)	2.363(16)	154.420,00
20	418(*)	5.620(22)	154.420,00	1.806(9)	2.626(18)	154.420,00

* - resultado obtido durante o primeiro processo de otimização (nível I)

OBS: - Configurações geradas de maneira aleatória, com dezoito posições de memória, obtendo-se um valor médio de 5.568 PL's
- Configurações geradas com metodologia Garver modificada, com doze posições, obtendo-se um valor médio de 2.660 PL's.
- Ambas, conforme Figura 5.9

Tabela 6.7: Resultados do sistema Sul brasileiro sem redespacho (*TS - AG*)

Híbrido - Sistema Sul brasileiro - caso sem redespacho						
Testes do algoritmo híbrido formado por **TS - AG**						
	Inicialização aleatória			Inicialização Garver modificado		
Caso Nº	Nº PL 1ª vez	Nº PL final	Custo (10^3US$)	Nº PL 1ª vez	Nº PL final	Custo ($10^3 US$$)
1	4.175(2)	5.942(4)	154.420,00	634(*)	2.812(4)	154.420,00
2	547(*)	6.151(4)	154.420,00	450(*)	2.726(4)	154.420,00
3	3.236(1)	5.494(4)	154.420,00	349(*)	2.580(4)	154.420,00
4	1.729(*)	4.998(4)	154.420,00	122(*)	2.386(4)	154.420,00
5	3.714(2)	5.389(4)	154.420,00	549(*)	2.564(4)	154.420,00
6	3.215(2)	4.891(4)	154.420,00	373(*)	2.833(4)	154.420,00
7	3.657(2)	5.078(4)	154.420,00	325(*)	2.599(4)	154.420,00
8	1.992(*)	5.275(4)	154.420,00	192(*)	2.352(4)	154.420,00
9	449(*)	5.204(4)	154.420,00	295(*)	2.558(4)	154.420,00
10	211(*)	5.208(4)	154.420,00	1.280(2)	2.303(4)	154.420,00
11	2.509(1)	5.086(4)	154.420,00	626(1)	2.429(4)	154.420,00
12	3.670(2)	5.459(4)	154.420,00	616(*)	2.415(4)	154.420,00
13	2.154(*)	5.339(4)	154.420,00	582(1)	2.148(4)	154.420,00
14	581(*)	5.018(4)	154.420,00	1.209(2)	2.690(4)	154.420,00
15	1.222(*)	5.025(4)	154.420,00	1.110(2)	2.058(4)	154.420,00
16	3.758(2)	5.324(4)	154.420,00	1.788(3)	2.434(4)	154.420,00
17	3.184(2)	5.202(4)	154.420,00	1.912(3)	2.385(4)	154.420,00
18	646(*)	5.177(4)	154.420,00	588(1)	2.138(4)	154.420,00
19	3.403(2)	4.756(4)	154.420,00	1.009(*)	2.607(4)	154.420,00
20	1.401(*)	4.852(4)	154.420,00	446(*)	2.515(4)	154.420,00

* - resultado obtido durante o primeiro processo de otimização (nível I)

OBS: - Configurações geradas de maneira aleatória, com doze posições de memória, obtendo-se um valor médio de 5.243 PL's
- Configurações geradas com metodologia Garver modificada, com oito posições, obtendo-se um valor médio de 2.477 PL's.
- Ambas, conforme Figura 5.10

Tabela 6.8: Resultados do sistema Sul brasileiro sem redespacho (*TS - SA - AG*)

Híbrido - Sistema Sul brasileiro - caso sem redespacho						
Testes do algoritmo híbrido formado por **TS - SA - AG**						
	Inicialização aleatória			Inicialização Garver modificado		
Caso Nº	Nº PL 1ª vez	Nº PL final	Custo (10^3US$)	Nº PL 1ª vez	Nº PL final	Custo ($10^3 US$)
1	2.164(*)	5.329(4)	154.420,00	1.605(2)	2.385(4)	154.420,00
2	146(*)	4.764(4)	154.420,00	1.689(2)	2.760(4)	154.420,00
3	2.679(1)	5.476(4)	154.420,00	761(1)	2.105(4)	154.420,00
4	558(*)	5.173(4)	154.420,00	1.427(3)	2.106(4)	154.420,00
5	772(*)	5.075(4)	154.420,00	616(1)	2.034(4)	154.420,00
6	1.597(*)	4.670(4)	154.420,00	215(*)	2.143(4)	154.420,00
7	2.355(1)	4.452(4)	154.420,00	1.440(2)	2.239(4)	154.420,00
8	3.862(2)	4.751(4)	154.420,00	1.202(2)	2.348(4)	154.420,00
9	604(*)	4.772(4)	154.420,00	410(*)	2.508(4)	154.420,00
10	860(*)	5.456(4)	154.420,00	1.553(3)	2.227(4)	154.420,00
11	4.411(2)	4.789(4)	154.420,00	1.992(3)	2.394(4)	154.420,00
12	705(*)	4.975(4)	154.420,00	1.442(2)	2.462(4)	154.420,00
13	3.379(2)	4.605(4)	154.420,00	785(3)	2.056(4)	154.420,00
14	2.544(1)	5.246(4)	154.420,00	213(*)	2.643(4)	154.420,00
15	153(*)	5.287(4)	154.420,00	548(*)	2.715(4)	154.420,00
16	1.896(*)	4.717(4)	154.420,00	128(*)	2.713(4)	154.420,00
17	365(*)	4.683(4)	154.420,00	1.981(3)	2.697(4)	154.420,00
18	1.709(*)	5.028(4)	154.420,00	759(*)	2.804(4)	154.420,00
19	1.658(*)	5.718(4)	154.420,00	2.441(3)	2.989(4)	154.420,00
20	510(*)	5.171(4)	154.420,00	2.049(3)	2.826(4)	154.420,00

$*$ - resultado obtido durante o primeiro processo de otimização (nível I)

OBS: - Configurações geradas de maneira aleatória, com doze posições de memória, obtendo-se um valor médio de 5.007 PL's
- Configurações geradas com metodologia Garver modificada, com oito posições, obtendo-se um valor médio de 2.458 PL's.
- Ambas, conforme Figura 5.10

6.7 TESTES EM SISTEMAS DE GRANDE PORTE

Para realizar estes testes foi utilizado o sistema Norte-Nordeste brasileiro, o qual é um sistema de grande porte com muitas barras isoladas da configuração inicial. Na Figura 3.12 é apresentado este sistema.

6.7.1 Sistema Norte - Nordeste Brasileiro - Plano 2008

Este sistema está constituído por 89 barras e 183 linhas candidatas e uma demanda total de 29.754 MW e cujos dados são apresentados no Apêndice A.

Para estas análises foi empregado o algoritmo híbrido formado por **TS**, **SA** e **AG**, em suas diferentes versões. Nestas, o processo foi inicializado com configurações geradas com a metodologia de Garver modificada. Como se mencionara anteriormente, quando o processo é inicializado com configurações geradas de maneira aleatória, pode-se ficar preso em um ótimo local de má qualidade e o esforço computacional se torna proibitivo. Isto devido ao grande número de nós isolados que contém este sistema, já que a formação dos caminhos ótimos que conectam estes nós é uma tarefa complexa. Foi observado então que os melhores resultados foram obtidos com populações iniciais geradas com a metodologia de Gaver.

A melhor configuração encontrada com o algoritmo híbrido tem um investimento de US$ 2.585.383.000,00 e as seguintes linhas adicionadas:

$n_{01-02} = 1$, $n_{02-87} = 1$, $n_{04-05} = 4$, $n_{04-81} = 2$, $n_{05-56} = 1$, $n_{05-58} = 3$, $n_{06-37} = 1$, $n_{12-13} = 1$, $n_{13-15} = 5$, $n_{14-59} = 1$, $n_{15-16} = 4$, $n_{15-46} = 1$, $n_{16-44} = 6$, $n_{16-61} = 1$, $n_{18-50} = 11$, $n_{18-74} = 6$, $n_{21-57} = 2$, $n_{22-37} = 1$, $n_{24-43} = 2$, $n_{25-55} = 4$, $n_{26-54} = 1$, $n_{30-31} = 1$, $n_{30-63} = 2$, $n_{35-51} = 1$, $n_{36-46} = 1$, $n_{39-42} = 1$, $n_{39-86} = 3$, $n_{40-45} = 1$, $n_{40-46} = 2$, $n_{41-64} = 2$, $n_{42-44} = 2$, $n_{43-55} = 3$, $n_{43-58} = 3$, $n_{48-49} = 1$, $n_{49-50} = 4$, $n_{52-59} = 1$, $n_{53-86} = 1$, $n_{54-55} = 1$, $n_{54-63} = 1$, $n_{56-57} = 1$, $n_{61-64} = 1$, $n_{61-85} = 2$, $n_{67-69} = 2$, $n_{67-71} = 3$, $n_{68-83} = 1$, $n_{69-87} = 1$, $n_{71-72} = 1$, $n_{72-73} = 2$, $n_{73-83} = 1$, $n_{73-74} = 2$, $n_{81-83} = 1$.

Além desta foram obtidas soluções de grande qualidade e que apresentam pequenos cortes de carga; assim a melhor solução encontrada apresenta um investimento de US$ 2.573.941.000,00 e um corte de carga de 4 MW, a qual tem as seguintes linhas adicionadas.

$n_{01-02} = 1$, $n_{02-87} = 1$, $n_{04-05} = 4$, $n_{04-81} = 2$, $n_{05-56} = 1$, $n_{05-58} = 3$, $n_{06-37} = 1$, $n_{12-15} = 1$, $n_{13-15} = 4$, $n_{14-59} = 1$, $n_{15-16} = 4$, $n_{15-46} = 1$, $n_{16-44} = 6$, $n_{16-61} = 1$, $n_{18-50} = 11$, $n_{18-74} = 6$, $n_{21-57} = 2$, $n_{22-37} = 1$, $n_{24-43} = 2$, $n_{25-55} = 4$, $n_{30-31} = 1$,

$n_{30-63} = 2$, $n_{35-51} = 1$, $n_{36-46} = 1$, $n_{39-42} = 1$, $n_{39-86} = 3$, $n_{40-45} = 1$, $n_{40-46} = 2$, $n_{41-64} = 2$, $n_{42-44} = 2$, $n_{43-55} = 3$, $n_{43-58} = 3$, $n_{48-49} = 1$, $n_{49-50} = 4$, $n_{52-59} = 1$, $n_{53-86} = 1$, $n_{54-55} = 1$, $n_{54-63} = 1$, $n_{56-57} = 1$, $n_{61-64} = 1$, $n_{61-85} = 2$, $n_{67-69} = 2$, $n_{67-71} = 3$, $n_{68-83} = 1$, $n_{69-87} = 1$, $n_{71-72} = 1$, $n_{72-73} = 2$, $n_{73-83} = 1$, $n_{73-74} = 2$, $n_{81-83} = 1$.

Na Tabela 6.9 apresentam-se estatísticas com soluções obtidas da análise do sistema Norte-Nordeste, para diferentes casos. Estes casos foram analisados com diferentes números de configurações iniciais. Também foram variados alguns parâmetros tais como: o número de configurações elite armazenadas, valor do corte de carga permitido, duração tabu e tamanho da lista, entre outros.

Tabela 6.9: Resultados do sistema Norte-Nordeste brasileiro, algoritmo híbrido (*TS - SA - AG*)

Híbrido - Sistema Norte-Nordeste - plano 2008				
Testes do algoritmo híbrido formado por **TS - SA - AG**				
	Inicialização Garver modificado			
Caso Nº	Valor da f.o.(10^3US$)	Custo (10^3US$)	Corte de carga (MW)	Nº de PL's
1	2.577.941,00	2.573.941,00	4,0	174.350
2	2.585.383,00	2.585.383,00	0,0	178.284
3	2.596.416,00	2.596.416,00	0,0	148.250
4	2.596.861,00	2.552.061,00	44,8	103.075
5	2.599.181,00	2.592.181,00	7,0	176.066
6	2.601.591,00	2.600.591,00	1,0	110.114
7	2.603.102,00	2.603.102,00	0,0	152.936
8	2.603.268,00	2.600.268,00	3,0	123.317
9	2.604.536,00	2.595.536,00	9,0	123.317
10	2.605.416,00	2.596.416,00	9,0	139.781
11	2.607.446,00	2.590.046,00	17,4	85.450
12	2.607.749,00	2.607.749,00	0,0	172.142
13	2.607.842,00	2.606.842,00	1,0	81.965
14	2.610.842,00	2.607.842,00	3,0	117.280
15	2.613.556,00	2.606.556,00	7,0	191.143
16	2.622.557,00	2.617.557,00	5,0	84.300
17	2.624.376,00	2.624.376,00	0,0	83.371
18	2.624.726,00	2.624.726,00	0,0	86.670
19	2.626.376,00	2.624.376,00	2,0	83.371
20	2.627.826,00	2.611.026,00	16,8	105.477
21	2.627.736,00	2.596.736,00	31,0	81.965
22	2.646.626,00	2.646.626,00	0,0	65.740

6.8 CONCLUSÕES

Três métodos diferentes de otimização não convexa foram estudados na solução do problema do planejamento da transmissão a longo prazo, ou seja, "simulated annealing" (**SA**), algoritmos genéticos (**AG**) e busca tabu (**TS**). Baseados no **TS** foram incorporadas eficientes características dos algoritmos **SA** e **AG** para formar o algoritmo híbrido.

Os diferentes métodos de otimização são comparados, observando velocidade computacional e qualidade da resposta final. O método híbrido foi implementado em várias versões e comparados seus desempenhos, ou seja, inicialmente ao método **TS** foi incorporado características do **SA**, depois, ao **TS** foi incorporado características dos **AG** e finalmente ao **TS** incorporou-se características do **SA** e **AG**.

O algoritmo híbrido é formado por dois procedimentos básicos, diversificação e intensificação; assim, para se obter um bom desempenho, foram selecionadas as melhores características dos três métodos no desempenho destes procedimentos. Nos **AG** o "crossover" realiza um procedimento de diversificação muito efetivo; o **TS** realiza um procedimento efetivo de intensificação; **SA** em suas primeiras etapas e à temperaturas altas, basicamente é um procedimento de diversificação; à temperaturas baixas, o procedimento torna-se de intensificação.

No algoritmo híbrido, observa-se que o melhor desempenho do **SA** está na etapa de intensificação. Assim, quando no procedimento de intensificação uma proposta apresenta pior função objetivo, a mesma pode ser aceita segundo um critério de probabilidade de **SA**. Apesar de **TS** aceitar a menos pior, observa-se que esta pode ser uma melhor maneira de sair dos ótimos locais no procedimento de intensificação.

Foram feitas análises com os diferentes algoritmos implementados. Ou seja, foram analisados os sistemas Garver e Sul brasileiro, com e sem redespacho. São apresentados resultados das diferentes versões dos algoritmos formados com **TS**, **SA** e **AG**. Das análises com estes dois sistemas se deduz que a melhor versão híbrida é a formada pelos três métodos mencionados anteriormente. Assim, foi selecionado este algoritmo para analisar o sistema Norte-Nordeste o qual apresentou excelentes resultados quando comparados com os obtidos com algoritmos aproximados; sua melhor configuração apresentou um investimento de US$ 2.573.941.000,00, o qual é US$ 280 milhões inferior com relação a encontrada com os algoritmos aproximados. Quando a solução anterior é comparada com a obtida com os algoritmos **TS**, **SA** ou **AG** isoladamente, é a que apresenta menor investimento.

Os resultados obtidos na aplicação do algoritmo **híbrido** no problema do planejamento da transmissão, confirmam um melhor desempenho comparados com os obtidos com **SA** ou **AG** e de qualidade semelhante quando comparado com **TS**.

Capítulo 7

CONCLUSÕES

O problema do planejamento de sistemas de transmissão é um típico problema combinatorial onde o número de possíveis configurações a serem analisadas cresce exponencialmente com o tamanho da rede. Este é um tipo de problema de programação não linear inteira mista onde frações de linha não são permitidas. É um problema de difícil resolução devido a que o espaço de soluções está representado por uma grande quantidade de possíveis configurações candidatas e cuja enumeração é excessivamente grande, sendo praticamente infinita em sistemas de grande porte. Neste caso cada alternativa possível é analisada através de um PL, o qual torna ainda mais demorada a resolução do problema. Portanto, com os algoritmos propostos somente são analisadas uma pequena parcela das configurações candidatas.

Foram estudados os algoritmos combinatoriais **SA**, **AG**, **TS** e **híbrido**. O primeiro faz uso de conceitos da mecânica estatística e da técnica de Monte Carlo, o segundo usa técnicas de busca baseadas no mecanismo da seleção natural e da genética, o terceiro nasce do campo da inteligência artificial e o quarto é constituído usando **TS** como base e adicionando as melhores características de **SA** e **AG**. Inicialmente estas técnicas foram validadas através de comparações com métodos aproximados e técnicas de otimização baseadas em decomposição de Benders, estas últimas encontram a resposta ótima para sistemas de pequeno e médio porte.

Os quatro métodos mencionados encontram facilmente as soluções ótimas conhecidas de sistemas de pequeno e médio porte em um grande número dos casos que foram testados, sendo que os algoritmos **híbrido** e **TS** são os mais freqüentes em encontrar a solução ótima. Entretanto, para sistemas de grande porte os quatro algoritmos encontram soluções de qualidade semelhantes, sendo que a diferença principal está no esforço computacional. O **SA** precisa de um maior número de PL's, e em ordem inversa seguem: **AG**, **TS** e o **híbrido**.

Estes métodos apresentam características diferentes, sendo alguns deles altamente

aleatórios e outros altamente determinísticos. Na primeira classificação aparece **SA** que é um método no qual a configuração vizinha é escolhida com base em seleções aleatórias dos elementos candidatos; no outro extremo está **TS** no qual a própria vizinhança é definida com base nas memórias de curto e de longo prazo, selecionando a maioria dos vizinhos de maneira determinística e uma pequena parcela de forma aleatória, passando posteriormente para o melhor vizinho ou melhor configuração candidata. Com relação aos **AG**, estes são altamente aleatórios, a parte determinística deste algoritmo consiste em conservar os blocos construtivos e transferi-los para outras configurações. O híbrido é construído tomando **TS** como algoritmo base, assim o híbrido é um método basicamente determinístico.

Existem duas maneiras de determinar as configurações iniciais para serem usadas pelos algoritmos aqui analisados; a primeira é gerar as configurações de uma maneira aleatória e a segunda usando uma metodologia modificada de Garver. Com relação a isto, pode-se concluir que em sistemas pequenos e de médio porte, tanto a estratégia com populações iniciais geradas aleatoriamente assim como com populações geradas com a metodologia de Garver modificada, chega-se à resposta ótima com uma taxa de freqüência relativamente alta. Quando em sistemas de pequeno e médio porte se emprega a inicialização baseada na metodologia de Garver modificada, consegue-se ganhos na diminuição do número de iterações com relação a inicialização aleatória.

Para sistemas de grande porte, como é o caso da rede Norte - Nordeste brasileira, não é adequado usar populações iniciais geradas aleatoriamente devido ao grande número de nós isolados contido neste sistema; além do grande número de regiões no espaço de soluções que as contém. A formação dos caminhos ótimos que conectam estes nós é uma tarefa muito complicada, assim o processo pode ficar preso em regiões que contém soluções de pobre qualidade e poderia permanecer por muitas iterações nestas regiões antes de sair para regiões de configurações de melhor qualidade, o que torna o problema de difícil resolução.

Quando em sistemas de grande porte se utiliza a metodologia de Garver modificada para gerar as configurações iniciais, observa-se que as configurações geradas conectam adequadamente as barras ilhadas do sistema, isto é, os novos caminhos quando são atrativos aparecem nas configurações geradas pelo algoritmo de Garver modificado e, logicamente, corretamente ligados, especialmente na alocação adequada das grandes linhas de transmissão. Esta metodologia de Garver leva em conta somente a primeira lei de Kirchhoff; assim as configurações obtidas não adicionam linhas pequenas, permitindo que esta parte do trabalho de alocação seja adequadamente executada pelos algoritmos combinatoriais.

Nos algoritmos desenvolvidos foram usados procedimentos eficientes para a entrada e saída de caminhos completos que ligam as partes ilhadas do sistema com a configuração base, evitando que durante o processo sejam adicionadas linhas supérfluas e sem capacidade de transporte de energia elétrica (espécie de linha ligando duas barras isoladas da configuração base). Além disso foi usado o conceito de bloco construtivo na parte da rede

que inicialmente está isolada da configuração base, assim um conjunto de linhas formam um bloco construtivo os quais são construídos, modificados e/ou eliminados durante o processo e também são integralmente transferidos de uma configuração a outra.

Na presente pesquisa foram analisados três algoritmos combinatoriais. Entretanto, além dos algoritmos aqui estudados existem outros tais como o Grasp e as Redes Neurais. Todos estes algoritmos, junto com as heurísticas construtivas, podem formar uma equipe, para que trabalhando cooperativamente possam resolver o problema de planejamento da transmissão. Esta metodologia é denominada Times Assíncronos, e consiste na utilização de diferentes algoritmos como agentes de um time, trabalhando de maneira autônoma e distribuída.

Bibliografia

[1] *R. Fischl*: "Optimal System Expansion Planning: A Critical Overview", ERDA & EPRI Conf., Henniker, 1975.

[2] *M. Srinivas, L.M. Patnaik*: "Genetic Algorithms", IEEE Computer, 1994.

[3] *D.E. Goldberg*: "Genetics Algorithms in Search, Optimization and Machine Learning", Addison Wesley, Reading, Mass., 1989.

[4] *S.A. Kazarlis, A.G. Bakirtzis, V. Petridis*: "A Genetic Algorithm Solution to the Unit Commitment Problem", IEEE Transactions on Power Systems, Vol. 11, No. 1, february 1996.

[5] *C.C. Kuo, H.C. Chang*: "Applying a Refined Genetic Algorithm to Network Reconfiguration for Loss Reduction", IEE of Japan Power & Energy, 1994.

[6] *R. Romero, R.A. Gallego, A. Monticelli*: "Transmission System Expansion Planning by Simulated Annealing", IEEE Transactions on Power Systems, Vol. 11, No. 1, pp. 364 - 369, February 1996.

[7] *R.A. Gallego, A.B. Alves, A. Monticelli, R. Romero*: "Parallel Simulated Annealing Applied to Long Term Transmision Network Expansion Planning", IEEE Transactions on Power Systems, Vol. 12, No. 1, pp. 181 - 188, February 1997.

[8] *Gallego R.A., Monticelli A. Romero R.*: "Comparative Studies on Non-Convex Optimization Methods for Transmission Network Expansion Planning", IEEE Transactions on Power Systems, Vol. 13, 1998.

[9] *L.L. Garver*: "Transmission Network Estimation Using Linear Programming", IEEE Trans. Power App. Syst., vol. PAS-89, pp. 1688-1697, Sept/Oct 1970.

[10] *M.V.F. Pereira, L.M.V.G. Pinto, S.H.F. Cunha, G.C. Oliveira*: "A Decomposition Approach to Automated Generation/Transmission Expansion Planning", IEEE Trans. Power App. Systems, Vol. PAS-104, No. 11, November 1985.

[11] *R. Romero and A. Monticelli*: "A Zero-One Implicit Enumeration Method for Optimizing Investments in Transmission Expansion Planning", IEEE Transactions on Power Systems, Vol. 9, No. 3, August 1994.

[12] *A. Monticelli, A. Santos Jr., M.V.F. Pereira, S. Cunha, J.G. Praça, B. Park*: "Interactive Transmission Network Planning Using a Least-Effort Criterion", IEEE Transactions PAS, Vol. 101, No. 10, pp 3919-3925, October 1982.

[13] *H. Rudnick, R, Palma, E. Cura and C. Silva*: "Economically Adapted Transmission systems in Open Access Schemes – Application of Genetic Algorithms", IEEE Transactions on Power Systems, Vol. 11, No. 3, pp. 1427 - 1440, august 1996.

[14] *K.P. Wong, Y.W. Wong*: "Combined Genetic Algorithms/Simulated Annealing/Fuzzy Set Approach to Short-Term Generation Scheduling with Take-or-Pay Fuel Contract", IEEE Transactions on Power Systems, Vol. 11, No. 1, february 1996.

[15] *Aarts E.H.L., De Bont F.M.J., Habers E.H.A. and Van Laarhoven P.J.M.:* "Parallel Implementations of the Statistical Cooling Algorithm", Integration, the VLSI Journal 4, pp. 209-238, 1986.

[16] *Mori H., Takeda K.*: "Parallel Simulated Annealing for Power System Decomposition", IEEE Transaction on Power Systems, Vol. 9, No. 2, May 1994.

[17] *Johnson D.S., Aragon C.R., McGeoch L.A., Schevon C.*: "Optimization by Simulated Annealing: An Experimental Evaluation; Part I, Graph Partitioning", Operations Research, Vol. 37, No. 6, Nov.-Dec. 1989.

[18] *D.S. Johnson, C.R. Aragon, L.A. McGeoch, C. Schevon*: "Optimization by Simulated Annealing: An Experimental Evaluation; Part II, Graph Coloring and Number Partitioning", Operations Research, Vol. 39, No. 3, May-June 1991.

[19] *Aarts E., Korst J.*: "Simulated Annealing and Boltzmann Machines", John Wiley & Sons, 1989.

[20] Van Laarhoven P.J.M., Aarts E.H."Simulated Annealing: Theory and Applications", D. Reidel Publishing Company, Holand, 1987.

[21] *Fischl R.*: "Optimal System Expansion Planning: A Critical Overview", ERDA & EPRI Conf., Henniker, 1975.

[22] Ilic'-Spong M. et al. "Secondary Voltage Control Systems Pilot Point Information", IEEE Trans. on Power Systems, Vol. 3, No. 2, pp. 660-668, May 1988.

[23] Hsiao Y. T., Chiang H.D., Liu C.C., Chen Y.L. "A Computer Package for Optimal Multi-objective VAR Planning in Large Scale Power Systems", IEEE Transactions on Power Systems, Vol. 9, No.2, May 1994.

[24] Drexl A. "A Simulated Annealing Approach to the Multiconstraint Zero-One Knapsack Problem", Computing 40, pp. 1-8, 1988.

[25] Kirkpatrick S., Gelatt Jr. C.D., Vecchi M.P. "Optimization by simulated annealing", Science 220, pp. 671-680, 1983.

[26] *R. Villanasa, L.L. Garver, S.J. Salon:* "Transmission Network Planning Using Linear Programming", IEEE Trans. Power App. Systems, vol. PAS-104, No. 2, February 1985.

[27] R. Romero and A. Monticelli, "A Hierarchical Decomposition Approach for Transmission Network Expansion Planning", IEEE Trans. on Power Systems, Vol.9, No.1, pp. 373-380, Feb. 1994.

[28] Cerny V. "Thermodynamical Approach to the Traveling Salesman Problem: An Efficient Simulation Algorithm", Journal of Optimization Theory and Applications, Vol. 45, No. 1, 1985.

[29] Connolly D.T. "An improved annealing scheme for the QAP", European Journal of Operation Research 46, pp. 93-100, 1990.

[30] Aardal K., Larson T. "A Benders decomposition based heuristic for the hierarchical production planning problem", European Journal of Operation Research 45, pp. 4-14,1990.

[31] Rutenbar R.A. "Simulated Annealing Algorithms: An Overview", IEEE Circuits and Devices Magazine, pp. 19-26, 1989.

[32] Hajek B. "Cooling schedules for optimal annealing", Mathematics of Operations Research 13, pp. 311-329, 1988.

[33] Mitra D., Romeo F., Sangiovanni-Vincentelli A.L. "Convergence of finite-time behavior of simulated annealing", Advances in Applied Probability 18, pp. 747-771, 1986.

[34] Granville S., Pereira M.V.F.: "Analysis of the Linearized Power Flow Model in Benders Decomposition", EPRI-Report RP 2473-6, Stanford University, February 1985.

[35] Pereira M.V.F., Pinto L.M.V.G.: "Aplication of Sensitivity Analysis of Load Supplying Capability to Interactive Transmission Expansion Planning"; IEEE Trans. Power App. Systems, vol. PAS-104, No. 2, february, 1985.

[36] Said A.: "Um Método de Síntese Estática de Redes de Transmissão para Auxilio ao Planejamento a Longo Prazo", Tese de Mestrado, Unicamp, julho de 1988.

[37] Chiang H.D., Wang J.C., Cockings O.R. and Shin H.D."Optimal Capacitor Placements in Distribution Systems, Parte 1: A New Formulation and the Overall Problem", IEEE Trans. on Power Delivery, Vol. 5, No. 2, April 1990, pp. 634-642.

[38] Chiang H.D., Wang J.C., Cockings O.R. and Shin H.D."Optimal Capacitor Placements in Distribution Systems, Parte 1: Solution Algorithms and Numerical Results", IEEE Trans. on Power Delivery, Vol. 5, No. 2, April 1990, pp. 643-649.

[39] Chiang H.D., Jean-Jumeu R. "Optimal Network Reconfigurations in Distribution Systems, Parte 1: A New Formulation and a Solution Methodology", IEEE Trans. on Power Delivery, Vol. 5, No. 4, October 1990, pp. 1902-1909.

[40] Chiang H.D., Jean-Jumeu R. "Optimal Network Reconfigurations in Distribution Systems, Parte 2: Solution Algorithms and Numerical Results", IEEE Trans. on Power Delivery, Vol. 5, No. 3, July 1990, pp. 1568-1574.

[41] Siao Y.T., Lin C.C., Chiang H.D. and Chen Y.L. "A New Approach for Optimal VAR Sources Planning in Large Scale Electric Power Systems", IEEE Trans. on Power Systems, Vol. 8, No. 3, August 1993, pp. 988-996.

[42] Eglese R.W. "Simulated Annealing: A Tool for Operational Research", European Journal of Operation Research 46, pp. 271-281,1990.

[43] Lundy M., Mees A. "Convergence of an annealing algorithm", Mathematical Programming 34, pp. 111-124, 1986.

[44] Chams M., Hertz A., de Werra D. "Some experiments with simulated annealing for coloring graphs", European Journal of Operation Research 32, Vol. 39, pp. 260-266,1987.

[45] Metropolis N., Rosenbluth A., Rosenbluth M., Teller A., Teller E. "Equation os state calculations by fast computing machines", Journal of Chemical Physics 21, pp. 1087-1092, 1953.

[46] Tovey C.A. "Simulated Annealing", American Journal of Mathematical and Management Sciences 8, pp. 389-407, 1988.

[47] Battiti, R., Tecchiolli, G. "Parallel biased search for combinatorial optmization: genetic algorithms and tabu", *Microprocessors and Microsystems*, n.16, p.351-367, 1992.

[48] Battiti, R., Tecchiolli, G. "The reactive tabu search", *ORSA Journal on Computing*, n.6, p.126-140, 1994.

[49] Eck, B., Pinedo, M. "Good solutions to job scheduling problems via tabu search". In: Orsa/Tims Meeting, May 10 1989, Vancouver. *Proceedings...* Vancouver: 1989

[50] Glass, C.A., Potts, C.N., SHADE, P. "Unrelated parallel machine scheduling using local search", *Mathematical and Computer Modelling*, n.20, p.41, 1994.

[51] Glover, F. "Tabu Search" - Part I. *ORSA Journal Computing*, v.1, n.3, Summer 1989.

[52] Glover, F. "Tabu Search" - Part II. *ORSA Journal Computing*, v.2, n.1, Winter 1990.

[53] Husbscher, R., Glover, F. "Applying tabu search with influential diversification to multiprocessor scheduling", *Computers & Operations Research*, n.21, p.867, 1994.

[54] Laguna, M. "Tabu search tutorial". In: Escuela DE Verano Latino-Americana de Investigación Operativa, 2, 1995. *Tutorial.* 1995, 112p.

[55] Glover, F. "Tabu Search Fundamental and Uses", Graduate School of Business, University of Colorado, April 1995.

[56] Lokketangen, A., Glover, F."Probabilistic move selection in tabu search for 0-1 mixed integer programming". In: Osman, I.H., Kelly, J.P., *Metaheuristics: theory and applications.* Boston: Kluwers Academic Publishers, 1995.

[57] Osman, I.H. "Metastrategy simulated annealing and tabu search algorithms for the vehicle routing problems", *Annals of Operations Research*, n.41, p.421-451, 1993.

[58] Osman, I.H. "Heuristics for the generalized assignment problem: simulated annealing and tabu search approaches", *OR Spectrum*, n.17, p.211-225, 1995.

[59] Osman, I.H., Christofides, N. "Capacitated clustering problems by hybrid simulated annealing and tabu search", *International Transactions in Operational Research*, n.1, p.317-336, 1994.

[60] Soriano, P., Gendreau, M. "Diversification strategies in tabu search algorithm for the maximum clique problem", *Annals of Operations Research*, n.21, p.885, 1994.

[61] Woodruff, D.L. & Zemel, Z. "Hashing vectors for tabu search", *Annals of Operations Research*, n.41, p.123-138, 1993.

[62] Pereira M.V.F., Pinto L.M.V.G., Oliveira G.C., Cunha S.H.F., "Composite Generation-Transmission Expansion Planning", EPRI-Research Project 2473-9, S-tanford University, 1987.

[63] Romero R. "Um Método de Decomposição para Planejamento a longo Prazo de Sistemas de Transmissão",Ph.D, Tese, Universidade Estadual de Campinas Brasil, 1993.

[64] Bayona G., Pérez Arriaga J.I., CHOPIN, "A Heuristic Model For Long Term Transmission Expansion Planning", presented at the IEEE/PES Winter Meeting, New York, NY, Jan/Fev, 1994.

[65] Oliveira G.C., Costa A.P., Binato S., "Large Scale Transmission Network planning Using Optimization and Heuristic Techniques", presented at the IEEE/PES Winter Meeting, New York, NY, Jan/Fev, 1995.

[66] *Hsiao Y. T., Chiang H.D., Liu C.C., Chen Y.L.*: "A Computer Package for Optimal Multi-objective VAR Planning in Large Scale Power Systems", IEEE Transactions on Power Systems, Vol. 9, No. 2, May 1994.

[67] *Díaz, A., Glover, F. , Ghaziri, H.M., González, J.L. , Laguna, M. , Moscato, P. , Tseng, F.T.* : "Optimización Heurística y Redes Neuronales", Editorial Paraninifo, S.A., 1996.

[68] *R.A. Gallego, A. Monticelli, R. Romero*: " Transmission System Expansion Planning by Extended Genetic Algorithm", submitted for publication.

[69] *R.A. Gallego, A. Monticelli, R. Romero*: " Tabu Search Algorithm Applied to Transmission Network Expansion Planning ", submitted for publication.

Apêndice A

Dados dos Sistemas Testados

- **Sistema de 06 Barras de Garver**
- **Sistema Sul Brasileiro de 46 Barras**
- **Sistema Norte-Nordeste Brasileiro de 87 Barras**

A.1 Sistema de 06 Barras de Garver

Níveis de Geração e Carga

Barra	Cap. de Geração	Ger. Atual	Carga (MW)
1	150	50	80
2	0	0	240
3	360	165	40
4	0	0	160
5	0	0	240
6	600	545	0

Características das Linhas

Nº	Linha	Linhas existentes	Reatância (pu)	Capacidade (MW)	Custo 10^3 dol.
1	1-2	1	0.40	100	40
2	1-3	0	0.38	100	38
3	1-4	1	0.60	80	60
4	1-5	1	0.20	100	20
5	1-6	0	0.68	70	68
6	2-3	1	0.20	100	20
7	2-4	1	0.40	100	40
8	2-5	0	0.31	100	31
9	2-6	0	0.30	100	30
10	3-4	0	0.59	82	59
11	3-5	1	0.20	100	20
12	3-6	0	0.48	100	48
13	4-5	0	0.63	75	63
14	4-6	0	0.30	100	30
15	5-6	0	0.61	78	61

A.2 Sistema Sul Brasileiro de 46 Barras

Níveis de Geração e Carga

Barra	Cap. de Geração	Ger. Atual	Carga (MW)
1	0.0	0.0	0.0
2	0.0	0.0	443.1
3	0.0	0.0	0.0
4	0.0	0.0	300.7
5	0.0	0.0	238.0
6	0.0	0.0	0.0
7	0.0	0.0	0.0
8	0.0	0.0	72.2
9	0.0	0.0	0.0
10	0.0	0.0	0.0
11	0.0	0.0	0.0
12	0.0	0.0	511.9
13	0.0	0.0	185.8
14	1257.0	944.0	0.0
15	0.0	0.0	0.0
16	2000.0	1366.0	0.0
17	1050.0	1000.0	0.0
18	0.0	0.0	0.0
19	1670.0	773.0	0.0
20	0.0	0.0	1091.2
21	0.0	0.0	0.0
22	0.0	0.0	81.9
23	0.0	0.0	458.1
24	0.0	0.0	478.2
25	0.0	0.0	0.0
26	0.0	0.0	231.9
27	220.0	54.0	0.0
28	800.0	730.0	0.0

Níveis de Geração e Carga - Continuação

Barra	Cap. de Geração	Ger. Atual	Carga (MW)
29	0.0	0.0	0.0
30	0.0	0.0	0.0
31	700.0	310.0	0.0
32	500.0	450.0	0.0
33	0.0	0.0	229.1
34	748.0	221.0	0.0
35	0.0	0.0	216.0
36	0.0	0.0	90.1
37	300.0	212.0	0.0
38	0.0	0.0	216.0
39	600.0	221.0	0.0
40	0.0	0.0	262.1
41	0.0	0.0	0.0
42	0.0	0.0	1607.9
43	0.0	0.0	0.0
44	0.0	0.0	79.1
45	0.0	0.0	86.7
46	700.0	599.0	0.0

Características das Linhas

Nº	Linha	Linhas existentes	Reatância (pu)	Capacidade (MW)	Custo 10^3 dol.
1	01-07	1	0.0616	270	4349
2	01-02	2	0.1065	270	7076
3	04-09	1	0.0924	270	6217
4	05-09	1	0.1173	270	7732
5	05-08	1	0.1132	270	7480
6	07-08	1	0.1023	270	6823
7	04-05	2	0.0566	270	4046
8	02-05	2	0.0324	270	2581
9	08-13	1	0.1348	240	8793
10	09-14	2	0.1756	220	11267
11	12-14	2	0.0740	270	5106
12	14-18	2	0.1514	240	9803
13	13-18	1	0.1805	220	11570
14	13-20	1	0.1073	270	7126
15	18-20	1	0.1997	200	12732
16	19-21	1	0.0278	1500	32632
17	16-17	1	0.0078	2000	10505
18	17-19	1	0.0061	2000	8715
19	14-26	1	0.1614	220	10409
20	14-22	1	0.0840	270	5712
21	22-26	1	0.0790	270	5409
22	20-23	2	0.0932	270	6268
23	23-24	2	0.0774	270	5308
24	26-27	2	0.0832	270	5662
25	24-34	1	0.1647	220	10611
26	24-33	1	0.1448	240	9399

Características das Linhas - Continuação

Nº	Linha	Linhas existentes	Reatância (pu)	Capacidade (MW)	Custo 10^3 dol.
27	33-34	1	0.1265	270	8288
28	27-36	1	0.0915	270	6167
29	27-38	2	0.2080	200	13237
30	36-37	1	0.1057	270	7025
31	34-35	2	0.0491	270	3591
32	35-38	1	0.1980	200	12631
33	37-39	1	0.0283	270	2329
34	37-40	1	0.1281	270	8389
35	37-42	1	0.2105	200	13388
36	39-42	3	0.2030	200	12934
37	40-42	1	0.0932	270	6268
38	38-42	3	0.0907	270	6116
39	32-43	1	0.0309	1400	35957
40	42-44	1	0.1206	270	7934
41	44-45	1	0.1864	200	11924
42	19-32	1	0.0195	1800	23423
43	46-19	1	0.0222	1800	26365
44	46-16	1	0.0203	1800	24319
45	18-19	1	0.0125	600	8178
46	20-21	1	0.0125	600	8178
47	42-43	1	0.0125	600	8178
48	02-04	0	0.0882	270	5965
49	14-15	0	0.0374	270	2884
50	46-10	0	0.0081	2000	10889
51	04-11	0	0.2246	240	14247
52	05-11	0	0.0915	270	6167

Características das Linhas - Continuação

Nº	Linha	Linhas existentes	Reatância (pu)	Capacidade (MW)	Custo 10^3 dol.
53	46-06	0	0.0128	2000	16005
54	46-03	0	0.0203	1800	24319
55	16-28	0	0.0222	1800	26365
56	16-32	0	0.0311	1400	36213
57	17-32	0	0.0232	1700	27516
58	19-25	0	0.0325	1400	37748
59	21-25	0	0.0174	2000	21121
60	25-32	0	0.0319	1400	37109
61	31-32	0	0.0046	2000	7052
62	28-31	0	0.0053	2000	7819
63	28-30	0	0.0058	2000	8331
64	27-29	0	0.0998	270	6672
65	26-29	0	0.0541	270	3894
66	28-41	0	0.0339	1300	39283
67	28-43	0	0.0406	1200	46701
68	31-41	0	0.0278	1500	32632
69	32-41	0	0.0309	1400	35957
70	41-43	0	0.0139	2000	17284
71	40-45	0	0.2205	180	13994
72	15-16	0	0.0125	600	8178
73	46-11	0	0.0125	600	8178
74	24-25	0	0.0125	600	8178
75	29-30	0	0.0125	600	8178
76	40-41	0	0.0125	600	8178
77	02-03	0	0.0125	600	8178
78	05-06	0	0.0125	600	8178
79	09-10	0	0.0125	600	8178

A.3 Sistema Nordeste Brasileiro de 87 Barras

Níveis de Geração e Carga

Barra	Geração em 2002 (MW)	Carga em 2002 (MW)	Geração em 2008 (MW)	Carga em 2008 (MW)
1	0	1857	0	2747
2	4048	0	4550	0
3	0	0	0	0
4	517	0	6422	0
5	0	0	0	0
6	0	0	0	0
7	0	31	0	31
8	403	0	82	0
9	465	0	465	0
10	538	0	538	0
11	2200	0	2260	0
12	2257	0	4312	0
13	4510	0	5900	0
14	542	0	542	0
15	0	0	0	0
16	0	0	0	0
17	0	0	0	0
18	0	0	0	0
19	0	86	0	125
20	0	125	0	181
21	0	722	0	1044
22	0	291	0	446
23	0	58	0	84
24	0	159	0	230
25	0	1502	0	2273
26	0	47	0	68
27	0	378	0	546
28	0	189	0	273
29	0	47	0	68

Níveis de Geração e Carga - Continuação

Barra	Geração em 2002 (MW)	Carga em 2002 (MW)	Geração em 2008 (MW)	Carga em 2008 (MW)
30	0	189	0	273
31	0	110	0	225
32	0	0	0	0
33	0	0	0	0
34	0	28	0	107
35	1635	0	1531	0
36	0	225	0	325
37	169	0	114	0
38	0	0	0	0
39	0	186	0	269
40	0	1201	0	1738
41	0	520	0	752
42	0	341	0	494
43	0	0	0	0
44	0	4022	0	5819
45	0	0	0	0
46	0	205	0	297
47	0	0	0	0
48	0	347	0	432
49	0	777	0	1124
50	0	5189	0	7628
51	0	290	0	420
52	0	707	0	1024
53	0	0	0	0
54	0	0	0	0
55	0	0	0	0
56	0	0	0	0
57	0	0	0	0
58	0	0	0	0

Níveis de Geração e Carga - Continuação

Barra	Geração em 2002 (MW)	Carga em 2002 (MW)	Geração em 2008 (MW)	Carga em 2008 (MW)
59	0	0	0	0
60	0	0	0	0
61	0	0	0	0
62	0	0	0	0
63	0	0	0	0
64	0	0	0	0
65	0	0	0	0
66	0	0	0	0
67	1242	0	1242	0
68	888	0	888	0
69	902	0	902	0
70	0	0	0	0
71	0	0	0	0
72	0	0	0	0
73	0	0	0	0
74	0	0	0	0
75	0	0	0	0
76	0	0	0	0
77	0	0	0	0
78	0	0	0	0
79	0	0	0	0
80	0	0	0	0
81	0	0	0	0
82	0	0	0	0
83	0	0	0	0
84	0	0	0	0
85	0	487	0	705
86	0	0	0	0
87	0	0	0	0

Características das Linhas - Nordeste Brasileiro

Nº	Linha	Linhas existentes	Reatância (pu)	Capacidade (MW)	Custo 10^3 dol.
1	01-02	2	0.0374	1000	44056
2	02-04	0	0.0406	1000	48880
3	02-60	0	0.0435	1000	52230
4	02-87	1	0.0259	1000	31192
5	03-71	0	0.0078	3200	92253
6	03-81	0	0.0049	3200	60153
7	03-83	0	0.0043	3200	53253
8	03-87	0	0.0058	1200	21232
9	04-05	1	0.0435	1000	52230
10	04-06	0	0.0487	1000	58260
11	04-32	0	0.0233	300	7510
12	04-60	0	0.0215	1000	26770
13	04-68	0	0.0070	1000	10020
14	04-69	0	0.0162	1000	20740
15	04-81	0	0.0058	1200	21232
16	04-87	1	0.0218	1000	26502
17	05-06	1	0.0241	1000	29852
18	05-38	2	0.0117	600	8926
19	05-56	0	0.0235	1000	29182
20	05-58	0	0.0220	1000	27440
21	05-60	0	0.0261	1000	32130
22	05-68	0	0.0406	1000	48880
23	05-70	0	0.0464	1000	55580
24	05-80	0	0.0058	1200	21232
25	06-07	1	0.0288	1000	35212
26	06-37	1	0.0233	300	7510
27	06-67	0	0.0464	1000	55580
28	06-68	0	0.0476	1000	56920
29	06-70	0	0.0371	1000	44860
30	06-75	0	0.0058	1200	21232
31	07-08	1	0.0234	1000	29048
32	07-53	0	0.0452	1000	54240
33	07-62	0	0.0255	1000	31460
34	08-09	1	0.0186	1000	23420
35	08-12	0	0.0394	1000	47540
36	08-17	0	0.0447	1000	53570

Características das Linhas - Continuação

Nº	Linha	Linhas existentes	Reatância (pu)	Capacidade (MW)	Custo 10^3 dol.
37	08-53	1	0.0365	1200	44190
38	08-62	0	0.0429	1000	51560
39	08-73	0	0.0058	1200	21232
40	09-10	1	0.0046	1000	7340
41	10-11	1	0.0133	1000	17390
42	11-12	1	0.0041	1200	6670
43	11-15	1	0.0297	1200	36284
44	11-17	1	0.0286	1200	35078
45	11-53	1	0.0254	1000	31326
46	12-13	1	0.0046	1200	7340
47	12-15	1	0.0256	1200	31594
48	12-17	1	0.0246	1200	30388
49	12-35	2	0.0117	600	8926
50	12-84	0	0.0058	1200	21232
51	13-14	0	0.0075	1200	10690
52	13-15	0	0.0215	1200	26770
53	13-17	0	0.0232	1200	28780
54	13-45	1	0.0290	1200	35480
55	13-59	1	0.0232	1200	28780
56	14-17	0	0.0232	1200	28780
57	14-45	0	0.0232	1200	28780
58	14-59	0	0.0157	1200	20070
59	15-16	2	0.0197	1200	24760
60	15-45	0	0.0103	1200	13906
61	15-46	1	0.0117	600	8926
62	15-53	0	0.0423	1000	50890
63	16-44	4	0.0117	600	8926
64	16-45	0	0.0220	1200	27440
65	16-61	0	0.0128	1000	16720
66	16-77	0	0.0058	1200	21232
67	17-18	2	0.0170	1200	21678
68	17-59	0	0.0170	1200	21678
69	18-50	4	0.0117	600	8926
70	18-59	1	0.0331	1200	40170
71	18-74	0	0.0058	1200	21232
72	19-20	1	0.0934	170	5885

Características das Linhas - Continuação

Nº	Linha	Linhas existentes	Reatância (pu)	Capacidade (MW)	Custo 10^3 dol.
73	19-22	1	0.1877	170	11165
74	20-21	1	0.0715	300	6960
75	20-21	1	0.1032	170	6435
76	20-38	2	0.1382	300	12840
77	20-56	0	0.0117	600	8926
78	20-66	0	0.2064	170	12210
79	21-57	0	0.0117	600	8926
80	22-23	1	0.1514	170	9130
81	22-37	2	0.2015	170	11935
82	22-58	0	0.0233	300	7510
83	23-24	1	0.1651	170	9900
84	24-25	1	0.2153	170	12705
85	24-43	0	0.0233	300	7510
86	25-26	2	0.1073	300	29636
87	25-26	3	0.1691	170	10120
88	25-55	0	0.0117	600	8926
89	26-27	2	0.1404	300	25500
90	26-27	3	0.2212	170	12760
91	26-29	1	0.1081	170	6710
92	26-54	0	0.0117	600	8926
93	27-28	3	0.0826	170	5335
94	27-35	2	0.1367	300	25000
95	27-53	1	0.0117	600	8926
96	28-35	3	0.1671	170	9900
97	29-30	1	0.0688	170	4510
98	30-31	1	0.0639	170	4235
99	30-63	0	0.0233	300	7510
100	31-34	1	0.1406	170	8525
101	32-33	0	0.1966	170	11660
102	33-67	0	0.0233	300	7510
103	34-39	2	0.1160	170	7150
104	34-39	2	0.2968	80	6335
105	34-41	2	0.0993	170	6215
106	35-46	4	0.2172	170	12705
107	35-47	2	0.1327	170	8085
108	35-51	3	0.1602	170	9625
109	36-39	2	0.1189	170	7315

Características das Linhas - Continuação

Nº	Linha	Linhas existentes	Reatância (pu)	Capacidade (MW)	Custo 10^3 dol.
110	36-46	2	0.0639	170	4235
111	39-42	1	0.0973	170	6105
112	39-86	0	0.0233	300	7510
113	40-45	1	0.0117	600	8926
114	40-46	3	0.0875	170	5500
115	41-64	0	0.0233	300	7510
116	42-44	2	0.0698	170	4565
117	42-85	2	0.0501	170	3465
118	43-55	0	0.0254	1000	31326
119	43-58	0	0.0313	1000	38160
120	44-46	3	0.1671	170	10010
121	47-48	2	0.1966	170	11660
122	48-49	1	0.0757	170	4895
123	48-50	2	0.0256	170	2090
124	48-51	2	0.2163	170	12760
125	49-50	1	0.0835	170	5335
126	51-52	2	0.0560	170	3795
127	52-59	1	0.0117	600	8926
128	53-54	0	0.0270	1000	32120
129	53-70	0	0.0371	1000	44860
130	53-76	0	0.0058	1200	21232
131	53-86	0	0.0389	1000	46870
132	54-55	0	0.0206	1000	25028
133	54-58	0	0.0510	1000	60940
134	54-63	0	0.0203	1000	25430
135	54-70	0	0.0360	1000	43520
136	54-79	0	0.0058	1200	21232
137	56-57	0	0.0122	1000	16050
138	58-78	0	0.0058	1200	21232
139	60-66	0	0.0233	300	7510
140	60-87	0	0.0377	1000	45530
141	61-64	0	0.0186	1000	23420
142	61-85	0	0.0233	300	7510
143	61-86	0	0.0139	1000	18060
144	62-67	0	0.0464	1000	55580
145	62-68	0	0.0557	1000	66300
146	62-72	0	0.0058	1200	21232

Características das Linhas - Continuação

Nº	Linha	Linhas existentes	Reatância (pu)	Capacidade (MW)	Custo 10^3 dol.
147	63-64	0	0.0290	1000	35480
148	65-66	0	0.3146	170	18260
149	65-87	0	0.0233	300	7510
150	67-68	0	0.0290	1000	35480
151	67-69	0	0.0209	1000	26100
152	67-71	0	0.0058	1200	21232
153	68-69	0	0.0139	1000	18060
154	68-83	0	0.0058	1200	21232
155	68-87	0	0.0186	1000	23240
156	69-87	0	0.0139	1000	18060
157	70-82	0	0.0058	1200	21232
158	71-72	0	0.0108	3200	125253
159	71-75	0	0.0108	3200	125253
160	71-83	0	0.0067	3200	80253
161	72-73	0	0.0100	3200	116253
162	72-83	0	0.0130	3200	149253
163	73-74	0	0.0130	3200	149253
164	73-75	0	0.0130	3200	149253
165	73-84	0	0.0092	3200	107253
166	74-84	0	0.0108	3200	125253
167	75-76	0	0.0162	3200	185253
168	75-81	0	0.0113	3200	131253
169	75-82	0	0.0086	3200	101253
170	75-83	0	0.0111	3200	128253
171	76-77	0	0.0130	3200	149253
172	76-82	0	0.0086	3200	101253
173	76-84	0	0.0059	3200	70953
174	77-79	0	0.0151	3200	173253
175	77-84	0	0.0115	3200	132753
176	78-79	0	0.0119	3200	137253
177	78-80	0	0.0051	3200	62253
178	79-82	0	0.0084	3200	98253
179	80-81	0	0.0101	3200	117753
180	80-82	0	0.0108	3200	125253
181	80-83	0	0.0094	3200	110253
182	81-83	0	0.0016	3200	23253
183	82-84	0	0.0135	3200	155253

Apêndice B

Artigos Publicados

- **R. Romero, R.A. Gallego, A. Monticelli**: "Transmission System Expansion Planning by Simulated Annealing", IEEE Transactions on Power Systems, Vol. 11, No. 1, February 1996.

- **R.A. Gallego, A.B. Alves, A. Monticelli, R. Romero**: "Parallel Simulated Annealing Applied to Long Term Transmision Network Expansion Planning", IEEE Transactions on Power Systems, Vol. 12, No. 1, pp. 181 - 188, February 1997.

- **R.A. Gallego, A. Monticelli, R. Romero**: "Comparative Studies on Non-Convex Optimization Methods for Transmission Network Expansion Planning", IEEE Transactions on Power Systems, Vol. 13, 1998.

- **R.A. Gallego, A. Monticelli, R. Romero**: " Transmission System Expansion Planning by Extended Genetic Algorithm", submitted for publication.

Artigo 1

- **R. Romero, R.A. Gallego, A. Monticelli**: "Transmission System Expansion Planning by Simulated Annealing", IEEE Transactions on Power Systems, Vol. 11, No. 1, February 1996.

TRANSMISSION SYSTEM EXPANSION PLANNING BY SIMULATED ANNEALING

R. Romero, R.A. Gallego and A. Monticelli
UNICAMP-Campinas-Brazil

Abstract: This paper presents a simulated annealing approach to the long term transmission expansion planning problem which is a hard, large scale combinatorial problem. The proposed approach has been compared with a more conventional optimization technique based on mathematical decomposition with a zero-one implicit enumeration procedure. Tests have been performed on three different systems. Two smaller systems for which optimal solutions are known have been used to tune the main parameters of the simulated annealing process. The simulated annealing method has then been applied to a larger example system for which no optimal solutions are known: as a result an entire family of interesting solutions have been obtained with costs about 7% less than the best solutions known for that particular example system.

Keywords: simulated annealing, network static expansion planning, large scale optimization.

INTRODUCTION

The long term transmission expansion planning problem is a large, complex mixed integer nonlinear programming problem which has represented a major challenge to all known heuristic, optimization approaches [1]. The main difficulties are due to the combinatorial nature of the planning process, which normally leads to an explosive increase in the number of alternatives that have to be analyzed even for moderately sized systems. Moreover the planning problem presents a typical multimodal landscape and so solutions can very easily be poor local optima. To cope with these problems a variety of approaches have been suggested in the literature, including a mathematical decomposition approach linked with the zero-one implicit enumeration method suggested by the authors [6], [7]. Although the method shown in [7] performs well in providing optimal solutions for small and medium size problem, it is not applicable to larger problems such as the Brazilian North-Northeastern network discussed later in the paper.

This paper proposes a new approach to the transmission expansion problem based on the concept of simulated annealing, which has been found to be a highly suitable method for solving hard combinatorial optimization methods [8]. Simulated annealing tries to avoid local optima by allowing temporary, limited deteriorations of actual solutions. In that regard it radically differs from conventional algorithms that always proceed by deterministic exchanges, which as we know may lead to local optima. Thus in the simulated annealing approach state transitions leading to actual increases in the objective function can be accepted with a certain probability.

Simulated annealing mimics the physical process of annealing in solids and has originally been proposed in the domain of statistical mechanics as a means of modeling the natural process of solidification and formation of crystals. It is known that defect free crystals, i.e. minimum energy solids, are more likely to be formed under a slow cooling process. The two main features of the simulated annealing process are (1) the transition mechanism between states and (2) the cooling scheme. The parameters defining these two key features have been determine through testing with small and medium scale problems for which optimal solutions are known [7], and then the approach has been scaled up to solve larger problems with unknown optima.

A detailed definition of the transmission expansion problem can be found in [1] and [6]. A brief summary follows. Given the network configuration for a certain year and the peak generation/demand for the next year (along with other data such as network operating limits, costs, and investment constraints), one wants to determine the expansion plan with the lowest cost, i.e. one wants to determine *where* and *what* type of new equipment should be installed [1]. Of course this is a subproblem of a more general case, called *dynamic* expansion planning, where, in addition to the questions of *what* and *where*, one wants to know *when* to install new pieces of equipment. In this paper we will be interested in the initial stages of expansion planning studies during which the basic topology of the future network is studied. Network topologies synthesized by the proposed approach will be further analyzed and improved by testing their performances using other analysis tools such as power flow, short circuit, transient, and dynamic stability analysis tools.

The paper is organized as follows. Initially the formulation of the transmission expansion problem as a mixed integer nonlinear programming problem is discussed. Next the simulated annealing approach is overviewed. Next the proposed algorithm is presented. Then the results of validation tests and comparisons with a mathematical decomposition method are reported. Next tests performed on a large scale study case are presented. Finally the relevant conclusions are summarized.

PROBLEM FORMULATION

In the following the static transmission expansion planning problem is formulated as a mixed integer nonlinear programming problem in which the power network is represented by a dc power flow model.

$$\begin{aligned} \min \quad v = & \sum_{ij} c_{ij} n_{ij} + \sum_{i} \alpha_i r_i \qquad (1) \\ & B(x+\gamma^{\circ})\theta + g + r = d \\ & (x_{ij} + \gamma^{\circ}_{ij})\,|\theta_i - \theta_j| \le (x_{ij} + \gamma^{\circ}_{ij})\overline{\phi}_{ij} \\ & 0 \le g \le \overline{g} \\ & 0 \le r \le d \\ & 0 \le n_{ij} \le \overline{n}_{ij} \end{aligned}$$

where:

c_{ij} - Cost of the addition of a circuit in branch i-j.

x_{ij} - Total susceptance added in branch i-j.

$B(.)$ - Susceptance matrix.

θ - Vector of nodal voltage angles.

γ^o - Vector of initial susceptances, whose elements are γ^o_{ij}, i.e. the summation of the susceptances in branch i-j at the beginning of the optimization.

n_{ij} - Number of circuits added in branch i-j: $n_{ij} = x_{ij}/\gamma_{ij}$; where γ_{ij} is the susceptance of the new circuits.

$\overline{\phi}_{ij}$ - Defined as the ratio: $\overline{\phi}_{ij} = \overline{f}_{ij}/\gamma_{ij}$; where $\overline{f}_{ij}$ is the maximum flow in a circuit i-j.

d - Vector of liquid demand.

g - Generation vector.

$\overline{g}$ - Vector of maximum generation capacity.

r - Vector of artificial generations.

α - Penalty parameter associated with loss of load caused by lack of transmission capacity.

Problem 1 is a mixed integer nonlinear problem. It is a typical hard combinatorial problem prone to combinatorial explosion as the number of decision variables increases. An extra complication relates to the fact that there are cases in which planning does not simply mean the reinforcement of an existing network. Sometimes we have to start from scratch, at least for parts of the network due to the addition of new load, transition and generation buses (Please see Fig.1). In the case of the Brazilian example systems, the simple addition of one or two circuits (lines or transformers) may not be enough to guarantee network connectivity: it is not uncommon to find situations in which entire paths have to be built to put all the pieces together in a single network. Under these circumstances the combinatorial burden is even heavier than it would be in simpler, reinforcement only, types of problems.

For a given set of x_{ij}, which may be a tentative solution, Problem 1 is transformed into a linear programming problem, i.e. Problem 2, which is solved for testing the adequacy of a candidate solution; adequacy is indicated by zero loss of load. Notice that Problem 1 is always feasible due to the presence of the loss of load factor $\sum_i \alpha_i r_i$ in the objective function; thus whenever a tentative solution set x_{ij} is inadequate, feasibility is achieved by the use of artificial generators (loss of load). It has been observed that this feature facilitates the annealing process to emerge from local minima, by temporarily moving through regions of inadequate solutions, but still keeping the feasibility of the mathematical problem.

$$\begin{aligned} \min \quad w = & \sum_i \alpha_i r_i \qquad (2) \\ & B(x^k + \gamma^o)\theta + g + r = d \\ & (x^k_{ij} + \gamma^o_{ij})\,|\theta_i - \theta_j| \le (x^k_{ij} + \gamma^o_{ij})\overline{\phi}_{ij} \\ & 0 \le g \le \overline{g} \\ & 0 \le r \le d \end{aligned}$$

COMBINATORIAL OPTIMIZATION BY SIMULATED ANNEALING

For most hard combinatorial problems normally found in practice only approximate solutions can be obtained by conventional optimization methods. Transmission network expansion planning for large scale system is a classical example for that matter. Simulated annealing is a stochastic optimization procedure which has been successfully applied to a range of combinatorial problems in electrical engineering including power systems [2, 9]. A basic characteristic of simulated annealing is that the quality of the final solution does not depend on initial configuration. It can be shown mathematically that SA converges asymptotically to the global optimal solution with probability one. Although this may turn out to be computationally expensive, it is a valuable feature of the approach. Normally in practice faster solutions can be obtained with faster cooling schemes, which may yield a bunch of near optimal solutions or even optimal solutions.

Generally speaking SA solves a combinatorial optimization problem formulated as a pair (G, v), where G is a finite set of configurations, possibly a huge one, also called a space of configurations, and v is an objective function which associates a real value to each possible configuration. Thus basically what has to be done is to search for the configuration(s) with minimum cost. Departing from an initial configuration, SA generates a series of configurations eventually leading to the configuration with minimum cost. The transition between two successive configurations is managed by a stochastic mechanism. The acceptance of newly generated configurations is based on the value of the objective function: configurations with decreasing objectives are always accepted, whereas configurations with higher costs can be occasionally accepted with a certain probability. The possibility of accepting higher cost solutions avoids a sequence of solutions getting trapped in local minima.

Simulated annealing models the process of annealing in solids. It is known that defect free crystals, i.e. minimum energy solids, are more likely to be formed under a slow cooling process. At a given temperature level, SA sequentially moves from one to the next configuration until thermal equilibrium is reached. At that point temperature is decreased and a new sequence of configurations is generated. The two main features of the simulated annealing process are then (1) the transition mechanism between states and (2) the cooling scheme. These two key features will be discussed below.

Basic Transition Mechanism

The transition mechanism makes the current solution move between two successive configurations: from configuration G_k to configuration G_{k+1}. This is achieved through a small modification randomly introduced into the current configuration. A simple way of doing this is based on the method suggested in [12]. Given a configuration G_k, a new configuration G_{k+1} can be obtained in the following ways:

- *Adding a new circuit:* a candidate circuit is chosen randomly and its state is increased by 1;
- *Swapping two circuits:* two candidate circuits are chose randomly; the state of one of them is increased by 1 and the state of the other is decreased by 1;
- *Removing a circuit:* a circuit of the current configuration G_k is chosen randomly and its state is decreased by 1.

This mechanism can be used in two different sequences: *add-swap-remove* or *remove-swap-add*. The first one being more adequate for the early stages of the expansion process, whereas the second one is applied to later stages of the simulation. For instance, when the *add-swap-remove* option is used, additions are tried first until success or eventual failure; in case of failure, swap is then tried; if swap fails, then removal is finally tested. When the *remove-swap-add* option is used first removals are tried until success or eventual failure;

in case of failure swaps are tested; and finally additions are tried.

In some practical applications, expansion planning involves more than the simple reinforcing of an existing network. It may occur that parts of the network have to be actually built from scratch due to the addition of new load, transition and generation buses. For two of the example systems studied in this paper, the simple addition of one or two circuits does not necessarily guarantee the connectivity of the resulting network. A special method for generating alternative connecting paths are required. In these cases the simple transition mechanism discussed above does not work properly and some improvements have to be made, as discussed below.

Modified Transition Mechanism

The network shown in Fig.1 illustrates the connectivity problem mentioned earlier in the paper. The addition of a circuit between some of the nodes may turn out to be ineffective in the sense that this circuit will connect two isolated nodes without any benefits regarding reductions in the loss of load. The modified transition mechanism has the extra requirement that any configuration obtained by *add-swap-remove* should not produce ineffective investments of the type described above. As a result, rather than single circuits, entire paths can be added-swapped-removed. For a more detailed discussion on path building please refer to [17].

Cooling Scheme

The cooling scheme is defined by the following four parameters:

- Initial temperature T_o;
- Final temperature T_f;
- Number of transitions, N_k, at temperature T_k.
- Temperature rate of change, $T_{k+1} = \beta.T_k$.

The efficiency of the algorithm regarding both the quality of the final solutions as well as the number of iterations will depend on the choice of these parameters. The procedures used in the calculation of parameters are based on the idea of thermal equilibrium and are detailed below:

- T_o

T_o is determined is such a way that practically all possible transitions could be made, i.e. T_o is a temperature for which $exp[-\frac{\Delta v_{ij}}{T_o}] \cong 1$ for virtually all pairs of configurations i, j. More specifically, in this paper we have adopted the method of [8] which suggests the following,

$$T_o = \frac{\overline{\Delta V}^{(+)}}{ln(1/X_o)} \tag{3}$$

where $\overline{\Delta V}^{(+)}$ is the average increase in cost for a certain number of random transitions, X_o is the rate of acceptance, normally chosen between 0.8 and 0.95. This of course requires the previous determination of $\overline{\Delta V}^{(+)}$ which in turn is based on the simulation of a certain number of configuration transitions (in this paper we have used $X_o = 0.85$).

- T_f

The stopping criterion is determined in such a way that at the optimal point the expected improvements in the objective function become negligible. There are at least two simple stopping criteria which have been used in practice: the first one assumes a fixed number of temperature levels T_k for the cooling process, say a number between 6 and 50; and the second is based on the rate of improvement of the objective function – say, if the objective does not improve for two consecutive iterations, the process stops. In this paper we have adopted a slightly more elaborate criterion suggested in [8]: for a given temperature level T_k, the algorithm compares the difference between the maximum and the minimum costs associated with the configurations accepted for that temperature level, with the maximum individual transition cost observed in the same temperature level; if these two numbers are within a certain tolerance, the process terminates. Thus, let ΔC_{max} be maximum cost of an accepted transition; C_{min} be the minimum cost of an accepted configuration; C_{max} be the maximum cost of an accepted configuration; then the algorithm stops whenever

$$\Delta C_{max} = C_{max} - C_{min} \tag{4}$$

- N_k

N_k should be such that a state of thermal near-equilibrium could be reached at a given temperature level T_k. This will depend not only on N_k itself but also on the difference between two consecutive temperature levels, T_k - T_{k+1}; if the temperature steps are too big, thermal equilibrium would be reached only with an abnormally high N_k. Two different rules have been used in the research reported in this paper: (1) N_k is a function of the number of branches where circuit additions are allowed (so in this case N_k does not vary with the temperature level); (2) a number N_o determined as in the first criterion is used at the initial temperature level and is corrected according to the rule

$$N_{k+1} = \rho.N_k \tag{5}$$

Notice that N_k represents the number of add-swap-remove trials, which means that at a given temperature level we can have up to three times as many individual trials. For higher temperatures the number of individual trials is closer to the lower bound, N_k, whereas for lower temperatures the number of individual trials tends to increase towards the upper bound, $3N_k$.

- $T_k \rightarrow T_{k+1}$

The rate of cooling has a direct effect on the number of iterations required at each temperature level. A simple rule which has been widely used in practice is

$$T_{k+1} = \beta.T_k \tag{6}$$

where β is a constant varying between 0.50 and 0.99. The results reported in the paper have been obtained with $\beta = 0.7$.

THE PROPOSED ALGORITHM

The simulated annealing algorithm for the transmission expansion problem is:

Step 1:
Determine initial temperature using eq. (3).

Step 2:
Define other control parameters: ρ, β, ξ, w_o and ε_w. Define the initial value of the number of iterations for each temperature level: $N_o = \xi.N_t$; N_t is the total number of branches where new circuits can be added.

Step 3:
Increase the iteration counter IT; If $IT > N_k$ then go to Step 7; Otherwise go to Step 4.

Step 4:
If loss of load for the current configuration, w_o^k, is smaller than tolerance, w_o, go to Step 5; Otherwise go to Step 6.

Step 5:
a) Simulate the *removal* of a circuit i randomly chosen among the previously added circuits (belonging to the current configuration). Test the adequacy of the resulting configuration by solving Problem 2. If the objective function decreases, i.e. $\Delta v < 0$ or the probability given by the Boltzmann distribution, P_{an}, is greater than $P_r(0,1)$, then accept the new configuration and update pointers and w_o^k, and go to Step 3; otherwise go to 5(b).

b) Simulate a *swap* of circuits, i.e. the removal of circuit i selected in 5(a) and the addition of a randomly chosen circuit j. Test the adequacy of the resulting configuration by solving Problem 2. If the objective function decreases, i.e. $\Delta v < 0$ or the probability given by the Boltzmann distribution, P_{an}, is greater than $P_r(0,1)$, then accept the new configuration and update pointers and w_o^k, and go to Step 3; otherwise go to 5(c).

c) Simulate the *addition* of circuit j selected in 5(b). Test the adequacy of the resulting configuration by solving Problem 2. If the objective function decreases, i.e. $\Delta v < 0$ or the probability given by the Boltzmann distribution, P_{an}, is greater than $P_r(0,1)$, then accept the new configuration and update pointers and w_o^k. Go to Step 3.

Step 6:
a) Simulate the *addition* of a circuit i randomly chosen among the previously added circuits (belonging to the current configuration). Test the adequacy of the resulting configuration by solving Problem 2. If the objective function decreases, i.e. $\Delta v < 0$ or the probability given by the Boltzmann distribution, P_{an}, is greater than $P_r(0,1)$, then accept the new configuration and update pointers and w_o^k, and go to Step 3; otherwise go to 6(b).

b) Simulate a *swap* of circuits, i.e. the addition of circuit i selected in 6(a) and the removal of a randomly chosen circuit j. Test the adequacy of the resulting configuration by solving Problem 2. If the objective function decreases, i.e. $\Delta v < 0$ or the probability given by the Boltzmann distribution, P_{an}, is greater than $P_r(0,1)$, then accept the new configuration and update pointers and w_o^k, and go to Step 3; otherwise go to 6(c).

c) Simulate the *removal* of circuit j selected in 6(b). Test the adequacy of the resulting configuration by solving Problem 2. If the objective function decreases, i.e. $\Delta v < 0$ or the probability given by the Boltzmann distribution, P_{an}, is greater than $P_r(0,1)$, then accept the new configuration and update pointers and w_o^k. Go to Step 3.

Step 7:
Test stopping criterion; if the criterion is satisfied, go to Step 9; otherwise go to Step 8.

Step 8:
Update N_k, T_k: $N_{k+1} = \rho.N_k$; $T_{k+1} = \beta.T_k$; Go to Step 3.

Step 9:
Perform local search.

Remarks:
In the current implementation local search performed at Step 9 consists basically in repeating the simulation for the last temperature level. At this level there is a family of solutions in thermal equilibrium around the optimal solution(s). By repeating the simulation we can actually find the optimal solution(s) or the n best solutions, for a specified number n.

VALIDATION TESTS AND PARAMETER ADJUSTMENT

The quality of the solutions produced by the simulated annealing methods depends on the determination and adjustment of a set of parameters as discussed in the previous section: basically we have to define a transition mechanism between feasible configurations and the cooling scheme. More specifically, we have to know where to start and when to stop iterating at a given temperature level, i.e. it is necessary to have an efficient, reliable method for detecting that thermal equilibrium has been reached, and also we have to determine the rate of cooling. Although there is a relatively wide range of values for which these parameters will produce good performances for the simulated annealing algorithm, they are not known a priori for this particular application. The parameters defining these two key features have been tested and adjusted with small and medium scale problems for which optimal solutions are known [7] and then the approach has been scaled up to solve a larger problem with unknown optimal solutions.

The small system is the classical Garver's 6-bus example system which has been widely used and referred to in the literature [3]. The medium size system represents the Southern Brazilian network, for which an optimal solution can be found in [6]. For both networks rescheduling of generation levels within specified limits was allowed, the optimal generation levels being a by-product of the global optimization process. Several situations have been analyzed by setting generation limits at different levels (from a no rescheduling situation to maximum rescheduling); in all the cases optimal solution has been achieved. A summary of the results obtained are given below.

Garver's 6-bus system without generation rescheduling

In this case the algorithm has found the same optimal solution originally given by Garver [3], i.e. a configuration with cost of $v = 200$ and additions: $n_{26} = 4$ circuits, $n_{35} = 1$ circuits and $n_{46} = 2$ circuits.

Garver's 6-bus system with generation rescheduling

In this case the algorithm has found the same optimal solution given in [6], i.e. a configuration with cost of $v = 110$ and additions: $n_{35} = 1$ circuit and $n_{46} = 3$ circuits.

Southeastern system without generation rescheduling

In this case the algorithm has found the same solution given in [6], i.e. a configuration with cost of $US\$154,420,000.00$ and additions: $n_{20-21} = 1$, $n_{42-43} = 2$, $n_{46-06} = 1$, $n_{19-25} = 1$, $n_{31-32} = 1$, $n_{28-30} = 1$, $n_{26-29} = 3$, $n_{24-25} = 2$, $n_{29-30} = 2$ and $n_{05-06} = 2$.

A good characteristic of the simulated annealing algorithm is that in addition to the optimal solution it provides a number of interesting local solutions found during the simulation process. These near-optimal solutions can be further analyzed by the planner to take into account factors other than expansion costs. To illustrate the point, for this test case 20 solutions where obtained within a range of 5% of optimal costs.

Southeastern system with generation rescheduling

In this case the algorithm has found the same solution given in [6], i.e. a configuration with a cost of $US\$70,289,000.00$ and additions: $n_{13-20} = 1$, $n_{20-23} = 1$, $n_{46-06} = 1$, $n_{20-21} = 2$, $n_{42-43} = 1$ and $n_{05-06} = 2$. For this test case 10 solutions where obtained within a range of 5% of the optimal costs.

APPLICATION TO A LARGE SYSTEM

The third test system is given in Fig.1 and represents the part of the Brazilian network corresponding to the North and Northeastern regions (the system has 89 buses and 183 candidate branches). This case has represented a challenge for a variety of approaches over the years mainly due to (1) the large number of possible additions which have to be considered, (2) combined with the fact that for part of the system we have a typical case of reinforcement of an existing network whereas for the other part an entirely new network has to be built from scratch. Typical cost estimates for 10 to 15 years horizons are above $US\$2.0$ *billion*. Total generation is 29,754 MW. This same problem has been previously studied by the authors using different methods, including interactive heuristics, expert systems, and mathematical programming based on the Benders decomposition. The expansion costs for the networks synthesized by the simulated annealing approach are about 7% less than the best solutions obtained by the other methods (savings of $US\$$ 170.0 *million*).

Allowing for $17MW$ (out of a total load of 29,754 MW) of loss of load a solution cost of $US\$2,664,109.00$ was obtained. The best solution found with the requirement of no loss of load presents a cost of $US\$2,676,704.00$. The circuit additions corresponding to this solution are: $n_{01-02} = 1$, $n_{02-04} = 1$, $n_{04-05} = 5$, $n_{04-06} = 3$, $n_{04-68} = 1$, $n_{05-56} = 1$, $n_{05-58} = 5$, $n_{06-07} = 2$, $n_{07-08} = 2$, $n_{08-17} = 2$, $n_{12-15} = 1$, $n_{12-17} = 1$, $n_{13-14} = 1$, $n_{13-15} = 4$, $n_{14-59} = 1$, $n_{15-16} = 3$, $n_{15-45} = 1$, $n_{15-46} = 1$, $n_{16-44} = 6$, $n_{17-18} = 2$, $n_{18-50} = 11$, $n_{18-74} = 3$, $n_{21-57} = 2$, $n_{22-23} = 1$, $n_{22-58} = 2$, $n_{24-43} = 1$, $n_{25-55} = 3$, $n_{26-54} = 3$, $n_{30-31} = 4$, $n_{30-63} = 3$, $n_{31-34} = 2$, $n_{35-51} = 1$, $n_{36-46} = 2$, $n_{40-45} = 2$, $n_{41-64} = 2$, $n_{42-44} = 2$, $n_{42-85} = 1$, $n_{43-55} = 2$, $n_{43-58} = 2$, $n_{48-49} = 1$, $n_{49-50} = 4$, $n_{52-59} = 1$, $n_{53-86} = 1$, $n_{54-58} = 2$, $n_{54-63} = 1$, $n_{56-57} = 1$, $n_{61-64} = 1$, $n_{61-85} = 2$, $n_{61-86} = 1$, $n_{67-68} = 1$, $n_{67-69} = 1$, $n_{67-71} = 3$, $n_{71-72} = 1$, $n_{72-73} = 1$, $n_{73-74} = 1$.

Computing times for this test system are in the range from 18. to 19. hours, whereas for the Southeastern system the computing times vary between 1.1 and 1.2 hours (Workstation Sun Sparc10). To improve perfomance two alternatives are being investigated: (1) a faster, specialized LP solver and (2) parallel/distributed computing.

CONCLUSIONS

The long term transmission expansion planning problem is a typical combinatorial problem in which the number of options to be analyzed increases exponentially with the size of the network, i.e. it is a NP (for *Non Polynomial* time) problem. This is due to integrality constraints (there are no fractional transmission lines) which turns the problem into a complex mixed integer nonlinear programming problem. Previous studies by the authors have shown that mathematical programming methods based on Benders decomposition and binary search are effective in obtaining optimal solutions for small and medium size problems. In this paper simulated annealing is proposed as a means of solving large scale problems such as the ones found in the planning of the integration of the future Amazon basin hydro developments to the existing Brazilian power grid of the North and Northeast regions. This problem is a mix of classical network reinforcement problems combined with the building of an entirely new network for significant parts of the planned system.

The simulated annealing is an optimization technique based on a well known process found in nature and which has been successfully applied to a number of electrical engineering problems. The results obtained with the new approach have been validated through comparison with the results given by an optimization approach based on Benders decomposition. Two example systems have been tested in the validation process. Then the simulated annealing method have been applied to a large, difficult test case for which no optimal solution is know: a number of interesting solutions has been obtained with costs about 7% less than the best solutions known for that example system obtained by optimization and heuristic methods. The potential drawback of the approach, as has already been reported regarding other types of power system applications, is the computational effort required to solve large scale problems. Thus future research will be aimed at (1) developing faster algorithms for verifying the feasibility of candidate solutions as well as (2) the exploitation of the inherent parallelism which is characteristic of the simulating annealing approach.

References

[1] *R. Fischl*: "Optimal System Expansion Planning: A Critical Overview", ERDA & EPRI Conf., Henniker, 1975.

[2] Ilic'-Spong M. et al. "Secondary Voltage Control Systems Pilot Point Information", IEEE Trans. Power Systems, Vol. 3, No. 2, pp. 660-668, May 1988.

[3] *L.L. Garver*: "Transmission Network Estimation Using Linear Programming", IEEE Trans. Power App. Syst., vol. PAS-89, pp. 1688-1697, Sept/Oct 1970.

[4] *R. Villanasa, L.L. Garver, S.J. Salon*: "Transmission Network Planning Using Linear Programming", IEEE Trans. Power App. Systems, vol. PAS-104, No. 2, February 1985.

[5] *M.V.F. Pereira, L.M.V.G. Pinto, S.H.F. Cunha, G.C. Oliveira*: "A Decomposition Approach to Automated Generation/Transmission Expansion Planning"; IEEE Trans. Power App. Systems, vol. PAS-104, No. 11, November 1985.

[6] R. Romero and A. Monticelli, "A Hierarchical Decomposition Approach for Transmission Network Expansion Planning", IEEE Trans. on Power Systems, Vol.9, No.1, pp. 373-380, Feb. 1994.

[7] R. Romero and A. Monticelli, "A Zero-one Implicit Enumeration Method for Optimizing Investments in Transmission Expansion Planning", IEEE PES Summer Meeting, Vancouver, July 1993.

[8] Van Laarhoven P.J.M., Aarts E.H."Simulated Annealing: Theory and Applications", D. Reidel Publishing Company, Holand, 1987.

[9] Hsiao Y. T., Chiang H.D., Liu C.C., Chen Y.L. "A Computer Package for Optimal Multi-objective VAR Planning in Large Scale Power Systems", IEEE Transactions on Power Systems, Vol. 9, No.2, May 1994.

[10] Cerny V. "Thermodynamical Approach to the Traveling Salesman Problem: An Efficient Simulation Algorithm", Journal of Optimization Theory and Applications, Vol. 45, No. 1, 1985.

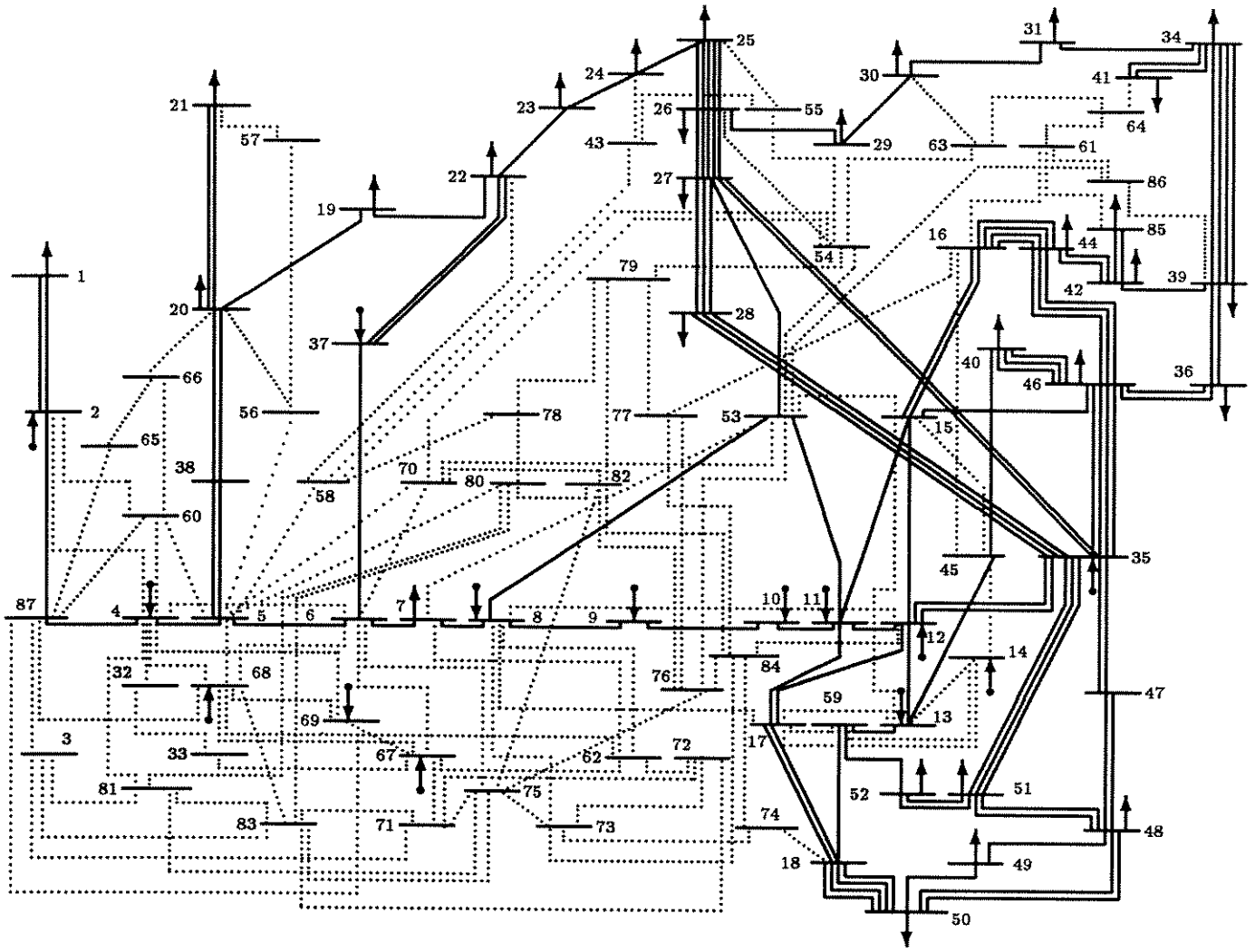


Figure 1: North-Notheastern Network.

[11] Connolly D.T. "An improved annealing scheme for the QAP", European Journal of Operation Research 46, pp. 93-100, 1990.

[12] Drexl A. "A Simulated Annealing Approach to the Multiconstraint Zero-Um Knapsack Problem", Computing 40, pp. 1-8, 1988.

[13] Kirkpatrick S., Gelatt Jr. C.D., Vecchi M.P. "Optimization by simulated annealing", Science 220. pp. 671-680, 1983.

[14] Rutenbar R.A. "Simulated Annealing Algorithms: An Overview", IEEE Circuits and Devices Magazine, pp. 19-26, 1989.

[15] Hajek B. "Cooling schedules for optimal annealing", Mathematics of Operations Research 13, pp. 311-329, 1988.

[16] Mitra D., Romeo F., Sangiovanni-Vincentelli A.L. "Convergence of finite-time behavior of simulated annealing", Advances in Applied Probability 18, pp. 747-771, 1986.

[17] A. Monticelli, A. Santos Jr., M.V.F. Pereira, S. Cunha, J.G. Praça, B. Park, "Interactive transmission network planning using a least-effort criterion", IEEE Transactions PAS, Vol.101, No.10, pp 3919-3925, Oct. 1982.

ACKNOWLEDGEMENTS

This project was partly supported by FAPESP, Fundação de Amparo à Pesquisa do Estado de São Paulo, grant 90/3940-0, and by CNPq, Conselho Nacional de Desenvolvimento Científico e Tecnológico. R.A. Gallego is on leave from Universidad Tecnológica de Pereira, Colombia, and gratefully acknowledges the support of Colciencias.

BIOGRAPHIES

Rubén Romero (Member, IEEE) received his B.Sc. degree in 1978 and P.E. degree in 1984 from the National University of Engineering, in Lima, Peru, his M.Sc. and Ph.D. degrees in 1990 and 1993 from Unicamp, in Brazil. Currently he is CNPq researcher at Unicamp. (e-mail address: *ruben@dsee.fee.unicamp.br)*

Ramón A. Gallego received his B.S. degree in Electrical Engineering in 1981 from Universidad Tecnológica de Pereira and his M.Sc. degree in 1985 from Universidad Nacional de Colombia. Currently he is working towards his Ph.D. degree at Unicamp. (e-mail address: *ramon@dsee.fee.unicamp.br)*

Alcir J. Monticelli (Senior Member, IEEE) received his Ph.D. degree from Unicamp in 1975 where he is currently a Professor of Electrical Engineering. (e-mail address: *alcir@dsee.fee.unicamp.br).*

Artigo 2

- **R.A. Gallego, A.B. Alves, A. Monticelli, R. Romero**: "Parallel Simulated Annealing Applied to Long Term Transmision Network Expansion Planning", IEEE Transactions on Power Systems, Vol. 12, No. 1, pp. 181 - 188, February 1997.

PARALLEL SIMULATED ANNEALING APPLIED TO LONG TERM TRANSMISSION NETWORK EXPANSION PLANNING

R.A. Gallego, A.B. Alves, A. Monticelli
UNICAMP-Campinas-Brazil

R. Romero
FEIS-UNESP-Brazil

Abstract:
The simulated annealing optimization technique has been successfully applied to a number of electrical engineering problems, including transmission system expansion planning. The method is general in the sense that it does not assume any particular property of the problem being solved, such as linearity or convexity. Moreover, it has the ability to provide solutions arbitrarily close to an optimum (i.e. it is asymptotically convergent) as the cooling process slows down. The drawback of the approach is the computational burden; finding optimal solutions may be extremely expensive in some cases. This paper presents a Parallel Simulated Annealing, PSA, algorithm for solving the long term transmission network expansion planning problem. A strategy which does not affect the basic convergence properties of the sequential simulated annealing algorithm have been implementeded and tested. The paper investigates the conditions under which the parallel algorithm is most efficient. The parallel implementations have been tested on three example networks: a small 6-bus network, and two complex real-life networks. Excellent results are reported in the tests section of the paper: in addition to reductions in computing times, the parallel simulated annealing algorithm proposed in the paper has shown significant improvements in solution quality.

Keywords: combinatorial optimization, simulated annealing, network static expansion planning, large scale optimization, parallel computing.

INTRODUCTION

There are two basic reasons for parallelizing the simulated annealing algorithm: the obvious one is computing time; the other one is solution quality. In hard combinatorial problems such as the transmission network expansion planning problem, both the number of alternatives to be analyzed as the number of local minimum points increases with the dimension of the network. The increase in the number of alternatives badly affects computing time, whereas the increase in the number of local minima increases the chances for the solution being caught at poor local optimum points. It has been shown that parallel simulated annealing both speeds-up the solution process [1] as well as increases the chances of global minimization [3].

Simulated annealing is modeled as a Markov chain, i.e. a sequence of trials, where the probability of the outcome of a given trial depends only on the outcome of the previous trial [6, 7]. In terms of the transmission network expansion problem, we have a sequence of network configurations generated by the algorithm, where the successor of a given configuration depends only on the given configuration (its predecessor), and not on other previous elements of the the sequence. To each trial is associated a cost function (the cost of the expansion plan). The method tries to avoid local minima by allowing, with a certain probability, occasional increases in cost.

A simulated annealing optimization algorithm applied to the transmission system expansion planning problem is described in [2]. The main advantage of the method is its ability to reach global, or near global, optimal solutions under certain circumstances (slow cooling schedules). Another characteristic of the approach is generality; for instance, there is no restrictions regarding convexity and thus the method can handle cost functions with multiple local minima (the so called multimodal landscapes). The main disadvantage is the computational burden, since slow cooling normally means slow convergence. To obtain good solutions for one of the test systems discussed in the paper, running on a Sun workstation, it may take a few days of computing time, depending on the control parameters utilized in the algorithm.

The paper discusses the use of parallel computing methods for speeding-up simulated annealing optimization of transmission networks. The main difficulty associated with the parallelization of the simulated annealing algorithm is due to the fact that simulated annealing is modeled by a Markov chain, which is a typical sequential entity; the parallelization should be done in such a way that this sequential structure is not affected when the algorithm is mapped onto a parallel architecture. The paper investigates the conditions under which the parallel algorithms are most efficient. The so called *division* method has been chosen for implementation. In this method, for each temperature level, a set of N_k trials is broken down into a number of sub-sets equal to the number of processors of the parallel machine, each one performing N_k/np trials, where np is the number of processors in the parallel machine. From time to time processors exchange information about the best solutions found so far. For large, complex networks the proposed method gave not only faster solutions but also better ones.

The paper is organized as follows. Firstly the mathematical formulation of the transmission expansion problem as a mixed integer nonlinear programming problem is discussed. Secondly the simulated annealing approach is overviewed the Parallel Simulated Annealing algorithm is described. Next, tests performed with three example system are presented: a 6-bus system and two real-life systems representing the Southern and the North-Northeastern parts of the Brazilian network.

MATHEMATICAL FORMULATION OF THE PROBLEM

The long term transmission expansion planning problem is a typical combinatorial problem in which the number of

options to be analyzed increases exponentially with the size of the network, i.e. it is a np (for *non polynomial* time) problem. This is due to integrality constraints (there are no fractional transmission lines) which turns the problem into a complex mixed integer nonlinear programming problem. Previous studies by the authors have shown that mathematical programming methods based on Benders decomposition and binary search are effective in obtaining optimal solutions for small and medium size problems. In this paper simulated annealing is proposed as a means of solving large scale problems such as the ones found in the planning of the integration of the future Amazon basin hydro developments to the existing Brazilian power grid of the north and northeast regions. This problem is a mix of classical network reinforcement problems combined with the building of an entirely new network for significant parts of the planned system.

A detailed definition of the transmission expansion problem can be found in [8] and [11]. A brief summary follows. Given the network configuration for a certain year and the peak generation/demand for the next year (along with other data such as network operating limits, costs, and investment constraints), one wants to determine the expansion plan with the lowest cost, i.e. one wants to determine *where* and *what* type of new equipment should be installed [8]. Of course this is a subproblem of a more general case, called *dynamic* expansion planning, where, in addition to the questions of *what* and *where*, one wants to know *when* to install new pieces of equipment. In this paper we will be interested in the initial stages of expansion planning studies during which the basic topology of the future network is studied. Network topologies synthesized by the proposed approach will be further analyzed and improved by testing their performances using other analysis tools such as power flow, short circuit, transient, and dynamic stability analysis tools.

The long term transmission expansion planning problem can be formulated as follows

$$\begin{aligned} \min \quad v = & \sum_{ij} c_{ij} n_{ij} + \sum_{i} \alpha_i r_i \qquad (1) \\ & B(x+\gamma^o)\theta + g + r = d \\ & (x_{ij} + \gamma^o_{ij})\,|\theta_i - \theta_j| \le (x_{ij} + \gamma^o_{ij})\overline{\phi}_{ij} \\ & 0 \le g \le \overline{g} \\ & 0 \le r \le d \\ & 0 \le n_{ij} \le \overline{n}_{ij} \end{aligned}$$

where: c_{ij} is the cost of the addition of a circuit in branch $i-j$, x_{ij} is the total susceptance added in branch $i-j$, $B(.)$ is the susceptance matrix, θ is the vector of nodal voltage angles, γ^o is the vector of initial susceptances, whose elements are γ^o_{ij}, i.e. the summation of the susceptances in branch $i-j$ at the beginning of the optimization, n_{ij} is the number of circuits added in branch $i-j$, $n_{ij} = x_{ij}/\gamma_{ij}$; where γ_{ij} is the susceptance of the new circuits, $\overline{\phi}_{ij}$ are defined as the ratio: $\overline{\phi}_{ij} = \overline{f}_{ij}/\gamma_{ij}$; where $\overline{f}_{ij}$ is the maximum flow in a circuit $i-j$, d is the vector of liquid demand, g is the generation vector, $\overline{g}$ is the vector of maximum generation capacity, r is the vector of artificial generations and α is the penalty parameter associated with loss of load caused by lack of transmission capacity.

For a given set of x^k_{ij}, which may be a tentative solution, Problem 1 is transformed into a linear programming problem which is solved for testing the adequacy of a candidate solution; adequacy is indicated by zero loss of load. Notice that Problem 1 is always feasible due to the presence of the loss of load factor $\sum_i \alpha_i r_i$ in the objective function; thus whenever a tentative solution set x^k_{ij} is inadequate, feasibility is achieved by the use of artificial generators (loss of load).

This problem has represented a major challenge to all known heuristic, optimization approaches [8, 12]. A number of methods have been suggested in the literature. A mathematical decomposition approach (Benders decomposition with the zero-one implicit enumeration) proposed in [13] performs well in providing optimal solutions for small and medium size problems. As for larger systems some type of heuristic has to be added to the approach. It is not applicable to larger problems such as the Brazilian north-northeastern network discussed later in the paper.

PARALLEL SIMULATED ANNEALING

It is known that defect free crystals are more likely to be formed under slow cooling. During the crystalization process the free energy of a solid tends to be minimized; depending on the speed of cooling, local or global minimum can be reached. Simulated annealing mimics the physical process of annealing in solids (i.e. heating up a solid, and cooling it down until it crystalizes). It has been originally proposed as a model for studying crystalization. More recently [16], it has been used to solve large scale combinatorial problems such as the one posed by the long term expansion planning of transmission networks [2]. The analogy between annealing and optimization is established by associating the cost function of the optimization problem to the free energy of a crystal.

In this section the simulated annealing algorithm is briefly overviewed and the parallelization of the algorithm is discussed. The key requirement imposed on the parallelization process is that it does not affects the ability of the simulated annealing method to obtain optimal solutions.

Sequential Version of the Algorithm

The network transmission expansion problem is formulated as a pair (G, v), where G is a finite set of configurations, called a space of configurations, and v is a cost function which associates a real value to each possible configuration. Departing from an initial configuration, the SA algorithm generates a series of configurations eventually leading to the configuration with minimum cost. The transition between two successive configurations is managed by a stochastic mechanism. The acceptance of newly generated configurations is based on the value of the objective function: configurations with decreasing objectives are always accepted, whereas configurations with higher costs can be occasionally accepted with a certain probability. The possibility of accepting higher cost solutions avoids a sequence of solutions getting trapped in local minima. The sequential version of the SA algorithm is summarized below:

Sequential Algorithm:

Step 1: Initialize current solution x_o, temperature T_o, $k = 0$ and make $x_{opt} = x_o$.

Step 2: Evaluate cost function $v(x_j)$; If $v(x_j) < v(x_{opt})$, update incumbent solution $x_{opt} = x_j$.

Step 3: Determine state transition Δx_j.

Step 4: Evaluate the cost of the candidate solution, $v(x_j + \Delta x_j)$, and determine the cost variation,

$\Delta v = v(x_j + \Delta x_j) - v(x_j)$.

Step 5: If $\Delta v < 0$, go to Step 7.

Step 6: If $exp(-\Delta v/T_k) < R$, where R is a random number in the range $[0, 1]$, go to Step 7; Otherwise, go to Step 8.

Step 7: Update the current state, $x = x + \Delta x$; Test whether the system reached thermal equilibrium; If so go to Step 8, otherwise go to Step 2.

Step 8: Reduce temperature and make $k = k + 1$.

Step 9: If $T_k < T_{min}$, stop; Otherwise, go to Step 2.

Implementation Details:

According to Step 7 of the *Sequential Algorithm* given above a certain number of trials is performed at each temperature level until a near thermal equilibrium condition is reached. This sequence of trials form a Markov chain of length N_k. In practice, rather than actually testing whether thermal equilibrium has been reached, it is easier to specify a limit for N_k, i.e. the number of trials which is performed at temperature level T_k. Normally this is done by given a N_o for the initial temperature and a control parameter ρ; the number of trials increases with the temperature level according to $N_{k+1} = \rho.N_k$ (in this paper we have used $\rho = 1.2$).

Remarks:

a) The two main features of the simulated annealing process are: (1) the transition mechanism between states; and (2) the cooling schedule. The transition mechanism is define in Steps 3. Cooling is carried out in Step 8 of the algorithm, where temperature T_k is decreased by a factor β (e.g., $\beta = .75$) [2].

b) The transition mechanism (Steps 3) can be carried out in two different sequences: ***add-swap-remove*** or ***remove-swap-add***, where the operation of ***adding***, ***swapping*** and ***removing*** involves a single circuit or a group of candidate circuits. The first sequence is more commonly found in the early stages of the expansion process, whereas the second one is more frequent in the later stages of the simulation. For instance, when the ***add-swap-remove*** option is used, additions are tried first until success or eventual failure; in case of failure, swap is then tried; if swap fails, then removal is finally tested. When the ***remove-swap-add*** option is used first removals are tried until success or eventual failure; in case of failure swaps are tested; and finally additions are tried [2].

c) In Step 2 the algorithm keeps track of the best current solution; for every new accepted configuration the corresponding cost is compared with the cost of the incumbent solution and the incumbent solution is updated whenever a solution with a smaller cost is reached [2, 4].

d) Simulated annealing tries to avoid local optima by allowing temporary, limited deteriorations of actual solutions; this is performed in Step 6 where state transitions leading to cost increases are accepted with a certain probability. In that regard it radically differs from conventional algorithms that always proceed by deterministic exchanges, which as we know may may lead to local optima.

Parallelization of the SA Algorithm

Let np denote the number of processors in our parallel machine. A simple, effective way of parallelizing the SA algorithm consists in dividing the effort of generating a Markov chain over the np processors: each processor generates a subchain of length N_k/np. In order to keep the main characteristics of the SA algorithm, at each temperature level, when all processor finish processing their individuals tasks, the incumbent optimal solutions are sent to $node - 0$, which selects the best one and broadcasts the results to the individual processor.

Parallel Algorithm:

Step 1: If $p = 0$: (a) initialize x_o and T_o; and (b) broadcasts x_o. If $p = 0, \ldots, np-1$: (a) receive x_o; and (b) make $x_o^p = x_o$ and $x_{opt}^p = x_o$.

Step 2: Evaluate cost function $v(x_j^p)$; If $v(x_j^p) < v(x_{opt}^p)$, update incumbent solution at node p, $x_{opt}^p = x_j^p$.

Step 3: Determine state transition at node p, Δx_j^p.

Step 4: Evaluate the cost of the candidate solution, $v(x_j^p + \Delta x_j^p)$, and determine the cost variation, $\Delta v^p = v(x_j^p + \Delta x_j^p) - v(x_j^p)$.

Step 5: If $\Delta v^p < 0$, go to Step 7.

Step 6: If $exp(-\Delta v^p/T_j) < R^p$, where R^p is a random number in the range $[0, 1]$, go to Step 7; Otherwise, go to Step 8.

Step 7: Update the current state, $x^p = x^p + \Delta x^p$; Test thermal equilibrium conditions at node p; If so go to Step 8, otherwise go to Step 2.

Step 8: If $p = 1, \ldots, np - 1$: (a) send x_{opt}^p to $node - 0$; and (b) receive new global x_{opt} from $node - 0$; (c) make $x_{opt}^p = x_o$; (d) go to Step 2. If $p = 0$: (a) update temperature $T_{k+1} = \beta.T_k$; (b) make $k = k + 1$; (c) receive x_{opt}^p from nodes $p = 1, \ldots, np - 1$; (d) find $x_{opt} = min[x_{opt}^p]$, for $p = 0, \ldots, np - 1$; (e) broadcasts x_{opt}; and (f) go to Step 9.

Step 9: If $T_k < T_{min}$, stop; Otherwise, go to Step 2.

Implementation Details:

The same comments made above regarding near thermal equilibrium conditions hold for the parallel version of the simulated annealing algorithm. At Step 8, when $node - 0$ determines x_{opt}, it also checks whether optimal solutions coming from all nodes, $p = 1, \ldots, np-1$, happen to be the same; if so, the process stops and that solution is called the optimal one. This condition normally is satisfied before the minimum temperature T_{min} is reached, which guarantees faster convergence when compared with the *Sequential Algorithm*. Also, due to the fact that this is a multiple trial algorithm, the probability of reaching the global optimal solution is higher than in the *Sequential Algorithm*. These two conditions combined are responsible for speed-ups which sometimes are much higher than the ones we could expect from the simple division of the effort of generating a Markov chain over the parallel machine processors.

Remarks

a) Fig.1 illustrates how the division algorithm works. In the horizontal line we have the evaluation of the temperature and the number of trials per temperature level, whereas in the vertical line we have the np processors of the parallel machine. At temperature $T(k)$ each process performs its own sequence of N_k/np trials.

b) Processors communicate when they finish analyzing the trials assigned to them. When this happens all processors receive the new best configuration and restart from that point on.

c) When np increases, the number of trials performed by a processors decreases and if the number becomes too small this the performance of the algorithm may be negatively affected.

The number of processors for which this type of problem occurs depends on the size of the problem being solve (number of candidate circuits). A strategy which can be used to reduce the impact of this type of problem consists in increasing the number of trials, N_k, per temperature level and/or increase control parameter cooling. The idea is to guarantee that each processor will have a number of trials big enough to simulate near thermal equilibrium conditions as close as possible.

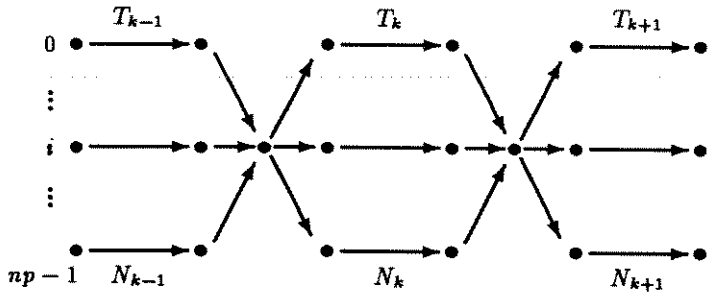


Figure 1: Parallelization by the division algorithm

TEST RESULTS

Garver's 6-bus System

This is the classical Garver's 6-bus example system illustrated in Fig.2 which has been widely used and referred to in the literature [10]. Generation rescheduling within specified limits was allowed, the optimal generation levels being a by-product of the global optimization process. Without generation rescheduling the optimal solution is the same as given in [10], i.e. a configuration with cost of $v = 200$ and additions: $n_{26} = 4$ circuits, $n_{35} = 1$ circuits and $n_{46} = 2$ circuits. With generation rescheduling the optimal solution is the same as given in [13], i.e. a configuration with cost of $v = 110$ and additions: $n_{35} = 1$ circuit and $n_{46} = 3$ circuits. There is a range of parameters β and ρ for which the *Sequential SA Algorithm* converges to the optimal solution. Typical values have been used for the *Parallel SA Algorithm*: $\beta = .75$ and $\rho = 1.2$, i.e. $T_{k+1} = .75T_k$ and $N_{k+1} = 1.2N_k$, with $N_o = 4.nl$, where $nl = 15$ is the total number of candidate branches for this example system.

Table 1 shows how the performance of the parallel algorithm varies with the dimension, np, of the parallel machine. The computing times given in the table are typical (median) values since the number of iterations may with the seed used for generating the sequence of random numbers (Markov Chain). Table 2 illustrate how the seed value may affect computing times. In all cases given in Tables 1 and 2 the same optimal solution has been obtained ($v = 110$); this is not always the case however, mainly with larger systems for which the quality of the solution may depend on np.

Southern Brazilian System

This is a medium size system modeling part of the Southern Brazilian network, for which an optimal solution is known [13]. Generation rescheduling within specified limits was allowed, the optimal generation levels being a by-product of the global optimization process.

The investment at optimal solution is US$ 154 420 000, which corresponds to the following 16 circuit additions: one circuit in branch 20-21, two circuits in branch 42-43, one circuit in branch 46-6, one circuit in branch 19-25, one circuit

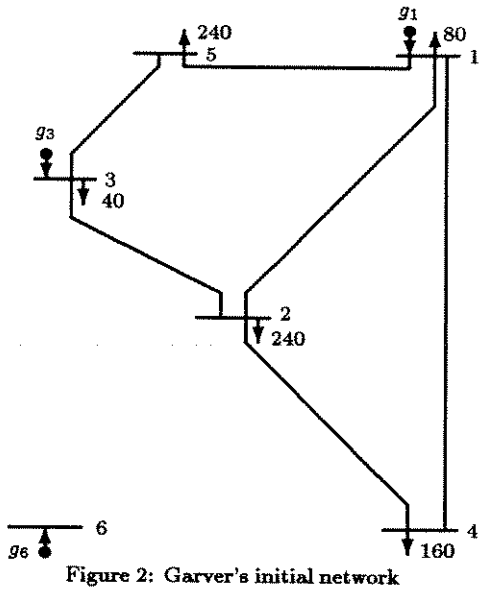


Figure 2: Garver's initial network

Table 1: Results for Garver's Example System.

Processor	No. iter.	time (min.)
1	5253	41.3
2	2171	16.6
4	1262	10.2
8	971	7.6

Control parameters: $\rho = 1.2$; $\beta = 0.75$; $N_o = 4.nl$; $nl = 15$; $nb = 6$; $v_{op} = 110$.

Table 2: The effect of the *seed* on the convergence for a 4-processor parallel machine.

Tests	No. iter.	time (min.)
1	1568	12.60
2	1178	9.67
3	1262	10.25
4	1458	11.80
5	1566	12.67
6	1019	8.40

Control parameters: the same as in Table 1.

in branch 31-32, one circuit in branch 28-30, three circuits in branch 26-29, two circuits in branch 24-25, two circuits in branch 29-30, and two circuits in branch 5-6. Table 3 shows how the performance of the parallel algorithm varies with the dimension, np, of the parallel machine. Table 4 shows how the configuration changes with cooling for the case of a 4-processor parallel machine.

North-Northeastern Brazilian System

This example system represents the North-Northeastern regions of the Brazilian network. The planning model used in these studies has 89 buses and 183 candidate branches (A illustration of this system can be found in Fig.1 of Ref. [2]). This is case involving the reinforcement of an existing network combined with the building of an entirely new network for a significant part of the system. In some parts of the network it may be necessary to start from scratch due to the addition of new load, transition and generation buses. To a lesser degree, this also happens with the previous example system (Southeastern network). In these cases the addition of one

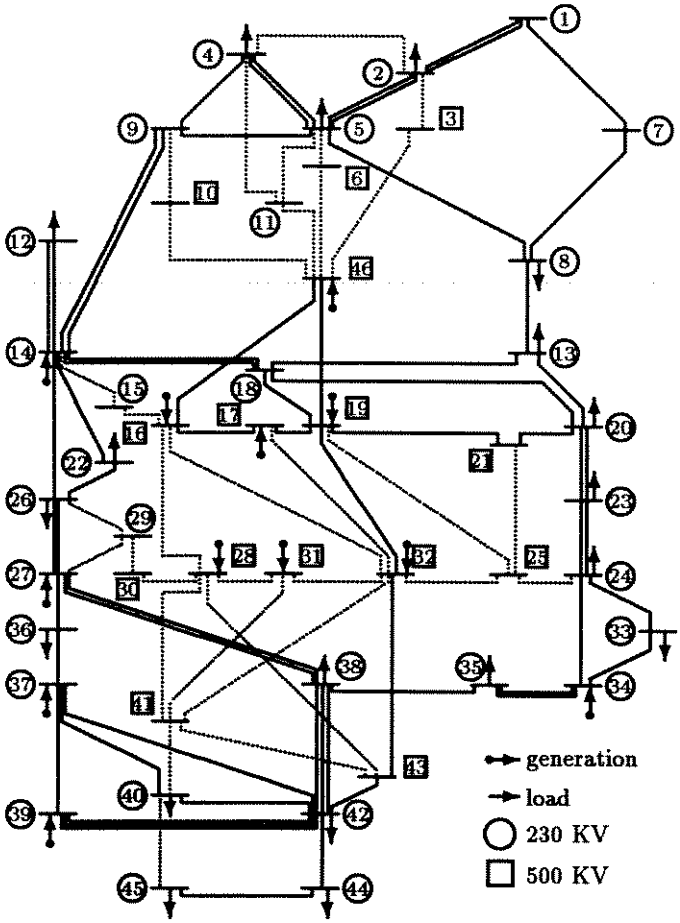


Figure 3: Southern Brazilian System

or two circuits (lines or transformers) may not be enough to guarantee network connectivity: entire paths may have to be built to put all the pieces together in a single network. Expansion cost estimates for 10 to 15 years horizons are above $US\$2.0$ ***billion***. Total generation is 29,754 MW.

Table 5 shows how the quality of the solution found by the parallel simulated annealing algorithm may vary with the dimension, np, of the parallel machine. Contrary to what happens with the previous examples, for this case an optimal solution is not known. However, as predicted in the theory of simulated annealing it is clear from the results that the chances of reaching an optimal solution increases with np. The solution obtained with $np = 32$ is best known solution for this example network.

The best configuration found in the tests ($np = 16$ and $np = 32$) has costs of US$ 2,630,290,000.00, with no loss of load, and requires the following circuit additions:

$n_{01-02} = 1$, $n_{02-04} = 1$, $n_{04-05} = 4$, $n_{04-06} = 4$, $n_{04-68} = 1$, $n_{05-56} = 1$, $n_{05-58} = 4$, $n_{06-07} = 2$, $n_{07-08} = 2$, $n_{08-17} = 2$, $n_{12-17} = 1$, $n_{13-15} = 4$, $n_{14-59} = 1$, $n_{15-16} = 3$, $n_{15-46} = 1$, $n_{16-44} = 6$, $n_{17-18} = 3$, $n_{18-50} = 11$, $n_{18-74} = 3$, $n_{21-57} = 2$, $n_{22-37} = 1$, $n_{22-58} = 1$, $n_{24-43} = 1$, $n_{25-55} = 3$, $n_{26-54} = 2$, $n_{30-31} = 3$, $n_{30-63} = 3$, $n_{31-34} = 2$, $n_{35-46} = 1$, $n_{35-51} = 1$, $n_{36-46} = 2$, $n_{40-45} = 1$, $n_{40-46} = 2$, $n_{41-64} = 2$, $n_{42-44} = 4$, $n_{42-85} = 2$, $n_{43-55} = 2$, $n_{43-58} = 2$, $n_{48-49} = 1$, $n_{49-50} = 4$, $n_{52-59} = 1$, $n_{53-86} = 1$, $n_{54-58} = 2$, $n_{54-63} = 1$, $n_{56-57} = 1$, $n_{61-64} = 1$, $n_{61-85} = 2$, $n_{61-86} = 1$, $n_{67-68} = 1$, $n_{67-69} = 1$, $n_{67-71} = 3$, $n_{71-72} = 1$, $n_{72-73} = 1$, $n_{73-74} = 1$.

Table 3: Results for the Southern system.

Processor	No. iter.	time (min.)
1	40514	1790
2	11387	503
4	9396	402
8	8164	361
16	6445	285

Control parameters: $\rho = 1.2$; $\beta = 0.75$; $N_o = 4.nl$; $nl = 79$; $nb = 46$; $v_{op} = US\$\ 154,420,000.00$

Table 4: A typical Cooling Process for a 4-processor parallel machine

Temperature	Time (min)	No. of iter.	w (MW)	Invest. $(10^6)US\$$)
100334	—	—	3450	—
75250	10	241	0	647.505
56438	23	525	35	565.823
42328	37	867	21	358.684
31746	55	1277	0	287.816
23809	76	1768	0	275.575
17857	101	2358	0	214.799
13393	131	3065	0	203.793
10044	168	3918	0	173.173
7533	211	4940	0	178.068
5650	263	6144	0	154.420
4237	327	7634	0	158.314
3178	402	9396	0	154.420

Control parameters: the same as in Table 3.

Table 5: Results for the North-Northeastern System

Processor	Number iter.	time (hours)	Invest. $10^6 US\$$
1	201580	441.0	2668.242
16	14045	30.7	2640.330
32	9597	21.0	2632.825

Control parameters: $\rho = 1.2$; $\beta = 0.75$; $N_o = 4.nl$; $nl = 183$; $nb = 89$; $v_{best} = US\$\ 2,630,290,000.00$

This configuration presents a cost reduction of US$ 220,000,000.00 when compared with the best solutions found with other heuristic methodologies [2], and a cost reduction of US$ 46,000,000.00 when compared with the best solution obtained by a sequential simulated annealing algorithm [2]. If we accept some moderate level of loss of load a variety of solutions with reduced costs can be obtained: for instance a solution with costs of US$ 2,600,745,000.00 and loss of load of 98 MW and another one with costs of US$ 2,626,825,000.00 and loss of load of 5 MW.

Remarks:

a) As it could be expected the Parallel Simulated Annealing algorithm was able to find better solution than the Sequential Simulated Annealing algorithm [2] for the Brazilian North-Northeastern network. This is due to the large number of local optimal solutions existing in this problem (highly multimodal landscape). The reason for such behavior is related to the fact that the Parallel Simulated Annealing algorithm tests a larger

variety of alternatives searching simultaneously in differents areas of the solution space. This same behavior has been observed and reported in [3] where the Parallel Simulated Annealing algorithm was applied to the voltage optimization problem.

b) The tests with the North-Northeastern network have been performed with using a family of configurations with costs varying from 2650 to 2750 million US dollars. As it happens in Tables 1 and 3, the values given in Table 5 represent typical (median) values, since the solution found may vary with initialization and with the random number seed. Best results were found with $np = 16$ and $np = 32$. The conclusion is that for larger networks it is advisable to use a larger number of processors, not only for speed, but mainly for solution quality.

CONCLUSIONS

A parallel simulated annealing optimization technique has been successfully implemented and tested. The method is general in the sense that it does not assume any particular property of the problem being solved, such as linearity or convexity. Excellent results are reported in the tests section of the paper: in addition to reductions in computing times, the parallel simulated annealing algorithm proposed in the paper has shown significant improvements in solution quality.

The parallelization was done in such a way that sequential structure os the simulated annealing algorithm was not affected. The paper investigates the conditions under which the parallel algorithms are most efficient. For each temperature level, a set of N_k trials is broken down into a number of sub-sets equal to the number of of processors of the parallel machine, each one performing N_k/np trials, where np is the number of processors in the parallel machine. From time to time processors exchange information about the best solutions found so far. For large, complex networks such as the Brazilian North-Northeastern system, the proposed method gave not only faster solutions but also better ones.

References

[1] *Aarts E.H.L., De Bont F.M.J., Habers E.H.A. and Van Laarhoven P.J.M.:* "Parallel Implementations of the Statistical Cooling Algorithm", Integration, the VLSI Journal 4, pp. 209-238, 1986.

[2] *Romero R., Gallego R.A., Monticelli A.*: "Transmission System Expansion Planning by Simulated Annealing", Power Industry Computer Applications - PICA 95, Salt Lake City, May 1995.

[3] *Mori H., Takeda K.*: "Parallel Simulated Annealing for Power System Decomposition", IEEE Transaction on Power Systems, Vol. 9, No. 2, May 1994.

[4] *Johnson D.S., Aragon C.R., McGeoch L.A., Schevon C.*: "Optimization by Simulated Annealing: An Experimental Evaluation; Part I, Graph Partitioning", Operations Research, Vol. 37, No. 6, Nov.-Dec. 1989.

[5] *D.S. Johnson, C.R. Aragon, L.A. McGeoch, C. Schevon*: "Optimization by Simulated Annealing: An Experimental Evaluation; Part II, Graph Coloring and Number Partitioning", Operations Research, Vol. 39, No. 3, May-June 1991.

[6] *Aarts E., Korst J.*: "Simulated Annealing and Boltzmann Machines", John Wiley & Sons, 1989.

[7] Van Laarhoven P.J.M., Aarts E.H."Simulated Annealing: Theory and Applications", D. Reidel Publishing Company, Holand, 1987.

[8] *Fischl R.*: "Optimal System Expansion Planning: A Critical Overview", ERDA & EPRI Conf., Henniker, 1975.

[9] Ilic'-Spong M. et al. "Secondary Voltage Control Systems Pilot Point Information", IEEE Trans. on Power Systems, Vol. 3, No. 2, pp. 660-668, May 1988.

[10] *Garver L.L.*: "Transmission Network Estimation Using Linear Programming", IEEE Trans. Power App. Syst., Vol. PAS-89, pp. 1688-1697, Sept/Oct 1970.

[11] Monticelli A., Santos Jr. A., Pereira M.V.F., Cunha S., Praça J.G., Park B., "Interactive Transmission Network Planning Using a Least-Effort Criterion", IEEE Transactions PAS, Vol.101, No.10, pp 3919-3925, Oct. 1982.

[12] *Pereira M.V.F., Pinto L.M.V.G., Cunha S.H.F., Oliveira G.C.*: "A Decomposition Approach to Automated Generation/Transmission Expansion Planning"; IEEE Trans. Power App. Systems, Vol. PAS-104, No. 11, November 1985.

[13] *Romero R., Monticelli A.*:"A Zero-One Implicit Enumeration Method for Optimizing Investments in Transmission Expansion Planning", IEEE Transactions on Power Systems, Vol. 9 No. 3, August 1994.

[14] Hsiao Y. T., Chiang H.D., Liu C.C., Chen Y.L. "A Computer Package for Optimal Multi-objective VAR Planning in Large Scale Power Systems", IEEE Transactions on Power Systems, Vol. 9, No.2, May 1994.

[15] Drexl A. "A Simulated Annealing Approach to the Multiconstraint Zero-One Knapsack Problem", Computing 40, pp. 1-8, 1988.

[16] Kirkpatrick S., Gelatt Jr. C.D., Vecchi M.P. "Optimization by simulated annealing", Science 220, pp. 671-680, 1983.

ACKNOWLEDGEMENTS

This project was partly supported by FAPESP, Fundação de Amparo à Pesquisa do Estado de São Paulo, grant 90/3940-0, and by CNPq, Conselho Nacional de Desenvolvimento Científico e Tecnológico. R.A. Gallego is on leave from Universidad Tecnológica de Pereira, Colombia, and gratefully acknowledges the support of Colciencias.

BIOGRAPHIES

Biographies of Ramón A. Gallego, Rubén Romero and Alcir J. Monticelli can be found in Ref. [2].

Antonio Cesar Baleeiro received his B.Sc. degree from Federal University of Goias (UFG) in 1983, and his M.Sc. degree from Federal University of Uberlandia in 1991 (UFU), all EE. He is currently working towards his Ph.D degree at UNICAMP. His general research interests are in power systems real-time control, optimal power flow and harmonic elimination.

Artigo 3

- **R.A. Gallego, A. Monticelli, R. Romero**: "Comparative Studies on Non-Convex Optimization Methods for Transmission Network Expansion Planning", IEEE Transactions on Power Systems, Vol. 13, 1998.

COMPARATIVE STUDIES ON NON-CONVEX OPTIMIZATION METHODS FOR TRANSMISSION NETWORK EXPANSION PLANNING

R.A. Gallego, A. Monticelli
UNICAMP-Campinas-Brazil

R. Romero
FEIS-UNESP-Ilha Solteira-Brazil

Abstract:

We have investigated and extensively tested three families of non-convex optimization approaches for solving the transmission network expansion planning problem: simulated annealing (SA), genetic algorithms (GA), and tabu search algorithms (TS). The paper compares the main features of the three approaches and presents an integrated view of these methodologies. A hybrid approach is then proposed which presents performances which are far better than the ones obtained with any of these approaches individually. Results obtained in tests performed with large scale real-life networks are summarized.

Keywords: simulated annealing, genetic algorithm, tabu search, network static expansion planning, combinatorial optimization.

INTRODUCTION

Network transmission expansion planning is a hard, large scale combinatorial problem in which the number of options to be analyzed increases exponentially with the size of the network. This problem presents a multimodal landscape in which the chances of being trapped in a local minimum solution increases with the size of the problem. On the other hand, the allocation of transmission costs in a competitive environment requires the determination of optimal transmission network expansion plans [1]. Most of the conventional optimization algorithms used in practice however are unable to generate optimal solutions for larger, complex networks.

In this paper three non-convex optimization approaches to combinatorial optimization are discussed: simulated annealing (SA), genetic algorithms (GA), and tabu search algorithms (TS). To certain extent, all the three approaches are based on processes found in nature: simulated annealing (SA) mimics the physical process of solidification and formation of perfect crystals; genetic algorithms (GA) are based on natural genetics and evolutionary processes; and tabu search (TS) is based and generalizes concepts from the domain of artificial intelligence. In addition to discussing the application of these methodologies to the transmission network expansion planning problem, this paper proposes a hybrid approach which draws upon the most interesting features of each one of the three approaches.

The paper is organized as follows: firstly, an overview of the three combinatorial approaches is presented (SA, GA and TS approaches) with emphasis on the network expansion problem; secondly, the basic mechanisms behind each approach are discussed and compared; thirdly, a new hybrid approach is proposed; next, test results obtained with large, difficult real-life cases are presented; finally, the relevant conclusions are summarized.

OVERVIEW OF COMBINATORIAL OPTIMIZATION METHODS

In this section the main characteristics of the three non-convex optimization approaches studied in the paper are summarized.

Simulated Annealing

Simulated Annealing starts the optimization process with a given initial configuration from which a set of candidate configurations are generated through a cooling process. A candidate configuration is accepted if its investment cost is smaller than the cost of the current configuration; with a certain probability, however, configurations presenting costs higher than the cost of the current configuration can be accepted as well. This feature is crucial since it allows the simulated annealing process to avoid local optimal solutions. Also essential to simulated annealing is the cooling scheme which basically controls how temperature is decreased aiming at obtaining perfect "crystals" [12, 13]. Two other important aspects of simulated annealing are the search space representation and a transition mechanism which governs the way the search space is explored [10].

Genetic Algorithms

Genetic Algorithms start the optimization process from a initial population (a set of configurations). New populations (or new generations) are generated by an evolution mechanism comprising selection, crossover and mutation, and which gradually leads to individuals (configurations) with higher performance indices (evolutionary process). The selection mechanism tends to guide the solution to local optimal points since it selects configurations presenting superior performance indices. In the transmission network expansion problem the crossover mechanism introduces *diversification* in the search for better configurations allowing the algorithm to escape from local optimal points. The mutation mechanism on the other hand helps exploring the neighborhood of a configuration and thus has an important role in searching for local optimal solutions (this process is also called *intensification*, as opposed to *diversification* mentioned above) [4].

Tabu Search

Tabu Search algorithms are initialized with a configuration (or a set of configurations when search is performed in parallel) and the search is performed in a more aggressive way than in the cases of simulated annealing and genetic algorithms, in the sense that the TS algorithm is greedier than

SA and GA algorithms. In the tabu search approach for each candidate configuration a neighborhood structure is defined; the algorithm then moves to the best configuration in such neighborhood, i.e. it switches to the configuration presenting the smallest cost. Normally only the most attractive neighbors are evaluated, otherwise the problem could become intractable. An essential feature of tabu search algorithms is the direct exclusion of search alternatives temporarily classed as forbidden. As a consequence the use of memory become crucial in these algorithms, as opposed to simulated annealing, for example. Other mechanisms of tabu search are *intensification* and *diversification*: by the intensification mechanism the algorithm does a more comprehensive exploration of attractive regions which may lead to a local optimal point; by the diversification mechanism, on the other hand, the search is moved to previously unvisited regions, which is important in avoiding getting trapped into local minimum points [3, 4].

COMPARATIVE ANALYSIS

In this section the main mechanisms behind each of the three combinatorial algorithms (SA, GA and TS) are discussed and compared. Specifically, we will discuss the use of memory and the intensification and diversification mechanisms.

Short Term and Long Term Memories

The simulated annealing algorithm is based on the Boltzmann distribution (statistical process) and as such makes little use of memory. Thus some configurations may be visited more than once during the optimization process. Although the best configuration so far is stored (incumbent optimal solution) this information is not used to redirect the search. A more elaborate implementation of SA would calculate and keep track of the standard deviations of the configuration costs for each temperature level; this information could then be used in determining the rate of cooling.

Genetic algorithms make implicit use of memory by keeping track of attractive building blocks. Also, the fact that algorithm works with populations (a sequence of generations) can be related to the use implicit memory. Explicit memory is present when a subset of a population (the current generation) is not subject to crossover and thus kept as is for the next generation in order to preserve valuable genetic information.

In the tabu search approach memory structures are used to direct to search process. Four different dimensions are considered in memory structures: *recency*, *frequency*, *quality*, and *influence*. *Recency*-based memory is one of the most important features of tabu search: it is a type of *short term memory* which keeps track of solution attributes that have changed during the most recent moves made by the algorithm [3]. The information contained in this memory allows to label as tabu-active selected attributes of the solutions recently visited; this feature avoids revisiting solution already visited in recent past. Tabu lists are used for managing recency-based memory. The *frequence* dimension is used in connection with long term memory mechanisms; for instance, in the transmission planning problem one could keep track of the frequences with which line/transformer additions appeared in the configurations visited by the algorithm. This information could be used later on in diversifying the search by changing the choice rules in such a way that configurations containing still unused features could be visited by the algorithm. The *quality* dimension refers to the costs associate to different solutions searched by the algorithm: for instance, there are certain groups of lines/transformers that greatly improve candidate solutions when added together to the current configuration; these building blocks can be used as part of the *path relinking* strategy, which allows the creation of new configuration from high quality building blocks (such building blocks are in elite configurations store in long term memory). Finally, the *influence* dimension takes account of the impact a line/transformer addition may have on the loss of load: the removal of these lines/transformers will then remain tabu-active as long as possible. Summarizing, the TS algorithm makes extensive use of memory structures to direct the search to attractive regions and for avoiding revisiting solutions which have already been considered.

Intensification

The mechanism of intensification in simulated annealing is less systematic than in genetic algorithms and tabu search algorithms. SA uses a probabilistic acceptance test such that at the early stages of the cooling process (higher temperatures) the search moves around more freely, whereas at lower temperatures (closer to the optimal solution) the algorithm becomes more conservative and intensification takes place; i.e. the search for the best solution in a neighborhood, which hopefully is the neighborhood containing the optimal solution).

As with simulated annealing, in genetic algorithm intensification is carried out in a non-systematic way. The mechanism of mutation allows to perform relatively small changes in candidate configurations an so it is a means of searching in a neighborhood, i.e. is a mechanism for implementing intensification. As it happens with simulated annealing intensification is more likely to happen at the late stages of the optimization process when the populations are formed by a relatively uniform set of configurations (hopefully, elite configurations).

Tabu search algorithms explore intensification in a more systematic way. Configurations found during the search are stored and their neighborhoods are then explored more thoroughly; the local optimal solutions closest to these solutions are found in the intensification phase. Later on in the paper we will describe Strategy B which is used in searching for local optimal solutions (a simple way of implementing intensification). Intensification can also be performed based on building blocks which are present in elite solutions stored in memory. By a mechanism called path relinking we can produce new attractive configurations utilizing such building blocks. As a matter of fact *path-relinking* is a means of implementing both intensification and diversification, depending on the amplitude of the changes introduced by this mechanism in the current configuration, as discussed in the following.

Diversification

In simulated annealing diversification occurs by the action of the probabilistic acceptance test. Configurations with higher costs can occasionally be accepted and when this happens usually the search is moved to a new region in the search space (diversification). As mentioned above, this is more likely to happen during the early stages of the optimization process (i.e. at higher temperature levels). At lower temperatures the probability of such type of moves are greatly reduced and the algorithm becomes more intensification-oriented.

The crossover mechanism is the main responsible for the introduction of diversification in genetic algorithms. During crossover two high performance configurations exchange parts of their elements (chromosomes) generating new configura-

tions which usually are located in unvisited regions of the search space. Diversification is guaranteed by the random selection of the configurations which are used for crossover and by the random choice of the crossover point (or points) where the strings that represent the configurations are broken.

In tabu search algorithms diversification is performed in a more planned way that it is in simulated annealing and genetic algorithms. As before, the objective of diversification is to move the search to unvisited regions of the search space. In the transmission network expansion planning problem this is carried out by temporarily changing the rules for finding new configurations: for instance, certain lines are removed from a configuration and become tabu-active, i.e. they will not be allowed to be reintroduced for a certain time. The removal of key lines/transformers usually change the architecture of a solution in such a way that forces the algorithm to visit unexplored regions as desired.

Problem Modeling

Problem representation comprises a configuration and an objective function. In the transmission network expansion problem a configuration is characterized by a list of line/transformer additions, whereas the objective function has two components: investment costs and loss of load penalized by an appropriate factor. The same representation is used in the three approaches (SA, GA and TS). A configuration can be seen as a point in the search space. To each configuration corresponds a neighborhood structure.

In the simulated annealing approach a neighbor configuration is obtained by the addition or removal of a line/transformer (or a set of lines/transformers defining a path) or by exchanging two lines/transformers (or paths). Although the size of a neighborhood in the SA approach normally is very large, this does not represent a problem since in this approach only one neighbor is select for the current configuration; the quality of the other neighbor solutions is not taken into account in the process.

In genetic algorithms there is no formal definition of neighborhood since transitions are defined in a selection-crossover-mutation cycle by which an entirely new generation is found from the current one.

Since tabu search is greedier than simulated annealing and genetic algorithms, neighborhood definition is more critical for this method: one of the main strategies of tabu search consists in moving to the best configuration in a given neighborhood. Usually the sizes of neighborhoods are much larger than can be evaluated by the algorithm, and thus only the most attractive part of a neighborhood is actually explored. Having the means of efficiently finding such attractive parts is a key issue in the TS methodology. This is particularly important in the transmission network expansion problem as will be discussed later on in the paper.

Implementation Details

There are a number of algorithms and heuristics that have been used in the past to generate approximate solutions to the transmission network expansion problem, e.g., [14]-[16]. These procedures are based on special features of the problem and may provide satisfactory performance under the right conditions. There are three basic issues in applying combinatorial algorithms for which approximate algorithm can be of some help: (a) generating sound initial configurations; (b) representing specific knowledge about the problem; and (c) using special strategies from approximate algorithms to guide and simplify the search process.

Both SA, GA and TS require the knowledge of initial configurations (a single configuration in the cases of SA and TS and a initial generation in the case of GA). Two alternative ways of generating such initial configurations have been tested in connection with the transmission network expansion problem: by random selection and by approximate algorithms [14, 15, 16]. Although in principle the initial point should not influence the final solutions yielded by SA, GA and TS, in practice we have observed that both computing times and quality of final solutions are improved when the algorithms are properly initialized using information from approximate methods. A modified version Garver's algorithm has shown to be particularly good in generating such initial configurations [9, 14].

Regarding specific knowledge (or expert knowledge) about the problem there is at least one issue wortwhile mentioning: in certain situations transmission network expansion may involve not only reinforcement but the addition of new generation/load buses which may require more than a single branch to become connected to the existing network (there are cases, for example, in which a path formed by, say, two transformers and three line sections have to be added in order to have any impact on the network power flow distribution). A strategy based on the use of such building blocks is particularly useful in connection with genetic algorithms.

Tabu search, more than simulated annealing and genetic algorithms, is particularly suited for incorporating special strategies from approximate algorithms. As has been mentioned above, one of the main difficulties found in using TS regards the need for reducing the size of a solution neighborhood without loosing quality, since TS tries to find the best configuration in the neighborhood before making a move during the intensification phase. In dealing with practical problems, dramatic reductions in neighborhood sizes are normally required, and this should be done adaptively. Approximate algorithms can be extremely useful at this stage. For instance, when the current configuration is infeasible, i.e. it presents some loss of load due to lack of transmission capacity, approximate methods such as the ones given in [14, 15, 16, 18] can provide lists of attractive line/transformer additions which can be used to adaptively guide the search in a neighborhood; in addition to reducing the size of the neighborhood significantly (say, to one tenth or so of its actual size).

HYBRID ALGORITHM FOR TRANSMISSION NETWORK EXPANSION PLANNING

Based on the best characteristics of simulated annealing, genetic algorithms and tabu search, a hybrid approach was developed which presents performances far better than the ones obtained with any one of these approaches used isolately. Figure 1 will be used for illustrating the workings of the proposed method.

Mathematical Formulation of the Problem

The static (one stage) transmission expansion planning problem can be formulated as the following mixed integer nonlinear program:

$$\min \quad v = \sum_{ij} c_{ij} n_{ij} + \sum_{i} \alpha_i r_i \qquad (1)$$

$$B(x + \gamma^o)\theta + g + r = d$$

$$(x_{ij} + \gamma^o_{ij})\,|\theta_i - \theta_j| \le (x_{ij} + \gamma^o_{ij})\overline{\phi}_{ij}$$

$$0 \le g \le \overline{g}$$

$$0 \leq r \leq d$$
$$0 \leq n_{ij} \leq \overline{n}_{ij}$$

where:

c_{ij} - Cost of the addition of a circuit in branch i-j.

x_{ij} - Total susceptance in branch i-j.

$B(.)$ - Susceptance matrix.

θ - Vector of nodal voltage angles.

γ^{o} - Vector of initial susceptances, whose elements are γ^{o}_{ij}, i.e. the summation of the susceptances in branch i-j at the beginning of the optimization.

n_{ij} - Number of circuits added in branch i-j: $n_{ij} = x_{ij}/\gamma_{ij}$; where γ_{ij} is the susceptance of the new circuits.

$\overline{\phi}_{ij}$ - Defined as the ratio: $\overline{\phi}_{ij} = \overline{f}_{ij}/\gamma_{ij}$; where $\overline{f}_{ij}$ is the maximum flow in a circuit i-j.

d - Vector of liquid demand.

g - Generation vector.

$\overline{g}$ - Vector of maximum generation capacity.

r - Vector of artificial generations.

α - Penalty parameter associated with loss of load caused by lack of transmission capacity.

For a given set of decisions variables x_{ij}, Problem 1 becomes linear program:

$$\min \quad w = \sum_{i} \alpha_i r_i \qquad (2)$$
$$B(x^k + \gamma^o)\theta + g + r = d$$
$$(x^k_{ij} + \gamma^o_{ij})\,|\theta_i - \theta_j| \leq (x^k_{ij} + \gamma^o_{ij})\overline{\phi}_{ij}$$
$$0 \leq g \leq \overline{g}$$
$$0 \leq r \leq d$$

which is solved for testing the adequacy of a candidate solution; adequacy is indicated by zero loss of load. Notice that Problem 1 is always feasible due to the presence of the loss of load factor $\sum_i \alpha_i r_i$ in the objective function; thus whenever a tentative solution set x_{ij} is inadequate, feasibility is achieved by the use of artificial generators (loss of load). It has been observed that this feature facilitates the optimization process to come out from local minima, by temporarily moving through regions of inadequate solutions, but still keeping the feasibility of the mathematical problem.

Outline of the Hybrid Approach

The hybrid algorithm draws upon a basic tabu [3, 4] search algorithm to which some features of simulated annealing [10, 11], genetic algorithms [5, 9] and approximate algorithms [14, 15, 16] have been added.

Figure 1 shows the workings of the proposed approach with its three strategies. The figure also shows a linear objective (dotted lines) and a non-convex feasible region. The search process begins either randomly or by using an approximate algorithm. Usually this initial configuration is an infeasible one (the starting point appearing out of the feasible region in Figure 1), i.e. there are overloads which require certain degree of loss of load. From this starting point strategy A is activated; this strategy will bring the configuration to the feasible region. Next, strategy B performs local search by visiting neighboring configurations inside the feasible region. Then strategy C is activated by means of which the search moves across the feasibility region. And then the cycle is repeated as indicated in the figure. Notice that the transition rules are different for each one of the three strategies; i.e. in each one of the strategies we have intensification, diversification and memory structures; what varies from one strategy to the other is the degree of utilization of these mechanisms.

Rather than working with a single configuration, a population can be used instead (as in genetic algorithms); in this case cycles comprising strategies A, B and C are carried out independently for each configuration of the population. From time to time, after a certain number of iterations, a new generation of configurations is formed from the current configurations, as in GA. In forming a new generation two basic mechanisms are applied to the best configurations of the current population: (a) guided crossover and (b) path relinking. Both mechanisms operate on high quality building blocks which form part of the current configurations. (Notice that guided crossover has been adopted as a basic mechanism for diversification, rather than the more conventional *frequency* dimension normally used in TS algorithms.)

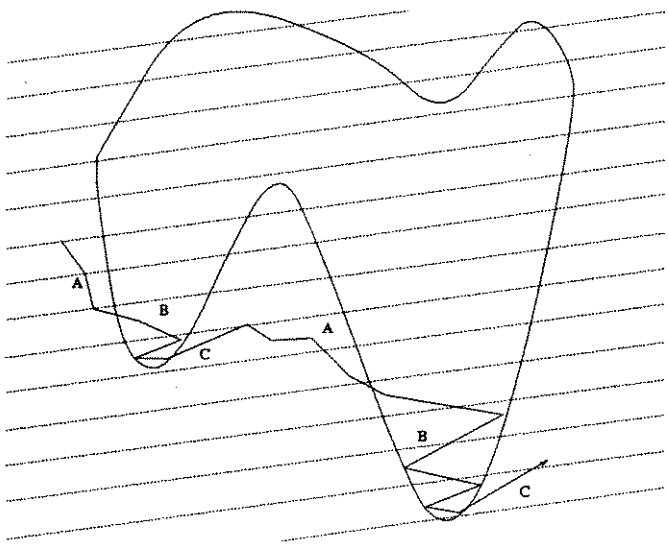


Figure 1: Search strategies of the hybrid algorithm.

Set of Initial Configurations

A family of initial configurations is obtained with a modified Garver's algorithm: the idea is to obtain a first configuration using the algorithm as is, and then turning tabu-active some essential attributes of this configuration leading to other configuration belonging to different regions of the search space. Garver's algorithm uses a transportation model, i.e. a model in which only Kirchhoff's current law is taken into account. By doing so the algorithm, although approximate, is able to pinpoint the main transmission trunks in the expanded network. These configurations normally contains the most interesting attributes which will eventually be found in optimal solutions which makes Garver's approach an extremely handful initializer for the optimal search process. Although the proposed approach (as well as SA, GA and TS algorithms in general) can be randomly initialized, Garver's initialization has proved to be more effective both regarding

computational speed and final solution quality.

Strategy A

Strategy A is first applied to the initial configurations (initial generation) which in turn was obtained via modified Garver's algorithm [14]. Usually such configurations are infeasible since only Kirchhoff's current law is taken into account in Garver's transportation model; some loss of load due to lack of transmission capacity are normally observed when the linearized power flow model ($P = B\theta$, the so called dc power flow model) is imposed. The evaluation of each neighboring configuration requires the solution of a specialized LP.

The most important feature of strategy A is the characterization of the neighborhoods of the current configurations (i.e. the configuration belonging to the current generation). A neighborhood is defined as formed by a list of additions or swaps obtained by: (a) the top additions suggested by the approximate methods [14, 15, 16] (Garver's algorithm based on a transportation model of the transmission network [14], least effort method based on a dc power flow method [16], and the minimum loss of load method based on a specialized linear program [15]); (b) line, transformer, or paths additions to branches adjacent to buses, or groups of buses, with loss of load/generation [18]; (c) a short list of randomly defined additions; and finally (d) a short list of randomly chosen line/transform/path swaps. A transition is made to the configuration in the reduced neighborhood presenting maximum decrease in the objective function. Whenever the current configuration is a local minimum point, i.e. none of the configurations in the reduced neighborhood produces a decrease in the objective function, the k best configurations (typically $k \leq 5$) are analyzed and one of the two following alternatives is randomly taken: (a) chose the best one (as in a plain TS algorithm); (b) among these k configurations, excluding the best one, accept a randomly chosen one provided it passes the SA criterion (i.e. for the computed cost increase $\Delta v > 0$, calculate the probability given by the Boltzmann distribution, $P = P_0 e^{-\Delta v/kT}$, and if it is greater than a random number in the range [0,1], accept the configuration). This strategy balances the greedness of a TS algorithm and diversification as it normally occurs in SA and GA algorithms.

Each one of the approximate algorithms [14, 15, 16] gives a set of attractive additions requiring a relatively small computational effort (a LP solution). During the execution of strategy A two lists of tabu-active attributes are kept by the algorithm: if an attribute becomes tabu-active, for a certain number of iterations the corresponding additions or removals are forbidden. An *aspiration criterion* [3, 4] is also used in strategy A: an attribute becomes tabu-inactive whenever leading to a configuration which is better than any other found during the application of strategy A. Most of the computational effort of the algorithm is taken by strategy A.

Strategy B

Strategy B is a form of intensification by means of which, given a feasible configuration belonging to the current population and its reduce neighborhood, its is found the best local feasible solution in that neighborhood. Neighboring feasible solutions are found by small perturbations introduced in the current configuration, i.e. by the removal or swapping of lines/transformers causing small impact the power flow distribution. The list of small impact changes are formed by: (a) removal of low cost lines/transformers; (b) removal of randomly chosen lines/transformers; (c) swapping of low cost line/transformers; and (d) swapping of of randomly chosen lines/transformers. Whenever a new feasible solution is found the list is updated; the strategy stops whenever there is no configuration in the list presenting objective function better than the one of the current configuration under consideration. During the application of strategy B there is no tabu-active constraints. In addition to strategy B there are other types of intensification: e.g. intensification occurs in connection with path relinking as discussed below.

Strategy C

This strategy consists in removing from the current configuration the less critical lines/transformers (typically the least expensive ones), excluding the additions added during the last application of strategy B (to avoid cycling). These additions become tabu-active for a relatively high number of iterations (an iteration is a transition between two consecutive configurations made by one of the three strategies); for the Northern-Northeastern network we have use a tabu-tenure varying from 18 to 36 iterations). Strategy C will eventually lead the current family of configurations back to the infeasible region. If properly implemented, strategy C normally takes only one step.

Stopping Criterion

Two types of stopping criteria are needed: one for individual strategies and other for the global process (cycling through the three strategies as shown in Figure 1). For instance, strategy A stops when during a given number of iterations (transitions between configurations) no further reductions in the objective function are obtained. Whenever this happens the optimal incumbent solutions are stored in long-term memory. The global stopping criterion is similar to the stopping criterion used in the genetic algorithm [9], i.e. the process stops when the objective of the incumbent optimal solutions does not improve in a given number of complete cycles (cycles through strategies A, B, and C).

Long Term Memory and Path Relinking

During the optimization process two sets of configurations are kept in memory: one is the set of configurations forming the current population and other is the set of elite solutions, i.e. the list of best solutions so far (typically from two to five solutions). The configurations in these lists can be used both in intensification as in diversification: a typical use is in a mechanism called *path relinking*.

Path relinking is a strategy which allows further exploration of known solutions. In the current implementation of the approach we take two configurations: one belonging to the current generation and other in the list of elite solutions. Taking the elite solutions as a goal, i.e. assuming the attributes of the elite solution can improve the current solution, a current solution is modified by adding to it attributes of the elite configuration. Path relinking has been adopted for producing part of the configurations which form part of the next generation (the other configurations are generated by guided crossover as discussed above). As a result of path relinking we can get either a new superior solution in the neighborhood of the current solution (intensification) or a solution which will be far apart from the original one, thus allowing exploring unvisited regions of the search space (diversification). Although both effects are always present, in the network transmission expansion planning problem, it has been observed that the diversification effect is more pronounced than the intensification

one.

Final Remarks

Three alternate diversification procedures have been described. The first one in connection with strategy C, which is a simple mechanism used to take a current configuration to an adjacent infeasible region. A second type of diversification is aimed at taking the current configuration to more remote regions of the search space; this is done by performing guided crossover within the current population; care is taken in order to avoid destroying attractive paths that form part of such elite solutions. Finally, an additional diversification mechanism is path relinking as discussed above; in this case elite configurations exchange attractive attributes with configurations of the current population.

TEST RESULTS

The test results reported in this section illustrate the relative performance of the three methods (SA,GA and Hybrid). The impact of the initial configurations obtained via Garver's algorithm and by random choice is discussed as well.

Garver's 6-bus system

In this case only the random initialization has been considered since the modified Garver's algorithm is able to find the optimal solution. Table 1 gives the performances of the three methods tested for the case in which generation/load levels are fixed (no redispatch is allowed): N_{op} is the number of LP solutions required to reach the optimum for the first time and N_t is the number of LP solutions to achieve convergence. Twenty different cases have been tested for each method; the percentage of times the Hybrid method was able to hit the optimal solution averaged 85% (which is around 10% higher than the rate of success observed with SA and GA algorithms).

Algorithm	N_{op}	N_t
SA	594	881
GA	598	754
Hybrid	470	528

Table 1: Results for Garver's 6-bus example system.

The optimal configuration for this case is: investment cost $v = 200$; and additions $n_{26} = 4$ circuits, $n_{35} = 1$ circuit and $n_{46} = 2$ circuits. In the SA algorithm the following parameters have been used: $N_k = 2.n_l$ (n_l is the number of lines), $T_o = 400$, $\beta = 0.7$ e $\rho = 1.2$, [10]. In the GA algorithm the control parameters were: $\alpha = 10$, $\rho_c = 0.9$, variable mutation rate with $\rho_m = [0.04, 0.08]$ per bit, $k_{max} = 6$, [9]. In the Hybrid algorithm a population of four configurations along with two elite configurations have been kept in long term memory. No path relinking has been used in this case.

Southern System

If redispatch is allowed (a less demanding situation) the best configuration found by the modified Garver's initializer is practically the optimal solution, with minor differences. Thus this situation will not be considered in the comparative tests. If no redispatch is allowed (fixed generation/load levels) a larger number of additions is required which makes the problem more difficult for the algorithm. Table 2 shows the results obtained with two different initializations: Garver's and random choice (twenty different cases have been tested). As with Garver's example system, the rate of success of the Hybrid algorithm was slightly better than the ones obtained with GA and SA algorithms. In the SA algorithm the following parameters have been used: $N_k = 2.n_l$, $T_o = 70000$ for random initialization and $T_o = 20000$ when the modified Garver's initializer is used; $\beta = 0.7$ and $\rho = 1.2$ for both initializations [10]. In the GA algorithm the control parameters were: $\alpha = 400$, population of de 50 configurations, $\rho_c = 0.9$, variable mutation rate with $\rho_m = [0.02, 0.04]$ per bit, $k_{max} = 30$, [9]. In the Hybrid algorithm a population of six configurations have been used with Garver's initializer and eight configurations with random initializer. Two elite configurations have been kept in both cases. Path relinking have been used in both cases as well; two configurations of the current population are randomly chosen for path relinking with the two elite configurations.

Random Initialization

Algorithm	N_{op}	N_t
SA	6831	10291
GA	6775	8275
Hybrid	1883	4835

Modified Garver's Initialization

Algorithm	N_{op}	N_t
SA	4018	5820
GA	2850	4039
Hybrid	896	1969

Table 2: Results for the Southern example system.

The optimal configuration for this case is: investment cost of $US\$154,420,000.00$; and additions $n_{20-21} = 1$, $n_{42-43} = 2$, $n_{46-06} = 1$, $n_{19-25} = 1$, $n_{31-32} = 1$, $n_{28-30} = 1$, $n_{26-29} = 3$, $n_{24-25} = 2$, $n_{29-30} = 2$ and $n_{05-06} = 2$.

North-Northeastern system without generation rescheduling

The third test system represents the part of the Brazilian network corresponding to the North and Northeastern regions; the system has 89 buses and 183 candidate branches (The complete data set for this test system is available from the authors). Total demand is 29,754 MW and there is no known optimal solution for this system.

Regarding solution quality all the three algorithms presented comparable performances. The best configuration found by SA has investments of US$ 2,630,290,000 [11]; the best configuration found by GA has investments of US$ 2,600,595,000 [9]; and the best configuration obtained with the hybrid algorithm has costs of US$ 2,596,416 and the following list of additions:

$n_{01-02} = 1$, $n_{02-04} = 1$, $n_{04-05} = 4$, $n_{04-06} = 1$, $n_{04-81} = 3$, $n_{05-56} = 1$, $n_{05-58} = 4$, $n_{12-15} = 1$, $n_{13-14} = 1$, $n_{13-15} = 4$, $n_{14-59} = 1$, $n_{15-16} = 3$, $n_{15-46} = 2$, $n_{16-44} = 6$, $n_{18-50} = 11$, $n_{18-74} = 5$, $n_{21-57} = 2$, $n_{22-23} = 1$, $n_{22-58} = 2$, $n_{24-43} = 1$, $n_{25-55} = 3$, $n_{26-54} = 1$, $n_{30-31} = 2$, $n_{30-63} = 2$, $n_{35-51} = 1$, $n_{36-39} = 2$, $n_{36-46} = 4$, $n_{40-45} = 1$, $n_{40-46} = 1$, $n_{41-64} = 2$, $n_{42-44} = 2$, $n_{43-55} = 2$, $n_{43-58} = 2$, $n_{48-49} = 1$, $n_{49-50} = 4$, $n_{52-59} = 1$, $n_{53-86} = 1$, $n_{54-58} = 1$, $n_{54-63} = 1$, $n_{56-57} = 1$, $n_{61-64} = 1$, $n_{61-85} = 2$, $n_{61-86} = 1$, $n_{67-68} = 1$, $n_{67-69} = 1$, $n_{67-71} = 3$, $n_{71-72} = 1$, $n_{72-73} = 1$, $n_{73-74} = 2$, $n_{73-75} = 1$, $n_{75-81} = 1$.

In the SA algorithm the following parameters have been used: $N_k = 4.n_l$, $\beta = 0.75$ and $\rho = 1.2$, [11]. In the GA

algorithm the control parameters were: a population of 160 configurations, $\rho_c = 0.9$, variable mutation rate with $\rho_m = [0.02, 0.04]$ per bit, $k_{max} = 80$ [9]. In the Hybrid algorithm a population of 24 configurations have been used along with three elite configurations. Six of the configuration of each new generation were was obtained by path relinking; the rest by guided crossover.

CONCLUSIONS

Three families of non-convex optimization approaches for solving the transmission network expansion planning problem have been investigated: simulated annealing (SA), genetic algorithms (GA), and hybrids based on tabu search algorithms (TS) which incorporates efficient features of SA and GA algorithms. Performance of the approaches have been compared both regarding computational speed as well as final solution quality. The results obtained in applying the hybrid algorithm to the transmission network expansion problem confirms the claims made in the literature regarding the superiority of this approach in dealing with complex, large scale combinatorial problems in which the size of search space increases exponentially with the dimensions of the network.

References

[1] *H. Rudnick, R. Palma, E. Cura and C. Silva*: "Economically Adapted Transmission Systems in Open Access Schemes – Application of Genetic Algorithms", IEEE PES Summer Meeting, July 1995.

[2] *R. Fischl*: "Optimal System Expansion Planning: A Critical Overview", ERDA & EPRI Conf., Henniker, 1975.

[3] *F. Glover*: "Tabu Search Fundamental and Uses", Graduate School of Business, University of Colorado, April, 1995.

[4] *M. Laguna*: "Tabu Search Tutorial", II Escuela de Verano Latino-Americana de Investigaci'on Operativa, 1995.

[5] *M. Srinivas, L.M. Patnaik*: "Genetic Algorithms", IEEE Computer, 1994.

[6] *D.E. Goldberg*: "Genetics Algorithms in Search, Optimization and Machine Learning", Addison Wesley, Reading, USA, 1989.

[7] *S.A. Kazarlis, A.G. Bakirtzis, V. Petridis*: "A Genetic Algorithm Solution to the Unit Commitment Problem", IEEE Transactions on Power Systems, Vol. 11, No. 1, February 1996.

[8] *K.P. Wong, Y.W. Wong*: "Combined Genetic Algorithms/Simulated Annealing/Fuzzy Set Approach to Short-Term Generation Scheduling with Take-or-Pay Fuel Contract", IEEE Transactions on Power Systems, Vol. 11, No. 1, February 1996.

[9] *R.A. Gallego, A. Monticelli, R. Romero*: " Transmission System Expansion Planning by Extended Genetic Algorithm", submitted for publication.

[10] *R. Romero, R.A. Gallego, A. Monticelli*: "Transmission System Expansion Planning by Simulated Annealing", IEEE Transactions on Power Systems, Vol. 11, No. 1, February 1996.

[11] *R.A. Gallego, A.B. Alves, A. Monticelli, R. Romero*: "Parallel Simulated Annealing Applied to Long Term Transmission Network Expansion Planning", IEEE PES Winter Meeting, Baltimore, Maryland, January 1996.

[12] *Aarts E., Korst J.*: "Simulated Annealing and Boltzmann Machines", John Wiley & Sons, 1989.

[13] *Van Laarhoven P.J.M., Aarts E.H.*:"Simulated Annealing: Theory and Applications", D. Reidel Publishing Company, Holand, 1987.

[14] *L.L. Garver*: "Transmission Network Estimation Using Linear Programming", IEEE Trans. Power App. Syst., Vol. PAS-89, pp. 1688-1697, Sept/Oct 1970.

[15] *M.V.F. Pereira, L.M.V.G. Pinto, S.H.F. Cunha, G.C. Oliveira*: "A Decomposition Approach to Automated Generation/Transmission Expansion Planning", IEEE Trans. Power App. Systems, Vol. PAS-104, No. 11, November 1985.

[16] *A. Monticelli, A. Santos Jr., M.V.F. Pereira, S. Cunha, J.G. Praça, B. Park*: "Interactive Transmission Network Planning Using a Least-Effort Criterion", IEEE Transactions PAS, Vol. 101, No. 10, pp 3919-3925, October 1982.

[17] *R. Romero and A. Monticelli*: "A Zero-One Implicit Enumeration Method for Optimizing Investments in Transmission Expansion Planning", IEEE Transactions on Power Systems, Vol. 9, No. 3, August 1994.

[18] *C.J. Baldwing, C.A. deSalvo, C.H. Hoffman and W.S. Ku*, "A Model for Transmission Planning by Logic", Trans. AIEE Power App. and Systemms, Vol. 78, pt B, 1638-1645, Feb. 1960.

ACKNOWLEDGEMENTS

This project was partly supported by FAPESP, Fundação de Amparo à Pesquisa do Estado de São Paulo, and by CNPq, Conselho Nacional de Desenvolvimento Científico e Tecnológico. R.A. Gallego is on leave from Universidad Tecnológica de Pereira, Colombia, and gratefully acknowledges the support of Colombian Ministry of Education-Colciencias.

BIOGRAPHIES

Ramon A. Gallego received his B.Sc. degree in Electrical Engineering in 1981 from Universidad Tecnologica de Pereira and his M.Sc. degree in 1985 from Universidad Nacional de Colombia. Currently he is working towards his Ph.D. degree at Unicamp. (e-mail address: *ramon@dsee.fee.unicamp.br)*

Alcir J. Monticelli (Fellow IEEE) received his Ph.D. degree from Unicamp in 1975 where he is currently a Professor of Electrical Engineering. (e-mail address: *alcir@dsee.fee.unicamp.br).*

Rubén Romero received his B.Sc. degree in 1978 and P.E. degree in 1984 from the National University of Engineering, in Lima, Peru, his M.Sc. and Ph.D. degrees from Unicamp in 1990 and 1993. Currently he is a Professor of Electrical Engineering at Feis-Unesp-Ilha Solteira. (e-mail address: *ruben@isldee.uesp.ansp.br)*

Artigo 4

- **R.A. Gallego, A. Monticelli, R. Romero**: " Transmission System Expansion Planning by Extended Genetic Algorithm", submitted for publication.

TRANSMISSION SYSTEM EXPANSION PLANNING BY EXTENDED GENETIC ALGORITHM

R.A. Gallego, A. Monticelli
UNICAMP-Campinas-Brazil

R. Romero
FEIS-UNESP-Ilha Solteira-Brazil

Abstract: This paper presents an extended genetic algorithm for solving the optimal transmission network expansion planning problem. Two main improvements have been introduced in the genetic algorithm: (a) initial population obtained by conventional optimization based methods; (b) mutation approach inspired in the simulated annealing technique. The proposed method is general in the sense that it does not assume any particular property of the problem being solved, such as linearity or convexity. Excellent results are reported in the test results section of the paper for a difficult large scale real-life problem: substantial reduction in investment costs has been obtained with regard to previous solutions obtained via conventional optimization methods and simulated annealing algorithms.

Keywords: genetic algorithm, network static expansion planning, combinatorial optimization.

INTRODUCTION

It is recognized that the allocation of transmission costs in a competitive environment will require more careful evaluation of alternative transmission expansion plans [1]. As a result, the need for methods that are able to synthesize optimal transmission expansions plans has become more important than ever. Unfortunately, however, practice has shown that conventional optimization procedures are unable to produce optimal solutions for larger networks such as the ones studied in [8,9]. The reason is that the long term transmission expansion planning problem is a hard, large scale combinatorial problem. The number of options to be analyzed increases exponentially with the size of the network, i.e. it is a NP (for non polynomial time) problem. There is a larger number of local optimal solutions (a highly multimodal landscape) which makes the chances of being trapped in one of them very large.

As an alternative to conventional optimization methods, two different heuristic search algorithms rooted in natural and physical processes have been investigated: simulated annealing and genetic algorithms. Genetic Algorithms are based on the mechanisms of evolution and natural genetics (the survival of the fittest individuals); simulated annealing mimics the physical process of annealing in solids and has originally been proposed in the domain of statistical mechanics as a means of modeling the natural process of solidification and formation of crystals. The results obtained with the simulated annealing approach have been presented in [8,9].

Genetic Algorithms are based on the mechanisms of evolution and natural genetics (the survival of the fittest individuals). The proposed approach has been compared with a more conventional optimization technique based on mathematical decomposition with a zero-one implicit enumeration procedure. Tests have been performed on three different systems. Two smaller example systems for which optimal solutions are known have been used to tune the main parameters of the genetic algorithm. The genetic algorithm method then has been applied to a larger example system for which no optimal solutions are known: as a result interesting solutions have been obtained with significant savings in investment costs.

Genetic algorithms generate a sequence of populations (generations) by using three basic mechanisms: selection, crossover and mutation. Crossover is a mechanism of exchange of information among members of a generation. Evolutionary strategies use mutation as an important search mechanism. Despite the algorithm's success, some open issues remain: (1) the choice of control parameters; (2) the exact roles of crossover and mutation; (3) the characterization of search landscapes amenable to optimization; (4) convergence properties; (5) ability to solve large scale practical problems. All these issues are fully addressed in the paper and the answers found using the transmission network expansion problem as an example are reported.

The paper is organized as follows. First, an overview of the genetic algorithm applied to the transmission expansion problem is presented. Secondly, some practical aspects of genetic algorithm are described. Thirdly, the proposed algorithm is summarized. Next, tests performed with three example systems are presented: a 6-bus system and two real-life systems representing the Southern and the Northern-Northeastern parts of the Brazilian network.

OVERVIEW OF THE APPROACH

Figure 1 illustrates the basic features of the genetic algorithm. For simplicity, a binary representation is used (alternative, more efficient representations are discussed in the next section). Part (a) of the figure shows two configurations of the current population. A six digit binary number is used to represent line additions: two digits per branch. For instance, in the first configuration of (a) branch 1-2 has two new lines added (binary 10), whereas in the second configuration we have added three additions in branch 1-2 (binary 11). Part (a) of the figure also indicates a crossover point. By the crossover operation, the left side of the first configuration is attached to the right side of the second configuration, and vice-versa, resulting in the two new configurations (offsprings) given in part (b) of Figure 1. Part (b) of the figure indicates where mutation occurs: by mutation a single bit is changed (0 to 1 or 1 to 0). The resulting configurations are given in part (c).

COMBINATORIAL OPTIMIZATION BY GENETIC ALGORITHM

Codification of the Problem

Objective Function:

The objective function has two components: investment costs and penalties associated with the loss of load. Investment costs are functions of the decision variables (integer variables representing the addition of new transmission equipment); penalties are functions of the continuous operation variables (power flows). In genetic algorithms, the objective function represents the quality of a configuration (the value associated to an individual of a population). Higher quality configurations have better chances to generate offsprings. Conventional genetic algorithms are formulated as maximization problems. Minimization problems are transformed in equivalent maximization problems. This is a critical step since the selectivity of the genetic algorithm may be affected by a bad choice of the transformed objective function. Two alternative transformations have been implemented and tested. The first one is similar to the one proposed in [7]:

$$min\ v \iff max\ [K - v]$$

where K is a parameter which can be adjusted during the solution process. When the algorithm is initialized with a population of configurations obtained with other algorithms, e.g. Garver's, K can be kept constant: that is, K is set at a value 20% higher than the cost of a typical configuration of the initial population. The second type of transformation is proposed in [6]:

$$min\ v \iff max\ [1/v]$$

Tests performed show that both alternatives yield satisfactory results.

Codification:

Three alternative codification approaches were studied and compared: (a) binary codification, (b) codification by independent bits, and c) decimal codification. Figure 2 shows a network branch (branch k - m) at which up to seven new lines can be added. Figure 3 illustrates the three alternative codification approaches for a case where, in the current configuration, three lines have been added to branch k - m. Although the binary representation is normally favored in conventional genetic algorithm applications, there is at least one good reason for not using it in the transmission expansion planning problem: mutation and crossover can generate offsprings which are too different from their parent configurations which sometimes creates chaotic behavior in the genetic algorithm.

This is illustrated in Figure 4. Situation before crossover: a configuration with three lines added to branch k - m and a second one with four branches added. Situation after crossover: a configuration with seven lines added to branch k - m and a second one with zero additions. Situation after mutation: a configuration with seven lines added to branch k - m and a second one with two additions. One related difficulty associated with the binary coding is known as Hamming cliffs: integer numbers 3 and 4 are represented by 011 and 100, respectively; which means that moving from 3 to 4 requires changing all the three digits (a big difference in terms of Hamming distance). On the other hand, when the binary representation is used, a single mutation would change 000 into 100, meaning

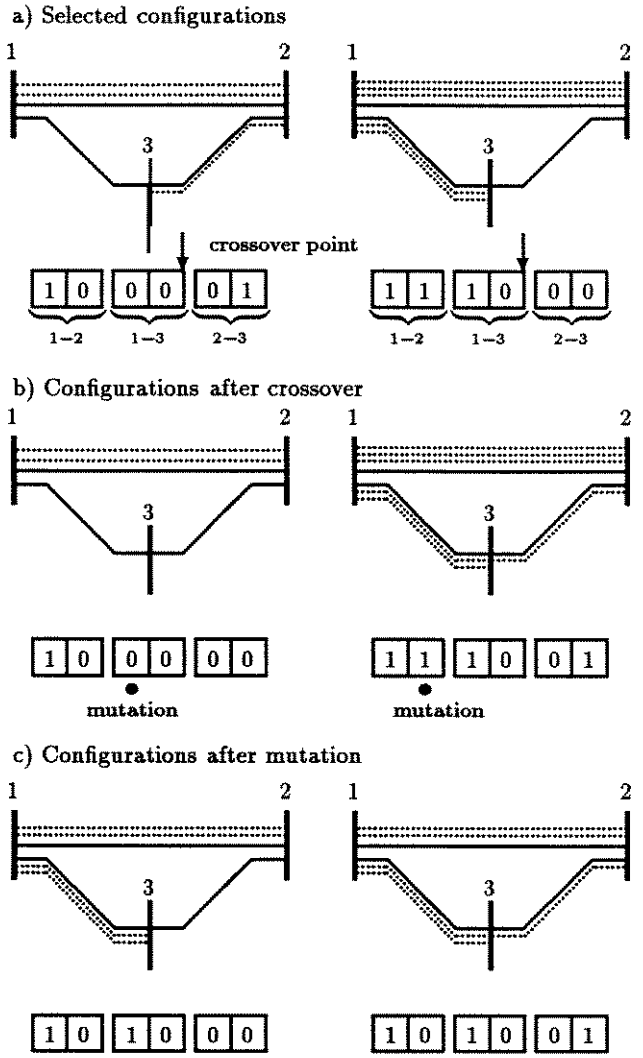


Figure 1: Example of crossover and mutation using binary representation

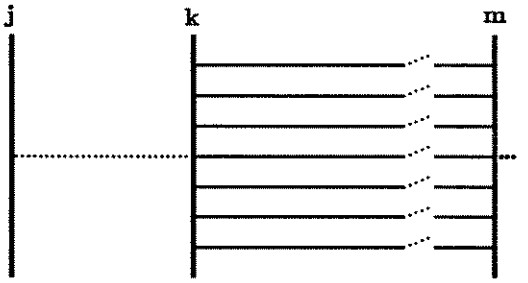


Figure 2: Example of codification

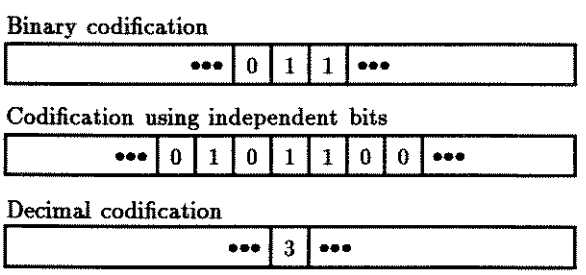


Figure 3: Types of codification

the simultaneous addition of 4 circuits! Thus, although, the binary representation (including the representation by independent bits) provides a natural way for implementing the mechanisms of mutation and crossover, the decimal representation is favored for the application we have in mind.

The decimal representation was finally adopted for it provides a more uniform behavior of the genetic algorithm. For instance, in this representation, mutation increases or decreases by one the number of additions in a given branch. And the same is true regarding crossover, in which case we always have the correct transfer of information from parent to child configurations. To compensate for the conservativeness of this representation, the mutation rate has to be increased accordingly.

Selection-Crossover-Mutation Cycle

From the current generation a new generation is produced by three basic mechanisms: selection, crossover and mutation. Selection models the concept of survival of the fittest in the natural world. Based on the value of the objective function, each configuration of a population is allowed a certain number of offsprings. In the simplest selection criterion, each configuration is allowed to participate in reproduction in proportion to its objective function. Thus configurations with high values of objective function will have their genes in a larger number of offsprings.

There are three basic types of crossover: single-point crossover, two-point crossover, and multipoint crossover. In the first case (single-point crossover), the crossover point is randomly chosen: if L is the number of elements in a configuration, a random number Lp (crossover point) is selected between 1 and L-1. Then the two configurations selected for crossover swap their first Lp positions producing the configurations given in fig.1, part (b). Two-point crossover is a generalization of the previous case in which the swapped string is between two points randomly chosen in the string. In the multipoint (or continuous) crossover each binary element of the two matching configurations is independently tested. The crossover rate, ρ_c, is a key parameter in defining a genetic algorithm: once two configurations are selected for crossover, a random number is drafted from the [0-1] interval; if this number is smaller then ρ_c, then crossover is performed; otherwise the two selected configurations pass unchanged to the next generation.

The last operation in a cycle is mutation. When a binary representation is used, mutation simply means changing a gene (an element of a string representing a configuration) from 0 to 1 and vice versa. The mutation rate is controlled by a parameter ρ_m. In conventional genetic algorithms mutation restores part of the variety of genetic material which is destroyed by selection and crossover operations (we can say that while selection and crossover tend to homogenize the population based on the survival of the fittest principle, mutation recovers part of the genes which disappear in the process and which may be essential for avoiding local optimal solutions). Practical experience has shown that in the transmission expansion planning application, the rate of mutation has to be larger than the ones reported in the literature for other applications of the genetic algorithm.

Implementation Details

Initial Population:

Two alternative approaches have been used for generating the initial population: (a) controlled random generation; and (b) modified sub-optimal algorithms such as Garver's method with a heuristic for generating a family of alternative feasible networks. In a well posed transmission expansion problem, normally 70 to 90% of the decision variables are equal to zero at the optimal solution point; this information is used for controlling the random initialization procedure. Garver's approach will be discussed in the next section.

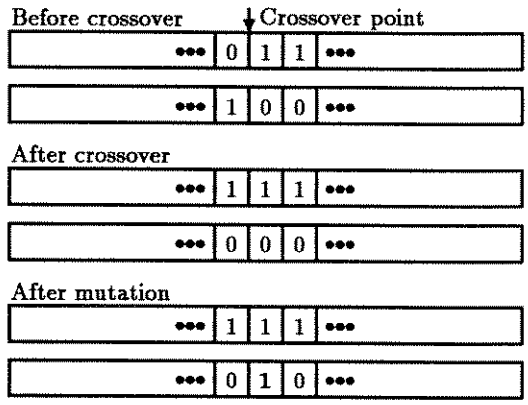


Figure 4: Crossover and mutation

Control Parameters:

Typical values for the control parameters reported in the literature are: population size between 30 and 200; crossover rate ρ_c between 0.5 and 1.0; mutation rate in the range 0.001 and 0.050. There is no agreement, however, on the optimal way of tuning such parameters and a particular choice of parameter values is highly dependent on the application. Moreover, these parameters are closely related and the value chosen for one of them may affect the optimal values of the others. In this research, parameter values have been determined in a number of tests with smaller networks and this information was used for extrapolating parameters values for larger networks (more details in the test results section).

Stopping Criterion:

The process stops whenever the incumbent solution (least cost configuration) does not improve after a specified number of generations (typically from 5 to 100 generations).

Extended Genetic Algorithm

In addition to the modifications described in the previous section, some extensions were required in order to develop a genetic algorithm able to solve large scale practical cases, as described in the following.

Variable Mutation Rate:

Mutation is the key feature used by a genetic algorithm for avoiding local minima (premature convergence). The need for mutation increases in the final phases of the process when populations tend to be more homogeneous and dominated by the more efficient genes. Mutation introduces variety which in turn allows exploring new zones in the optimization landscape.

Mutation Rate Controlled by Simulated Annealing:

In [6] it is suggested that the mutation rate can be controlled using the Boltzmann distribution in the same way it is done in simulated annealing algorithms,

$$P(\Delta v) = exp(-\Delta v/T)$$

where Δv is the change in the objective function produced by the mutation being considered. Given a configuration and a candidate mutation, Δv is determined first. If $\Delta v < 0$ the proposed mutation is accepted. Otherwise, i.e., when the proposed mutation increases the objective function, it will be accepted with a certain probability $Pr(\Delta v) > P(0,1)$, where $Pr(.)$ is a random number in the range [0,1]. This feature (the possibility of increasing the objective function) will help avoiding local minima. This type of strategy leads to improved performance of the genetic algorithm applied to the transmission expansion planning problem and has also been reported in [6,7].

Limited Selection:

Rates of reproduction based on the value of the objective function (the proportional criterion described above) may lead to premature convergence (*local minima*). During the initial phase, generations will present configurations with poor objective functions as a rule. If it happens that by chance one or a few configurations have relatively better objectives, these configurations will tend to reproduce at a much higher rate than the others, and this may cause convergence to occur at a poor local minimum. With this in mind, a variety of safeguards have been suggested in the literature [4]. An efficient safeguard consists in limiting the number of offsprings of a certain configuration to no more than, say, 2 or 3% of the total population. Although limited selection is normally not activated when dealing with smaller networks, it may become critical for larger practical systems such as the ones studied in the test results section.

Building Blocks and Unconnected Networks:

The ability of genetic algorithms to find optimal or near-optimal solutions is closely associated with the creation and manipulation of so-called ***building blocks*** as part of the configurations of a population. In order to clarify this property we need the concept of ***schemata*** which simply is a set of strings presenting the same bits (genes) in a given set of positions, i.e., a schemata is a family of configurations sharing a set common genes. For example, the schemata $**000$ represents the subset of all 5-digit binary numbers which share three digits in common; thus the following strings belong to such schemata {00000, 01000, 10000, 11000}.

The average fitness of a schemata is the average of the values of the objective functions associated with the configurations belonging to the schemata. The ***defining length*** of a schemata is the distance between the genes at the two extremes of the invariant part of the strings of a schemata. For example, schemata $**011*$ has defining length equal to 2 and schemata $1*0**1$ has defining length equal to 5.

A ***building block*** is a ***schemata*** containing configurations with high fitness values and at the same time small defining lengths. The importance of high fitness values is self explanatory; the interest in small defining lengths, on the other hand, is linked to the fact that schematas with small defining lengths are less likely to be destroyed by crossover operations, i.e., their genes will have greater chances to pass to the next generations. In the current research, in addition to the more conventional use of building blocks, we have used this concept to implement ***transmission paths*** which are subsets of transmission lines and transformers whose addition only makes economical sense if made together (Ref. [13] presents some practical hints on the importance of transmission paths). This feature of the algorithm is particularly useful in dealing

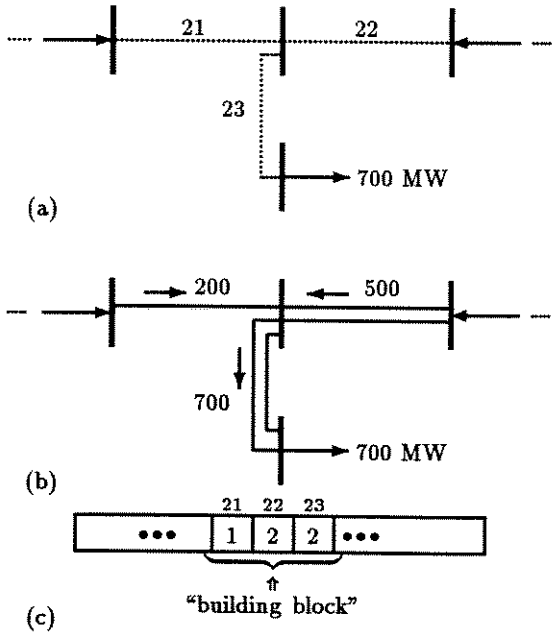


Figure 5: Building blocks

with new generation/load sites, as illustrated in fig. 5.

Determination of the Initial Population:

There are several ways of determining the initial population for a genetic algorithm. In this paper we have used a simple yet very efficient initialization procedure based on the well known Garver's algorithm [10]. The transmission network is modeled by a transportation model (Kirchhoff Voltage Law, KVL, is not represented in the model, and in this sense, although faster, it is less accurate than the DC power flow model). Garver's algorithm have been modified in such a way that radically different configurations are generated, which means that a wide variety of genetic material is produced in this initial stage of the optimization process. The algorithm used to generate the initial population is summarized below:

1. Determine an optimal expansion plan using continuous decision variables.
2. Add the integer part of all lines/transformers added in step 1 and form the *basic configuration.*
3. Using the basic configuration, list the NL lines/transformers with fractional investments and carrying the largest power flows. (A configuration will be obtained from each element of this list).
4. For each one of the NL elements in the list, do:
   (a) Add the integer line/transformer in the list to the basic configuration thus creating a *current configuration.*
   (b) Test feasibility of the current configuration. If feasible, add the current configuration to the initial population. Otherwise, proceed to 4.c.
   (c) Obtain a new current configuration by adding to the current configuration the line/transformer corresponding to the branch carrying the largest power flow, and go to 4.b

For example, $NL = 20$ has been used in the tests performed with the Northern-Notheastern Brazilian network.

The Penalty Parameter α:

Parameter α is used to penalize the loss of load in the objective function. As a general rule, the value of α is such that loss of load is less attractive than investments in new transmission equipment. Towards the end of the optimization process, the value of α should be high enough to discourage any type of loss of load. During the initial stages (initial populations), however, allowing some degree of loss of load is very effective in the sense that it makes it easier for the genetic algorithm to move through the search space. Thus, in order to keep a larger degree of heterogeneity during the initial phases of the simulated evolution process, it is important to allow the survival of some configurations with a certain degree of loss of load. A similar approach has been applied with success to the short-term generation scheduling problem [6].

THE PROPOSED ALGORITHM

Based on the ideas discussed above, the following algorithm has been developed for solving the transmission network expansion planning problem:

Step 1: Define k_{max}, the stopping criterion, k_{stop}, maximum number of generations, T_p, the population size and the members of the initial population.

Step 2: Determine the fitness (value of the objective function) of each configuration of the current population. Determine the incumbent configuration (string) as the configuration with the highest fitness. Update counter k_2 which indicates the number of generations, and the counter k_1 which indicates the number the generations during which the incumbent configuration remains unchanged.

Step 3: If $k_1 > k_{max}$, stop, the algorithm converged. If $k_2 > k_{stop}$, stop, the maximum number of generations have been exceeded. If neither of the two happens, go to step 4.

Step 4: Determine the number of offsprings allocated to each configuration using the proportional criterion with limited selection.

Step 5: Perform crossover between each pair of selected configurations until the number of offsprings reach the specified population size. Parameter ρ_c, the crossover rate, statistically controls the number of configurations taking part in the crossover process.

Step 6: Perform mutation: for each candidate configuration it is carried out up to n_m mutations (single or double), where n_m is proportional to the mutation rate, ρ_m. Go to step 2.

TEST RESULTS

Garver's 6-bus system without generation rescheduling

All the relevant data for this system can be found in [10]. The following parameters have been used: penalty factor $\alpha = 10$, crossover rate $\rho_c = 0.9$, variable mutation rate $\rho_m = [0.04, 0.08]$ per bit, $k_{max} = 8$ (i.e., process stops after 8 consecutive generations without improvement in the incumbent solution). The initial population has been generated randomly (this normally gives good results for small systems such as with Garver's 6-bus system). With population varying from 16 to 24, configurations convergence to the optimal solution was typically obtained after 25 to 35 generations (the initial population was generated randomly; the convergence rate depends on the average fitness of the initial generation). The optimal solution known for this case is: cost of $v = 200$; and additions $n_{26} = 4$ circuits, $n_{35} = 1$ circuit, and $n_{46} = 2$ circuits.

Garver's 6-bus system with generation rescheduling

With the same paramenters as above, the algorithm took 15% less iterations to converge than in the previous case where rescheduling was not allowed. In this case the optimal solution is: cost of $v = 110$; and additions, $n_{35} = 1$ circuits and $n_{46} = 3$ circuits.

Southeastern system without generation rescheduling

All the relevant data about this system can be found in [11]. The following parameters have been used: $\alpha = 400$, $\rho_c = 0.9$, variable mutation rate with $\rho_m = [0.02, 0.04]$ per bit, $k_{max} = 30$. The initial population was generated randomly. With population varying from 50 to 60 configurations convergence to the optimal solution was typically obtained after 150 to 180 generations. The number of generations depends on the average fitness of the initial population (generated randomly); for good starting points, it was observed that the optimal solution can be reached in far less generations: i.e., 38 generations for finding the optimal solution for the first time in one of the test cases, plus 30 generations for assuring convergence). The optimal solution known for this case is: cost of US154,420,000.00; and additions $n_{20-21} = 1$, $n_{42-43} = 2$, $n_{46-06} = 1$, $n_{19-25} = 1$, $n_{31-32} = 1$, $n_{28-30} = 1$, $n_{26-29} = 3$, $n_{24-25} = 2$, $n_{29-30} = 2$ and $n_{05-06} = 2$. In this case, 4 circuit additions are permitted in each one of the 79 candidate right-of-ways which means that the number of possible configurations is $(4+1)^{79} = 2^{183}!$. It is worth mentioning that although the genetic algorithm explores only a small fraction of the solution space still it is able to find the optimal solution. As observed with the Garver's example system, when generation rescheduling is allowed the number of generations required to reach the optimal solution is reduced by about 15% (to 140 generations on average).

North-Northeastern system without generation rescheduling

The third test system is given in fig.6 and represents the part of the Brazilian network corresponding to the North and Northeastern regions; the system has 89 buses and 183 candidate branches (the complete data set for this test system is available from the authors). Total demand is 29,754 MW and there is no known optimal solution for this system. The following parameters have been used: $\alpha = 1000$, $\rho_c = 0.9$, variable mutation rate with $\rho_m = [0.01, 0.03]$ per bit, $k_{max} = 80$ and a population size of 160 configurations. Convergence was typically obtained after 350 to 500 generations. The final solution depends on the initial population. The best solution found using the modified Garver's approach for generating the initial population presents investment costs of US$ 2,600,593,000.00 (the best known solution for this example system) and the following additions (Typically converged solutions have costs in the range US$ 2,600,593,000.00 to US$ 2,628,773,000.00):

$n_{01-02} = 1$, $n_{02-04} = 1$, $n_{04-05} = 4$, $n_{04-81} = 3$, $n_{05-56} = 1$, $n_{05-58} = 4$, $n_{06-75} = 1$, $n_{12-15} = 1$, $n_{13-15} = 4$, $n_{14-59} = 1$, $n_{15-16} = 3$, $n_{15-46} = 2$, $n_{16-44} = 6$, $n_{18-50} = 11$, $n_{18-74} = 5$, $n_{21-57} = 2$, $n_{22-58} = 2$, $n_{24-43} = 1$, $n_{25-55} = 4$, $n_{30-31} = 1$, $n_{30-63} = 2$, $n_{35-51} = 2$, $n_{36-46} = 2$, $n_{39-86} = 1$, $n_{40-45} = 1$, $n_{40-46} = 1$, $n_{41-64} = 2$, $n_{42-44} = 1$, $n_{43-55} = 2$, $n_{43-58} = 3$, $n_{48-49} = 1$, $n_{49-50} = 4$, $n_{52-59} = 1$, $n_{53-86} = 1$, $n_{54-58} = 1$, $n_{54-63} = 1$, $n_{56-57} = 1$, $n_{61-85} = 3$, $n_{61-86} = 1$, $n_{63-64} = 1$, $n_{67-68} = 1$, $n_{67-69} = 1$, $n_{67-71} = 3$, $n_{69-87} = 1$, $n_{71-72} = 1$,

$n_{72-73} = 1$, $n_{73-74} = 2$, $n_{73-75} = 1$, $n_{75-81} = 1$.

CONCLUSIONS

Genetic algorithms are optimization techniques based on a well known process found in nature and which has been successfully applied to a number of electrical engineering problems. We have successfully extended and applied the approach to large scale transmission network expansion planning problems. The new methodology has proved to be more efficient than other methods tried before such as optimization techniques and simulated annealing. More specifically, for the North-Northeastern Brazilian network, for which optimal solutions are unknown, a family of interesting solutions have been obtained with costs about 8.8% less than the best solution obtained by conventional optimization, 2.84% less than the best solution obtained by sequential simulated annealing, and 1.13% less than the best solution obtained by parallel simulated annealing.

The integrality constraints (there is no fractional transmission lines) make the transmission expansion planning problem a very complex mixed integer nonlinear programming problem. Previous papers by the authors have shown that mathematical programming methods based on Benders decomposition and binary search are effective in obtaining optimal solutions for small and medium size problems. Still another study by the authors has shown that simulated annealing was able to provide solutions at costs that had never been obtained before with any other method. This was a major motivation for continuing investigating the family of methods based on natural processes and the present paper on genetic algorithms (which in many senses generalizes the simulated annealing approach) is the result of that effort.

REFERENCES

[1] *H. Rudnick, R, Palma, E. Cura and C. Silva*: "Economically Adapted Transmission systems in Open Access Schemes – Application of Genetic Algorithms", IEEE PES Summer Meeting, July 1995.

[2] *R. Fischl*: "Optimal System Expansion Planning: A Critical Overview", ERDA & EPRI Conf., Henniker, 1975.

[3] *M. Srinivas, L.M. Patnaik*: "Genetic Algorithms", IEEE Computer, 1994.

[4] *D.E. Goldberg*: "Genetics Algorithms in Search, Optimization and Machine Learning", Addison Wesley, Reading, Mass., 1989.

[5] *S.A. Kazarlis, A.G. Bakirtzis, V. Petridis*: "A Genetic Algorithm Solution to the Unit Commitment Problem", IEEE Transactions on Power Systems, Vol. 11, No. 1, February 1996.

[6] *K.P. Wong, Y.W. Wong*: "Combined Genetic Algorithms/Simulated Annealing/Fuzzy Set Approach to Short-Term Generation Scheduling with Take-or-Pay Fuel Contract", IEEE Transactions on Power Systems, Vol. 11, No. 1, february 1996.

[7] *C.C. Kuo, H.C. Chang*: "Applying a Refined Genetic Algorithm to Network Reconfiguration for Loss Reduction", IEE of Japan Power & Energy, 1994.

[8] *R. Romero, R.A. Gallego, A. Monticelli*: "Transmission System Expansion Planning by Simulated Annealing", IEEE Transactions on Power Systems, Vol. 11, No. 1, February 1996.

[9] *R.A. Gallego, A.B. Alves, A. Monticelli, R. Romero*: "Parallel Simulated Annealing Applied to Long Term Transmission Network Expansion Planning", IEEE PES Winter Meeting, Baltimore, Maryland, January 1996.

[10] *L.L. Garver*: "Transmission Network Estimation Using Linear Programming", IEEE Trans. Power App. Syst., Vol. PAS-89, pp. 1688-1697, Sept/Oct 1970.

[11] *M.V.F. Pereira, L.M.V.G. Pinto, S.H.F. Cunha, G.C. Oliveira*: "A Decomposition Approach to Automated Generation/Transmission Expansion Planning", IEEE Trans. Power App. Systems, Vol. PAS-104, No. 11, November 1985.

[12] *R. Romero and A. Monticelli*: "A Zero-One Implicit Enumeration Method for Optimizing Investments in Transmission Expansion Planning", IEEE Transactions on Power Systems, Vol. 9, No. 3, August 1994.

[13] *A. Monticelli, A. Santos Jr., M.V.F. Pereira, S. Cunha, J.G. Praça, B. Park*: "Interactive Transmission Network Planning Using a Least-Effort Criterion", IEEE Transactions PAS, Vol. 101, No. 10, pp 3919-3925, October 1982.

ACKNOWLEDGEMENTS

This project was partly supported by FAPESP, Fundação de Amparo à Pesquisa do Estado de São Paulo, and by CNPq, Conselho Nacional de Desenvolvimento Científico e Tecnológico. R.A. Gallego is on leave from Universidad Tecnologica de Pereira, Colombia, and gratefully acknowledges the support of Colombian Ministry of Education.

APPENDIX: MATHEMATICAL FORMULATION

In the following, the static transmission expansion planning problem is formulated as an mixed integer nonlinear programming problem in which the power network is represented by a dc power flow model.

$$
\begin{aligned}
\min \quad v = & \sum_{ij} c_{ij} n_{ij} + \sum_{i} \alpha_i r_i \qquad (1)\\
& B(x+\gamma^{\circ})\theta + g + r = d \\
& (x_{ij} + \gamma^{\circ}_{ij})\,|\theta_i - \theta_j| \le (x_{ij} + \gamma^{\circ}_{ij})\overline{\phi}_{ij} \\
& 0 \le g \le \overline{g} \\
& 0 \le r \le d \\
& 0 \le n_{ij} \le \overline{n}_{ij}
\end{aligned}
$$

where:

- c_{ij} - Cost of the addition of a circuit in branch i-j.
- x_{ij} - Total susceptance in branch 1-j.
- $B(.)$ - Susceptance matrix.
- θ - Vector of nodal voltage angles.
- γ° - Vector of initial susceptances, whose elements are γ°_{ij}, i.e., the summation of the susceptances in branch i-j at the beginning of the optimization.
- n_{ij} - Number of circuits added in branch i-j: $n_{ij} = x_{ij}/\gamma_{ij}$; where γ_{ij} is the susceptance of the new circuits.
- $\overline{\phi}_{ij}$ - Defined as the ratio: $\overline{\phi}_{ij} = \overline{f}_{ij}/\gamma_{ij}$; where $\overline{f}_{ij}$ is the maximum flow in a circuit i-j.
- d - Vector of liquid demand.
- g - Generation vector.
- $\overline{g}$ - Vector of maximum generation capacity.

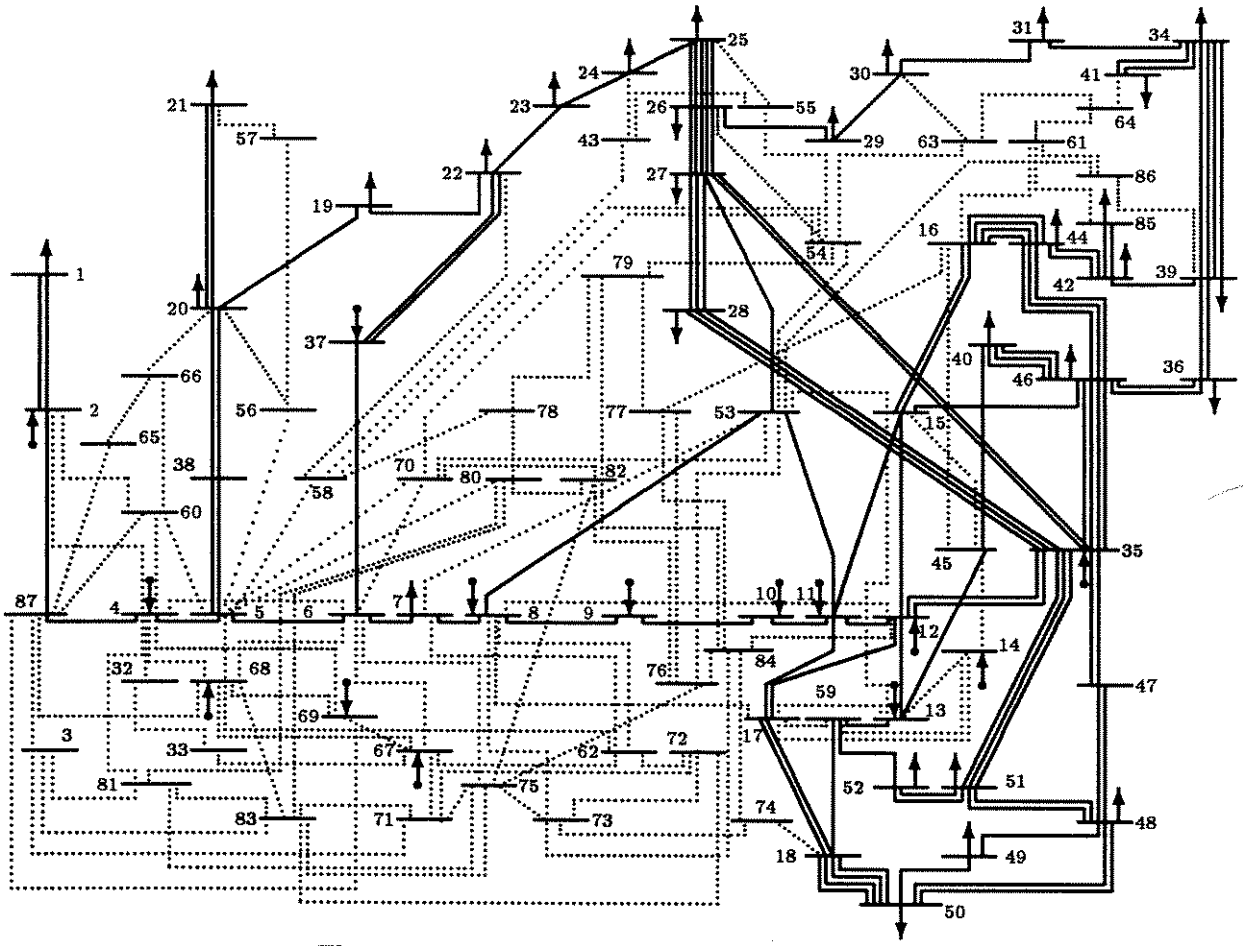


Figure 6: North-Notheastern Network.

r - Vector of artificial generations.

α - Penalty parameter associated with loss of load caused by lack of transmission capacity.

Problem 1 is a mixed integer nonlinear problem. It is a typical hard combinatorial problem prone to combinatorial explosion as the number of decision variables increases. An extra complication regards the fact that there are cases in which planning does not simply mean the reinforcement of an existing network. Sometimes we have to start from scratch, at least for parts of the network due to the addition of new load, transition and generation buses (Please refer to fig.1). In the case of the Brazilian example systems, the simple addition of one or two circuits (lines or transformers) may not be enough to guarantee network connectivity: situations are not uncommon in which entire paths have to be built to put all the pieces together in a single network. In these cases, the combinatorial burden is even heavier than it would be in a simpler, reinforcement only, type of problems.

For a given set of x_{ij}, which may form a tentative solution, Problem 1 becomes a linear programming problem, Problem 2, which is solved testing the adequacy of a candidate solution; adequacy is indicated by zero loss of load. Notice that Problem 1 is always feasible due to the presence of the loss of load factor $\sum_i \alpha_i r_i$ in the objective function; thus whenever a tentative solution set x_{ij} is inadequate, feasibility is achieved by the use of artificial generators (loss of load). It has been observed that this feature facilitates the evolution of the process in order to come out from local minima, by temporarily moving through regions of inadequate solutions, but still keeping the feasibility of the mathematical problem.

$$\begin{aligned} \min \quad w = & \sum_i \alpha_i r_i \qquad (2) \\ & B(x^k + \gamma^\circ)\theta + g + r = d \\ & (x^k_{ij} + \gamma^\circ_{ij})\,|\theta_i - \theta_j| \le (x^k_{ij} + \gamma^\circ_{ij})\overline{\phi}_{ij} \\ & 0 \le g \le \overline{g} \\ & 0 \le r \le d \end{aligned}$$

BIOGRAPHIES

Ramon A. Gallego received his B.S. degree in Electrical Engineering in 1981 from Universidad Tecnologica de Pereira and his M.S. degree in 1985 from Universidad Nacional de Colombia. Currently he is working towards his Ph.D. degree at Unicamp. (e-mail address: *ramon@dsee.fee.unicamp.br)*

Alcir J. Monticelli (Senior Member, IEEE) received his Ph.D. degree from Unicamp in 1975 where he is currently a Professor of Electrical Engineering. (e-mail address: *alcir@dsee.fee.unicamp.br).*

Rubén Romero received his B.S. degree in 1978 and P.E. degree in 1984 from the National University of Engineering, in Lima, Peru, his M.S. and Ph.D. degrees from Unicamp in 1990 and 1993. Currently he is a Professor of Electrical Engineering at Feis-Unesp-Ilha Solteira. (e-mail address: *ruben@isldee.uesp.ansp.br)*