UMA NOVA METAHEURÍSTICA PARA PROBLEMAS COMBINATÓRIOS APLICADA AO PLANEJAMENTO DA EXPANSÃO DE SISTEMAS DE TRANSMISSÃO DE ENERGIA ELÉTRICA

Haroldo de Faria Junior

TESE SUBMETIDA AO CORPO DOCENTE DA COORDENAÇÃO DOS PROGRAMAS DE PÓS-GRADUAÇÃO DE ENGENHARIA DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE DOUTOR EM CIÊNCIAS EM ENGENHARIA ELÉTRICA.

Aprovada por:

Prof. Djalma Mosqueira Falcão, Ph.D.

Dr. Silvio Binato, D.Sc.

Prof. Alexandre Pinto Alves da Silva, Ph.D.

Prof. Gerson Couto de Oliveira, D.Sc.

Prof. Rubén Augusto Romero Lázaro, D.Sc.

Prof. Felipe Martins Müller, D.Sc.

RIO DE JANEIRO, RJ - BRASIL
SETEMBRO DE 2005

DE FARIA JUNIOR, HAROLDO

Uma Nova Metaheurística para Problemas Combinatórios Aplicada ao Planejamento da Expansão de Sistemas de Transmissão de Energia Elétrica [Rio de Janeiro] 2005

XVIII, 154 p. 29,7 cm (COPPE/UFRJ, D.Sc., Engenharia Elétrica, 2005)

Tese – Universidade Federal do Rio de Janeiro, COPPE

1. Planejamento da Expansão de Sistemas de Transmissão de Potência
2. Problemas de Escalonamento de Tarefas
3. Métodos Metaheurísticos
4. Otimização Discreta
5. Algoritmos Paralelos

I. COPPE/UFRJ II. Título (série)

A meus pais, Haroldo e Vanda.

Agradecimentos

A todos os amigos e colegas que, de alguma forma, contribuiram para a realização deste trabalho de tese e especialmente,

– ao meu amigo e orientador Silvio Binato pela dedicação, incentivo e paciência na orientação prestada a mim durante o desenvolvimento deste trabalho;

– ao professor e orientador Djalma Mosqueira Falcão pelas sugestões, orientações e incentivos dispensados desde a época de mestrado;

– aos pesquisadores Maurício G. C. Resende e Renata Machado Aiex pelas discussões e esclarecimentos de minhas dúvidas a respeito da metaheurística GRASP;

– ao professor José Luiz Rezende Pereira pelo incentivo e apoio dispensado desde a graduação;

– ao CEPEL, que disponibilizou os recursos necessários à realização desta tese;

– ao Núcleo de Atendimento em Computação de Alto Desempenho (NACAD) da COPPE/UFRJ, que disponibilizou os recursos computacionais necessários para a conclusão desta tese;

– à secretaria do departamento de Engenharia Elétrica da Coppe/UFRJ, em especial à Solange que sempre me ajudou em questões burocráticas;

– aos amigos Erick, Camilo, Itaniel, João Alberto, Varrichio, Sérgio Gomes, Ricardo, Júlio, Alex, Javier, entre muitos outros, pela amizade dispensada durante todos estes anos e a amiga Mirela;

– à minha família, em especial minha mãe Vanda, meu pai Haroldo e meu irmão Hugo.

Resumo da Tese apresentada à COPPE/UFRJ como parte dos requisitos necessários para a obtenção do grau de Doutor em Ciências (D.Sc.)

UMA NOVA METAHEURÍSTICA PARA PROBLEMAS COMBINATÓRIOS APLICADA AO PLANEJAMENTO DA EXPANSÃO DE SISTEMAS DE TRANSMISSÃO DE ENERGIA ELÉTRICA.

Haroldo de Faria Junior

Setembro/2005

Orientadores: Djalma Mosqueira Falcão
Silvio Binato

Programa: Engenharia Elétrica

Esta tese apresenta uma nova metaheurística para solução de problemas de otimização combinatória denominado de GRAPR (Greedy Randomized Adaptive Path Relinking). Este novo método herda suas características das metaheurísticas GRASP (Greedy Randomized Adaptive Search Procedure) e Path Relinking, que são muito eficientes na resolução de problemas combinatórios. O método é aplicado ao importante problema de Planejamento da Expansão de Sistemas de Transmissão de Energia Elétrica e também a um problema clássico de otimização denominado de Escalonamento de Tarefas, conhecido na literatura como Job Shop Scheduling Problem (JSP). Uma investigação experimental da distribuição de probabilidade do tempo de solução em metaheurísticas do tipo GRASP é realizada. A paralelização do método proposto utilizando a biblioteca de passagem de mensagens MPI é feita na tentativa de se reduzir o tempo computacional extra oriundo da implementação do GRAPR.

Abstract of Thesis presented to COPPE/UFRJ as a partial fulfillment of the requirements for the degree of Doctor of Science (D.Sc.)

A NEW METAHEURISTIC FOR COMBINATORIAL OPTIMIZATION PROBLEMS APPLIED TO TRANSMISSION NETWORK EXPANSION PLANNING

Haroldo de Faria Junior

September/2005

Advisors: Djalma Mosqueira Falcão
Silvio Binato

Department: Electrical Engineering

In this thesis we present a new metaheuristic to solve combinatorial optimization problems called GRAPR (Greedy Randomized Adaptive Path Relinking). This new approach inherits its characteristics from the metaheuristics GRASP (Greedy Randomized Search Procedure) and Path Relinking, which are very efficient at the solution of combinatorial problems. The method is applied to the important problem of Transmission Network Expansion Planning and to a classical optimization problem called Job Shop Scheduling Problem (JSP). An experimental investigation of the probability distribution of solution time in GRASP type metaheuristics is carried out. Parallelization of the proposed method is accomplished through the use of the message passing interface library MPI, in an attempt to reduce the extra computational time required by the implementation of GRAPR.

Conteúdo

Lista de Figuras

Lista de Tabelas

Notação

Nesta proposta de tese utilizaremos a seguinte notação:

A, B ...	letras maiúsculas para representar matrizes.
x, y ...	letras minúsculas para representar vetores (coluna).
A^t, x^t	representa o operador de transposição, aplicado à matrizes ou a vetores.
$[x]$	representa uma matriz diagonal cuja diagonal é o vetor x.
I	representa uma matriz identidade na dimensão apropriada.
e	é um vetor cujas componentes são unitárias.
$\mathcal{C}$, $\mathcal{E}$ etc.	letras "caligráficas" são utilizadas para representar conjuntos.
$\bar{\mathcal{C}}$	representa o conjunto complemento do conjunto $\mathcal{C}$.
$\|\mathcal{C}\|$	representa a cardinalidade do conjunto $\mathcal{C}$.

CAPÍTULO 1

INTRODUÇÃO

Muitos problemas de importância tanto teórica quanto prática requerem a obtenção de um conjunto de parâmetros ou uma configuração ótima para atingir um determinado objetivo. São denominados de problemas de programação matemática e, nas últimas décadas, várias classes desses problemas surgiram juntamente com uma coleção de técnicas para a sua solução. Um problema geral de programação matemática é um problema de otimização sujeito a restrições no $\mathbb{R}^n$ da forma:

$$Minimize\ f(x) \tag{1.1a}$$

$$sujeito\,a: \tag{1.1b}$$

$$g_i(x) \leq 0 \quad i = 1, \ldots, m \tag{1.1c}$$

$$x \in S \subset \mathbb{R}^n. \tag{1.1d}$$

onde f e g_i são funções do parâmetro $x \in \mathbb{R}^n$. A resolução de tal problema compreende encontrar o vetor x que seja factível, isto é, $g_i(x) \leq 0$ para $i = 1, \ldots, m$ e $x \in S$ e que minimize a função objetivo $f(x)$. As técnicas para solução desses problemas são quase sempre numéricas, baseadas em algoritmos.

A tabela 1.1 mostra a classificação dos vários tipos de problemas de otimização encontrados na prática de acordo com as propriedades da função objetivo f do problema, das restrições g_i e de acordo com a definição do conjunto S do $\mathbb{R}^n$ [65].

Os problemas de otimização a serem resolvidos nesta proposta de tese são classificados como problemas de programação linear inteira mista já que possuem variáveis inteiras

Tabela 1.1: Principais Classes de Problemas em Programação Matemática.

Função f	Funções g_i	Conjunto S	Terminologia usada
Arbitrária contínua e não linear		Contínuo, Compacto, $\subset \mathbb{R}$	Programação Matemática Contínua
Arbitrária não-linear (não necessariamente contínua)		Discreto (Ex.: conjunto de pontos com coordenadas inteiras dentro de um conjunto compacto)	Programação Matemática Discreta (se $S \subset Z^n$, programação não-linear inteira)
Arbitrária contínua e não-linear	$m = 0$	$S = \mathbb{R}^n$	Otimização não-linear contínua sem restrições
Arbitrária não-linear (não necessariamente contínua)	$m = 0$	$S = \mathbb{Z}^n$	Otimização não-linear inteira sem restrições
Arbitrária não-linear e convexa		$S \subset \mathbb{R}^n$ convexo	Programação matemática não-linear convexa
Linear		S subconjunto do $\mathbb{R}^n$ (ex. $\mathbb{R}^{n+}$)	Programação linear
Linear		$S \subset \mathbb{Z}^n$	Programação linear inteira

e reais em sua formulação. Problemas práticos, geralmente, são de grandes dimensões e métodos de solução exatos encontram dificuldades para a sua resolução. Também vale ressaltar que a complexidade para a resolução desses tipos de problemas por métodos de solução exatos cresce exponencialmente para aumentos lineares no tamanho do problema. Para o problema de escalonamento de tarefas (JSP), por exemplo, aqueles com dimensões 15×15 (15 tarefas e 15 máquinas) com 225 variáveis e 915 restrições são considerados estar além do alcance de métodos exatos [2].

As duas principais famílias de métodos exatos utilizados para a resolução de problemas de programação linear inteira são Branch and Bound (Enumeração Implícita) e de Planos Cortantes. O método de Decomposição de Benders (1962) nos permite reduzir os problemas de programação linear inteira mista em uma seqüência de problemas de programação linear inteira, onde todas as variáveis são inteiras e nos quais Enumeração Implícita e Planos Cortantes podem ser utilizados [65].

Os métodos tipo Branch and Bound são mais utilizados e permitiram a resolução de problemas difíceis de otimização combinatória como o do Caxeiro Viajante (Traveling Salesman Problem) e problemas de Escalonamento (Job Shop Scheduling). A eficiência

desses métodos depende de forma crítica na qualidade das restrições disponíveis em cada nó da árvore de busca do método [65].

Apesar de estudos indicarem melhoramentos na solução de problemas clássicos de otimização combinatória por métodos tipo Branch and Bound, isso é atribuído à tecnologia disponível e não à técnica utilizada [51]. Em geral, não podem ser aplicados a problemas de dimensão elevada e a sua execução depende de um conhecimento especializado do domínio do problema em estudo.

Heurísticas (da antiga palavra grega "heuriskein", que significa descobrir, inventar, ter uma idéia), para otimização, são métodos aproximados de busca. É um conjunto bem definido de passos para se identificar, rapidamente, uma solução viável de boa qualidade para um problema. Esses métodos freqüentemente obtém a solução ótima, porém nenhuma garantia de convergência é fornecida na maior parte dos casos.

Metaheurísticas são algoritmos que descrevem como se explorar o espaço de busca sem se ater a um problema específico. Coordenam heurísticas mais simples, como busca local, com o objetivo de encontrar soluções de melhor qualidade do que as obtidas pelas heurísticas sozinhas. Alguns desses métodos podem ser modificados de forma a incluir conhecimentos específicos de um problema e se tornarem mais eficientes, mas, em geral, não usam informações específicas. Houve um grande esforço nessa área nos últimos anos, comprovado pelo grande número de artigos publicados. Entre as metaheurísticas mais conhecidas podemos citar Busca Tabu [43, 44], Algoritmos Genéticos [47, 45], Recozimento Simulado [39], GRASP [32, 25], entre outras. É fato estabelecido que esses métodos devem ser considerados complementares ao invés de competirem entre si por um melhor desempenho [51].

Recentemente, muito esforço tem sido concentrado no estudo de métodos híbridos para resolver problemas de otimização combinatória, já que uma única técnica, geralmente, não é capaz de obter soluções satisfatórias. Métodos híbridos são métodos que combinam dois ou mais métodos metaheurísticos, na tentativa de se tornarem mais eficientes. Podemos ter também combinação de metaheurísticas com métodos exatos. Atenção especial tem sido dada a algoritmos de natureza "míope", adequados à estrutura particular do problema, combinados com uma meta-estratégia que guia a busca para fora de ótimos locais.

Uma nova metaheurística para solução de problemas combinatórios é apresentada

nesta tese. Ela herda suas características de dois métodos utilizados em vários trabalhos e de eficácia comprovada na resolução de problemas de otimização de natureza combinatória. Esses métodos são o GRASP (Greedy Randomized Adaptive Search Procedure) e o Path Relinking, cujas características de exploração deram origem ao método híbrido apresentado, denominado de GRAPR (Greedy Randomized Adaptive Path Relinking). O método foi aplicado a dois tipos de problemas de otimização distintos. O primeiro é um problema de fundamental importância para o setor de planejamento da expansão de sistemas de energia elétrica, que é o planejamento da expansão de sistemas de transmissão de energia elétrica. Sistemas teste já utilizados em vários estudos na literatura são usados nos estudos.

O outro desafio é um problema clássico de otimização, que está entre os mais difíceis de serem solucionados eficientemente. Apesar de não ter ligação com a área de Engenharia Elétrica, foi utilizado nos estudos como uma forma de testar de forma mais efetiva a viabilidade de utilização do novo método. O problema é chamado de Escalonamento de Tarefas ou JSP (Job Shop Scheduling Problem) e consiste em se obter uma seqüência ótima de execução de certas operações em determinadas máquinas de forma que o tempo total de execução seja mínimo. Várias instâncias encontradas na literatura de otimização combinatória foram utilizadas nos estudos.

1.1 Revisão Bibliográfica

1.1.1 Problemas de Planejamento da Expansão de Sistemas de Transmissão de Energia Elétrica

Durante muito tempo, as únicas ferramentas disponíveis para a síntese de redes de transmissão eram os softwares de análise como os utilizados no cálculo de fluxo de carga, estudos de estabilidade, curto-circuito, etc. O planejador do sistema de energia elétrica era o responsável por determinar onde instalar novos equipamentos para suprir as novas cargas do sistema, resultando em uma configuração que deveria ser analisada através dos métodos mencionados anteriormente. Com o crescimento das dimensões das redes de transmissão, este procedimento se torna inviável. O trabalho pioneiro de Knight [56] teve o mérito de propor a distinção entre os métodos de análise e métodos matemáticos de projeto (síntese) de sistemas de transmissão de energia elétrica.

Um dos primeiros trabalhos propostos para a solução deste problema é datado de 1970 [41]. Garver formulou o problema como um problema de fluxo de potência e usou algoritmos de programação linear para identificar as rotas mais diretas entre os geradores e as cargas. Todos os candidatos a adição poderiam transportar potência mas eram penalizados para favorecer o fluxo nos circuitos existentes. As adições eram realizadas nos circuitos mais sobrecarregados e um novo fluxo de carga linearizado era computado.

Kaltenbatch et al. [54], também no ano de 1970, propuseram combinar programação linear com programação dinâmica. Programação linear era usada para encontrar o mínimo incremento da capacidade da rede para atender as variações de demanda e geração nas barras do sistema. Após essa etapa, programação dinâmica era utilizada para achar a melhor seqüência de investimentos (contínuos) para o período de planejamento. Este trabalho é o pioneiro para problemas de planejamento de expansão de redes de transmissão considerando múltiplos estágios.

Um algoritmo "puro" de programação dinâmica foi proposto por Dusonchet e El-Abiad em 1973 [28]. Esta proposta parecia contornar as dificuldades em obter a solução ótima dos trabalhos anteriores. Contudo, devido aos altos recursos computacionais requeridos, resultado do formalismo da programação dinâmica, simplificações ou relaxações de importantes restrições eram necessárias em aplicações práticas.

Tendo em vista as desvantagens da programação dinâmica foi proposto em 1973 por Gonzaga [48] um algoritmo de busca em grafos. Este algoritmo, uma versão de algoritmo $A*$, procura encontrar um caminho de custo mínimo em grafos de expansão utilizando heurísticas para reduzir o número de alternativas a serem analisadas. Com base em tal algoritmo, foi implementado um programa computacional chamado `Tânia`, que foi muito utilizado na solução de problemas de planejamento da expansão de redes com sistemas Brasileiros.

O conceito de rede adjunta combinada com o modelo de fluxo linearizado foi a proposta de Fischl e Puntel [33]. Este trabalho procurava pela variação contínua das susceptâncias dos circuitos que minimiza o custo de reforços na rede de transmissão. Posteriormente, um procedimento heurístico chamado método do vizinho mais próximo seria utilizado para obter os valores discretos das susceptâncias dos circuitos.

A primeira proposta de algoritmos do tipo "Branch-and-Bound" para este problema é devida a Lee et al. [60] em 1974. Contudo, assim como nos métodos de programação

dinâmica, a utilização de algoritmos combinatórios tipo “Branch-and-Bound” fica restrita a aplicações a sistemas de pequeno porte face aos recursos computacionais exigidos.

Em 1979, Monticelli et al. [66] propôs o uso de ferramentas interativas para o planejamento da transmissão. Para ordenar as possibilidades de adições era utilizado o índice de “Mínimo Esforço”, que consiste de uma análise de sensibilidade em relação as susceptâncias dos circuitos em um problema de otimização correlato cujo resultado é idêntico ao modelo de fluxo de carga linearizado.

O uso de análise de sensibilidade no problema de planejamento da rede de transmissão foi inicialmente proposta pelo trabalho de De Champs et al. [27]. Eles utilizaram análise de sensibilidade em relação as susceptâncias a partir de um problema de programação linear cujas restrições são as equações do modelo de fluxo de carga linearizado em conjunto com limites de transporte nos circuitos e de capacidade nos geradores. O objetivo do problema era obter o mínimo corte de carga necessário para eliminar todas as violações operacionais na rede elétrica. O uso de análise de sensibilidade também foi proposto por Pereira et al. [74].

Em 1981, Bennon et al. [10] utilizaram análise de sensibilidade com relação às susceptâncias dos circuitos em conjunto com o modelo linearizado de fluxo de potência, com o objetivo de determinar o caminho mais efetivo para a minimização de um índice de performance do sistema. Um fator de coerência, que relaciona mudanças em um fluxo em relação a alterações na capacidade de um circuito, é utilizado para determinar o “vetor de eficiência” que serve para escolher o caminho mais efetivo para reforço na rede de transmissão.

Em 1984, Villasana [90] propôs duas diferentes metodologias para serem aplicadas ao planejamento da expansão de redes de transmissão. A primeira foi formulada combinando o modelo de fluxo de carga linearizado com um modelo de Transporte. Enquanto o modelo linearizado calcula o fluxo de potência para os circuitos existentes, o modelo de Transporte era utilizado para computar o fluxo “sobrecarregado”. Este trabalho consistia de um aperfeiçoamento do trabalho proposto por Garver [41]. O segundo trabalho utilizava uma formulação linear inteira mista.

O uso de esquemas de decomposição para este problema teve início com o trabalho de Pereira [73]. Naquele trabalho, um esquema de decomposição de Benders [9] foi aplicado para decompor o problema global de planejamento de redes em dois subproblemas: um

de investimento, que tem por objetivo propor um plano de expansão; e outro de operação, que deve analisar o plano proposto e expressar as restrições operacionais em termos das variáveis de investimento através de restrições lineares chamadas de cortes de Benders. Esta nova restrição deve ser adicionada ao subproblema de investimento e novas iterações de Benders são repetidas até a obtenção da convergência. O modelo adotado para formular o problema de planejamento da expansão de redes de transmissão é não linear e não convexo, o que pode trazer sérias dificuldades para métodos de cortes como o algoritmo de decomposição de Benders. A aplicação de métodos de planos cortantes a um problema não linear e não convexo pode não ser bem sucedida pois os cortes produzidos podem excluir partes da região de viabilidade do problema, inclusive a região que contém a solução ótima.

Com o objetivo de contornar esta deficiência do método de decomposição de Benders em relação ao modelo não linear e não convexo, foi proposto na tese de mestrado de Romero [79, 82] uma metodologia de decomposição hierárquica composta por três fases distintas. Na primeira fase o problema de planejamento deve ser solucionado por decomposição de Benders considerando somente o modelo de transporte para o subproblema de operação. Além disso, as integralidades das variáveis de investimento deveriam ser relaxadas. Na segunda fase o modelo do subproblema de operação deve ser trocado por um modelo Híbrido (mais apurado) que consiste do modelo de fluxo de potência linearizado para os circuitos existentes e um modelo de transporte para computar o fluxo nos circuitos planejados. Finalmente, na terceira fase deste trabalho, o modelo de carga linearizado era utilizado para o cálculo do fluxo de carga em todos os circuitos da rede de transmissão. O subproblema de investimento considera as variáveis de investimento discretas e utiliza um algoritmo especializado de enumeração implícita desenvolvido em 1993 na tese de doutorado de Romero [80, 83].

Em 1990, Pinto e Nunes [77] usaram o esquema de decomposição de Benders combinado com um algoritmo de enumeração implícita. Com o objetivo de reduzir o esforço computacional, que pode ser muito grande, eles utilizaram duas técnicas: redução por inviabilidade e por custo.

Outro método de decomposição proposto para o problema de planejamento, foi proposto por Levi e Calovic [61] em 1991. Este trabalho propôs dividir o problema de planejamento em dois problemas menores, um tratando somente com questões de investimento e outro considerando somente problemas relacionados à operação. O problema de in-

vestimento foi especificado como um problema de fluxo de custo mínimo em rede. Este problema era decomposto novamente em dois subproblemas, o primeiro para computar o fluxo inicial que utiliza um algoritmo de programação linear para calcular o mínimo corte de carga. O segundo subproblema utilizava o modelo de rede marginal para obter o fluxo de carga "sobrecarregado".

Em 1994, Lattore-Bayona e Péres-Arriaga [58] propuseram uma metodologia heurística que toma vantagem da decomposição natural do problema em subproblemas de operação e investimento. O subproblema de investimento era solucionado utilizando-se um procedimento heurístico de busca em árvore iniciada a partir de uma solução viável obtida por outros modelos. As variáveis de investimento (ramos da árvore de busca) poderiam ser classificadas de três maneiras: as variáveis questionáveis (circuitos incluídos na solução viável inicial mas que o usuário pensa não pertencer ao plano ótimo), as variáveis atrativas (circuitos que o usuário pensa pertencer ao planejamento ótimo) e as variáveis congeladas (circuitos que não serão testados no processo de busca). Esta classificação das variáveis já consiste de um critério de truncamento utilizado por este trabalho com o objetivo de redução do tempo computacional. Os outros critérios utilizados eram limites na profundidade e na largura do processo de busca na árvore, limite no número de resoluções do subproblema de operação e limite no número de "passos errados" do processo de busca na árvore.

Em 1995, Binato e Oliveira [13] propuseram um método de busca, "backward-forward" para o problema de planejamento da expansão de redes de transmissão multi-estágio. Neste método são definidos passos para uma análise de planejamento a dois estágios: o passo "backward", que consiste de um planejamento retornando no tempo, buscando antecipações de circuitos já definidos para o segundo ano, e o passo "forward", que faz uma análise no sentido correto do tempo. Utilizando de uma maneira organizada estes dois passos, o método explora a região de viabilidade do problema em busca de economias de escala quando são considerados vários estágios durante o horizonte de planejamento.

Também em 1995, Oliveira et al. [69] utilizaram um esquema de decomposição hierárquica, mas composto por duas fases ao invés de três fases como no trabalho de Romero [79]. A primeira fase, da mesma forma que no trabalho de Romero, considera somente o modelo de Transporte, porém não relaxa a integralidade das variáveis de investimento, enquanto que a segunda fase é igual à terceira do trabalho de Romero. A maior

diferença entre estes dois trabalhos não vem da decomposição hierárquica utilizada e sim da maneira como o subproblema de investimento era solucionado. Enquanto que o trabalho anterior resolvia o subproblema de investimento até obter a solução ótima utilizando um algoritmo de enumeração implícita especializado, neste trabalho, utiliza um algoritmo de "branch and bound" com o objetivo de achar somente a primeira solução viável. Com isso, é possível se obter considerável redução do esforço computacional.

Na tese de doutorado de Binato [11], foi proposto uma aplicação computacional utilizando Decomposição de Benders que assegura que a solução ótima obtida pelo método de decomposição, é o plano ótimo de expansão da rede de transmissão. Isso está diretamente relacionado com a utilização do modelo linear $(0-1)$ disjuntivo, que pôde ser aplicado a problemas testes com sistemas reais devido à obtenção de valores mínimos para a constante disjuntiva. Uma nova heurística para determinar a convergência do problema *Mestre* da Decomposição de Benders resultou também em grandes economias em termos de tempo computacional gasto. Esse trabalho teve o mérito de ser o primeiro método exato implementado para resolução do problema de planejamento da expansão da transmissão. Entretanto, muitas vezes, o número elevado de candidatos de um caso de planejamento da expansão impede a aplicação com sucesso de técnicas de decomposição. Portanto, é necessário o desenvolvimento de técnicas heurísticas que são capazes de prover "boas" soluções para o problema. No trabalho [7], que utilizou um modelo linear inteiro misto disjuntivo para resolução do problema de planejamento, soluções fornecidas pelo método Grasp foram utilizadas como limites superiores para o método de "Branch and Bound" utilizado no estudo. Com essa combinação, a árvore de busca do "Branch and Bound" foi bastante reduzida, tornando possível sua aplicação a um problema de grande dimensão e possibilitando a verificação da otimalidade das soluções encontradas.

A utilização de métodos de busca mais elaborados, denominados de metaheurísticas, para o problema de planejamento da expansão de redes de transmissão teve início com o trabalho de Romero et al., que propuseram um método de Recozimento Simulado ("Simulated Annealing") [81], que posteriormente foi paralelizado [38]. A qualidade dos resultados publicados nestes dois artigos mostraram que tais métodos têm um excelente potencial para este problema.

Mais tarde, outras metaheurísticas também foram propostas. GRASP (Greedy Randomized Adaptive Search Procedure) foi utilizado por Binato et al. [16]. As melhores soluções já conhecidas para os dois sistemas teste reais brasileiros utilizados no estudo

foram obtidas pela metaheurística, assim como melhoramentos na solução do caso de planejamento da expansão do sistema sudeste, mostrando o potencial do método.

Algoritmos de Busca Tabu foram utilizados nos trabalhos [39, 70]. Dois Algoritmos Genéticos foram propostos para resolver problemas de planejamento em ("Extended Genetic Algorithms") [40] e ("Hybrid Genetic Algorithms") [18].

Em 2001, Edson Luiz da Silva et al. [86] utilizaram Busca Tabu para resolver o problema de planejamento estático de redes de transmissão. Foram utilizados vários conceitos da Busca Tabu como memória de curto-prazo, lista tabu e critério de aspiração. Uma fase de intensificação, que explora regiões do espaço de busca onde boas soluções devem existir, foi implementada juntamente com uma fase de diversificação, que direciona a busca para regiões não exploradas, usando conceitos de memória de médio e longo-prazo.

Uma metodologia híbrida combinando GRASP com Religamento de Caminhos foi desenvolvida em [25]. Religamento de Caminhos é um método que surgiu como uma estratégia de intensificação para melhorar a qualidade da solução de outras metaheurísticas. Nos poucos trabalhos em que foi utilizado, obteve grande sucesso. Neste estudo aprimorou a qualidade da busca por novas soluções, ajudando a obter a solução ótima dos problemas propostos em um número menor de iterações.

1.1.2 Problemas de Escalonamento de Tarefas (JSP)

Devido à importância econômica e logística dos problemas de escalonamento, muitos dos primeiros esforços concentravam-se em obter programações ótimas. O primeiro exemplo de um método eficiente que resolve o problema de forma ótima, exigindo um tempo computacional que cresce de acordo com um polinômio no tamanho do problema e, provavelmente, o primeiro trabalho em problemas de escalonamento é de Johnson [53], que desenvolveu um algoritmo para um caso especial de duas máquinas. Logo se descobriu que a geração de programações ótimas quase sempre requer um tempo computacional excessivo, independente da metodologia. Além disso, o problema de escalonamento de tarefas está entre os mais difíceis problemas de otimização combinatória e é NP-completo. Na prática, escalonamentos de tarefas são usualmente gerados por especialistas em escalonamento usando regras simples de despacho.

Vários métodos foram propostos. Dois métodos de otimização implementados foram

o de branch and bound [35] e programação dinâmica [75]. Métodos aproximados ou heurísticos não garantem a obtenção de soluções ótimas, mas são capazes de obter soluções de alta qualidade em tempos computacionais moderados e, portanto, mais adequados a problemas reais e maiores, concentrando a maioria dos esforços. Esses métodos foram primeiramente desenvolvidos tendo como base regras de despacho prioritárias, que são de implementação simples e requerem baixo esforço computacional [8, 37, 67]. A cada passo, todas as operações disponíveis para serem alocadas recebem um grau de prioridade e a operação com a maior prioridade é alocada. Geralmente, várias execuções do método são necessárias para se obter resultados válidos. Heurísticas mais aprimoradas realizam combinações probabilísticas de regras de despacho individuais como em [34]. O trabalho [59] mostra que uma combinação aleatória dessas regras fornece resultados superiores ao da utilização de regras individuais.

O surgimento de computadores mais velozes e a ênfase em técnicas mais aprimoradas possibilitaram o surgimento de métodos sofisticados como o Shifting Bottleneck Procedure de Adams et al. [1]. SBP é caracterizado pelos seguintes procedimentos: identificação do subproblema, seleção do bottleneck (gargalo), solução do subproblema e re-otimização do escalonamento. A estratégia atual relaxa o problema em m subproblemas de 1 máquina e resolve, iterativamente, cada subproblema por vez, para identificar a máquina gargalo.

Técnicas de Inteligência Artificial também têm sido utilizadas. Redes Neurais Artificiais também são uma alternativa para a resolução de problemas de escalonamento. Cheung [91] descreve algumas das principais arquiteturas de redes neurais aplicadas na solução de problemas de escalonamento: Rede de Hopfield, Multi-Layer Perceptron, Boltzmann machine, rede competitiva e rede auto-organizável. Redes de Hopfield e Multi-Layer Perceptron são os modelos mais utilizados. Essas implementações não são competitivas com as melhores heurísticas para qualquer classe de problema [71].

Um dos métodos mais eficazes na resolução de problemas difíceis de otimização combinatória é o método de busca local ou de vizinhança, apesar de serem conceitualmente simples. Métodos de busca local que usam estratégias para escapar de ótimos locais são o desenvolvimento mais recente aplicados na resolução desses problemas. Resende [78] apresenta um GRASP aplicado ao problema de escalonamento. Os trabalhos de Matsuo et al. [64] e Van Laarhooven et al. [89] mostram a aplicação de técnicas de busca local com resultados não muito satisfatórios. Quando outras técnicas são incorporadas,

como Recozimento Simulado e Algoritmos Genéticos, a qualidade das soluções melhora, como em Kolonko [57]. O tempo de computação elevado é uma das deficiências desses métodos e várias tentativas de se acelerar o processo foram tentadas. Vários algoritmos genéticos já foram testados em problemas de escalonamento, mas com resultados fracos devido à dificuldade dos operadores genéticos em gerar escalonamentos viáveis próximos a soluções ótimas. Uma forma de computação evolucionária conhecida como Busca Local Genética obteve bons resultados como no trabalho de Della Croce et al. [23]. Para instâncias grandes e difíceis porém, resultados sub-ótimos não são obtidos em tempos aceitáveis.

Busca Tabu fornece os melhores resultados dentre os métodos de aproximação, com tempos computacionais razoáveis. Os trabalhos [68, 85] obtiveram os melhores resultados, confirmando também que métodos híbridos são os mais eficientes. Entretanto, Busca Tabu, como a maioria das técnicas de busca local, precisa de ajustes nos seus parâmetros para cada problema estudado.

1.2 Organização da Tese

A tese está organizada da seguinte maneira:

O capítulo 2 introduz definições e conceitos das metaheurísticas GRASP, Religamento de Caminhos e GRAPR, que foram utilizadas nos estudos computacionais ilustrados neste trabalho.

O capítulo 3 descreve o problema de Planejamento Estático da Expansão de Sistemas de Transmissão de Energia Elétrica. Mostra como foi feita a implementação computacional da metaheurística GRAPR para esse problema e fornece pseudo-códigos dos algoritmos desenvolvidos, indicando cada passo do método proposto. Um pequeno exemplo de planejamento, usando um sistema de 3 barras, ilustra os passos do método. Resultados computacionais para o problema, utilizando algoritmos seqüenciais, são apresentados.

O capítulo 4 descreve o problema de Escalonamento de Tarefas (JSP). Mostra como foi feita a implementação computacional da metaheurística GRAPR para esse problema e fornece pseudo-códigos dos algoritmos desenvolvidos, indicando cada passo do método proposto. Um pequeno exemplo de JSP, com 3 tarefas e 3 máquinas, ilustra os passos do método em uma iteração completa. Resultados computacionais para o problema, uti-

lizando algoritmos seqüenciais, são apresentados.

Uma investigação experimental da distribuição de probabilidade do tempo de solução em metaheurísticas do tipo GRASP é realizada no capítulo 5. O método experimental de estudo é descrito em detalhes, e os resultados computacionais são mostrados tanto para o problema de Planejamento da Expansão, quanto para o de Escalonamento de Tarefas. O objetivo deste estudo é verificar se as distribuições de probabilidade do tempo de solução para a metaheurística GRAPR, se encaixam em uma distribuição exponencial a dois parâmetros.

No capítulo 6 se encontra uma pequena introdução sobre processamento paralelo, mostrando as vantagens da utilização de um cluster de PC's para computação de alto desempenho. É feita uma descrição da implementação paralela dos algoritmos utilizados nesta tese, usando o esquema não-colaborativo proposto. Resultados computacionais para o Planejamento da Expansão de Sistemas de Transmissão são apresentados.

Finalmente, o capítulo 7 apresenta conclusões e desenvolvimentos futuros para este trabalho.

Esse capítulo mostrou os tipos de problemas encontrados em programação matemática e em que classe se enquadram os problemas estudados neste trabalho. Citamos as dificuldades encontradas na solução destes problemas e quais métodos são utilizados para a sua resolução. Os trabalhos mais importantes aplicados na solução do Planejamento da Expansão de Sistemas de Transmissão e Escalonamento de Tarefas foram revisados. Os principais métodos exatos e aproximados de resolução foram citados, tentando mostrar o grau de sucesso de suas aplicações. No capítulo seguinte, daremos as definições e conceitos das metaheurísticas utilizadas neste trabalho.

CAPÍTULO 2

MÉTODO DE RESOLUÇÃO

2.1 Introdução

Um trabalho de pesquisa muito intenso tem sido realizado nos últimos anos com o intuito de desenvolver métodos de busca eficientes que não requerem a enumeração explícita de alternativas, e possuem uma maior capacidade de resolver problemas reais e complexos. Deve-se notar que a maioria dos problemas reais (real world problems) não podem ser tratados computacionalmente ou são muito grandes para serem resolvidos por métodos exatos. Nesse contexto, uma boa alternativa é a aplicação de métodos heurísticos para se encontrar boas soluções para o problema, mas sem a garantia de que essas soluções sejam ótimas. A eficiência desses métodos depende de alguns fatores como a capacidade de evitar ótimos locais, se adaptar à uma realização particular e tirar proveito da estrutura do problema. Vários métodos heurísticos ou metaheurísticos têm sido desenvolvidos e empregados com sucesso na solução de problemas reais de otimização combinatória. Métodos metaheurísticos são métodos de busca que combinam métodos heurísticos. Entre os mais conhecidos podemos citar Busca Tabu [43], Recozimento Simulado [55], Algoritmos Genéticos [50] e GRASP [31]. Neste trabalho, propomos uma nova metaheurística denominada de GRAPR (do inglês Greedy Randomized Adaptive Path Relinking), para a solução de problemas de otimização combinatória. O método recebeu esta denominação pois herdou suas características de duas metaheurísticas que têm se mostrado muito eficientes em vários trabalhos, GRASP e Religamento de Caminhos (do inglês Path Relinking).

```
função GRASP()
 1    LêDados();
 2    X* ← ∅;
 3    faça k = 1, ..., MaxIter
 4     X ← ConstróiSoluçãoGulosaAleatória();
 5     se X ≠ ∅
 6        X ← BuscaLocal(X);
 7        X* ← AtualizaSolução(X);
 8     fim se ;
 9    fim faça ;
10    retorna X*
fim GRASP;
```

Figura 2.1: GRASP, descrição genérica.

2.2 Metaheurísticas para Solução de Problemas de Otimização Combinatória

A seguir, descreveremos duas heurísticas que serviram de base para o desenvolvimento do método proposto neste trabalho.

2.2.1 GRASP

GRASP (do inglês Greedy Randomized Adaptive Search Procedure) é uma poderosa metaheurística para a solução de problemas de otimização combinatória. GRASP foi formalizado por Feo e Resende [32] em 1995 e é um processo iterativo em que cada iteração é constituída por duas fases; uma fase de construção e uma fase de busca local. Na fase de construção, uma solução viável para o problema é construída, um elemento por vez. Na fase de busca local, a vizinhança da solução viável é explorada, até a obtenção de um ótimo local. A melhor solução obtida por todas as iterações é mantida como resultado final. Um pseudo-código genérico de GRASP é ilustrado pela figura 2.1.

Na figura 2.1, $\mathcal{X}$ e $\mathcal{X}^*$ são, respectivamente, uma solução viável para o problema e a melhor solução obtida até uma dada iteração k. As principais funções da metaheurística GRASP são os procedimentos `ConstróiSoluçãoGulosaAleatória()` – passo 4 – fase de construção, onde uma solução viável é iterativamente construída e `BuscaLocal()` – passo 6 – fase de busca local, que se inicia na solução da fase de construção e procura por ótimos locais em uma determinada vizinhança.

O passo 7 – `AtualizaSolução` – simplesmente compara o custo da solução obtida na $k-$ésima iteração com o custo da melhor solução obtida até então e, se a solução corrente for melhor, atualiza a melhor solução. A seguir cada uma das fases da metaheurística GRASP será apresentada.

Fase de Construção

A fase de construção constrói, iterativamente e um elemento por vez, uma solução viável para o problema. Para ilustrar esta etapa, consideremos um problema geral de programação linear $(0-1)$ mista, conforme representado a seguir:

$$Minimize\ z = c^t x + d^t y \tag{2.1a}$$

$$sujeito\, a:$$

$$Ax + By \geq b \tag{2.1b}$$

$$x \in \{0,1\}^n, y \in \mathbb{R}^r. \tag{2.1c}$$

O primeiro passo dessa fase consiste em se inicializar uma solução tentativa fazendo, por exemplo, $\mathcal{X} = \{\emptyset\}$, onde o conjunto $\mathcal{X}$ é formado pelos índices das variáveis que tem valor igual a 1 na solução tentativa, isto é,

$$\mathcal{X} = \{i \mid \text{ se } x_i = 1, i = 1, \ldots, n\}.$$

Logo, na primeira iteração $x = 0$. Substituindo x no problema (2.1), obtém-se

$$Minimize\ z = d^t y \tag{2.2a}$$

$$sujeito\, a:$$

$$By \geq b \tag{2.2b}$$

$$y \in \mathbb{R}^r. \tag{2.2c}$$

A cada iteração da fase de construção, a escolha do próximo elemento que entrará no

conjunto $\mathcal{X}$ é determinada ordenando-se todos os elementos candidatos, isto é, aqueles que podem ser introduzidos na solução, segundo uma função de mérito $h(x)$. Essa função deve estar relacionada com o benefício (míope) de se adicionar cada elemento à solução $\mathcal{X}$, ou seja:

$$h(x) \propto \frac{\partial z}{\partial x}$$

A heurística utiliza esta função $h(x)$, definida acima, para selecionar a variável que será adicionada à solução tentativa, ou seja, $\mathcal{X} = \mathcal{X} + \{j\}$. No caso de o método ser do tipo guloso ("greedy") puro, a escolha desta variável seria, naturalmente, a variável de maior valor para a função $h(x)$. Os benefícios associados à adição de cada elemento devem ser atualizados a cada iteração para refletir os efeitos das adições realizadas anteriormente, isto é, a função h deve ser adaptativa, refletindo o efeito dessas adições.

Contudo, a fase de construção de um GRASP não é um método guloso "puro", e sim um algoritmo guloso aleatorizado. Para tanto, o elemento que será adicionado é escolhido de uma lista denominada de lista restrita de candidatos ou LRC, que contém os elementos com os melhores índices de mérito. Esta lista é construída a partir dos valores máximo,

$$h^{max} = \max_{i \in \bar{\mathcal{X}}} h(x_i),$$

e mínimo,

$$h^{min} = \min_{i \in \bar{\mathcal{X}}} h(x_i) \geq 0,$$

da função de mérito $h(x)$, onde $\bar{\mathcal{X}}$ é o complemento do conjunto $\mathcal{X}$ em relação a $\mathcal{C}$, ou seja, o conjunto das variáveis que ainda não foram adicionadas ao conjunto solução $\mathcal{X}$. Então, a LRC será formada por todas aquelas variáveis cujo índice de mérito estiver no intervalo $[h^{min}, h^{min} + \alpha(h^{max} - h^{min})]$, ou seja,

$$LRC = \left\{ i \in \bar{\mathcal{X}} \mid h^{min} \leq h(x_i) \leq h^{min} + \alpha(h^{max} - h^{min}) \right\}, \tag{2.3}$$

```
função ConstróiSoluçãoGulosaAleatória()
 1    X ← ∅;
 2    repita
 3      Construa LRC de acordo com (2.3);
 4      j ← SAMP(LRC);
 5      X = X ∪ j;
 6    até obter uma solução ou Falhar;
 7    se Falha
 8      X ← ∅;
 9    fim se
10    retorne X
fim ConstróiSoluçãoGulosaAleatória;
```

Figura 2.2: Fase de Construção, descrição genérica.

onde α é um parâmetro $(0 \leq \alpha \leq 1)$ fornecido pelo usuário que controla o quão "guloso" ou aleatório será o algoritmo.

Em um GRASP clássico, as variáveis são escolhidas da LRC de forma aleatória. Em [19], Bresina introduziu o conceito de função de probabilidade para a escolha das variáveis a serem incluídas na solução. Ao $i-$ésimo elemento da LCR, ordenado segundo a função de mérito $h(x)$ e possuindo, após a ordenação, a posição r_i, é associado um valor $bias(i)$. Essa função pode ser logarítmica ($bias(i) = log^{-1}(r_i + 1)$), linear ($bias(i) = 1/r_i$), polinomial de ordem n, ($bias(i) = r_i^{-n}$) ou mesmo aleatória ($bias(i) = 1$). A probabilidade de escolha do $i-$ésimo elemento é calculada por:

$$p_i = \frac{bias_i}{\sum_{k \in LRC} bias_k} \tag{2.4}$$

No trabalho [14] de planejamento de redes de transmissão de energia, a função de probabilidade linear mostrou melhores resultados.

Fase de Busca Local

Como na fase de construção utilizamos um algoritmo "guloso" aleatório, não podemos garantir que a solução obtida é um ótimo local. Por isso, é vantajoso aplicar uma fase de busca local após a fase de construção para melhorar a solução obtida.

A busca local é um dos métodos mais eficientes para se atacar problemas difíceis

de otimização combinatória, apesar de se basear em uma idéia simples. Em linhas gerais, o algoritmo de busca local trabalha em um esquema de trocas, isto é, trocando a solução corrente por uma melhor (isto é, também viável e de menor custo no caso de minimização) dentro de uma vizinhança da solução corrente. O algoritmo de busca local pára quando não existe mais nenhuma solução melhor do que a solução corrente pertencente à vizinhança dada desta solução. A vizinhança pode ser estabelecida com base no número de diferenças entre duas soluções.
Por exemplo, sejam x' e x'' duas soluções diferentes para o problema (2.1) e `Diff(`a`,`b`)` uma função que compara duas soluções quaisquer a e b, e retorna o número de diferenças entre elas. Então, dizemos que x'' pertence a vizinhança 2 de x' se `Diff(`x'`,`x''`)` ≤ 4. A vizinhança está diretamente relacionada à forma utilizada para codificar as soluções do problema. O exemplo dado refere-se, portanto, ao caso de vetores de pertinência, $x \in \{0,1\}^n$.

No exemplo do problema (2.1), considere uma estrutura de vizinhança, $N(x)$, conforme definido acima, para uma dada solução viável $\bar{x}$. Então, concluímos que uma solução $\bar{x}$ será uma solução ótima local na vizinhança $N(\bar{x})$ se, e somente se, nesta vizinhança não existir nenhuma outra solução melhor do que a solução $\bar{x}$, ou seja,

$$\bar{x} = \min_{x \in N(\bar{x})} (2.1) \Leftrightarrow \nexists x' \in N(s) \mid c^t x' < c^t \bar{x}.$$

O procedimento de busca local, ilustrado de maneira genérica na figura 2.3, percorre a vizinhança da solução obtida na fase de construção a procura de melhoramentos e, encontrando uma solução também viável, porém com um custo $c^t x$ menor que o da solução corrente, torna esta solução a solução corrente e reinicializa o procedimento de busca local. A idéia básica da metaheurística GRASP consiste em usar diferentes soluções iniciais como pontos de partida para a busca local. Uma solução x é dita como pertencente à *bacia de atração* de um ótimo local quando, a partir de uma busca local iniciada em x, é possível atingir este ótimo local. Caso uma das soluções iniciais esteja na bacia de atração de um ótimo global, a busca local irá encontrar este ótimo global. Caso contrário, a solução do algoritmo será um ótimo local.

Uma solução na bacia de atração de um ótimo global será eventualmente produzida, caso um número grande de soluções geradas aleatoriamente seja usado para iniciar

```
função BuscaLocal(s)
 1    Construa N(s) para s;
 3    Ache  y ∈ N(s) | Ay ≥ b ∧ c^t y < c^t s;
 4    se ∃y
 5       BuscaLocal(y);
 6    fim se
 8    retorne s
fim BuscaLocal;
```

Figura 2.3: Fase de Busca Local, descrição genérica.

a busca local. Porém, soluções produzidas aleatoriamente são, em geral, de baixa qualidade. Além disso, o número de movimentos necessários para que soluções geradas aleatoriamente (e que estejam na bacia de atração de um ótimo global) atinjam o ótimo possivelmente será muito elevado ou até mesmo exponencial no tamanho do problema. Por outro lado, algoritmos gulosos geralmente produzem soluções de melhor qualidade do que as geradas aleatoriamente. O uso de soluções gulosas como ponto de partida para uma busca local, em geral, levará a boas soluções, porém, sub-ótimas, isto é, soluções de qualidade inferior à qualidade dos ótimos globais. Isso acontece porque a diversidade das soluções produzidas por um algoritmo guloso é muito pequena. Para garantir diversidade de soluções e, ao mesmo tempo, um controle na qualidade das soluções produzidas, a metaheurística GRASP usa um algoritmo semi-guloso para produzir as soluções iniciais usadas em cada iteração.

O procedimento de busca local pode requerer esforço computacional exponencial para terminar. Entretanto, boas soluções iniciais, estruturas de dados especializadas e implementações cuidadosas podem melhorar muito o esforço computacional requerido.

A Figura 2.4 ilustra o padrão de exploração do espaço da metaheurística GRASP. Todos os pontos viáveis são mostrados e, em detalhe, aqueles gerados pela fase de construção que tiveram sua vizinhança explorada pela fase de busca local.

2.2.2 Religamento de Caminhos

O método de Religamento de Caminhos é uma generalização da estratégia evolucionária denominada de Scatter Search. As poucas aplicações práticas desses métodos mostraram um grande potencial para resolução de problemas de otimização discreta e não-lineares [46]. Religamento de Caminhos começou a ser utilizado como uma método

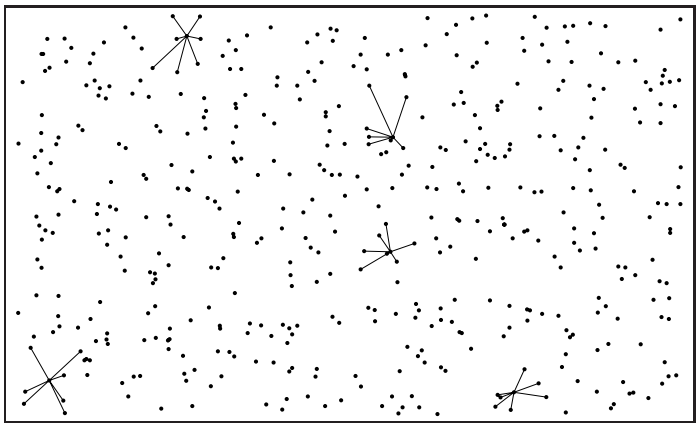

Figura 2.4: Padrão de exploração da metaheurística GRASP

para integrar estratégias de intensificação e diversificação no contexto de Busca Tabu, na procura por soluções de problemas de otimização. Intensificação e diversificação são características de métodos de busca. Intensificação significa explorar regiões onde soluções potencialmente boas devem existir. Diversificação significa direcionar a busca para regiões ainda não exploradas. Por exemplo, métodos puramente aleatórios são bons em diversificação, mas pobres em intensificação, enquanto um método de busca local tem características opostas.

As estratégias de combinação de regras de decisão que foram introduzidas com o objetivo de aprimorá-las para problemas de escalonamento de tarefas, e as de combinação de restrições, que geram restrições combinadas chamadas de *surrogate constraints*, levaram ao desenvolvimento da estratégia complementar de combinar soluções, presente nas metaheurísticas Scatter Search e Religamento de Caminhos [46]. Scatter Search trabalha em um conjunto de soluções denominadas *soluções de referência*, efetuando combinações lineares entre elas e utilizando heurísticas para gerar e selecionar soluções candidatas.

Características úteis adicionadas à metaheurística Scatter Search, por extensão da sua filosofia básica, estão presentes no método de Religamento de Caminhos. O processo de geração de combinações lineares entre um conjunto de soluções de referência pode ser caracterizado como gerador de caminhos entre essas soluções, onde pontos desses caminhos podem ser geradores de novas trajetórias. Com isso, a combinação entre soluções pode ter um significado mais amplo, podendo ser originada do processo de geração de trajetórias no espaço de busca. Pontos dessas trajetórias possuirão atributos das soluções geradoras em quantidades variadas dependendo da sua posição no espaço. Exemplifi-

cando, esses atributos podem ser vértices ou ramos de um grafo, valores de variáveis, posições de tarefas em uma programação etc. Logo, para gerar esses caminhos, devemos considerar uma solução inicial, uma solução final ou guia e, progressivamente, incorporar atributos da solução guia na inicial até alcançarmos a solução final. Normalmente, as soluções geradoras do caminho são de alta qualidade, pertencentes a um conjunto de elite.

A figura 2.5 ilustra possíveis caminhos gerados no espaço entre duas soluções pelo método de Religamento de Caminhos. Vemos que uma seqüência de pontos $x = x(1), x(2), \ldots, \hat{x}$ são gerados entre as soluções guia. A escolha da trajetória pode ser realizada de várias maneiras e segundo vários critérios. Considerando a função objetivo $f(x)$ a ser minimizada, Religamento de Caminhos escolhe, a cada passo, dentre todos os movimentos possíveis, aquele que resulte no menor valor para a função objetivo no ponto, mas escolhas diferentes podem ser feitas. De fato, podemos considerar o método de Religamento de Caminhos como sendo um algoritmo guloso para resolver o seguinte problema de otimização combinatória,

$$Otimize\ f(x), \tag{2.5a}$$

$$sujeito\ a: \tag{2.5b}$$

$$x \in \left\{ x_i \mid x_i = \prod_{i=1}^{n} \mathcal{R}_j(s), j, n \in \{1, 2, \ldots, |\mathcal{R}| - 1\} \right\} \tag{2.5c}$$

onde $f(x)$ é a função de custo a ser minimizada (ou maximizada), s e $t = \prod_{i=1}^{|\mathcal{R}|} \mathcal{R}_i(s)$ são, respectivamente, a solução inicial e final e $\mathcal{R}_i(x)$ é uma lista de movimentos locais a partir de x que diminuem a "distância" entre x e t, isto é, o conjunto de todas as soluções obtidas de movimentos $\mathcal{R}_i(x), i = 1, \ldots, |\mathcal{R}|$, que corresponde ao subconjunto da vizinhança de x. A restrição do problema (2.5) indica que x pode ser qualquer solução entre s e t, isto é, x pode ser obtido aplicando uma seqüência de movimentos $\mathcal{R}_i(s)$.

Alternativamente, podemos ter também múltiplas soluções guia que terão seus atributos reunidos em um conjunto para serem combinados e determinarem o próximo passo na construção da trajetória.

Finalmente, a implementação do método de Religamento de Caminhos requer a construção de um conjunto de elite $\mathcal{E}$, formado pelas soluções com os melhores custos, mas que contribuam para manter a diversidade do conjunto. Isto é, para pertencer ao

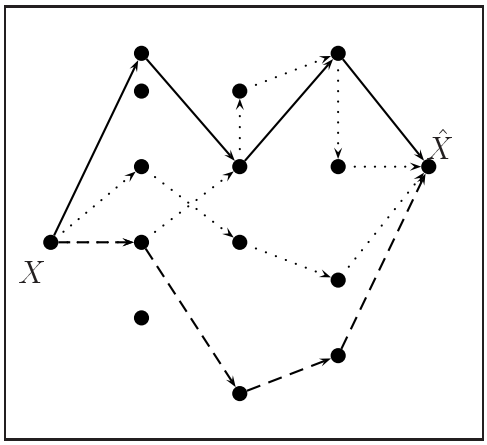

Figura 2.5: Possíveis caminhos gerados por Religamento de Caminhos

conjunto de elite, uma dada solução deve ter um custo ou um valor para a função objetivo do problema menor que a melhor solução do conjunto ou possuir um custo menor (minimização) que a pior solução do conjunto e ser suficientemente diferente das demais soluções. O grau de diferença entre duas soluções pode ser computado utilizando-se uma rotina que calcula o número de atributos diferentes entre duas soluções. O tamanho do conjunto de elite $\mid \mathcal{E} \mid$ e o grau de diferença mínimo entre as soluções que caracteriza o conjunto de elite, são parâmetros fornecidos ao programa. Essas soluções servirão como guias e terão seus atributos incorporados às soluções iniciais para a construção das trajetórias.

2.2.3 Greedy Randomized Adaptive Path Relinking (GRAPR)

O principal objetivo do método de Greedy Randomized Adaptive Path Relinking (GRAPR) é realizar uma exploração mais profunda do espaço entre a solução inicial e a solução guia do método de Religamento de Caminhos. Esse método foi inicialmente proposto no trabalho de Binato et al. [12]. Note que podem existir inúmeros caminhos entre as duas soluções. Cada caminho explora uma parte do espaço de busca e pode, eventualmente, conter boas soluções. Religamento de Caminhos, em sua forma original, gera apenas um caminho entre as duas soluções guia, utilizando movimentos que resultem sempre no menor valor para a função objetivo no ponto considerado o que classifica o método como um método “guloso” para exploração de caminhos entre duas soluções com uma habilidade limitada para a explorar o espaço de busca.

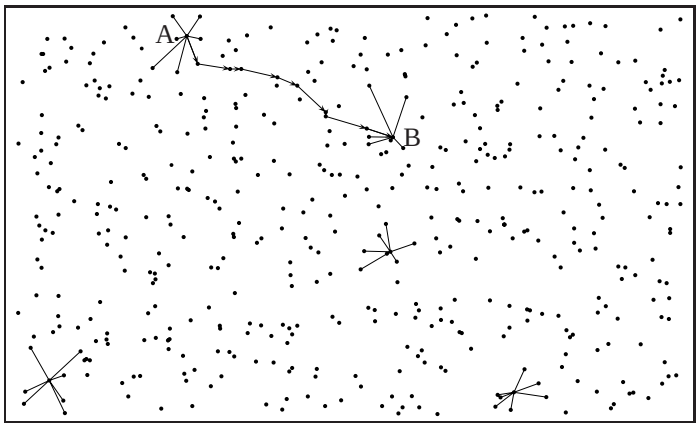

Figura 2.6: Caminhos gerados na fase de construção do GRAPR

A idéia do GRAPR é aplicar conceitos da metaheurística GRASP para resolver o problema (2.5), randomizando a escolha dos movimentos na construção das trajetórias. Para a sua implementação, precisamos generalizar as fases de Construção e de Busca Local do GRASP para o problema de geração de trajetórias no GRAPR. A seguir, descrevemos como isso deve ser feito.

Considerando que estamos na solução inicial e queremos realizar o primeiro movimento no espaço, na fase de construção, devemos montar a lista restrita de candidatos (LRC) com os melhores pontos vizinhos à solução inicial. Lembramos que esses pontos estão rankeados de acordo com o valor da função objetivo do problema. Uma função de distribuição de probabilidade pode ser utilizada para ser aplicada aos elementos da LRC. A seguir, um vizinho é escolhido aleatoriamente da LRC, e o movimento para esse ponto é realizado. Esse procedimento é diferente do método de Religamento de Caminhos, que escolhe sempre o melhor movimento local. Um padrão de exploração do espaço gerado pela fase de construção do GRAPR é ilustrado na figura 2.6.

Métodos híbridos como os que utilizam GRASP e Religamento de Caminhos têm sido utilizados recentemente com bons resultados na solução de problemas de otimização combinatória. A idéia é utilizar as boas características de cada um para se obter um método robusto. A utilização de GRASP com o inédito GRAPR é proposto neste trabalho. GRAPR entra como uma nova fase no algoritmo de GRASP, depois da fase de Busca Local, com o intuito de prover o método híbrido de um potencial de exploração do espaço de busca superior. Pretende-se com isso obter soluções de melhor qualidade ou soluções ótimas para os casos teste utilizados na pesquisa.

Este capítulo introduziu as definições e conceitos dos métodos metaheurísticos utilizados nesta tese. Mostrou como as estratégias GRASP e Religamento de Caminhos foram combinadas para gerar o método GRAPR, capaz de explorar o espaço entre duas soluções de alta qualidade de forma bastante eficiente.

CAPÍTULO 3

APLICAÇÃO AO PROBLEMA DE PLANEJAMENTO ESTÁTICO DA EXPANSÃO DE SISTEMAS DE TRANSMISSÃO DE ENERGIA ELÉTRICA

3.1 Introdução

O problema de planejamento da expansão de sistemas elétricos de potência é um problema de grande importância no setor elétrico, que deve ser solucionado pelos engenheiros de planejamento dos sistemas de energia, para garantir que os consumidores serão atendidos de forma econômica e confiável, com o aumento da demanda de energia com o passar dos anos. Apesar de o sistema elétrico estar passando por um processo de desregulamentação, transitando de uma estrutura centralizada para uma descentralizada, que tem como objetivo obter uma maior eficiência dos agentes participantes do setor, (agentes de geração, agentes de transmissão, agentes de distribuição, entre outros) que decidirão onde e quando investir seus recursos para expandir o sistema, este processo deverá sofrer a interferência de um agente central que deve funcionar como um plano de referência de forma a garantir uma expansão ótima global do sistema, otimizando a utilização dos recursos disponíveis e os custos para os consumidores. Novos parques geradores e novas rotas de transmissão devem ser construídos para atender esta nova carga do sistema. O Brasil passou recentemente por um grave problema de racionamento de energia devido à falta de investimentos na área de geração e transmissão de energia, agravado por um período de seca que baixou o nível dos reservatórios. Como 87% [1] da energia elétrica produzida no país

[1] Dado de 1999

é de origem hidroelétrica [52] e as usinas estão distantes dos grandes centros consumidores, torna-se necessário a construção de novos circuitos com a finalidade de transmitir a potência elétrica produzida nestas usinas, para aumentar a confiabilidade do sistema, otimizar recursos hídricos, etc. As decisões do processo de planejamento estão relacionadas à seleção das melhores unidades geradoras, das melhores rotas de transmissão e das melhores malhas para garantir um suprimento de energia de forma econômica e confiável. Este processo de tomada de decisão dá origem a um problema de otimização complexo de grande porte que deve ser resolvido pelos engenheiros de planejamento. É necessário o desenvolvimento de estratégias e técnicas de solução que assegurem que as decisões tomadas durante o processo de planejamento sejam as decisões ótimas. Este problema não pode ser resolvido sem que sejam feitas simplificações. Normalmente, o problema de planejamento é separado com relação aos seus principais agentes: o problema de planejamento da geração, que supõe conhecido o custo de expansão da transmissão; o problema de planejamento da transmissão, que supõe conhecida estimativas de crescimento da demanda e programas alternativos de expansão da geração, até o ano-horizonte de planejamento, e o planejamento da distribuição. Neste trabalho, estamos interessados no planejamento da expansão de sistemas de transmissão nos horizontes de médio e longo prazo (10 anos ou mais), que consiste em determinar onde e quando novos equipamentos de transmissão (linhas de transmissão, transformadores, etc.) devem ser instalados de forma a atender a carga de forma econômica e confiável. Devido tanto às incertezas como também às dimensões inerentes a este tipo de problema, métodos rápidos e aproximados de análise são requeridos.

3.2 Formulação do Problema Estático de Planejamento da Expansão de Sistemas de Transmissão

O problema de planejamento da expansão de redes de transmissão de potência consiste em se escolher, entre um conjunto pré-definido de circuitos candidatos, aqueles que devem ser construídos de forma a minimizar os custos de investimento e operação e atender a demanda de energia futura ao longo de um horizonte de planejamento com confiabilidade, que assume como conhecido o plano de geração. Esse problema tem uma natureza dinâmica, isto é, requer a consideração de múltiplos períodos de tempo, determinando-se uma seqüência (estágios) de planos de expansão de transmissão.

Quando o horizonte de planejamento reduz-se a apenas um ano (estágio) no futuro, o problema multi-estágio se transforma em um problema estático, em que o objetivo é determinar *onde* e que *tipo* de novos equipamentos de transmissão devem ser instalados de forma a minimizar os custos de investimento sujeito a uma série de restrições técnicas e operativas.

Os dados para este problema são a previsão de carga futura bem como o despacho dos geradores para atender ao mercado. Além disso, são necessários dados para a rede existente, também chamada de rede básica, e dados para os novos circuitos que podem ser adicionados à rede básica. Note que a rede básica não tem capacidade suficiente para o atendimento do mercado futuro.

O problema de planejamento da expansão de redes de transmissão neste trabalho, é modelado como um problema de programação não-linear inteira mista.

Sejam $\mathcal{N}$ o conjunto de barras da rede elétrica, $\mathcal{E}$ o conjunto de circuitos existentes, e $\mathcal{C}$ o conjunto dos circuitos candidatos, o problema de planejamento da expansão de redes de transmissão pode ser formulado da seguinte maneira:

$$Minimize\ z = \sum_{ij \in \mathcal{C}} c_{Iij} x_{ij} \tag{3.1a}$$

$$sujeito\, a:$$

$$-\sum_{j \in \Omega_i^0} f_{ij}^0 - \sum_{j \in \Omega_i^1} f_{ij}^1 + g_i = d_i,\ \forall i \in \mathcal{N} \tag{3.1b}$$

$$f_{ij}^0 - \gamma_{ij}^0(\theta_i - \theta_j) = 0,\ \forall (i,j) \in \mathcal{E} \tag{3.1c}$$

$$f_{ij}^1 - x_{ij}\gamma_{ij}^1(\theta_i - \theta_j) = 0,\quad \forall (i,j) \in \mathcal{C} \tag{3.1d}$$

$$-x_{ij}\overline{f}_{ij}^1 \leq f_{ij}^1 \leq x_{ij}\overline{f}_{ij}^1,\ \forall (i,j) \in \mathcal{C} \tag{3.1e}$$

$$0 \leq g_i \leq \overline{g}_i,\ \forall i \in \mathcal{N} \tag{3.1f}$$

$$-\overline{f}_{ij}^0 \leq f_{ij}^0 \leq \overline{f}_{ij}^0,\ \forall (i,j) \in \mathcal{E} \tag{3.1g}$$

$$f_{ij} = -f_{ji},\ \forall (i,j) \in \mathcal{E} \tag{3.1h}$$

$$x_{ij} \in \{0,1\},\ \forall (i,j) \in \mathcal{C} \tag{3.1i}$$

Esse modelo é denominado de modelo não-linear inteiro ou modelo DC onde: c_{Iij} é o custo de investimento associado a decisão de construção do circuito candidato, x_{ij}, entre as barras $i-j$. f_{ij}^0 e f_{ij}^1 são, respectivamente, os fluxos de potência nos circuitos existentes

e candidatos $i-j$, g_i e d_i são a geração de potência ativa e a demanda (carga ativa) a ser atendida na barra i, respectivamente. γ_{ij}^0 e γ_{ij}^1 são, respectivamente, as susceptâncias dos circuitos existente e candidato entre as barras $i-j$, e θ_i é o ângulo da tensão na barra i. Finalmente, $\overline{f}_{ij}$ é a capacidade de carregamento do circuito ij e $\overline{g}_i$ é a capacidade de geração do gerador conectado à barra i. Para a representação de circuitos em paralelo, seria necessário a inclusão de um novo índice nas variáveis que representam os fluxos, susceptâncias e circuitos candidatos, que ficariam como : f_{ij_k}, γ_{ij_k} e x_{ij_k}.

A função objetivo do problema (3.1) corresponde a minimização da soma dos custos de investimento para a construção de novos equipamentos de transmissão (linhas, transformadores etc.).

As restrições (3.1b), (3.1c) e (3.1d) correspondem às leis de Kirchhoff do modelo de fluxo linearizado para o sistema de transmissão. Note que as restrições correspondentes à segunda lei de Kirchoff sobre os circuitos candidatos devem existir somente para aqueles que venham a ser adicionados à rede elétrica. Quando o valor de x_{ij} em (3.1d) é 1, ou seja, quando decide-se construir o crcuito candidato $i-j$, o fluxo de potência no circuito $i-j$ é calculado por:

$$f_{ij}^1 - \gamma_{ij}^1(\theta_i - \theta_j) = 0,$$

Caso contrário, ou seja, quando $x_{ij} = 0$, $f_{ij} = 0$, pois

$$-x_{ij}\overline{f}_{ij}^1 \leq f_{ij}^1 \leq x_{ij}\overline{f}_{ij}^1,$$

assegura que que $f_{ij} = 0$.

As restrições (3.1f), (3.1g) e (3.1e) representam, respectivamente, os limites de capacidade de geração nas usinas do sistema e limites de transporte de potência para os circuitos (tanto os existentes como os candidatos) do sistema de transmissão. As restrições (3.1i) representam as condições de integralidade para as variáveis de decisão (variáveis de planejamento da expansão).

Implementações computacionais para tratar deste problema freqüentemente consideram uma variável adicional para resolver os problemas de inviabilidade. Tal variável, o "corte de carga" associado a cada barra de carga do sistema, pode ser vista como uma geração fictícia que torna o problema *sempre* viável pois, em último caso, pode-se cortar

a carga das barras e obter um sistema de transmissão viável. Inserindo as variáveis de corte de carga, o problema de planejamento da expansão de redes de transmissão passa a ser representado por:

$$Minimize\ z = \sum_{ij \in \mathcal{C}} c_{Iij} x_{ij} + \sum_{i \in \mathcal{N}} c_{Oi} r_i \tag{3.2a}$$

$$sujeito\,a:$$

$$-\sum_{j \in \Omega_i^0} f_{ij}^0 - \sum_{j \in \Omega_i^1} f_{ij}^1 + g_i + r_i = d_i,\ \forall i \in \mathcal{N} \tag{3.2b}$$

$$f_{ij}^0 - \gamma_{ij}^0(\theta_i - \theta_j) = 0,\ \forall (i,j) \in \mathcal{E} \tag{3.2c}$$

$$f_{ij}^1 - x_{ij}\gamma_{ij}^1(\theta_i - \theta_j) = 0,\quad \forall (i,j) \in \mathcal{C} \tag{3.2d}$$

$$-x_{ij}\overline{f}_{ij}^1 \leq f_{ij}^1 \leq x_{ij}\overline{f}_{ij}^1,\ \forall (i,j) \in \mathcal{C} \tag{3.2e}$$

$$0 \leq g_i \leq \overline{g}_i,\ \forall i \in \mathcal{N} \tag{3.2f}$$

$$0 \leq r_i \leq d_i,\ \forall i \in \mathcal{N} \tag{3.2g}$$

$$-\overline{f}_{ij}^0 \leq f_{ij}^0 \leq \overline{f}_{ij}^0,\ \forall (i,j) \in \mathcal{E} \tag{3.2h}$$

$$f_{ij} = -f_{ji},\ \forall (i,j) \in \mathcal{E} \tag{3.2i}$$

$$x_{ij} \in \{0,1\},\ \forall (i,j) \in \mathcal{C} \tag{3.2j}$$

onde c_O representa o vetor do custo do corte de carga nas barras, r é o vetor do montante de carga cortado. As restrições (3.2g) restringem o corte a carga própria de cada uma das barras do sistema.

A formulação não-linear insere uma dificuldade a mais para a solução de problemas de planejamento da expansão de redes de transmissão, devido às não linearidades no termo quadrático $[x][\gamma^1]S_1\theta$. A segunda dificuldade está relacionada às decisões de investimento que requerem a utilização de algoritmos combinatórios.

A título de ilustração, vamos considerar agora um sistema de três barras e verificar como fica a formulação não linear para o planejameno da expansão do seu sistema de transmissão. O sistema é mostrado na figura 3.1.

Nas tabelas 3.1 e 3.2 se encontram os dados de barra do sistema e os dados do sistema de transmissão (linhas de transmissão).

Para esse sistema, a formulação não-linear de acordo com (3.2) assume a forma:

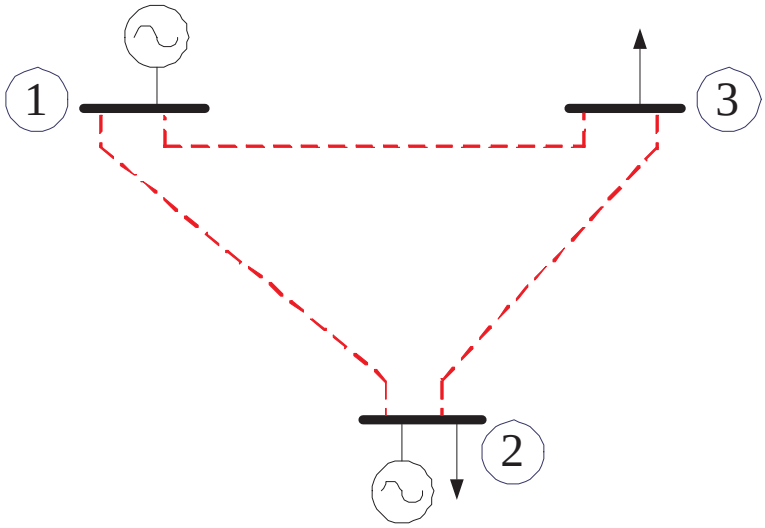

Figura 3.1: Sistema de 3 barras

Tabela 3.1: Dados de barra para o sistema de 3 barras

Barra	Geração (MW)	Carga (MW)
1	200	0
2	100	100
3	0	100

Tabela 3.2: Dados de circuito para o sistema de 3 barras

Circuito	Custo ($)	Reatância (Ω)	Limite de Transmissão (MW)
1-2	10	0,10	100
1-3	15	0,15	100
2-3	10	0,10	100

$$Minimize\ z = 10x_{12} + 15x_{13} + 10x_{23} + 1000r_2 + 1000r_3 \tag{3.3a}$$

$$sujeito\,a: \tag{}$$

$$-f_{12} - f_{13} + g_1 = 0 \tag{3.3b}$$

$$f_{12} - f_{23} + g_2 + r_2 = 100 \tag{3.3c}$$

$$f_{13} + f_{23} + r_3 = 100 \tag{3.3d}$$

$$f_{12} - x_{12}0.10(\theta_1 - \theta_2) = 0 \tag{3.3e}$$

$$f_{13} - x_{13}0.15(\theta_1 - \theta_3) = 0 \tag{3.3f}$$

$$f_{23} - x_{23}0.10(\theta_2 - \theta_3) = 0 \tag{3.3g}$$

$$-x_{12}\overline{f}_{12} \leq f_{12} \leq x_{12}\overline{f}_{12} \tag{3.3h}$$

$$-x_{13}\overline{f}_{13} \leq f_{13} \leq x_{13}\overline{f}_{13} \tag{3.3i}$$

$$-x_{23}\overline{f}_{23} \leq f_{23} \leq x_{23}\overline{f}_{23} \tag{3.3j}$$

$$0 \leq g_1 \leq 200 \tag{3.3k}$$

$$0 \leq g_2 \leq 100 \tag{3.3l}$$

$$0 \leq r_2 \leq 100 \tag{3.3m}$$

$$0 \leq r_3 \leq 100 \tag{3.3n}$$

$$x_{12}, x_{13}, x_{23} \in \{0, 1\} \tag{3.3o}$$

Vemos que as seis primeiras equações correspondem à aplicação da primeira e da segunda leis de Kirchhoff ao modelo linearizado do circuito. As demais correspondem a restrições operativas e a restrição de integralidade das variáveis de investimento.

Verifica-se que temos três variáveis de decisão no problema. Essas variáveis correspondem à decisão de se construir ou não um determinado circuito candidato. Como essas variáveis podem assumir o valor 0 ou 1, teríamos que analisar 2^3 soluções tentativa para escolher a melhor. Extrapolando essa análise para um problema maior, podemos tomar ciência da complexidade dos problemas combinatórios. Considerando agora um problema com 100 variáveis, qualquer técnica que examine todas as possíveis soluções teria que analisar 2^{100} possibilidades ($\approx 10^{30}$). Se um computador puder examinar 100 soluções por segundo, ele levaria 10^{18} anos para encontrar a solução. Imaginando que

esse problema pudesse ser resolvido em minutos, uma instância com 200 variáveis levaria milênios para ser solucionado. Esse fato é conhecido como explosão combinatória, e é o que estimula a pesquisa na área de metaheurísticas capazes de obter soluções ótimas para esse tipo de problema.

A busca pelo ótimo global de problemas de otimização, na prática, não é uma tarefa fácil. A própria incerteza nos dados pode tornar um ponto ótimo sem muito significado. Condições globais e soluções globais somente podem ser encontradas se o problema possuir propriedades de convexidade que garantam que qualquer mínimo (máximo) relativo é um mínimo (máximo) global. Metaheurísticas podem gerar muitas soluções boas diferentes, dando várias opções para o pesquisador tomar decisões sobre a melhor alternativa.

Vamos verificar agora que tipo de problema temos que resolver quando possuimos uma solução candidata. Essas soluções são geradas por métodos metaheurísticos a cada iteração, para análise de sua viabilidade. Considerando a construção dos três circuitos candidatos, essa solução seria representada pelo vetor $x = (111)$. Substituindo-se essa solução na formulação (3.3) o modelo se torna:

$$Minimize\ z = 1000r_2 + 1000r_3 + 35 \tag{3.4a}$$

$$sujeito\,a:$$

$$-f_{12} - f_{13} + g_1 = 0 \tag{3.4b}$$

$$f_{12} - f_{23} + g_2 + r_2 = 100 \tag{3.4c}$$

$$f_{13} - f_{23} + r_3 = 100 \tag{3.4d}$$

$$f_{12} - 0.10(\theta_1 - \theta_2) = 0 \tag{3.4e}$$

$$f_{13} - 0.15(\theta_1 - \theta_3) = 0 \tag{3.4f}$$

$$f_{23} - 0.10(\theta_2 - \theta_3) = 0 \tag{3.4g}$$

$$-\overline{f}_{12} \leq f_{12} \leq \overline{f}_{12} \tag{3.4h}$$

$$-\overline{f}_{13} \leq f_{13} \leq \overline{f}_{13} \tag{3.4i}$$

$$-\overline{f}_{23} \leq f_{23} \leq \overline{f}_{23} \tag{3.4j}$$

$$0 \leq g_1 \leq 200 \tag{3.4k}$$

$$0 \leq g_2 \leq 100 \tag{3.4l}$$

$$0 \leq r_2 \leq 100 \tag{3.4m}$$

$$0 \leq r_3 \leq 100 \tag{3.4n}$$

O problema (3.4) é um modelo de programação linear, de complexidade muito menor que o problema de programação não-linear inteira mista original. Podemos utilizar métodos como o Simplex para a sua solução, verificando a viabilidade da solução.

3.3 Planejamento da Expansão de Sistemas de Transmissão Usando GRAPR

Nesta seção, será mostrado em detalhes como foi feita a implementação computacional do método GRAPR para o planejamento da expansão de sistemas de transmissão. O primeiro passo para a construção do algoritmo consiste em se implementar o método de religamento de caminhos para o planejamento da expansão de sistemas. Primeiramente, usaremos como soluções guia para o método a $i-$ésima solução do método GRASP e uma solução de elite escolhida aleatoriamente de um conjunto de elite $\mathcal{E}$ ($|\mathcal{E}| =$ `TamanhoElite`), e incluir uma nova fase – `Religamento` – no loop principal do GRASP, como ilustrado na Fig. 3.2. Na linha 7, a solução do GRASP é religada com uma solução de elite e vice-versa na linha 8. Adicionalmente, o método de `GRASP+PR`, que consiste simplesmente em um algoritmo GRASP com uma fase extra de Religamento de Caminhos, precisa de um parâmetro adicional, que é o tamanho do conjunto de elite e duas novas funções. Uma para inserir novas soluções no conjunto de elite, linha 5, e outra para selecionar uma solução de elite do conjunto de elite, linha 6.

Para se tornar uma solução de elite, uma solução $\hat{x}$ deve ser melhor, ou seja, ter um custo de investimento menor que o melhor elemento de $\mathcal{E}$, ou melhor que o pior elemento de $\mathcal{E}$ e ser suficientemente diferente de todas as outras soluções de elite (esse grau de diferença é fornecido como um parâmetro pelo usuário). Inicialmente, esse conjunto está vazio e ao custo do pior elemento de elite é atribuído um valor muito grande, até que o conjunto de elite seja totalmente preenchido com `TamanhoElite` elementos ou planos de expansão da transmissão.

Quando se constrói um algoritmo de Religamento de Caminhos, devemos implementar uma função que monta uma estrutura com todos os movimentos possíveis que, quando aplicados à solução inicial, irão conduzir à solução final. Essa função, denominada de

```
procedimento GRASP+PR(MaxIter,Semente,TamanhoElite)
 1  LêDados();
 2  para k = 1,..., MaxIter faça
 3    x̂ ← ConstruçãoAleatória(Semente);
 4    x̂ ← BuscaLocal(x̂);
 5    ℰ ← AtualizaConjuntoElite(x̂);
 6    ℰ_i ← SelecionaSolução(ℰ);
 7    x̂^R ← Religamento(x̂, ℰ_i, x̂);
 8    x̂ ← Religamento(ℰ_i, x̂, x̂^R);
 9    x̄ ← AtualizaSolução(x̂);
 10   fim-para;
 11   retorna x̄
fim GRASP+PR;
```

Figura 3.2: Pseudo-código para o algoritmo de GRASP com Path Relinking.

(`Diff`), encontrada na linha 3 da Fig. 3.3, compara as soluções $\hat{x}^S$ e $\hat{x}^T$ e retorna dois vetores Δ^a e Δ^r que armazenam os índices de todos os circuitos candidatos que devem ser adicionados ou subtraídos da solução $\hat{x}^S$, respectivamente, para se atingir a solução final $\hat{x}^T$.

O segundo aspecto importante de qualquer implementação de um algoritmo de Religamento de Caminhos, é como deve ser feita a escolha dos movimentos realizados nas linhas 6 e 10. Na linha 6, um circuito candidato pertencente ao conjunto de circuitos que devem ser removidos da solução inicial $\hat{x}^S$, Δ^r, é selecionado para ser removido. O índice usado para ordenar esses movimentos é o custo de investimento, de tal forma que a remoção do circuito candidato mais custoso em Δ^r é o movimento "guloso" feito nas linhas 6 e 7. Depois da remoção do circuito candidato mais caro, a viabilidade da rede resultante deve ser verificada (linha 8). Se a rede permanece viável, ou seja, $\hat{z} = 0$, a solução corrente ou o plano de expansão da transmissão candidato $\hat{x}$, pode ser resultado do procedimento de `Religamento de Caminhos` nas linhas 14 e 15.

Se $\hat{z} > 0$, indicando que a rede é inviável, circuitos candidatos devem ser adicionados para reestabelecer a viabilidade. Para escolhermos qual movimento de adição será realizado na linha 10, usamos um índice baseado na sensibilidade do problema de operação (3.2) com respeito à susceptância do ramo, isto é, $\frac{\partial \hat{z}}{\partial \gamma}$. Foi mostrado

procedimento `Religamento(`$\hat{x}^S, \hat{x}^T, \hat{x}^R$`)`

1 $z_{min} = \min\left(\texttt{custo}(\hat{x}^S), \texttt{custo}(\hat{x}^T), \texttt{custo}(\hat{x}^R)\right)$;
$\hat{x}^{min} = \hat{x}^R$;
2 $\hat{x} = \hat{x}^S$;
3 $(\Delta^a, \Delta^r) \leftarrow \texttt{Diff}(\hat{x}^S, \hat{x}^T)$;
4 **enquanto** $|\Delta^a \cup \Delta^r| \geq 2$ **faça**
5 **se** $\Delta^r = \emptyset$ **retorne** $\hat{x}^{min}$;
6 $kl = \arg\max_{ij \in \Delta^r} \{c_{ij}\}$;
7 $\hat{x}_{kl} = 0$, $\Delta^r = \Delta^r \backslash kl$;
8 `Resolva programa` (3.2);
9 **enquanto** $\hat{z} > 0$ **e** $\Delta^a \neq \emptyset$ **faça**
10 $kl = \arg\max_{ij \in \Delta^a} \{h_{ij}\}$;
11 $\hat{x}_{kl} = 1$, $\Delta^a = \Delta^a \backslash kl$;
12 `Resolva programa` (3.2);
13 **fim-enquanto**;
14 **se** $\hat{z} = 0$ **e** $\texttt{custo}(\hat{x}) < z_{min}$
15 $z_{min} = \texttt{custo}(\hat{x})$; $x_{min} = \hat{x}$;
16 **fim-enquanto**;
17 **retorne** $\hat{x}^{min}$;

fim `Religamento`;

Figura 3.3: Pseudo-código para Religamento de Caminhos.

em [29] que podemos estimar esse índice de sensibilidade através da seguinte equação, $\pi_{kl}^{\gamma} = (\pi_l^d - \pi_k^d)(\theta_k - \theta_l), \forall kl \in \mathcal{C}$, onde π_k^d é o multiplicador de Lagrange (variáveis duais) da restrição (3.2b) no problema (3.2). Geralmente, esse índice é negativo, indicando o benefício marginal da inclusão de um novo ramo na rede. Para levarmos em conta os custos de investimento, definimos a função de mérito h_{kl} como a sensibilidade de viabilidade π_{kl}^{γ} dividida pelo custo de cada ramo candidato, dada por:

$$h_{kl} = -\frac{\pi_{kl}^{\gamma}}{c_{kl}}, \forall kl \in \mathcal{C}. \tag{3.5}$$

O movimento de adição no procedimento de `PathRelinking` consiste na seleção e adição do ramo candidato da lista Δ^a que apresenta o maior valor para h_{kl}, como indicado nas linhas 10 e 11 da figura 3.3.

Depois do movimento de adição, a viabilidade da rede resultante deve ser recalculada (linha 8) e, se a rede permanecer inviável, um novo movimento de adição é realizado. Se, por outro lado, $\hat{z} = 0$, indicando uma rede sem violações, a solução atual $\hat{x}$ é, novamente, candidata a ser solução do procedimento de `Religamento` nas linhas 14 e 15.

Ao invés de realizar sempre o melhor movimento, o que poderia prejudicar o potencial de exploração do método de Religamento de Caminhos, GRAPR seleciona os movimentos a serem executados aleatoriamente da lista de movimentos Δ^r ou Δ^a. No primeiro movimento (movimento de remoção), o circuito candidato a ser removido é aleatoriamente selecionado da lista restrita de candidatos ($\texttt{RCL}^r$),

$$\texttt{RCL}^r = \left\{ kl \in \Delta^r | \overline{c} - \alpha^r \left(\overline{c} - \underline{c} \right) \leq c_{kl} \leq \overline{c} \right\}, \tag{3.6}$$

onde $\alpha^r \in [0, 1]$ é um parâmetro, $\overline{c} = \max_{ij \in \Delta^r} \{c_{ij}\}$ e $\underline{c} = \min_{ij \in \Delta^r} \{c_{ij}\}$. Fixamos $\alpha^r = 1$.

No segundo movimento (movimento de adição), a seleção do movimento é feita baseada na função de mérito $h_{kl}, \forall kl \in \Delta^a$. Definindo $\overline{h} = \max_{kl \in \Delta^a}(h_{kl})$ e $\underline{h} = \min_{kl \in \Delta^a}(h_{kl})$, a lista restrita de candidatos de movimentos de adição ($\texttt{RCL}^a$) é calculada pela seguinte equação,

$$\texttt{RCL}^a = \left\{ kl \in \Delta^a | \overline{h} - \alpha^a \left(\overline{h} - \underline{h} \right) \leq h_{kl} \leq \overline{h} \right\}, \tag{3.7}$$

onde α^a é novamente fixado em 1.

Usando conceitos padrão de GRASP, o movimento de seleção é sempre feito aleatoriamente. Entretanto, foi mostrado em [14] que o uso de funções lineares de probabilidade ao invés de funções aleatórias, produz resultados melhores para o método GRASP aplicado ao problema de planejamento da expansão. Isto porque esse tipo de função guia a busca na direção dos melhores ramos candidatos. No algoritmo de GRAPR, também implementamos uma função linear de probabilidade definida como $bias(k) = \frac{1}{k}, k \in \texttt{RCL}$, onde $|\texttt{RCL}|$ é o tamanho da RCL. Sendo $rank(kl)$ e $bias(rank(kl))$, respectivamente, o rank e o valor da função linear de probabilidade para o ramo candidato (kl). A probabilidade de selecionar este ramo candidato da RCL é,

$$P_{kl} = \frac{bias(rank(kl))}{\sum_{(ij)\in\texttt{RCL}} bias(rank(ij))}. \tag{3.8}$$

Introduzindo essas modificações no pseudo-código de Religamento de Caminhos, mostrado na Fig. 3.3, obtemos o procedimento GRAPR, ilustrado na Fig. 3.4.

procedimento GRAPR ($bias$, $\hat{x}^S, \hat{x}^T, \hat{x}^R$)

1 $z_{min} = \min\left(\texttt{custo}(\hat{x}^S), \texttt{custo}(\hat{x}^T), \texttt{custo}(\hat{x}^R)\right)$;
$\hat{x}^{min} = \hat{x}^R$;

2 $\hat{x} = \hat{x}^S$;

3 $(\Delta^a, \Delta^r) \leftarrow \texttt{Diff}(\hat{x}^S, \hat{x}^T)$;

4 **enquanto** $|\Delta^a \cup \Delta^r| \geq 2$ **faça**

5 **se** $\Delta^r = \emptyset$ **retorne** $\hat{x}^{min}$;

6 Constrói $\texttt{RCL}^r$ de acordo com (3.6);

7 $kl = SelecaoRandomica(bias, \texttt{RCL}^r)$;

8 $\hat{x}_{kl} = 0$, $\Delta^r = \Delta^r \backslash kl$;

9 Resolve programa (3.2);

10 **enquanto** $\hat{z} > 0$ **e** $\Delta^a \neq \emptyset$ **faça**

11 Constrói $\texttt{RCL}^a$ de acordo com (3.7);

12 $kl = SelecaoRandomica(bias, \texttt{RCL}^a)$;

13 $\hat{x}_{kl} = 1$, $\Delta^a = \Delta^a \backslash kl$;

14 Resolve programa (3.2);

15 **fim-enquanto**;

16 **se** $\hat{z} = 0$ **e** $\texttt{custo}(\hat{x}) < z_{min}$

17 $z_{min} = \texttt{custo}(\hat{x})$; $x_{min} = \hat{x}$;

18 **fim-enquanto**;

19 **retorna** $\hat{x}^{min}$;

fim GRAPR;

Figura 3.4: Pseudo-código para GRAPR (Greedy Randomized Adaptive Path Relinking).

Agora, iremos descrever, passo a passo, a aplicação do método GRAPR ao problema de Planejamento da Expansão de Sistemas de Transmissão. O objetivo é esclarecer aspectos teóricos do algoritmo utilizando aplicações simples, que fornecem uma perspectiva e um entendimento melhor do algoritmo ao leitor.

Utilizaremos como caso exemplo de planejamento, um sistema de 3 barras, ilustrado na figura 3.5. Os dados relevantes para esse sistema podem ser encontrados na tabela 3.3.

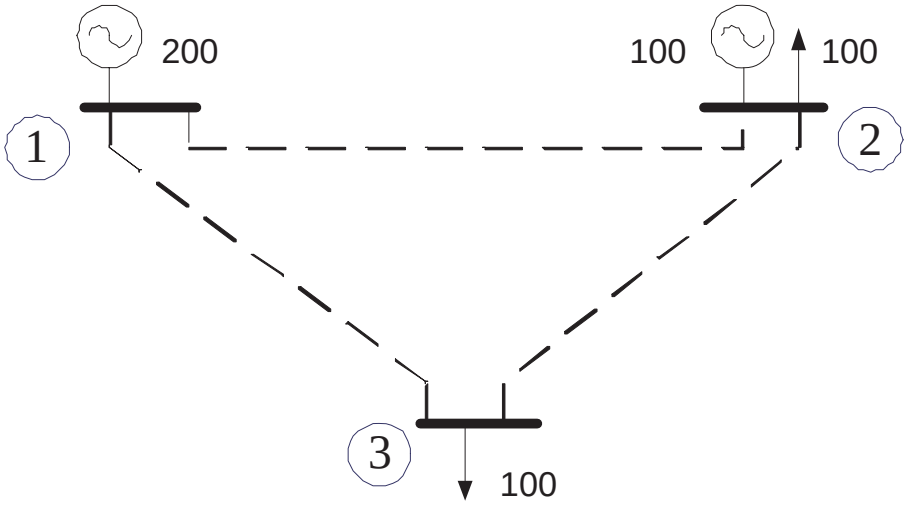

Figura 3.5: Sistema teste de 3 barras

Tabela 3.3: Dados de linha para o sistema de 3 barras.

Circuito	Custo ($)	Reatância (Ω)	Limite de Transmissão (MW)
1-2	10	0,10	100
1-3	15	0,15	100
2-3	10	0,10	100

Inicialmente, o plano de expansão da rede é um conjunto vazio. Chamando esse conjunto de $\mathcal{X}$, teremos $\mathcal{X} = \emptyset$. Os circuitos candidatos, na fase de construção, são ordenados de acordo com o índice de sensibilidade multiplicador de Lagrange, associado com o problema de corte de carga. Dividindo-se este índice pelo custo do circuito, temos a função gulosa utilizada na construção da LRC. Vamos supor que os multiplicadores de Lagrange, ou índices de sensibilidade do problema de corte de carga, para os circuitos candidatos, sejam aqueles mostrados na tabela 3.4.

Tabela 3.4: Multiplicadores para o sistema de 3 barras.

Multiplicadores		Índices/custo
π_{1-2}	-3,0	-0,3
π_{1-3}	-1,0	-0,07
π_{2-3}	-2,0	-0,2

Tabela 3.5: Multiplicadores para o sistema de 3 barras.

Multiplicadores		Índices/custo
π_{1-3}	-3,0	-0,3
π_{2-3}	-2,5	-0,17

Os índices negativos indicam variações marginais nos custos em decorrência de variações na susceptância dos ramos. De acordo com a fórmula de construção da LRC, um circuito candidato pode ser incluído na lista se a função gulosa, ou a razão índice/custo ($h(x_i)$) deste circuito, estiver entre os limites $[h^{min}, h^{min} + \alpha(h^{max} - h^{min})]$. Para este exemplo, com um parâmetro $\alpha = 0,5$, os candidatos $1-2$ e $2-3$ são incluídos na LRC e $1-2$, escolhido aleatoriamente da lista, adicionado à solução. Como essa solução não é viável, supomos agora que os multiplicadores para o novo problema de corte de carga sejam aqueles mostrados na tabela 3.5.

O candidato $1-3$ é incluído na LRC e adicionado à solução. Essa solução construída é viável e dada por $sol = [110]$.

Antes de prosseguirmos para a fase de busca local, vamos introduzir o conceito de vizinhança. Considere um grafo $G(\mathcal{N}, \mathcal{A})$, onde o conjunto de nós $\mathcal{N}$ é formado por todas as configurações de rede possíveis, e o conjunto de arcos $\mathcal{A}$ representam as trocas $1:1$ entre dois nós. A figura 3.6 representa o grafo G para o sistema exemplo de 3 barras e mostra o nó (círculo escuro) associado com a solução da fase de construção. Vamos supor uma vizinhança para a busca local de tamanho 1. Logo, a busca local tenta trocar um candidato adicionado por outro candidato, removendo inicialmente um circuito da solução da fase de construção, correspondendo a movimentos para os nós quadrados escuros $[010]$ e $[100]$. Esses nós não correspondem a soluções viáveis porque, ao final da fase de construção, todas as adições desnecessárias foram removidas. Para produzir uma solução melhor, temos que adicionar outros circuitos nessas duas soluções. Para o

exemplo dado, os nós quadrados brancos, $[011]$ e $[101]$, serão analisados.

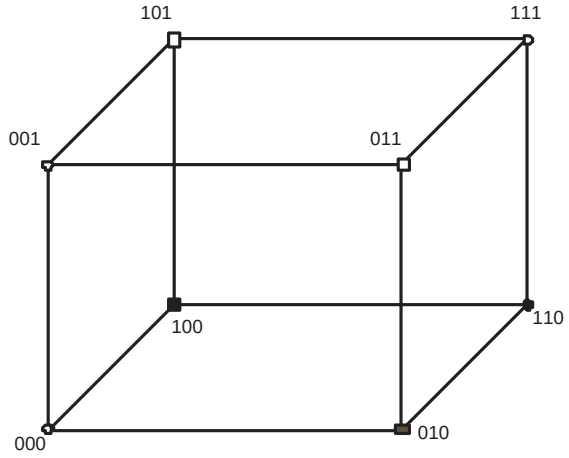

Figura 3.6: Grafo para o sistema teste de 3 barras

Após terem sido analisadas, e sendo constatado que a solução $[101]$ melhora a solução da fase de construção, $[101]$ é um ótimo local e solução da fase de busca local. Conseqüentemente, ela é incluída em um conjunto de elite.

Para ilustrar a fase de GRAPR, vamos supor que o conjunto de elite, com cardinalidade 2, após a segunda iteração do método, possui dois elementos. As iterações de GRAPR começam neste ponto, já que o conjunto de elite deve estar cheio para que as iterações se iniciem. Se a solução da fase de busca local na segunda iteração for $[110]$, ou seja, a solução guia, e $[011]$ for a solução escolhida do conjunto de elite como solução final, montaremos um vetor de diferenças simétricas entre as duas soluções, podendo visitar as soluções intermediárias $[010]$ e $[111]$. Supondo uma escolha aleatória da solução $[111]$, seu custo é calculado. Como seu custo é maior do que a melhor solução já analisada, ela é descartada. Em seguida, caminhamos para a solução final $[011]$. A melhor solução obtida nesta iteração corresponde à solução $[101]$, encontrada pela fase de busca local.

3.4 Resultados Computacionais para o Planejamento Estático da Expansão de Sistemas de Transmissão

Este capítulo mostra resultados computacionais para a implementação seqüencial dos algoritmos analisados nesta tese. O GRASP para o Planejamento da Expansão de Sistemas de Transmissão, que é a base do método proposto, foi implementado utilizando-se as lin-

guagens de programação Fortran e C. Os resultados apresentados neste capítulo foram obtidos em um computador PC-Pentium III, 500MHz com 192Mbytes de memória RAM e publicados em [26].

Dois casos reais de Planejamento da Expansão de Sistemas de Transmissão foram utilizados nos estudos. O primeiro caso corresponde ao sistema reduzido da região Sul Brasileira, possuindo dois níveis de alta tensão. Esse caso já foi discutido em vários trabalhos, incluindo [14, 17, 83]. O segundo caso corresponde ao sistema reduzido da região Sudeste, que já foi estudado em [14].

Os métodos de Religamento de Caminhos e GRAPR foram implementados além do algoritmo de GRASP com um conjunto de soluções de elite. As soluções guia eram compostas por uma solução obtida pelo GRASP e uma solução de elite escolhida aleatoriamente do conjunto de elite. Para descobrirmos o potencial de GRAPR, quatro tipos de estudos foram realizados. Cada um utilizava um algoritmo específico.

- **GRASP**
- GRASP com Religamento de Caminhos, **GRASP+PR**
- **GRAPR-10**
- **GRAPR-50**

Os dois últimos estudos utilizaram 10 iterações e 50 iterações de GRAPR respectivamente. Cada caso foi executado dez vezes, com funções de probabilidade linear e aleatória, e cinco sementes iniciais diferentes. Foram utilizadas 500 iterações para o GRASP. O tamanho do conjunto de elite foi fixado em 20, o parâmetro α do GRASP foi ajustado por um procedimento reativo usando $\delta = 1$, a cardinalidade do conjunto $\mathcal{A} = 10$ e valor do $k-$ésimo bloco igual a 50. A estrutura de vizinhança na fase de busca local do GRASP foi de trocas 1 por 1.

Todos os dados relevantes para a execução dos testes publicados a seguir podem ser obtidos no apêndice A.

3.4.1 Sistema Reduzido da Região Sul Brasileira

O sistema reduzido da região Sul Brasileira é composto por 46 nós (2 deles são novas unidades geradoras que devem ser conectadas à rede, nós 28 e 31), 62 ramos existentes

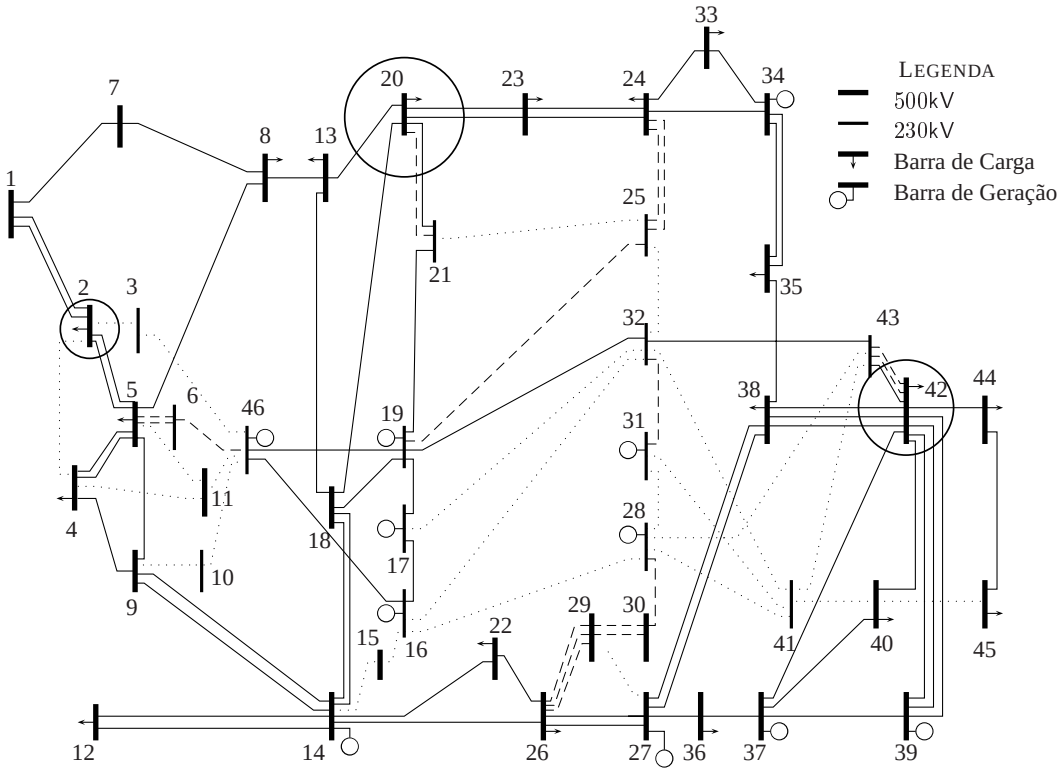

Figura 3.7: Sistema Reduzido da Região Sul Brasileira

e 17 novos caminhos (faixas de passagem). O número de circuitos candidatos é de 237 ($3 \times (62 + 17)$). A Fig. 3.4.1 fornece uma idéia da topologia desse sistema de potência e ilustra os circuitos existentes (linhas sólidas). Os centros de carga mais importantes são indicados por círculos na figura, e os geradores principais estão localizados nas barras $14, 16, 17$ e 19; novas unidades geradoras devem ser conectadas na rede principal nos barramentos 28 e 31. Se formulássemos esse problema como o problema (3.2), ele teria 237 variáveis binárias, 437 variáveis contínuas, e 345 restrições, excluindo os limites nas variáveis.

A solução ótima para esse caso foi publicada pela primeira vez, como o melhor limite superior conhecido em [82], mas a prova de otimalidade foi publicada em [17]. Seu custo de investimento é de US$\$154,26$ milhões, que corresponde à adição de 16 circuitos candidatos. A Tabela 3.6 e a Figura 3.4.1 apresentam, respectivamente, a lista de circuitos candidatos adicionados e a rede resultante onde todas as 16 adições são representadas por linhas pontilhadas. Os custos de investimento obtidos em todas as execuções desse caso teste são mostrados na tabela 3.7.

Tabela 3.6: Melhor Solução Conhecida para o Sistema Reduzido da Região Sul Brasileira.

De	**Para**	**#Ad**	**De**	**Para**	**#Ad**	**De**	**Para**	**#Ad**
26	29	3	5	6	2	28	30	1
42	43	2	19	25	1	20	21	1
24	25	2	46	6	1	31	32	1
29	30	2						

Tabela 3.7: Resultados para todos os Métodos para o Sistema Reduzido da Região Sul Brasileira

	Bias linear			Bias aleatório		
	média	melhor	variância	média	melhor	variância
GRASP	154	154	0.0	158	154	11.12
GRASP+PR	154	154	0.0	156	154	8.97
GRAPR-10	154	154	0.0	157	154	13.9
GRAPR-50	154	154	0.0	156	154	11.29

Note que em todos os casos a solução ótima foi obtida. Analisando os valores médios, podemos ver que o uso da função de probabilidade linear produz resultados melhores que aqueles obtidos quando do uso da função de probabilidade aleatória. GRASP com Religamento de Caminhos ou GRAPR produziram valores médios melhores do que o GRASP "puro". O método proposto, construindo 10 ou 50 caminhos no espaço de busca, achou a solução ótima antes dos outros métodos analisados quando a função de probabilidade linear foi utilizada.

O tempo médio de CPU necessário para processar todos os casos de GRASP foi de $8,5\mathsf{min}$ quando a função linear de probabilidade foi utilizada, e $11,5\mathsf{min}$ quando utilizada a função aleatória. GRASP+PR causa um pequeno aumento no tempo de CPU. No primeiro caso, foi de $10,7\mathsf{min}$ e , no segundo, em torno de $14,0\mathsf{min}$. GRAPR, construindo 10 caminhos a cada iteração, gastou em torno de $16,7\mathsf{min}$ e $20,0\mathsf{min}$ usando uma função linear e uma aleatória, respectivamente. Finalmente, GRAPR construindo 50 caminhos a cada iteração requer muito mais tempo que o caso anterior; foram necessários $20,0\mathsf{min}$ no caso de função linear de probabilidade e $23,0\mathsf{min}$ no caso da função aleatória. As diferenças observadas nos tempos de processamento utilizando funções lineares e aleatórias são devidas à maior eficiência da fase de construção quando utilizando a função linear.

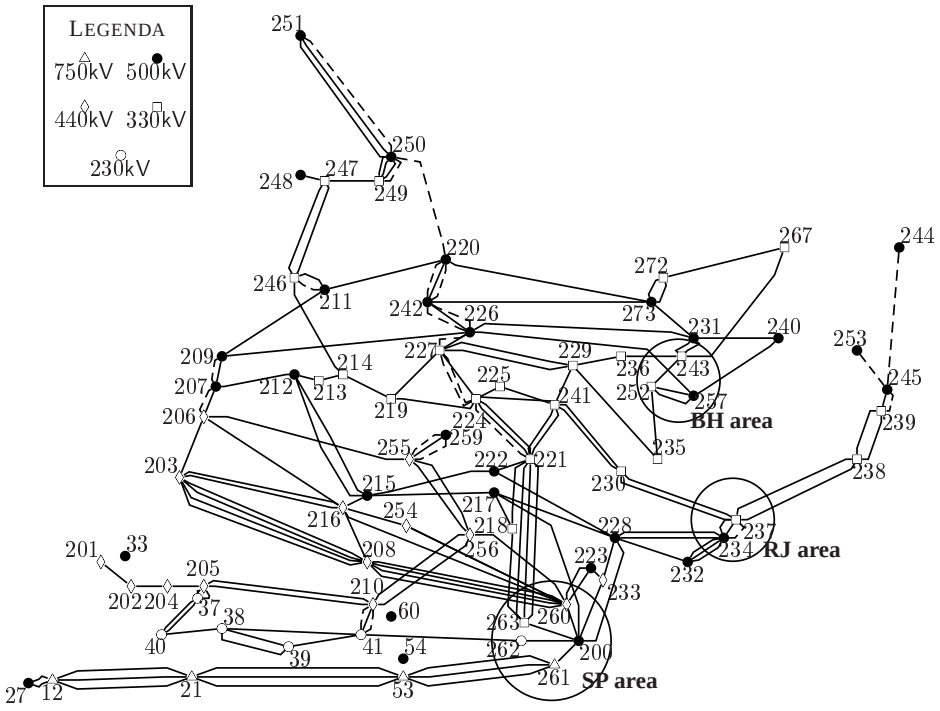

Figura 3.8: Sistema Reduzido da Região Sudeste Brasileira.

3.4.2 Sistema Reduzido da Região Sudeste Brasileira

O sistema reduzido da Região Sudeste tem 79 nós e 155 ramos existentes. A Figura 3.4.2 mostra esse sistema, ilustrando os circuitos existentes através de linhas sólidas e centros de consumo principais por círculos. As principais unidades geradoras estão localizadas nos nós $21, 27, 203, 211$ e 251, e novas unidades de geração devem ser integradas à rede principal nos nós 244 e 253. Formulando esse problema como o problema (3.2), ele teria 429 (3×143) variáveis binárias, 821 variáveis contínuas 663 restrições, excluindo limites nas variáveis. Essa instância é muito mais difícil de se resolver, em comparação com o sistema da Região Sul, devido não somente ao maior número de circuitos candidatos incluídos na solução ótima, como também à necessidade de seleção desses circuitos em cinco diferentes níveis de tensão (750kV, 500kV, 440kV, 330kV e 230kV).

A solução ótima para esse caso tem um custo de investimento de US$\$422$ milhões, obtido com a construção de 24 circuitos candidatos. Essa solução foi publicada pela primeira vez (como um limite superior) em [14], mas demonstrado ser ótima em [7]. A Tabela 3.8 e a Figura 3.4.2 apresentam, respectivamente, a lista de circuitos candidatos adicionados e a rede resultante, onde linhas pontilhadas representam as adições de cir-

Tabela 3.8: Melhor Solução Conhecida para o Sistema Reduzido da Região Sudeste Brasileira.

De	**Para**	**#Ad**	**De**	**Para**	**#Ad**	**De**	**Para**	**#Ad**
224	227	2	210	41	2	255	259	2
220	242	2	226	242	2	220	250	1
234	237	1	221	224	1	245	253	1
245	239	1	244	245	1	226	259	1
211	246	1	226	227	1	250	251	1
207	206	1	207	209	1	249	250	1
216	215	1						

Tabela 3.9: Resultados para todos os Métodos para o Sistema Reduzido da Região Sudeste Brasileira

	Bias linear			Bias aleatório		
	média	melhor	variância	média	melhor	variância
GRASP	431.8	424	124.2	454.0	443	65.5
GRASP+PR	429.0	424	68	446.4	430	144.8
GRAPR-10	427.6	424	10.8	447.6	443	67.7
GRAPR-50	423.6	422	0.8	445.8	443	9.2

cuitos necessárias. Custos de investimento obtidos em todos os estudos com esse caso são resumidos na Tabela 3.9.

Pode ser visto que a solução ótima foi encontrada apenas uma vez, usando o método GRAPR e construindo 50 caminhos a cada iteração, utilizando-se uma função linear de probabilidade. Com relação à função de probabilidade, os comentários são os mesmos que os do caso anterior, isto é, a fase de construção utilizando função linear produz melhores resultados do que o uso de uma função aleatória. Analisando os valores médios, podemos verificar que a junção de GRASP com soluções de elite, construindo somente um caminho (Religamento de Caminhos) ou vários caminhos (GRAPR) melhorou a qualidade dos resultados.

Com relação ao tempo de CPU, aproximadamente $25\mathsf{min}$ foram necessários, na média, para se processar todos os 5 casos de GRASP com função linear de probabilidade e $39\mathsf{min}$ para se processar os casos com função aleatória. Quando GRASP e Religamento de Caminhos foram utilizados, os tempos médios de CPU aumentaram para $26\mathsf{min}$ e $40\mathsf{min}$, respectivamente. O método de GRAPR, construindo 10 caminhos a cada iteração, requer, aproximadamente, $34\mathsf{min}$ no primeiro caso e $53\mathsf{min}$ no segundo. Finalmente, construindo-

se 50 caminhos no procedimento de GRAPR, o tempo de CPU gasto foi de, aproximadamente, 37min com a função linear e 56min com a função aleatória.

Com um aumento modesto de 12min no tempo de CPU, em média, para se processar todos os 5 casos, em comparação com o algoritmo de GRASP com função linear de probabilidade, o algoritmo de GRAPR com função linear foi capaz de obter a solução ótima e reduzir o custo total de investimento em $\text{US}\$2$ milhões. É importante ressaltar, que o algoritmo GRASP "puro", não foi capaz de encontrar a solução ótima neste trabalho devido à estrutura de vizinhança utilizada, que foi de trocas $1:1$. No trabalho [11], por exemplo, que utilizou uma vizinhança de trocas $2:2$, a solução ótima foi encontrada duas vezes, utilizando função de probabilidade linear e ajuste reativo do parâmetro α.

Apesar da estatística sobre a variância dos resultados obtidos em execuções sucessivas ser mostrada nas tabelas 3.7 e 3.9, não podemos tirar conclusões sobre a robustez do algoritmo devido ao tamanho reduzido da amostra utilizada nos estudos.

Este capítulo descreveu o problema de Planejamento Estático da Expansão de Sistemas de Transmissão de Energia Elétrica e a formulação matemática utilizada para modelar o problema. Os detalhes da implementação computacional do algoritmo de GRAPR para o planejamento são mostrados, juntamente com os pseudo-códigos do método. Uma aplicação do método a um sistema exemplo de 3 barras, esclarece os passos executados pelo algoritmo em uma iteração. Foram vistos os resultados da aplicação do método GRAPR nos problemas de planejamento da expansão dos sistemas reduzidos da região Sul e Sudeste Brasileiras. Foram feitas comparações com os métodos GRASP e GRASP com Religamento de Caminhos com relação a tempos computacionais e qualidade das soluções encontradas. GRAPR foi capaz de encontrar as soluções ótimas dos casos analisados, com aumentos modestos nos tempos computacionais.

CAPÍTULO 4

APLICAÇÃO AO PROBLEMA DE ESCALONAMENTO DE TAREFAS (JOB SHOP SCHEDULING PROBLEMS)

4.1 Introdução

O Problema de Escalonamento de Tarefas (JSP - do inglês Job Shop Scheduling) é um problema clássico de otimização combinatória que será utilizado neste trabalho como um "benchmark" para se verificar a eficiência da metodologia de resolução proposta. Muitas tarefas na indústria, em empresas, etc., exigem a realização de uma série de operações sujeitas à restrições temporais e de recursos. Restrições temporais exigem que certas operações sejam concluídas antes que outras sejam inicializadas; restrições de recursos exigem que duas operações alocadas no mesmo recurso (ex. uma máquina) não possam ser realizadas simultaneamente. Isso significa que a mesma máquina não pode efetuar duas operações ao mesmo tempo e, uma vez iniciada, a tarefa também não pode ser interrompida. O objetivo é criar uma programação especificando quando cada tarefa deve ser iniciada e que recursos ela usará satisfazendo todas as restrições e exigindo o mínimo tempo possível para sua execução. Esse problema, brevemente descrito acima, é chamado de Problema de Escalonamento de Tarefas.

Em sua forma mais geral, esse problema é NP-completo (consultar [72] para um estudo de complexidade de problemas de otimização), significando que, provavelmente, não existe um procedimento eficiente (por procedimento eficiente entenda-se um algoritmo que requer um número de operações para convergir limitado por um polinômio no tamanho do problema) para achar, exatamente, programações de tempo mínimo para

instâncias arbitrárias do problema. Em decorrência disso, JSP é usualmente resolvido por heurísticas que aproveitam propriedades especiais de cada instância específica.

4.2 Formulação do Problema

No problema de Escalonamento de Tarefas, um conjunto finito de tarefas é processado em um conjunto finito de máquinas. Cada tarefa é caracterizada por uma ordem fixa de operações que devem ser processadas em uma máquina específica, por um período de tempo especificado. Cada máquina só pode processar uma tarefa de cada vez e, uma vez iniciada, esta tarefa deve ser finalizada sem interrupção. Uma programação é uma alocação de operações em cada máquina. O objetivo do JSP é achar uma programação que minimiza o tempo necessário para completar as tarefas. Uma descrição formal do JSP é dada a seguir.

Dado um conjunto $\mathcal{M}$ de máquinas (o tamanho de $\mathcal{M}$ é dado por $|\mathcal{M}|$) e um conjunto $\mathcal{J}$ de tarefas (o tamanho de $\mathcal{J}$ é dado por $|\mathcal{J}|$), $\sigma_1^j \prec \sigma_2^j \prec \cdots \prec \sigma_{|\mathcal{M}|}^j$ é o conjunto ordenado de $\mathcal{M}$ operações da tarefa j, onde $\sigma_k^j \prec \sigma_{k+1}^j$ indica que a operação σ_{k+1}^j só pode ser processada depois que a operação σ_k^j for finalizada. Chamemos de $\mathcal{O}$ o conjunto de operações. Cada operação σ_k^j é definida por dois parâmetros: $\mathcal{M}_k^j$ é a máquina onde σ_k^j é processada e $p_k^j = p(\sigma_k^j)$ é o tempo de processamento da operação σ_k^j. Definindo $t(\sigma_k^j)$ como sendo o instante em que a operação $\sigma_k^j \in \mathcal{O}$ começa a ser processada, uma formulação disjuntiva para o JSP é dada a seguir:

$$
\begin{aligned}
&\text{minimize } C_{\max} \\
&\text{sujeito a: } C_{\max} \geq t(\sigma_k^j) + p(\sigma_k^j), \text{ para todos } \sigma_k^j \in \mathcal{O}, \\
&\qquad t(\sigma_k^j) \geq t(\sigma_l^j) + p(\sigma_l^j), \text{ para todos } \sigma_l^j \prec \sigma_k^j, && \text{(4.1a)} \\
&\qquad t(\sigma_k^j) \geq t(\sigma_l^i) + p(\sigma_l^i) \vee && \text{(4.1b)} \\
&\qquad t(\sigma_l^i) \geq t(\sigma_k^j) + p(\sigma_k^j), \text{ para todos } \sigma_l^i, \sigma_k^j \in \mathcal{O} \text{ satisfazendo } \mathcal{M}_{\sigma_l^i} = \mathcal{M}_{\sigma_k^j}, \\
&\qquad t(\sigma_k^j) \geq 0, \text{ para todos } \sigma_k^j \in \mathcal{O},
\end{aligned}
$$

onde $C_{\max}$ é o tempo a ser minimizado.

Uma solução viável do JSP pode ser construída a partir de uma permutação de $\mathcal{J}$ em cada uma das máquinas em $\mathcal{M}$, observando as restrições de precedência, a restrição que uma máquina só pode processar uma operação por vez e exigindo que uma vez iniciado,

o processamento de uma operação deve ser ininterrupto até a sua finalização. Uma vez obtida, a permutação de $\mathcal{J}$ pode ser analisada com respeito ao seu tempo de execução em $O(|\mathcal{J}| \cdot |\mathcal{M}|)$, veja [87]. Cada conjunto de permutações tem uma programação correspondente, logo, o objetivo do JSP é achar um conjunto de permutações com o menor tempo de execução.

O JSP tem desafiado continuamente pesquisadores de computação. Até instâncias com três máquinas e tempos de processamento unitários, assim como instâncias com três tarefas, são NP-completos. Métodos exatos [6, 20, 21, 22, 42] têm obtido sucesso na resolução de problemas de pequena dimensão, incluindo a instância 10×10 de Fisher e Thompson [34], proposta em 1963 e somente resolvida vinte anos depois. Problemas de dimensão 15×15 usualmente estão além do alcance dos métodos exatos. Para tais problemas, heurísticas eficientes são necessárias. Uma pesquisa de métodos heurísticos para o JSP é mostrada em [76, 88]. Estão incluídas neste estudo regras de despacho, revistas em [37],busca local [62, 63, 88], recozimento simulado [89, 62], busca tabu [87, 68, 63], e algoritmo genéticos [24]. Um estudo compreensivo de técnicas para o JSP pode ser visto em Jain e Meeran [51].

A maioria dos métodos usados para a resolução desse problema, sejam eles de otimização ou de aproximação, o representam usando o modelo de grafo disjuntivo $G = \{N, A, E\}$ de Roy e Sussmann [84]. Neste grafo há um nó para cada operação, onde N é o conjunto de nós (operações) a serem processadas no conjunto de máquinas $\mathcal{M}$. Incluídos dentro de N estão dois nós fictícios, S e T, que correspondem às operações iniciais e finais, respectivamente: $N = S, 1, 2, \ldots, T$. A cada nó do grafo disjuntivo G são associados pesos que correspondem ao tempo de processamento da operação correspondente. Os nós S e T têm peso zero, enquanto que o peso do nó $i \in \{1, \ldots, |\mathcal{J}| \cdot |\mathcal{M}|\}$ é o tempo de processamento da operação i. A é o conjunto de arcos orientados conectando operações consecutivas da mesma tarefa, representando as restrições (4.1a) do problema, de forma que $(i, j) \in A$ indica que a operação i é um predecessor imediato da operação j $(i \prec j)$ na cadeia de operações da tarefa. $E = \{(v, w) | \mathcal{M}_v = \mathcal{M}_w\}$ é o conjunto de arcos não orientados que conectam operações na mesma máquina. Os arcos de E correspondem às restrições (4.1b) da formulação do JSP. Um exemplo de um grafo disjuntivo para um problema de 3 tarefas e 3 máquinas é mostrado na figura 4.1.

Uma programação viável para o problema consiste em se definir uma orientação para os arcos em E. Dada uma orientação em E, podemos computar o tempo inicial de proces-

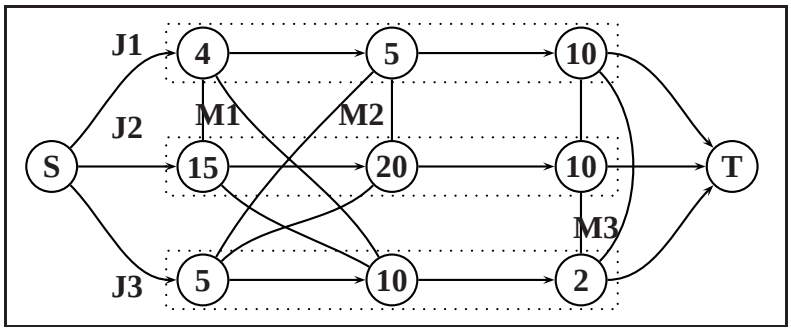

Figura 4.1: Grafo Disjuntivo : representação de uma programação

samento de cada operação computando o caminho mais longo do nó S ao nó correspondente da operação. Em conseqüência, o tempo para processamento de todas as operações será o caminho mais longo em G do nó S ao nó T.

4.3 Escalonamento de Tarefas Usando GRAPR

Nesta seção, entraremos em detalhes da implementação do método GRAPR para o problema de Escalonamento de Tarefas. Iremos descrever como o algoritmo híbrido de GRASP com Religamento de Caminhos é implementado. Na descrição do problema feita no início deste capítulo, uma programação é representada pela permutação das operações em $\mathcal{J}$ nas máquinas em $\mathcal{M}$. A programação é representada por $|\mathcal{M}|$ matrizes de permutação, cada uma com $|\mathcal{J}|$ operações. Cada permutação implica em uma ordem de processamento de todas as operações em cada máquina. Logo, uma solução do JSP é representada como:

$$S = \left\{ (j^S_{1,1}, j^S_{1,2}, \ldots, j^S_{1,|\mathcal{J}|}), (j^S_{2,1}, j^S_{2,2}, \ldots, j^S_{2,|\mathcal{J}|}), \ldots, (j^S_{|\mathcal{M}|,1}, j^S_{|\mathcal{M}|,2}, \ldots, j^S_{|\mathcal{M}|,|\mathcal{J}|}) \right\} \quad (4.2)$$

onde $j^S_{k,i}$ é a i-ésima operação executada na máquina k na solução S.

No método de Religamento de Caminhos original, a cada iteração, todos os movimentos que incorporam atributos da solução guia na solução inicial são analisados, e o melhor movimento é realizado. Para o JSP, Religamento de Caminhos é feito entre uma solução inicial

$$S = \left\{ (j^S_{1,1}, j^S_{1,2}, \ldots, j^S_{1,|\mathcal{J}|}), (j^S_{2,1}, j^S_{2,2}, \ldots, j^S_{2,|\mathcal{J}|}), \ldots, (j^S_{|\mathcal{M}|,1}, j^S_{|\mathcal{M}|,2}, \ldots, j^S_{|\mathcal{M}|,|\mathcal{J}|}) \right\}, \tag{4.3}$$

e uma solução guia

$$T = \left\{ (j^T_{1,1}, j^T_{1,2}, \ldots, j^T_{1,|\mathcal{J}|}), (j^T_{2,1}, j^T_{2,2}, \ldots, j^T_{2,|\mathcal{J}|}), \ldots, (j^T_{|\mathcal{M}|,1}, j^T_{|\mathcal{M}|,2}, \ldots, j^T_{|\mathcal{M}|,|\mathcal{J}|}) \right\}, \tag{4.4}$$

Um pseudo-código para este procedimento é mostrado na figura 3.3.

A diferença simétrica entre as soluções S e T é definida pelo conjunto de índices de cardinalidade $|\mathcal{M}|$ dado por:

$$\delta^{S,T}_k = \left\{ i = 1, \ldots, |\mathcal{J}| | j^S_{k,i} \neq j^T_{k,i} \right\}, k = 1, \ldots, |\mathcal{M}|. \tag{4.5}$$

Esses conjuntos são calculados nas linhas 2 a 4 do pseudo-código. Uma solução intermediária da trajetória de Religamento de Caminhos é visitada a cada passo do loop da linha 5 a 23. Durante um movimento, uma matriz de permutação em S dada por:

$$(\ldots, j^S_{k,i}, j^S_{k,i+1}, \ldots, j^S_{k,q-1}, j^S_{k,q}, \ldots), \tag{4.6}$$

é substituída por uma matriz de permutação

$$(\ldots, j^S_{k,q}, j^S_{k,i+1}, \ldots, j^S_{k,q-1}, j^S_{k,i}, \ldots), \tag{4.7}$$

substituindo as operações $j^S_{k,i}$ e $j^S_{k,q}$, onde $i \in \delta^{S,T}_k$ e q são tais que $j^T_{k,q} = j^S_{k,i}$. Note que a execução desses movimentos pode produzir soluções que violam as restrições de precedência, que são detectadas quando seus custos são computados, linha 11. O procedimento `CalculaCusto()` retorna o custo da solução intermediária $\bar{S}$, que é computado achando-se o caminho crítico do grafo disjuntivo associado com a programação, usando o algoritmo proposto em [87]. Se a função `CalculaCusto()` acha uma programação

procedimento `Religamento(`$\mathcal{M}, \mathcal{J}, \mathcal{O}, Custo, S, T$`)`

1 $S = \left\{(j^S_{1,1}, j^S_{1,2}, \ldots, j^S_{1,|\mathcal{J}|}), \ldots, (j^S_{|\mathcal{M}|,1}, j^S_{|\mathcal{M}|,2}, \ldots, j^S_{|\mathcal{M}|,|\mathcal{J}|})\right\}$;

$T = \left\{(j^T_{1,1}, j^T_{1,2}, \ldots, j^T_{1,|\mathcal{J}|}), \ldots, (j^T_{|\mathcal{M}|,1}, j^T_{|\mathcal{M}|,2}, \ldots, j^T_{|\mathcal{M}|,|\mathcal{J}|})\right\}$;

$c_{g,min} = Custo, S_{g,min} = S$,

2 **para** $k = 1, \ldots, |\mathcal{M}|$ **faça**

3 $\delta^{S,T}_k = \left\{i = 1, \ldots, |\mathcal{J}| \mid j^S_{k,i} \neq j^T_{k,i}\right\}$;

4 **fim-para**

5 **enquanto** $\sum_{k=1}^{|\mathcal{M}|} |\Delta^{S,T}_k| \geq 2$ **faça**

6 $c_{min} = \infty$;

7 **para** $k = 1, \ldots, |\mathcal{M}|$ **faça**

8 **para** $i \in \Delta^{S,T}_k$ **faça**

9 `Faça` q `de forma que` $j^T_{k,q} == j^S_{k,i}$;

10 $\bar{S} = \bar{S} \setminus \left\{\ldots, j^S_{k,i}, j^S_{k,i+1}, \ldots, j^S_{k,q-1}, j^S_{k,q}, \ldots\right\}$;
$\bar{S} = \bar{S} \cup \left\{\ldots, j^S_{k,q}, j^S_{k,i+1}, \ldots, j^S_{k,q-1}, j^S_{k,i}, \ldots\right\}$;

11 $\bar{c} = CalculaCusto(\bar{S})$;

12 **se** $\bar{c} \leq c_{min}$ **então**

13 $c_{min} = \bar{c}; S_{min} = \bar{S}$;

14 $i_{min} = i; k_{min} = k$;

15 **fim-se**

16 **fim-para**

17 **fim-para**

18 $S = S_{min}; Custo = c_{min}$;

19 $\Delta^{S,T}_{k_{min}} = \Delta^{S,T}_{k_{min}} \setminus \{i_{min}\}$;

20 **se** $Custo \leq c_{g,min}$ **então**

21 $c_{g,min} = Custo; S_{g,min} = S$;

22 **fim-se**

23 **fim-enquanto**

24 **retorne** $(S_{g,min})$;

fim `Religamento de Caminhos`;

Figura 4.2: Religamento de Caminhos entre a solução inicial S e a solução guia T para JSP.

inviável, ela retorna um número grande de forma a evitar que essa solução seja visitada em uma trajetória do Religamento de Caminhos.

A cada iteração do loop principal do método, linha 5 até a linha 23, o movimento que pertence ao conjunto $\Delta_k^{S,T}$ e produz as soluções de menor custo é selecionado (escolha "gulosa") para gerar a trajetória do Religamento de Caminhos. Os índices dos movimentos selecionados são obtidos do conjunto $\Delta_k^{S,T}$ (linha 19). Esse processo continua até que restem apenas dois índices no conjunto $\Delta_k^{S,T}$. Nesse ponto, o movimento obtido pela permutação desses dois índices produzirá a solução final T. A melhor solução $(S_{g,min})$ encontrada pela exploração do Religamento de Caminhos é retornada como a solução do procedimento de `Religamento de Caminhos`.

Na implementação proposta para o JSP, um conjunto P de soluções de elite é construído com as soluções produzidas pelo GRASP durante as primeiras $|P|$ iterações. Depois dessa fase inicial, uma solução S_g produzida pelo GRASP é religada com uma ou mais soluções de elite S_e em P. Religamento de Caminhos pode ser aplicado de uma solução do GRASP S_g para uma solução de elite S_e, de uma solução de elite S_e para uma solução do GRASP S_g, ou nas duas direções. Essas duas trajetórias frequentemente visitam soluções intermediárias diferentes.

Para a incorporação de soluções de elite no algoritmo GRASP, utiliza-se um procedimento desenvolvido por Fleurent e Glover [36]. Definimos C_{best} e C_{worst} como os valores da função objetivo da melhor e da pior solução em P, respectivamente. Dadas duas soluções

$$S = \left\{ (j_{1,1}^S, j_{1,2}^S, \ldots, j_{1,|\mathcal{J}|}^S), (j_{2,1}^S, j_{2,2}^S, \ldots, j_{2,|\mathcal{J}|}^S), \ldots, (j_{|\mathcal{M}|,1}^S, j_{|\mathcal{M}|,2}^S, \ldots, j_{|\mathcal{M}|,|\mathcal{J}|}^S) \right\}, \tag{4.8}$$

e

$$T = \left\{ (j_{1,1}^T, j_{1,2}^T, \ldots, j_{1,|\mathcal{J}|}^T), (j_{2,1}^T, j_{2,2}^T, \ldots, j_{2,|\mathcal{J}|}^T), \ldots, (j_{|\mathcal{M}|,1}^T, j_{|\mathcal{M}|,2}^T, \ldots, j_{|\mathcal{M}|,|\mathcal{J}|}^T) \right\}, \tag{4.9}$$

definimos

$$\Delta(S,T) = \sum_{k=1}^{|\mathcal{M}|} |\delta_k^{S,T}| \tag{4.10}$$

como a medida de não similaridade entre S e T. Uma solução S_{gmin} produzida pelo procedimento de Religamento de Caminhos é candidata a inserção no conjunto de elite. S_{gmin} será incluída no conjunto se ela satisfizer um dos seguintes critérios de aceitação:

1. $c_{gmin} < c_{best}$, isto é, S_{gmin} é a melhor solução encontrada até o momento;
2. $c_{best} \leq c_{gmin} < c_{worst}$ e, para todas as soluções de elite $S_p \in P, \Delta(S_{gmin}, S_p) > \Delta_{min}$, isto é, S_{gmin} é melhor que a pior solução em P e suficientemente diferente de todas as outras soluções de elite.

Uma vez aceita em P, S_{gmin} substituirá a pior solução de elite, que será descartada. No JSP, o custo de cada solução encontrada pelo Religamento de Caminhos pode ser calculado em $\mathcal{O}(|\mathcal{J}|.|\mathcal{M}|)$, usando o algoritmo proposto em [87]. Logo, é custoso computacionalmente aplicar Religamento de Caminhos depois de cada iteração do GRASP. A solução encontrada foi a de aplicar Religamento de Caminhos somente quando a solução do GRASP satisfaz um critério de qualidade. Esse critério proposto usa o valor médio μ_n e o desvio padrão σ_n dos custos das soluções do GRASP produzidas durante as primeiras n iterações. Uma solução S_i é utilizada para Religamento de Caminhos se

1. $c(S_i) \leq c_{worst}$, para $i \leq n$;
2. $c(S_i) \leq max(c_{worst}, \mu_n - 2 * \sigma_n)$, para $i > n$.

Religamento de Caminhos também pode ser usado como um esquema de intensificação para o conjunto de elite [4]. Isso é feito aplicando-se o método em cada par de soluções de elite em P e atualizando o conjunto quando necessário. Esse procedimento é repetido até que não ocorram mais alterações em P. Esse tipo de intensificação pode ser feito em uma fase de pós-otimização (usando-se o conjunto final de soluções de elite), ou periodicamente durante a otimização (usando-se o conjunto de elite corrente).

O procedimento de intensificação é executado depois de cada intervalo de `ifreq` iterações durante a otimização. Se nenhuma mudança acontece em P depois de no

mínimo `ifreq` iterações, depois de cada fase de intensificação, os custos das $|P|/2$ piores soluções em P recebem valores muito grandes para garantir que as soluções sejam renovadas. As $|P|/2$ piores soluções são substituídas por soluções geradas nas próximas iterações de GRASP com Religamento de Caminhos. Logo, soluções com custo elevado, mas suficientemente diferentes das soluções em P, são aceitas no conjunto de elite.

No método de GRAPR, cujo pseudo-código é mostrado na figura 4.3, a seleção dos movimentos que produzirão a trajetória não é "gulosa". A seleção é feita construindo-se, inicialmente, uma lista restrita de movimentos `(RCL)` que é um subconjunto de todos os movimentos possíveis. Um movimento escolhido aleatoriamente da `RCL` é executado. Para realizarmos esses procedimentos, precisamos computar a função h como

$$h_{k,i} = \texttt{CalculaCusto}(\bar{S}_{k,i}), i \in \Delta_k^{S,T}, k = 1, \ldots, \mathcal{M}, \tag{4.11}$$

onde $\bar{S}_{k,i}, i \in \Delta_k^{S,T}, k = 1, \ldots, \mathcal{M}$ representa todas as soluções (programações) que podem ser obtidas da solução inicial S trocando-se as operações $j_{k,i}^S$ e $j_{k,q}^S$ como indicado nas equações 4.6 e 4.7. Definindo $\bar{h} = max_{i \in \Delta_k^{S,T}, k=1,\ldots,\mathcal{M}}(h_{k,i})$ e $\underline{h} = min_{i \in \Delta_k^{S,T}, k=1,\ldots,\mathcal{M}}(h_{k,i})$, a lista restrita de candidatos do GRAPR (GRAPR-RCL) pode ser computada por

$$\texttt{GRAPR} - \texttt{RCL} = \left\{(k,i) \mid \bar{h} - \alpha^R(\bar{h} - \underline{h}) \leq h_i, i \in \Delta_k^{S,T}, k = 1, \ldots, \mathcal{M} \leq \bar{h}\right\}, \tag{4.12}$$

onde α^R é um parâmetro.

Uma vez selecionado, o índice é removido do conjunto correspondente $\Delta_k^{S,T}$, o que é feito na linha 12 da figura 4.3. Esse processo continua até que restem apenas dois índices em um dos conjuntos $\Delta_k^{S,T}$. Nesse ponto, o movimento obtido pela permutação desses elementos produzirá a solução final. Como GRAPR usa uma seleção aleatória para a construção da trajetória de Religamento de Caminhos, ele constrói vários caminhos (parâmetro $GRAPR_{MaxIter}$) entre uma solução produzida pelo GRASP e todas as soluções do conjunto de elite, melhorando a característica de exploração na região entre as duas soluções guia da trajetória de Religamento de Caminhos. A melhor solução $(S_{g,min})$ encontrada nesse processo é retornada como resultado do método.

procedimento GRAPR(Maxiter,Semente,$\mathcal{M}, \mathcal{J}, \mathcal{O}, Custo, S, T$)

1 $S = \left\{(j^S_{1,1}, j^S_{1,2}, \ldots, j^S_{1,|\mathcal{J}|}), \ldots, (j^S_{|\mathcal{M}|,1}, j^S_{|\mathcal{M}|,2}, \ldots, j^S_{|\mathcal{M}|,|\mathcal{J}|})\right\}$;

$T = \left\{(j^T_{1,1}, j^T_{1,2}, \ldots, j^T_{1,|\mathcal{J}|}), \ldots, (j^T_{|\mathcal{M}|,1}, j^T_{|\mathcal{M}|,2}, \ldots, j^T_{|\mathcal{M}|,|\mathcal{J}|})\right\}$;

$c_{g,min} = Custo, S_{g,min} = S$,

2 **para** $k = 1, \ldots, |\mathcal{M}|$ **faça**

3 $\delta^{S,T}_k = \left\{i = 1, \ldots, |\mathcal{J}|, j^S_{k,i} \neq j^T_{k,i}\right\}$;

4 **fim-para**

5 **para** $j = 1, \ldots, MaxIter$ **faça**

6 **enquanto** $\sum_{k=1}^{|\mathcal{M}|} |\Delta^{S,T}_k| \geq 2$ **faça**

7 Constrói RCL de acordo com (4.12;

8 $(id, kd) = SelecaoRandomica(Seed, RCL)$

9 Faça q de forma que $j^T_{k,q} == j^S_{k,i}$;

10 $\bar{S} = \bar{S} \setminus \left\{\ldots, j^S_{k,id}, j^S_{k,id+1}, \ldots, j^S_{k,q-1}, j^S_{k,q}, \ldots\right\}$;
$\bar{S} = \bar{S} \cup \left\{\ldots, j^S_{k,q}, j^S_{k,id+1}, \ldots, j^S_{k,q-1}, j^S_{k,id}, \ldots\right\}$;

11 $Custo = CalculaCusto(\bar{S})$;

12 $\Delta^{S,T}_{kd} = \Delta^{S,T}_{kd} \setminus \{id\}$;

13 **se** $Custo \leq c_{g,min}$ **então**

14 $c_{g,min} = Custo; S_{g,min} = S$;

15 **fim-se**

16 **fim-enquanto**

17 **fim-para**

18 **retorna** $(S_{g,min})$;

fim GRAPR;

Figura 4.3: Pseudo-código de GRAPR para o JSP

Agora, vamos dar um exemplo de aplicação do método de GRAPR ao problema de escalonamento de tarefas, mostrando como cada fase do algoritmo é executada. Consideraremos um problema com três tarefas e três máquinas.

Na fase de construção, na qual uma solução viável é construída um elemento por vez, as operações devem ser incluídas na solução observando-se todas as restrições do problema. A função gulosa corresponde ao tempo final para o processamento das tarefas, dada uma programação parcialmente construída, isto é : $h(\sigma \in \mathcal{O}_c) = C_{\max}$ para $\mathcal{O} = \{\mathcal{O}_s \cup \sigma\}$. A heurística gulosa é iniciada com um conjunto solução vazio, e escolhe para processamento uma operação retirada da LRC:

$$LRC = \left\{\sigma \in \mathcal{O}_c \mid h^{min} \leq h(\sigma) \leq h^{min} + \alpha(h^{max} - h^{min})\right\}, \tag{4.13}$$

onde $h^{min} = \min h(\sigma)$, $\sigma \in \mathcal{O}_c$ e $h^{max} = \max h(\sigma), \sigma \in \mathcal{O}_c$. Na fase de busca local, supomos uma vizinhança de permutações $1:1$ e parâmetro $\alpha = 0,5$.

Inicialmente, nosso conjunto solução é nulo e o estado inicial do grafo disjuntivo é dado na figura 4.4:

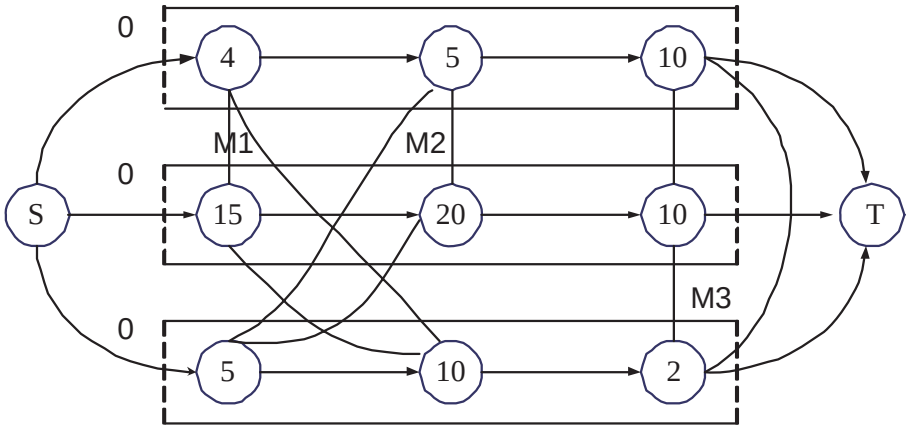

Figura 4.4: Grafo Disjuntivo inicial para o JSP.

No início da fase de construção temos: conjunto solução vazio, $\mathcal{X} = \emptyset$ e $\mathcal{O}_c = \{\sigma_{11}, \sigma_{21}, \sigma_{31}\}$. Consideremos que duas operações tenham atingido qualidade suficiente para entrar na lista, ou seja, $LRC(\alpha) = \{\sigma_{11}, \sigma_{31}\}$. Considerando que a operação σ_{11} tenha sido escolhida aleatoriamente da LRC para ser incluída na operação, teremos: $\mathcal{X} = \{\sigma_{11}\}$ e $\mathcal{O}_c = \{\sigma_{12}, \sigma_{21}, \sigma_{31}\}$, já que σ_{12} é a operação seguinte da primeira tarefa J_1.

Teremos o grafo disjuntivo da figura 4.5.

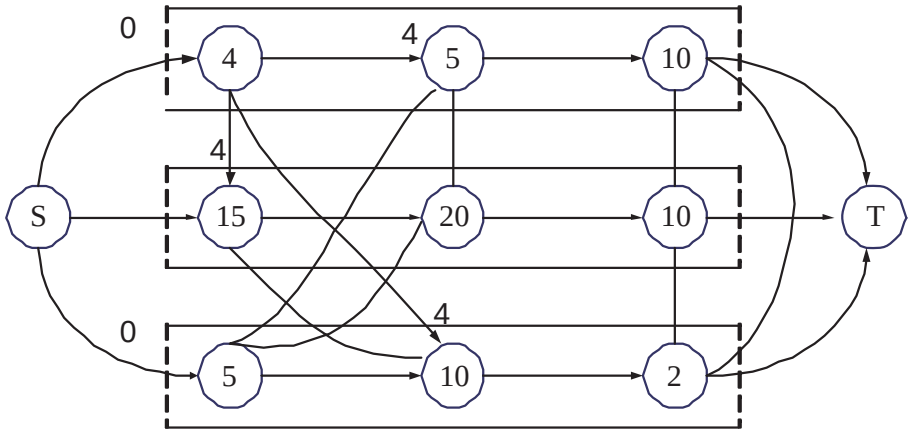

Figura 4.5: Grafo Disjuntivo intermediário para o JSP.

Com somente σ_{31} incluída na LRC, teremos $LRC(\alpha) = \{\sigma_{31}\}$. Essa solução será, necessariamente, incluída na solução. Na próxima iteração da fase de construção teremos: $\mathcal{X} = \{\sigma_{11}, \sigma_{31}\}$ e $\mathcal{O}_c = \{\sigma_{12}, \sigma_{21}, \sigma_{32}\}$, já que σ_{32} é a operação seguinte da tarefa J_3. $LRC(\alpha) = \{\sigma_{12}, \sigma_{21}, \sigma_{32}\}$. O grafo disjuntivo correspondente a essa programação parcial é dado na figura 4.6.

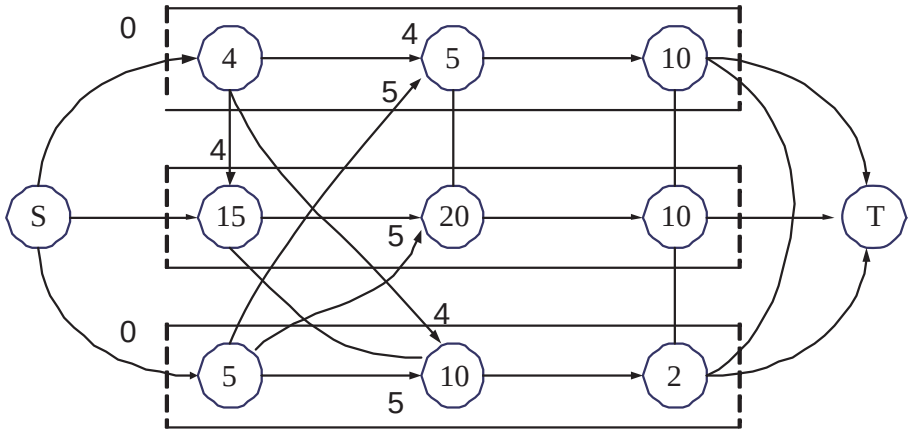

Figura 4.6: Grafo Disjuntivo intermediário para o JSP.

Com σ_{12} tendo sido escolhida para ser incorporada na solução, $\mathcal{X} = \{\sigma_{11}, \sigma_{31}, \sigma_{12}\}$, e $\mathcal{O}_c = \{\sigma_{13}, \sigma_{21}, \sigma_{32}\}$. Com todas essas soluções sendo admitidas na LRC, $LRC(\alpha) = \{\sigma_{13}, \sigma_{21}, \sigma_{32}\}$. Teremos o grafo da figura 4.7.

Na próxima iteração, $\mathcal{X} = \{\sigma_{11}, \sigma_{31}, \sigma_{12}, \sigma_{32}\}$ e $\mathcal{O}_c = \{\sigma_{13}, \sigma_{21}, \sigma_{33}\}$. $LRC(\alpha) = \{\sigma_{13}, \sigma_{33}\}$. O grafo é dado pela figura 4.8.

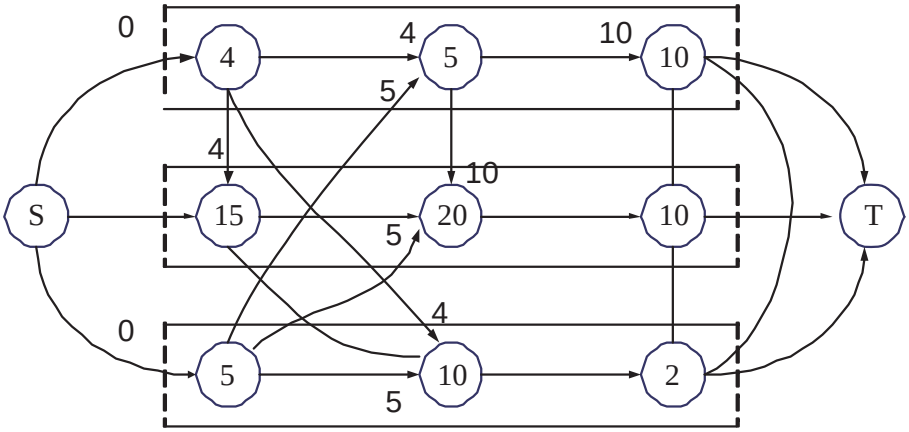

Figura 4.7: Grafo Disjuntivo intermediário para o JSP.

Considerando a admissão da operação σ_{33} na solução: $\mathcal{X} = \{\sigma_{11}, \sigma_{31}, \sigma_{12}, \sigma_{32}, \sigma_{33}\}$ e $\mathcal{O}_c = \{\sigma_{13}, \sigma_{21}\}$. $LRC(\alpha) = \{\sigma_{13}, \sigma_{21}\}$. A figura 4.9 mostra o grafo correspondente.

Com σ_{22} escolhida da LRC : $\mathcal{X} = \{\sigma_{11}, \sigma_{31}, \sigma_{12}, \sigma_{32}, \sigma_{33}, \sigma_{21}\}$ e $\mathcal{O}_c = \{\sigma_{13}, \sigma_{22}\}$. $LRC(\alpha) = \{\sigma_{13}\}$. O grafo é ilustrado na figura 4.10.

Com a escolha de σ_{13}: $\mathcal{X} = \{\sigma_{11}, \sigma_{31}, \sigma_{12}, \sigma_{32}, \sigma_{33}, \sigma_{21}, \sigma_{13}\}$ e $\mathcal{O}_c = \{\sigma_{22}\}$. $LRC(\alpha) = \{\sigma_{22}\}$. O grafo é mostrado na figura 4.11.

Com σ_{22} sendo escolhida, a última operação candidata é σ_{23}. $\mathcal{X} = \{\sigma_{11}, \sigma_{31}, \sigma_{12}, \sigma_{32}, \sigma_{33}, \sigma_{21}, \sigma_{13}, \sigma_{22}\}$ e $\mathcal{O}_c = \{\sigma_{23}\}$. $LRC(\alpha) = \{\sigma_{23}\}$. A figura 4.12 ilustra esse grafo.

Logo, a programação viável fornecida pela fase de construção é dada por:

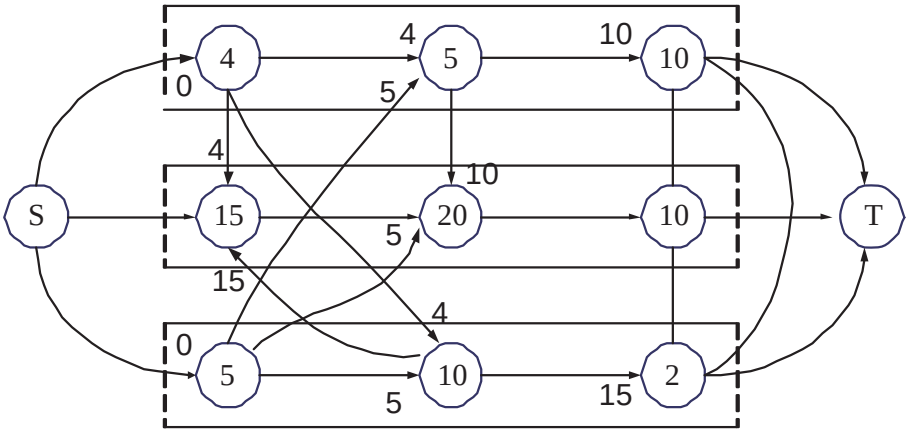

Figura 4.8: Grafo Disjuntivo para o JSP.

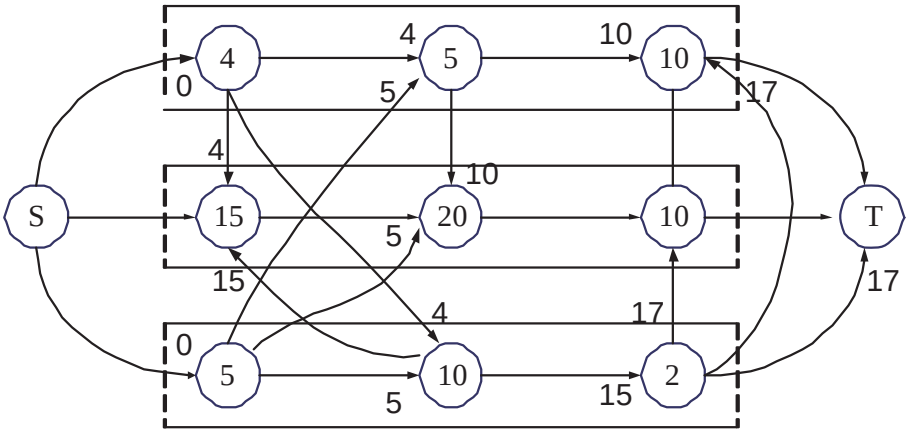

Figura 4.9: Grafo Disjuntivo intermediário para o JSP.

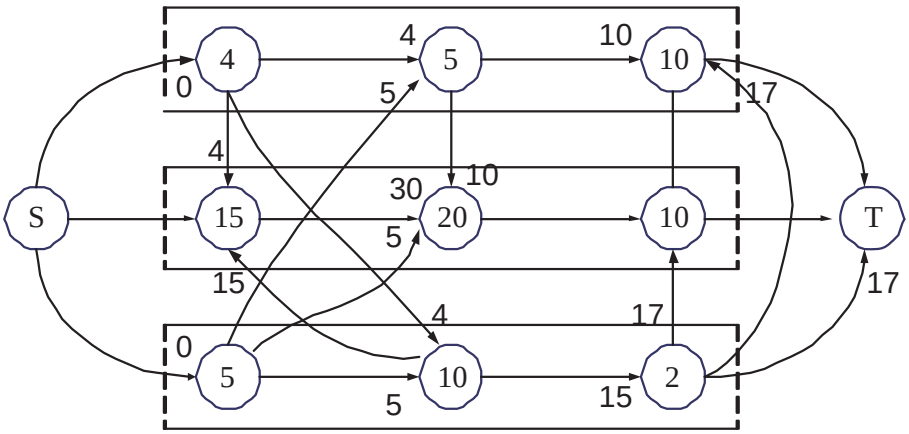

Figura 4.10: Grafo Disjuntivo intermediário para o JSP.

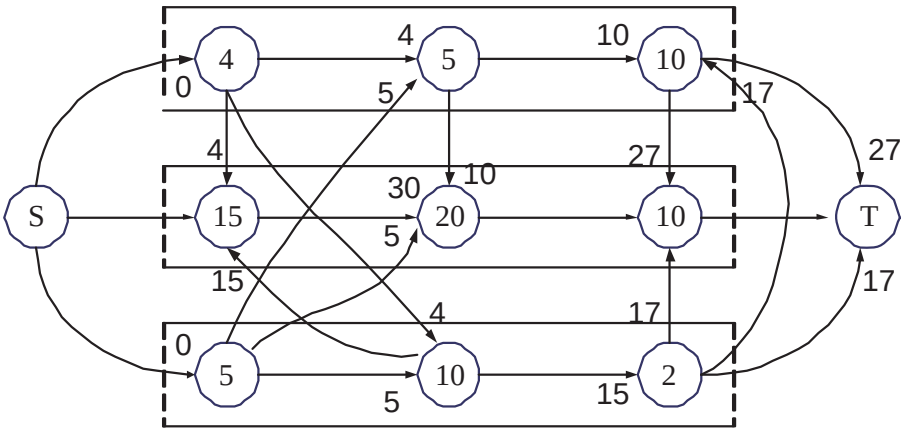

Figura 4.11: Grafo Disjuntivo intermediário para o JSP.

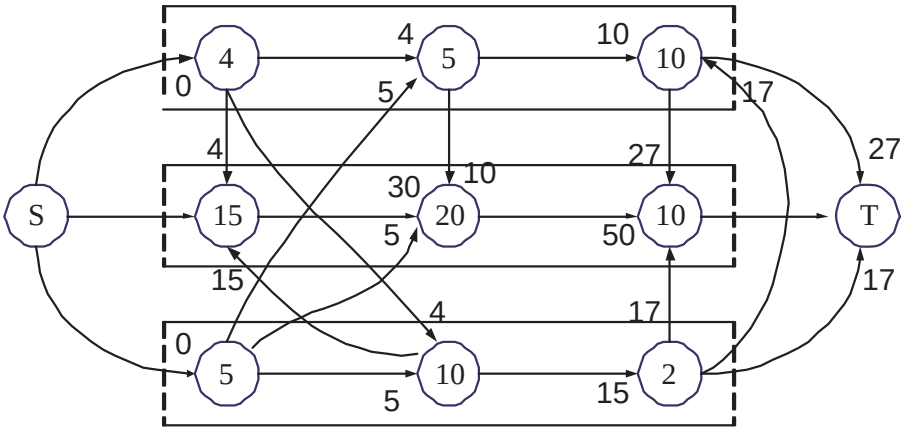

Figura 4.12: Grafo Disjuntivo intermediário para o JSP.

$\mathcal{X} = \{\sigma_{11}, \sigma_{31}, \sigma_{12}, \sigma_{32}, \sigma_{33}, \sigma_{21}, \sigma_{13}, \sigma_{22}, \sigma_{23}\}$. O grafo disjuntivo dessa programação, mostrando o tempo máximo de execução das tarefas é mostrado na figura 4.13. O tempo máximo corresponde ao caminho mais custoso, do nó S ao nó T, ou 60 unidades de tempo. Esse caminho de maior custo é chamado de caminho crítico.

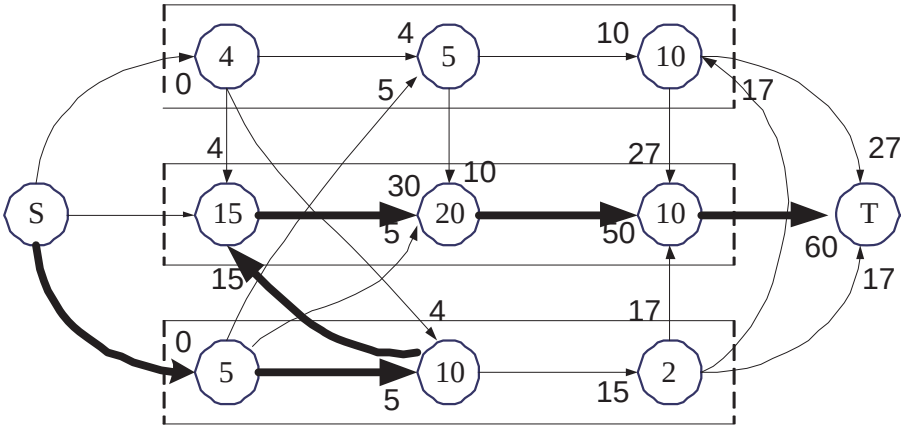

Figura 4.13: Grafo Disjuntivo viável para o JSP.

Na fase de busca local, todos os pares de operações consecutivas no caminho crítico, processadas na mesma máquina, são trocadas. Se a troca melhora o custo da solução, ela é aceita. Como o caminho crítico pode mudar depois da troca, um novo caminho deve ser identificado. Se nenhuma troca de operações consecutivas melhora o custo da solução, a programação é um ótimo local. O caminho crítico é dado por $\mathcal{C}(\mathcal{X}) = \{\sigma_{31}, \sigma_{32}, \sigma_{21}, \sigma_{22}, \sigma_{23}\}$. Trocando-se as operações σ_{21} e σ_{32}, a nova programação se torna $\mathcal{X} = \{\sigma_{11}, \sigma_{31}, \sigma_{12}, \sigma_{21}, \sigma_{32}, \sigma_{33}, \sigma_{13}, \sigma_{22}, \sigma_{23}\}$. O grafo disjuntivo da nova programação

é mostrado na figura 4.14.

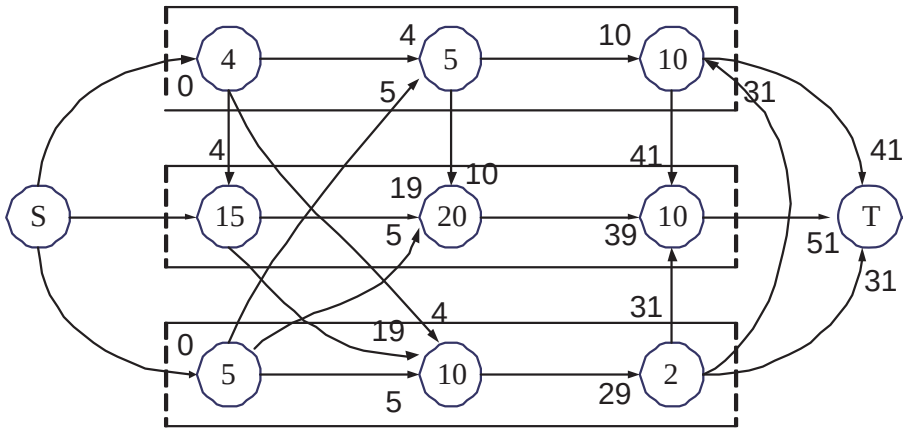

Figura 4.14: Grafo Disjuntivo intermediário para o JSP.

O novo caminho crítico é: $\mathcal{C}(\mathcal{X}) = \{\sigma_{11}, \sigma_{21}, \sigma_{32}, \sigma_{33}, \sigma_{13}, \sigma_{23}\}$, figura 4.15.

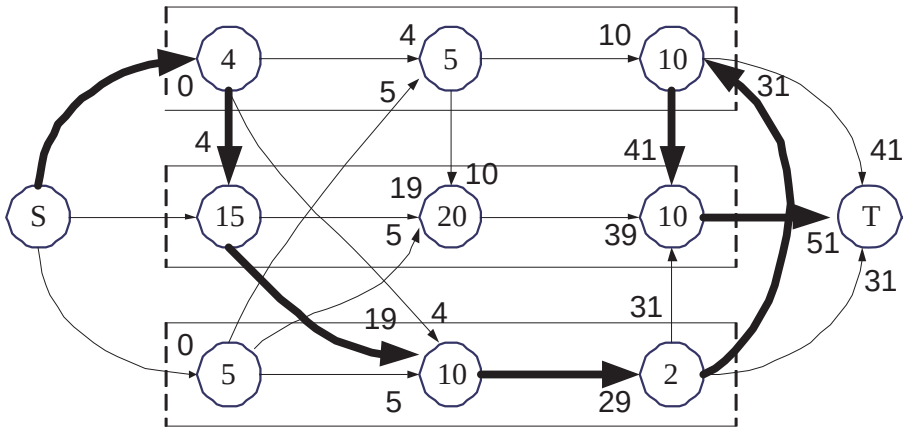

Figura 4.15: Grafo Disjuntivo para o JSP com caminho crítico assinalado.

Podemos realizar quatro permutações de operações de uma mesma máquina do caminho crítico, levando a quatro novas soluções para o problema, dadas por:

$\mathcal{X} = \{\sigma_{11}, \sigma_{31}, \sigma_{12}, \sigma_{21}, \sigma_{32}, \sigma_{33}, \sigma_{22}, \sigma_{23}, \sigma_{13}\}$

$\mathcal{X} = \{\sigma_{11}, \sigma_{21}, \sigma_{31}, \sigma_{12}, \sigma_{32}, \sigma_{13}, \sigma_{33}, \sigma_{22}, \sigma_{23}\}$

$\mathcal{X} = \{\sigma_{11}, \sigma_{31}, \sigma_{12}, \sigma_{32}, \sigma_{21}, \sigma_{33}, \sigma_{13}, \sigma_{22}, \sigma_{23}\}$

$\mathcal{X} = \{\sigma_{21}, \sigma_{11}, \sigma_{31}, \sigma_{21}, \sigma_{32}, \sigma_{33}, \sigma_{13}, \sigma_{22}, \sigma_{23}\}$

Todas tem custo maior que a solução corrente, indicando a otimalidade local da mesma. Os gráficos disjuntivos para essas soluções são mostrados nas figuras 4.16 a 4.19.

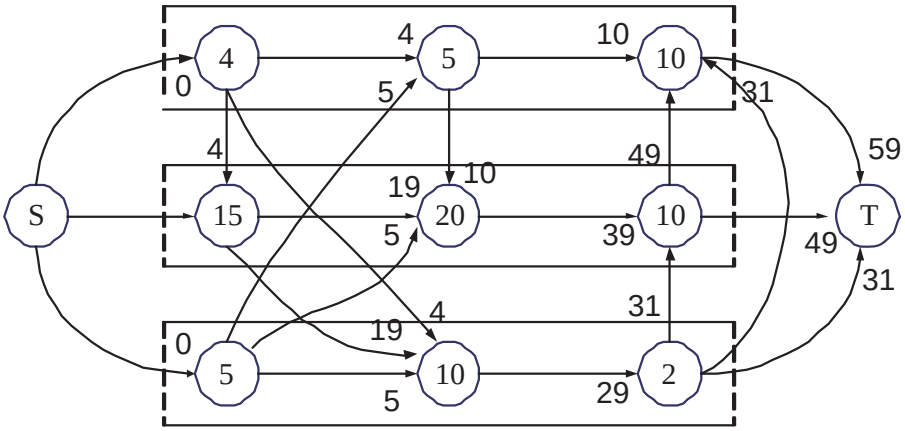

Figura 4.16: Grafo Disjuntivo intermediário para o JSP.

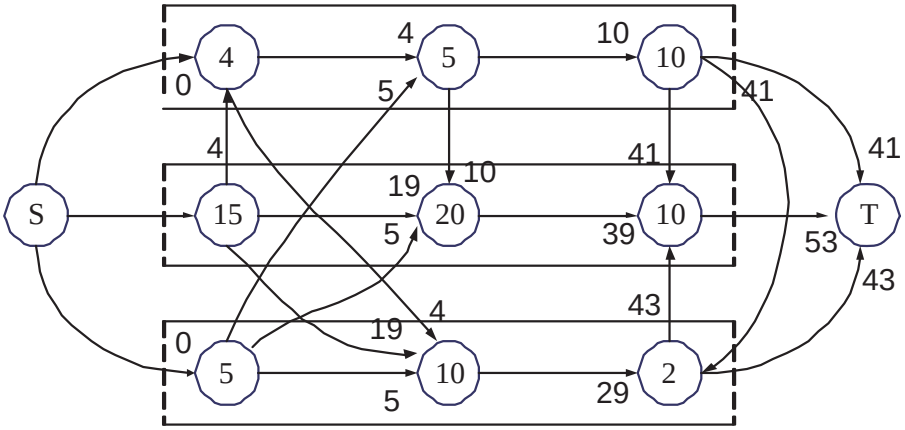

Figura 4.17: Grafo Disjuntivo intermediário para o JSP.

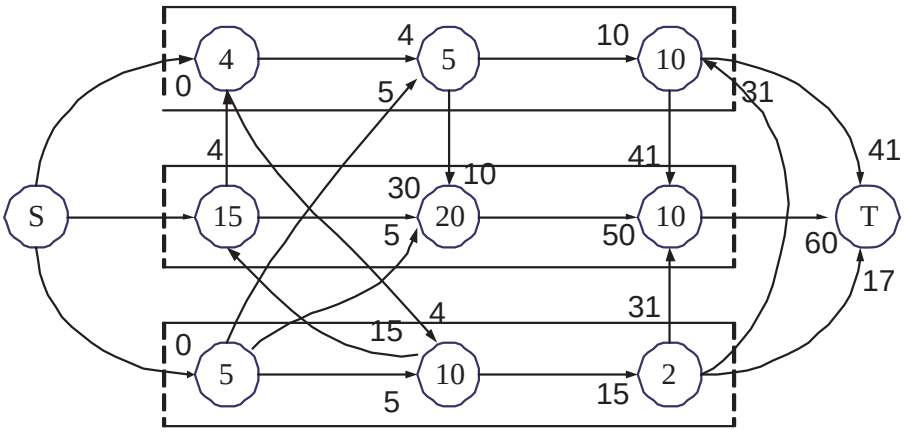

Figura 4.18: Grafo Disjuntivo intermediário para o JSP.

Na fase de Grapr, boas soluções encontradas pela fase de busca local são religadas com soluções de um conjunto de elite. Considerando o religamento de uma solução ótima local dada por: $\mathcal{X} = \{\sigma_{11}, \sigma_{31}, \sigma_{12}, \sigma_{21}, \sigma_{32}, \sigma_{33}, \sigma_{13}, \sigma_{22}, \sigma_{23}\}$ com uma solução escolhida aleatoriamente do conjunto de elite dada por: $\mathcal{X} = \{\sigma_{31}, \sigma_{21}, \sigma_{11}, \sigma_{12}, \sigma_{13}, \sigma_{22}, \sigma_{32}, \sigma_{23}, \sigma_{33}\}$, o vetor de diferenças simétricas entre as duas soluções será: $\delta = \{1, 2, 3, 4, 5, 6, 7, 8, 9\}$. Os movimentos que incorporam atributos da solução guia na solução inicial são incluídos em uma LRC. $LRC(\alpha) = \{3, 5, 6\}$, considerando que esses movimentos satisfazem o critério para entrada na LRC. Visitaríamos 3 soluções intermediárias do espaço de busca, incluindo esses atributos na solução inicial, dadas por:

$\mathcal{X}_1 = \{\sigma_{11}, \sigma_{31}, \sigma_{21}, \sigma_{12}, \sigma_{13}, \sigma_{22}, \sigma_{32}, \sigma_{23}, \sigma_{33}\}$

$\mathcal{X}_2 = \{\sigma_{11}, \sigma_{31}, \sigma_{12}, \sigma_{21}, \sigma_{13}, \sigma_{33}, \sigma_{32}, \sigma_{22}, \sigma_{23}\}$

$\mathcal{X}_3 = \{\sigma_{11}, \sigma_{31}, \sigma_{12}, \sigma_{21}, \sigma_{32}, \sigma_{23}, \sigma_{13}, \sigma_{22}, \sigma_{33}\}$

GRAPR pode escolher, agora, uma solução de forma aleatória ou utilizando funções de probabilidade. Supondo a escolha aleatória da solução 1 e, verificado a sua viabilidade, o seu grafo disjuntivo é dado pela figura 4.20.

O novo caminho crítico é: $\mathcal{C}(\mathcal{X}) = \{\sigma_{11}, \sigma_{21}, \sigma_{22}, \sigma_{23}\}$. Esse caminho é mostrado no gráfico 4.21.

Essa solução é melhor (menor custo) do que a fornecida pela fase de busca local e, se ainda não tiver sido encontrada uma outra de custo menor, em iterações anteriores, indica

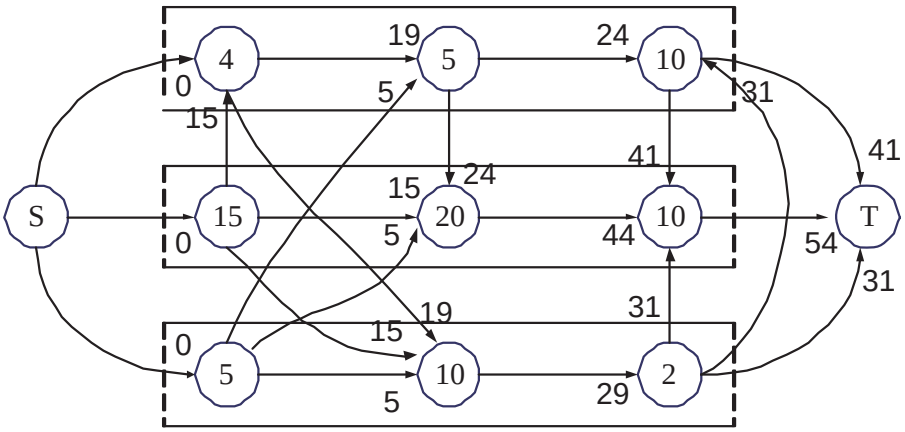

Figura 4.19: Grafo Disjuntivo intermediário para o JSP.

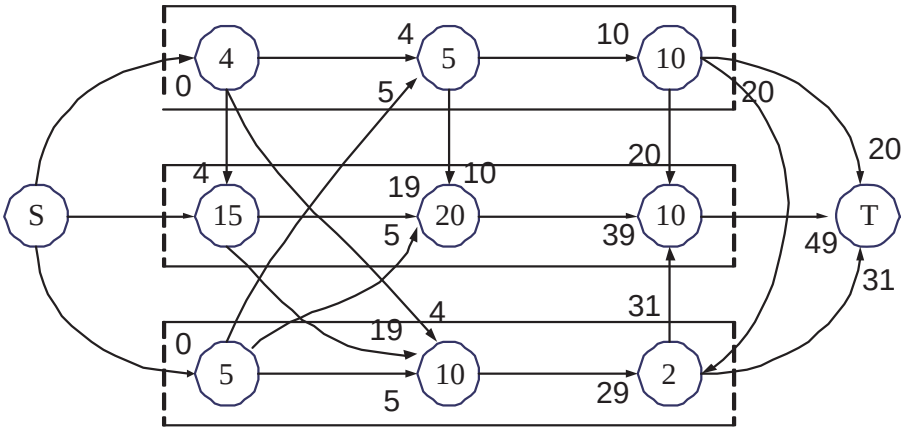

Figura 4.20: Grafo Disjuntivo intermediário para o JSP.

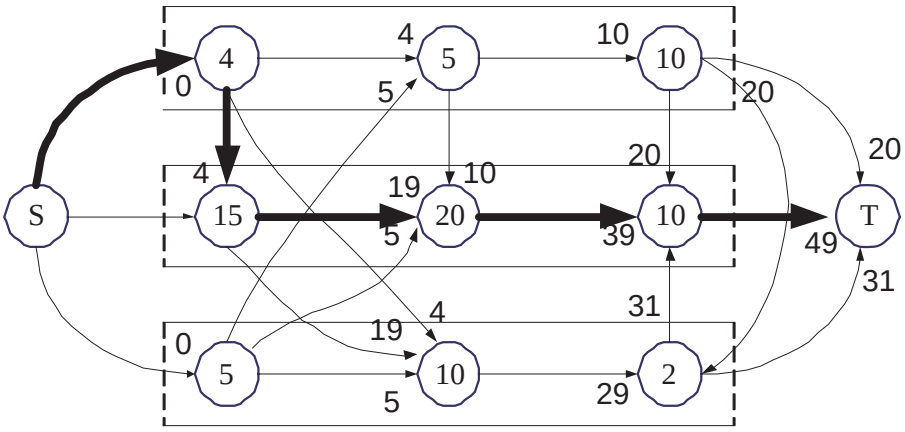

Figura 4.21: Grafo Disjuntivo ao final da iteração de GRAPR para o JSP.

um melhoramento na solução.

4.4 Resultados Computacionais para JSP

Os testes descritos nesta seção foram realizados em 66 instâncias de cinco classes de problemas de escalonamento de tarefas padrão: `abz`, `car`, `la`, `mt` e `orb`. As dimensões do problema variam de 6 a 30 tarefas e de 4 a 20 máquinas. Dois algoritmos foram implementados e testados. Um GRASP com Religamento de Caminhos (`GP+PR`) e o Greedy Randomized Adaptive Path Relinking (`GRAPR`), ambos específicos para problemas de escalonamento de tarefas. Nas duas implementações, o parâmetro α da lista restrita de candidatos `LRC` é escolhido aleatoriamente da distribuição uniforme no intervalo $[0, 1]$ a cada iteração do algoritmo, e é mantido fixo ao longo da mesma. Uma

função de probabilidade aleatória foi utilizada para a seleção dos candidatos da RCL. Para os dois algoritmos testados, (GP+PR) e (GRAPR), utilizamos um conjunto de elite P de tamanho $|P| = 30$ e um fator de diferenciação para inclusão no conjunto de elite de $dif = 25\%$. Os dois métodos também realizaram o procedimento de Religamento de Caminhos entre cada solução obtida pelo GRASP e todas as soluções do conjunto de elite. A cada iteração do GRAPR, 10 trajetórias foram construídas entre cada solução obtida pelo GRASP e todas as demais do conjunto de elite. Finalmente, o parâmetro da lista restrita de candidatos do GRAPR GRAPR-RCL, foi fixado em 1.0.

As tabelas 4.1 e 4.2 ilustram os resultados obtidos para cada classe dos problemas analisados. A tabela mostra o nome do problema, a sua dimensão (número de tarefas e de máquinas), a melhor solução conhecida (BKS), o número total de iterações executadas e o custo da solução obtida pelos dois métodos. Para GRAPR, também é mostrado o tempo de CPU em segundos para execução de 1000 (GRAPR) iterações e o erro percentual relativo da solução obtida pelo (GRAPR) com respeito ao BKS e à solução do (GP+PR).

Note que para verificar o comportamento do algoritmo de (GP+PR) em termos da qualidade da solução, ele foi executado extensivamente para todos os problemas teste considerados. O número de iterações, freqüentemente, está na casa dos milhões, onde cada iteração do GRASP usa uma semente diferente para o gerador de número aleatório. Conseqüentemente, (GP+PR) achou soluções de alta qualidade mais vezes que o (GRAPR). Das 66 instâncias testadas, (GP+PR) achou o BKS em 49 casos (74,2%). Ele achou uma solução dentro de $0,5\%$ do BKS para 50 instâncias ($75,7\%$). Em 56 instâncias ($84,8\%$), a solução do (GP+PR) está dentro de 2% do BKS e em 61 casos ($92,4\%$) ela está dentro de 2% do BKS. A solução do (GP+PR) está dentro de 5% do BKS em 64 instâncias (97%), enquanto que, para o restante, a solução está dentro de $7,5\%$ do BKS.

Nosso objetivo é verificar como as soluções obtidas por (GRAPR) se comparam as melhores soluções conhecidas para um conjunto de problemas teste padrão. Como todas as execuções do algoritmo de (GRAPR) usaram um número máximo de iterações menor que o algoritmo de (GP+PR), usado na comparação, nem todas as execuções de (GRAPR) alcançaram o BKS. Logo, nos casos em que os dois algoritmos alcançaram o BKS, o número de iterações requeridas por cada um é uma comparação de significância maior que pode mostrar a melhor capacidade de busca do novo método. (GRAPR) encontrou o BKS em 31 das instâncias testadas (47%) e a solução equivalente ao (GP+PR) diferente do BKS uma única vez. O número total de iterações requeridas pelo (GRAPR)

para alcançar o BKS ou a solução equivalente ao (GP+PR) foi menor que o requerido pelo (GP+PR) em 25 instâncias e igual em 2 instâncias. Desses resultados podemos concluir que (GRAPR) explora o espaço de busca mais eficientemente que o método de (GP+PR). Esse fato é esperado, já que (GRAPR) constrói vários caminhos entre as duas soluções guia, e não somente um, como faz (GP+PR). A construção desses múltiplos caminhos leva a um acréscimo moderado nos tempos computacionais para a maior parte dos casos.

Tabela 4.1: Resultados experimentais nas classes de problemas abz, car, mt e orb. A tabela mostra o nome do problema, a sua dimensão (tarefas e máquinas), a melhor solução conhecida (BKS), número total de iterações GP+PR realizadas e a melhor solução encontrada pelo GP+PR, o número total de iterações realizadas pelo GRAPR, o tempo de cpu em 1000 iterações de GRAPR e a melhor solução encontrada pelo GRAPR. Finalmente, as duas últimas colunas mostram o erro relativo do GRAPR com respeito ao BKS e a diferença relativa entre as soluções de GRAPR e GP+PR.

				GP+PR		GRAPR			BKS	GP+PR
Problema				itr	sol	itr	time	sol	erro	dif
Nome	$\|\mathcal{J}\|$	$\|\mathcal{M}\|$	BKS	($\times 10^6$)	solução	($\times 10^6$)	(10^3 iter)	solução	(%)	(%)
abz5	10	10	1234	1.0	1234	0.1	52.1	1238	0.3	0.3
abz6	10	10	943	0.3	943	0.1	28.2	947	0.4	0.4
abz7	15	20	665	50.0	692	0.1	1068.5	724	8.1	4.0
abz8	15	20	670	10.0	705	0.1	1343.1	728	8.0	3.2
abz9	15	20	691	1.0	740	0.03	568.1	781	11.5	5.2
car1	11	5	7038	0.001	7038	0.000234	18.0	7038	0.0	0.0
car2	13	4	7166	0.001	7166	0.00026	12.4	7166	0.0	0.0
car3	12	5	7312	50.7	7312	0.02	20.3	7531	2.9	2.9
car4	14	4	8003	0.01	8003	0.00151	24.1	8003	0.0	0.0
car5	10	6	7702	0.5	7702	0.02	49.7	7702	0.2	0.2
car6	8	9	8313	0.01	8313	0.001892	12.0	8313	0.0	0.0
car7	7	7	6558	0.001	6558	0.001592	14.4	6558	0.0	0.0
car8	7	7	8264	0.02	8264	0.000055	49.9	8264	0.0	0.0
mt06	6	6	55	0.00001	55	0.000005	****	55	0.0	0.0
mt10	10	10	930	2.5	930	0.1	77.3	946	1.8	1.8
mt20	20	5	1165	4.5	1165	0.1	327.1	1175	0.9	0.9
orb1	10	10	1059	1.2	1059	0.1	303.6	1075	1.5	1.5
orb2	10	10	888	1.1	888	0.1	95.2	889	0.1	0.1
orb3	10	10	1005	6.5	1005	0.1	251.3	1022	1.7	1.7
orb4	10	10	1005	100.0	1011	0.1	37.7	1040	3.4	2.8
orb5	10	10	887	20.0	889	0.1	113.4	889	0.2	0.0
orb6	10	10	1010	3.5	1012	0.1	390.0	1014	0.4	0.2
orb7	10	10	397	0.03	397	0.027966	167.9	397	0.0	0.0
orb8	10	10	899	1.6	899	0.1	170.0	922	3.6	3.6
orb9	10	10	934	11.1	934	0.1	67.1	948	1.5	1.5
orb10	10	10	944	0.3	944	0.077195	251.6	944	0.0	0.0

Neste capítulo, vimos uma descrição do problema de Escalonamento de Tarefas, a sua modelagem matemática, e como o método de GRAPR foi customizado e implementado computacionalmente para o JSP. Um exemplo de aplicação do método a um problema de 3 tarefas e 3 operações, ilustra os passos do algoritmo em uma iteração. Resultados computacionais da aplicação dos algoritmos de GRAPR e GRASP+PR são mostrados e comparados. Os testes foram feitos em instâncias padrões, encontradas na literatura, de problemas de Escalonamento de Tarefas.

Tabela 4.2: Resultados experimentais nas classes de problemas abz, car, mt e orb. A tabela mostra o nome do problema, a sua dimensão (tarefas e máquinas), a melhor solução conhecida (BKS), número total de iterações GP+PR realizadas e a melhor solução encontrada pelo GP+PR, o número total de iterações realizadas pelo GRAPR, o tempo de cpu em 1000 iterações de GRAPR e a melhor solução encontrada pelo GRAPR. Finalmente, as duas últimas colunas mostram o erro relativo do GRAPR com respeito ao BKS e a diferença relativa entre as soluções de GRAPR e GP+PR.

				GP+PR		GRAPR			BKS	GP+PR
Problema				itr	sol	itr	time	sol	erro	dif
Nome	$\|\mathcal{J}\|$	$\|\mathcal{M}\|$	BKS	($\times 10^6$)	solução	($\times 10^6$)	(10^3 iter)	solução	(%)	(%)
la01	10	5	666	0.0001	666	0.000019	110.1	666	0.0	0.0
la02	10	5	655	0.004	655	0.003382	39.3	655	0.0	0.0
la03	10	5	597	0.01	597	0.02	23.2	597	0.0	0.0
la04	10	5	590	0.001	590	0.001001	17.5	590	0.0	0.0
la05	10	5	593	0.0001	593	0.000008	680.0	593	0.0	0.0
la06	15	5	926	0.0001	926	0.000001	2838.8	926	0.0	0.0
la07	15	5	890	0.0001	890	0.000017	43.4	890	0.0	0.0
la08	15	5	863	0.0003	863	0.000223	92.7	863	0.0	0.0
la09	15	5	951	0.0001	951	0.000005	1642.6	951	0.0	0.0
la10	15	5	958	0.0001	958	0.000001	3219.8	958	0.0	0.0
la11	20	5	1222	0.0001	1222	0.000005	5366.6	1222	0.0	0.0
la12	20	5	1039	0.0001	1039	0.000007	1563.4	1039	0.0	0.0
la13	20	5	1150	0.0001	1150	0.000005	2655.0	1150	0.0	0.0
la14	20	5	1292	0.0001	1292	0.000001	****	1292	0.0	0.0
la15	20	5	1207	0.0002	1207	0.000263	491.3	1207	0.0	0.0
la16	10	10	945	1.3	945	0.02	136.0	961	1.6	1.6
la17	10	10	784	0.02	784	0.02	16.9	787	0.4	0.4
la18	10	10	848	0.05	848	0.02	103.5	848	0.0	0.0
la19	10	10	842	0.02	842	0.02	164.8	852	1.2	1.2
la20	10	10	902	17.0	902	0.02	101.4	911	1.0	1.0
la21	15	10	1047	100.0	1057	0.02	212.7	1116	6.2	5.3
la22	15	10	927	26.0	927	0.02	85.2	962	3.6	3.6
la23	15	10	1032	0.01	1032	0.01	831.7	1032	0.0	0.0
la24	15	10	935	125.0	954	0.02	192.0	967	3.3	1.3
la25	15	10	977	32.0	984	0.02	285.8	1020	4.2	3.5
la26	20	10	1218	3.5	1218	0.02	472.8	1250	2.6	2.6
la27	20	10	1235	10.5	1269	0.05	521.0	1324	6.7	4.1
la28	20	10	1216	20.0	1225	0.05	424.8	1281	5.1	4.4
la29	20	10	1157	50.0	1203	0.05	623.5	1234	6.2	2.5
la30	20	10	1355	3.0	1355	0.05	636.5	1370	1.0	1.0
la31	30	10	1784	0.01	1784	0.01	1828.0	1784	0.0	0.0
la32	30	10	1850	0.0001	1850	0.00005	10.0	1850	0.0	0.0
la33	30	10	1719	0.001	1719	0.000339	24216.5	1719	0.0	0.0
la34	30	10	1721	0.05	1721	0.03	2158.7	1721	0.0	0.0
la35	30	10	1888	0.01	1888	0.017679	5730.6	1888	0.0	0.0
la36	15	15	1268	51.0	1287	0.05	301.4	1318	3.8	2.3
la37	15	15	1397	20.0	1410	0.05	242.8	1413	1.1	0.2
la38	15	15	1196	20.0	1218	0.05	672.5	1259	5.0	3.3
la39	15	15	1233	6.0	1248	0.05	940.9	1316	6.3	5.2
la40	15	15	1222	2.0	1244	0.05	1080.7	1260	3.0	1.3

CAPÍTULO 5

INVESTIGAÇÃO EXPERIMENTAL DA DISTRIBUIÇÃO DE PROBABILIDADE DO TEMPO DE SOLUÇÃO EM METAHEURÍSTICAS DO TIPO GRASP

Este capítulo tem como objetivo estudar a distribuição de probabilidade do tempo de solução para se atingir um valor sub-ótimo de um problema de otimização, para as metaheurísticas analisadas nesta tese, no problema de planejamento da expansão de sistemas de transmissão de energia elétrica, e no problema de escalonamento de tarefas. Metaheurísticas iterativas do tipo GRASP têm um forte apelo à paralelização devido a independência das suas iterações. A paralelização pode ser obtida dividindo as iterações entre os processadores de uma máquina paralela ou de um cluster de PC's. Outro método de paralelização consiste em se fornecer um valor alvo τ sub-ótimo da função objetivo para cada processador e executar o algoritmo até que o primeiro processador ache uma solução com valor melhor ou igual a τ. Neste ponto, o processamento é interrompido.

Muitos trabalhos utilizando as estratégias acima constatam que "speedups" nos tempos de solução medidos foram proporcionais ao número de processadores utilizados nos estudos. Essa observação comum pode ser justificada se a variável aleatória tempo de solução para se atingir o valor alvo tiver uma distribuição de probabilidade exponencial de acordo com a seguinte proposição:

PROPOSIÇÃO 5.1 *Seja $P_\rho(t)$ a probabilidade de uma solução com um certo valor alvo não ter sido encontrada em t unidades de tempo com ρ processos independentes. Se*

$P_1(t) = e^{-t/\lambda}$ com $\lambda \in \mathbb{R}^+$, isto é, P_1 corresponde a uma distribuição exponencial, então $P_\rho(t) = e^{-\rho t/\lambda}$.

PROVA. Seja $F_\rho(t)$ a função de distribuição acumulada que representa a probabilidade de um dado valor alvo ter sido encontrado em até t unidades de tempo com ρ processos independentes. Analogamente, seja $F_1(t)$ a função de distribuição acumulada que representa a probabilidade de um dado valor alvo ter sido encontrado em t unidades de tempo com um processo, dada por $F_1(t) = 1 - e^{-t/\lambda}$, com $\lambda \in \mathbb{R}^+$.

Da definição da função de distribuição acumulada, tem-se que:

$$F_1(t) = P(X \leq t) = 1 - P(X > t)$$
$$F_\rho(t) = P(Y \leq t)$$

Por definição $Y = min(X_1, X_2, \ldots, X_\rho)$, onde $(X_1, X_2, \ldots, X_\rho)$ é uma amostragem de tamanho ρ da distribuição $F_1(t)$. Logo:

$$F_\rho(t) = P(X_1 < t \vee X_2 < t \vee \cdots \vee X_\rho < t) \tag{5.1}$$
$$= 1 - P(X_1 > t \wedge X_2 > t \wedge \cdots \wedge X_\rho > t) \tag{5.2}$$
$$= 1 - \prod_{i=1}^{\rho} P(X_i > t) \tag{5.3}$$

Calculando-se cada termo do produtório em 5.3, tem-se que:

$$P(X_i > t) = 1 - F_1(t) \tag{5.4}$$
$$= 1 - (1 - e^{-t/\lambda}) \tag{5.5}$$
$$= e^{-t/\lambda} \tag{5.6}$$

Substituindo-se 5.6 em 5.3:

$$F_\rho(t) = 1 - e^{-\rho t/\lambda} = F_1(\rho t) \tag{5.7}$$

Portanto, $P_\rho(t) = e^{-\rho t/\lambda} = P_1(\rho t)$. ■

A proposição acima vem da definição da distribuição exponencial. Ela implica em que a probabilidade de se encontrar uma solução com um determinado valor alvo em tempo ρt, usando-se um processo seqüencial, é igual à probabilidade de se encontrar uma solução com custo pelo menos tão bom quanto o valor alvo em tempo t usando-se ρ processos paralelos independentes. Portanto, é possível atingir uma aceleração linear no tempo de obtenção do valor alvo, usando-se vários processos independentes.

Uma proposição análoga pode ser definida para a distribuição exponencial de dois parâmetros (ou distribuição exponencial deslocada).

PROPOSIÇÃO 5.2 *Seja $P_\rho(t)$ a probabilidade de um dado valor alvo não ter sido encontrado em t unidades de tempo com ρ processos independentes. Se $P_1(t) = e^{-(t-\mu)/\lambda}$ com $\lambda \in \mathbb{R}^+$ e $\mu \in \mathbb{R}$, isto é, $P_1(t)$ corresponde a uma distribuição exponencial de dois parâmetros, então $P_\rho(t) = e^{-\rho(t-\mu)/\lambda}$.*

PROVA. Da definição da função de distribuição acumulada, tem-se que:

$$F_1(t) = P(X \leq t) = 1 - P(X > t)$$
$$F_\rho(t) = P(Y \leq t)$$

Por definição $Y = min(X_1, X_2, \ldots, X_\rho)$, onde $(X_1, X_2, \ldots, X_\rho)$ é uma amostragem de tamanho ρ da distribuição $F_1(t)$. Logo:

$$F_\rho(t) = P(X_1 < t \vee X_2 < t \vee \cdots \vee X_\rho < t) \tag{5.8}$$
$$= 1 - P(X_1 > t \wedge X_2 > t \wedge \cdots \wedge X_\rho > t) \tag{5.9}$$
$$= 1 - \prod_{i=1}^{\rho} P(X_i > t) \tag{5.10}$$

Calculando-se cada termo do produtório em 5.10, tem-se que:

$$P(X_i > t) = 1 - F_1(t) \tag{5.11}$$
$$= 1 - (1 - e^{-(t-\mu)/\lambda}) \tag{5.12}$$
$$= e^{-(t-\mu)/\lambda} \tag{5.13}$$

Substituindo-se 5.13 em 5.10:

$$F_\rho(t) = 1 - e^{-\rho(t-\mu)/\lambda} = F_1(\rho t) \tag{5.14}$$

Portanto, $P_\rho(t) = e^{-\rho(t-\mu)/\lambda}$ e $P_1(\rho t) = e^{-(\rho t-\mu)/\lambda}$. ■

Analogamente, essa proposição vem da definição da distribuição exponencial de dois parâmetros. Ela implica em que a probabilidade de se encontrar uma solução com um determinado valor alvo em tempo ρt com um processo seqüencial é igual a $1 - e^{-(\rho t-\mu)/\lambda}$, enquanto que a probabilidade de se encontrar uma solução com custo pelo menos tão bom quanto o valor alvo em tempo t, usando-se ρ processos paralelos independentes, é $1 - e^{-\rho(t-\mu)/\lambda}$. Note que se $\mu = 0$, as probabilidades são iguais e correspondem à proposição 5.1 com um parâmetro. Além disso, se $\rho\mu \ll \lambda$, as duas probabilidades são aproximadamente iguais e é possível se obter "speed-ups" aproximadamente lineares nos tempos de solução com múltiplos processos independentes. Esse comportamento tem sido observado em várias metaheurísticas como Recozimento Simulado e Busca Tabu partindo de um ótimo local. No trabalho [5], foi mostrado que os tempos de solução para a metaheurística GRASP possuem uma distribuição exponencial a dois parâmetros. Para se chegar a essa conclusão, foram utilizados cinco problemas clássicos de otimização combinatória encontrados na literatura para os quais haviam algoritmos GRASP implementados para resolver esses problemas. Para cada GRASP, foram utilizados quatro problemas teste e, para cada uma dessas instâncias, foram determinados três soluções alvo espalhadas entre o melhor e o pior valor produzido pelo GRASP. Para cada solução alvo, foram medidos tempos de solução para se achar um valor da função objetivo no mínimo igual a essa solução e a distribuição desses tempos foi estudada.

Neste capítulo pretende-se fazer um estudo semelhante e verificar se a metaheurística GRAPR proposta possui tempos de solução que se encaixam em uma distribuição exponencial de probabilidade e, conseqüentemente, nos possibilita obter "speed-ups" aproximadamente lineares nos tempos de solução com múltiplos processos independentes.

5.1 Método Experimental de Estudo

Nesta seção vamos mostrar como é feito o estudo experimental para se investigar a distribuição de probabilidade das metaheurísticas utilizadas nesta tese. Primeiramente, realizaremos o estudo para o problema de escalonamento de tarefas e, em seguida, para o

problema de planejamento da expansão de sistemas de transmissão de energia. As metaheurísticas analisadas serão as seguintes:

1. GRASP PURO
2. GRASP+Path Relinking (GRASP com uma fase de Religamento de Caminhos após a fase de busca local)
3. GRAPR

Medimos tempos de cpu para se achar um valor da função objetivo no mínimo igual ao valor da solução alvo. Isso é feito utilizando diferentes soluções alvo para cada problema teste. Esses valores estão entre uma solução distante do ótimo e o melhor valor produzido pelo método analisado. Cada método é executado $n = 200$ vezes para todas as combinações de instância/solução alvo. Para cada uma das 200 simulações de cada combinação, o gerador de números aleatórios é inicializado com um semente distinta e, conseq$\ddot{u}$entemente, as simulações são independentes. Para se comparar as distribuições empíricas e teóricas do estudo, usamos uma metodologia gráfica padrão para análise de dados.

Para cada par instância/solução alvo, os tempos de cpu são ordenados de forma crescente. Associamos com o $i-$ésimo tempo de solução (t_i) uma probabilidade $p_i = (i - \frac{1}{2})/n$, e plotamos os pontos $z_i = (t_i, p_i)$, para $i = 1, \ldots, n$. Essa escolha de p_i é justificada adiante nesta seção. A figura 5.1 mostra um gráfico com a distribuição de probabilidade acumulada empírica para um problema de JSP.

Para estimarmos os parâmetros da distribuição exponencial a dois parâmetros, desenhamos, inicialmente, o gráfico quantil-quantil teórico (ou gráfico Q-Q) para os dados. Para descrevermos os gráficos Q-Q, lembramos que a função de distribuição acumulada para a distribuição exponencial a dois parâmetros é dada por:

$$F(t) = 1 - e^{-(t-\mu)/\lambda}$$

onde λ é a média da distribuição de dados e μ é o deslocamento da distribuição com relação ao eixo das ordenadas. Para cada valor de $p_i, i = 1, \ldots, n$, associamos um quantil-p_i $Qt(p_i)$ da distribuição teórica. Para cada quantil-p_i, temos, por definição, que

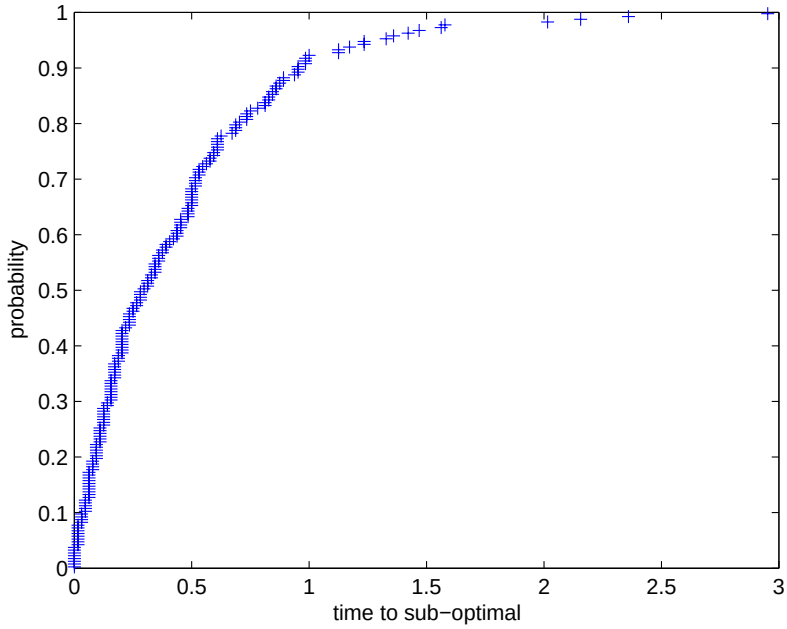

Figura 5.1: Gráfico de distribuição de probabilidade acumulada dos dados medidos

$$F(Qt(p_i)) = p_i$$

Logo, $Qt(p_i) = F^{-1}(p_i)$ e, portanto, para a distribuição exponencial a dois parâmetros, teremos

$$Qt(p_i) = -\lambda ln(1 - p_i) + \mu$$

Os quantis dos dados de uma distribuição empírica são, simplesmente, os dados medidos ordenados. Note que se as probabilidades p_i fossem calculadas como $p_i = i/n$, para $i = 1, \ldots, n$, $Qt(p_n)$ não estaria definida.

Um gráfico quantil-quantil teórico (ou gráfico Q-Q teórico) é obtido plotando-se os quantis dos dados de uma distribuição empírica contra os quantis de uma distribuição teórica. Isso envolve três passos. Primeiramente, os dados (tempos de cpu medidos) são ordenados de forma crescente. Depois, os quantis da distribuição exponencial teórica são obtidos. Finalmente, os tempos medidos são plotados contra os quantis teóricos. Em uma situação na qual a distribuição teórica é uma boa aproximação da distribuição empírica,

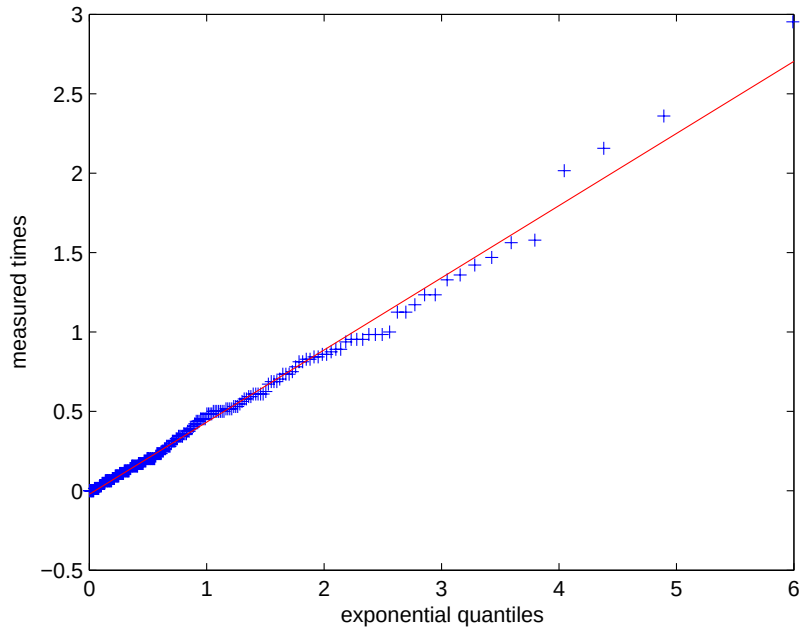

Figura 5.2: Gráfico Q-Q

os pontos em um gráfico Q-Q terão a forma aproximada de uma reta. Se os parâmetros λ e μ da distribuição teórica que melhor se aproximam dos dados medidos pudessem ser estimados a priori, os pontos em um gráfico Q-Q tenderiam a seguir a reta $x = y$. Alternativamente, em um gráfico dos dados (tempos medidos) contra uma distribuição exponencial a dois parâmetros com $\lambda = 1$ e $\mu = 0$, os pontos tenderiam a seguir a linha $y = \hat{\lambda}x + \hat{\mu}$. Isso significa que um gráfico Q-Q teórico compara um conjunto de dados não só com uma distribuição teórica, mas simultaneamente com uma família de distribuições. Portanto, os parâmetros λ e μ de uma distribuição exponencial com dois parâmetros pode ser estimada, respectivamente, pelo coeficiente angular $\hat{\lambda}$ e pelo ponto $\hat{\mu}$ da linha mostrada em um gráfico Q-Q.

O gráfico Q-Q mostrado na Fig. 5.2 é obtido plotando-se os tempos de cpu medidos no eixo das ordenadas contra os quantis de uma distribuição exponencial a dois parâmetros com $\lambda = 1$ e $\mu = 0$ na abscissa dado por $-ln(1 - p_i)$ para $i = 1, \ldots, n$. Para evitar possíveis distorções provocadas por pontos fora da distribuição, não estimamos a média da distribuição pela média dos dados medidos ou por regressão linear nos pontos do gráfico Q-Q. Ao invés disso, estimamos o coeficiente $\hat{\lambda}$ da linha $y = \lambda x + \mu$ usando o quartile superior q_u e o quartile inferior q_l dos dados. Os quartiles superior e inferior

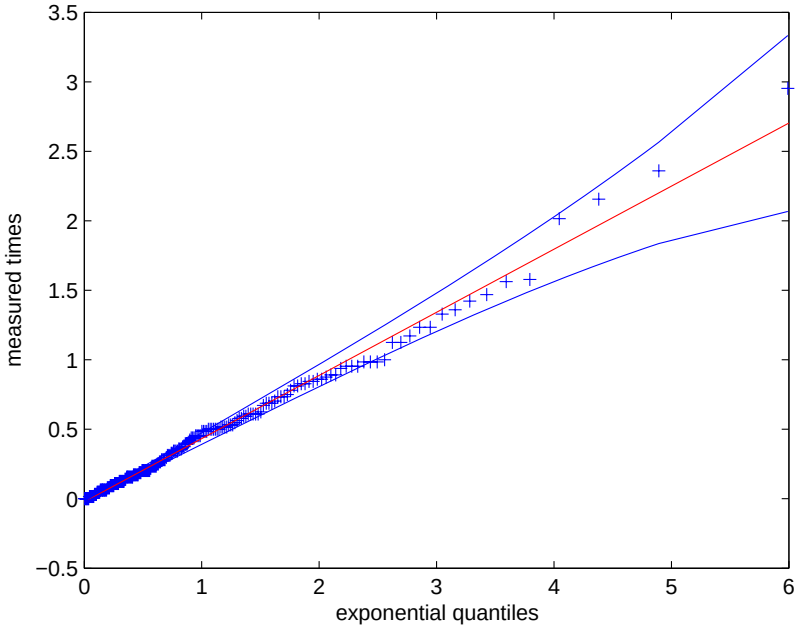

Figura 5.3: Gráfico Q-Q com informação de desvio padrão

são, respectivamente, os quantis $Q(1/4)$ e $Q(3/4)$. A estimativa do coeficiente $\hat{\lambda}$ é

$$\hat{\lambda} = (z_u - z_l)/(q_u - q_l)$$

onde z_u e z_l são o u-ésimo e l-ésimo pontos dos tempos medidos de cpu, respectivamente. Essa estimativa informal da distribuição da média dos dados medidos é robusta já que não será afetada por poucos pontos discrepantes. Para analisarmos a linearidade dos gráficos Q-Q, sobrepomos informações de desvio padrão. Para cada ponto plotado, mostramos o desvio padrão para mais e para menos da linha ajustada ao gráfico. Uma estimativa do desvio padrão para o ponto $z_i, i = 1, \ldots, n$, do gráfico Q-Q é

$$\hat{\sigma} = \hat{\lambda}\sqrt{\frac{p_i}{(1-p_i)n}}$$

A figura 5.3 mostra um exemplo de um gráfico Q-Q com informação de desvio padrão.

Um fato que deve ser salientado é que o desvio padrão natural dos dados gera desvios da linearidade no gráfico Q-Q téorico, mesmo se o modelo da distribuição é válido. A principal razão de se plotar o desvio padrão é o fato de ele nos dar uma noção da variação

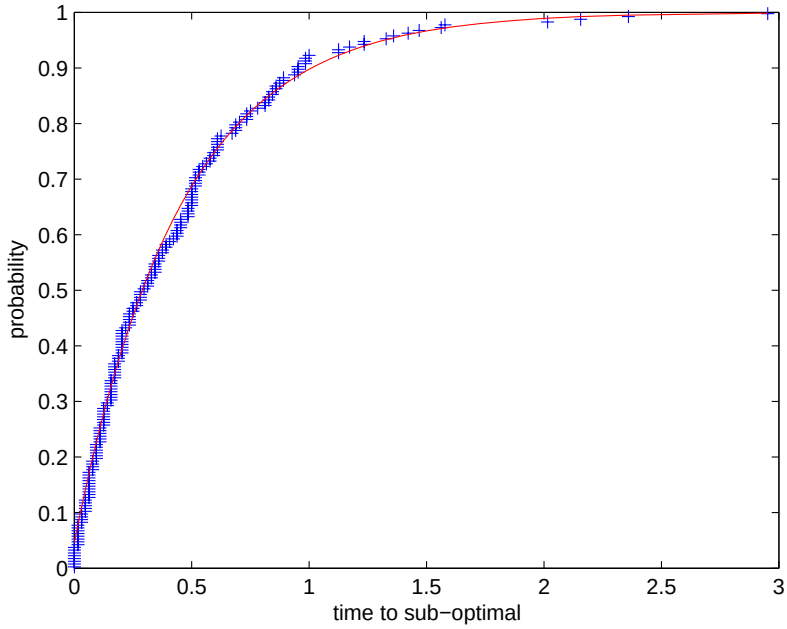

Figura 5.4: Distribuições empírica e teórica superpostas

relativa dos pontos em regiões diferentes do gráfico.

Observa-se que o desvio padrão varia substancialmente no gráfico Q-Q, como mostra a Fig. 5.3. Desvios padrões de pontos perto da metade superior do gráfico são maiores do que os da parte inferior. Depois que os dois parâmetros da distribuição são estimados, um gráfico superposto das distribuições empírica e teórica pode ser feito. A Fig. 5.4 ilustra um gráfico desse tipo.

5.2 Resultados Computacionais

Os Resultados computacionais dos testes realizados serão apresentados nesta seção. Primeiramente, vamos apresentar os resultados para as metaheurísticas GRASP, GRASP+PR e GRAPR aplicadas no problema de Escalonamento de Tarefas. Posteriormente, mostramos os resultados obtidos quando esses métodos são aplicados ao Problema de Planejamento da Expansão de Sistemas de Transmissão. O ambiente computacional usado nos testes foi um computador PC-PentiumIV 2.8GHz com 1MByte de cache L2 e 512MBytes de mémoria RAM.

5.2.1 Problema de Escalonamento de Tarefas

Os testes foram feitos em instâncias de quatro classes de problemas padrão de JSP: abz, mt, orb e la. As dimensões desses problemas variam de 6 a 30 trabalhos e de 4 a 20 máquinas.

O objetivo dos testes seq$\ddot{u}$ênciais foi observar o comportamento do método proposto e compará-lo com os outros métodos já implementados. Comparamos as seguintes heurísticas.

1. GRASP: GRASP puro utilizando, alternadamente, uma função de mérito baseada no tempo restante para finalização das tarefas e no tempo gasto para execução das tarefas.

2. GRASP+Path Relinking: GRASP com uma fase de Religamento de Caminhos após a fase de busca local

3. GRAPR

Queremos comparar os resultados obtidos com GRAPR com as melhores soluções conhecidas para esses problemas e com as soluções obtidas pelos outros métodos. Nas implementações, o parâmetro α da lista restrita de candidatos é escolhido aleatoriamente da distribuição uniforme $[0, 1]$ em cada iteração e é mantido constante durante a mesma. Os testes feitos com GRASP+PR e GRAPR usaram um conjunto de elite de tamanho $|P| = 30$ e um fator de diferenciação para entrar no conjunto de elite de $dif = 25\%$. Nos estudos com GRASP+PR, Religamento de Caminhos foi aplicado entre a solução obtida pelo GRASP e todas as soluções do conjunto de elite. Intensificação e pós-otimização não são aplicadas para GRASP+PR.

Nos estudos com GRAPR, o método foi aplicado cinco vezes entre a solução obtida pelo GRASP e todas as soluções do conjunto de elite. Os problemas teste foram o abz6, mt10, orb5, e la21. Os métodos foram executados duzentas vezes para cada um dos quatro problemas. Cada execução utilizou uma semente para o gerador de número aleatório diferente, o que garantiu a independência das mesmas. As execuções foram interrompidas quando uma solução com custo no mínimo igual a solução alvo foi encontrada. As soluções-alvo usadas para os problemas abz6, mt10, orb5, e la21 foram 970, 970, 930 e 1130, respectivamente. Esses valores estão distantes do melhor valor conhecido para esses

problemas de $2,8\%$; $4,3\%$; $4,8\%$ e $7,9\%$; respectivamente. As curvas de distribuição de probabilidade acumulada são mostradas a seguir para as quatro instâncias analisadas.

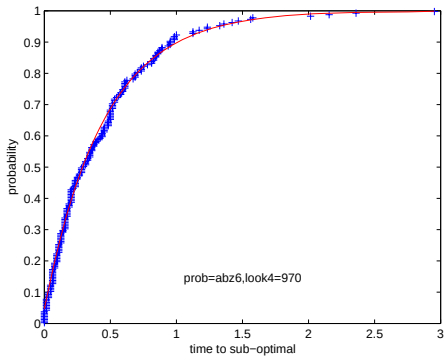

Figura 5.5: Problema abz6, Solução-alvo=970, Método GRASP

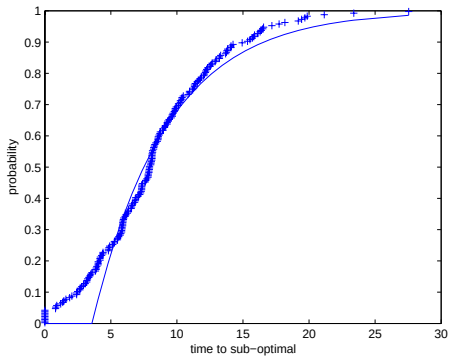

Figura 5.6: Problema abz6, Solução-alvo=970, Método GRASP+PR

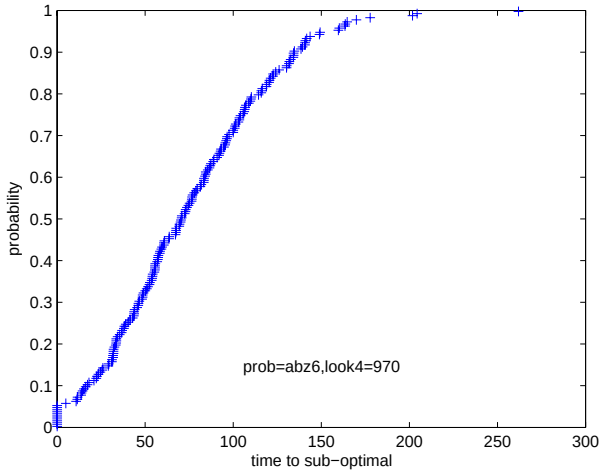

Figura 5.7: Problema abz6, Solução-alvo=970, Método GRAPR

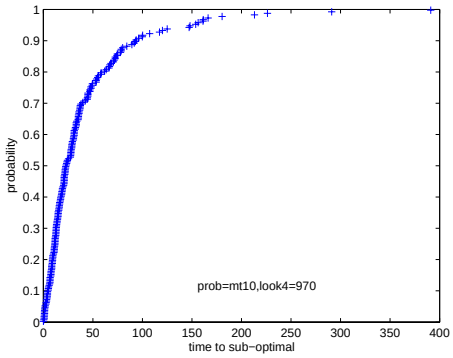

Figura 5.8: Problema mt10, Solução-alvo=970, Método GRASP

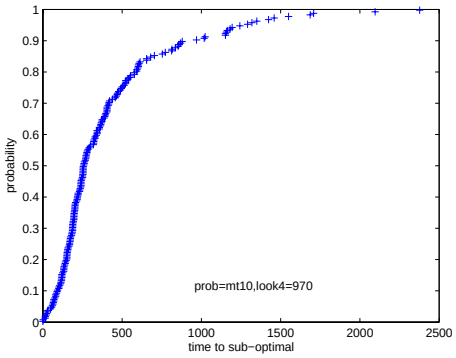

Figura 5.9: Problema mt10, Solução-alvo=970, Método GRASP+PR

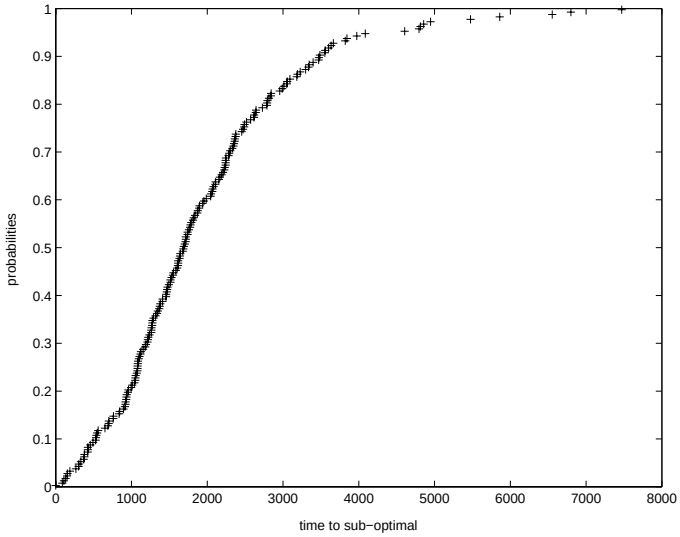

Figura 5.10: Problema mt10, Solução-alvo=970, Método GRAPR

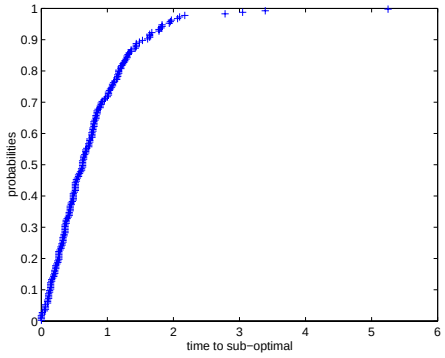

Figura 5.11: Problema orb5, Solução-alvo=930, Método GRASP

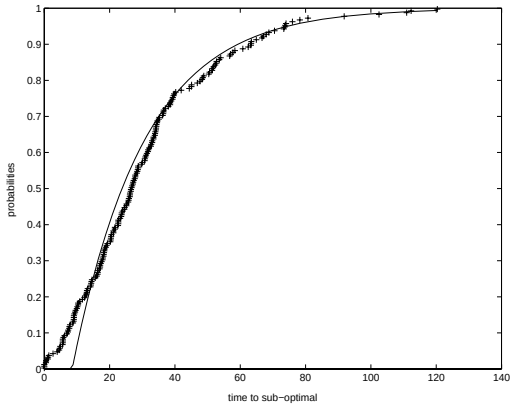

Figura 5.12: Problema orb5, Solução-alvo=930, Método GRASP+PR

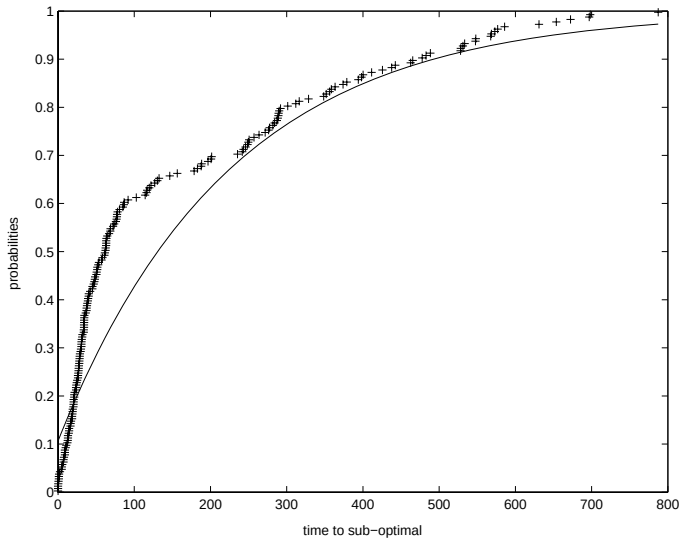

Figura 5.13: Problema orb5, Solução-alvo=930, Método GRAPR

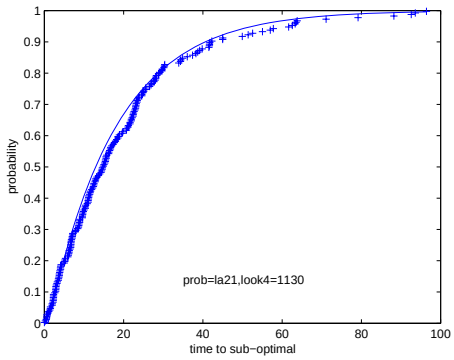

Figura 5.14: Problema la21, Solução-alvo=1130, Método GRASP

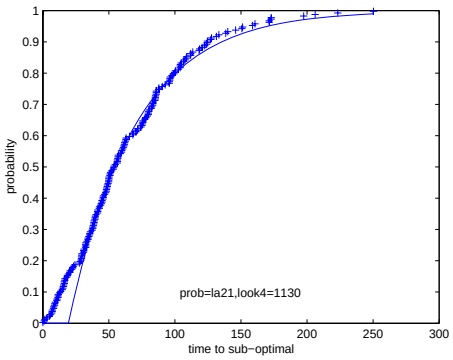

Figura 5.15: Problema la21, Solução-alvo=1130, Método GRASP+PR

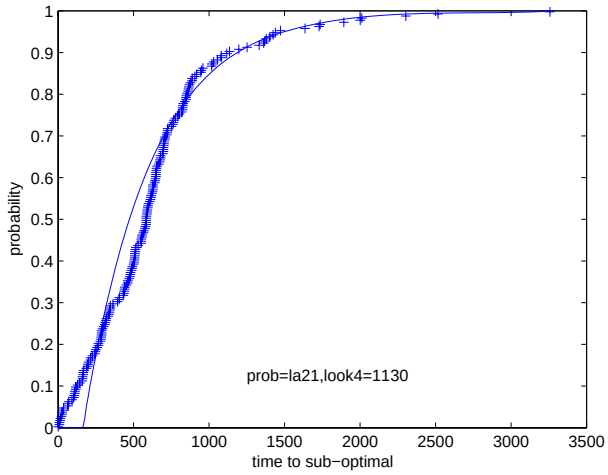

Figura 5.16: Problema la21, Solução-alvo=1130, Método GRAPR

Verifica-se, pelos gráficos, que o tempo de processamento para se encontrar as soluções-alvo dos casos analisados se encaixa em uma distribuição exponencial a dois parâmetros.

A seguir, são apresentados os gráficos Q-Q para as instâncias analisadas.

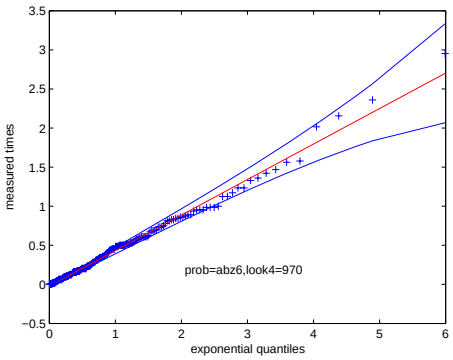

Figura 5.17: Problema abz6, Solução-alvo=970, Método GRASP

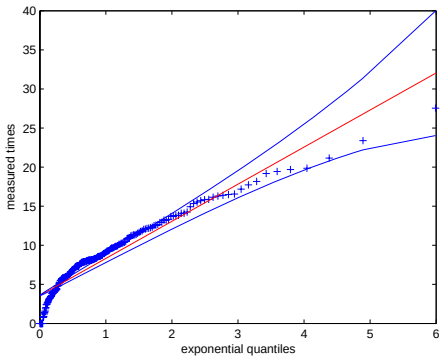

Figura 5.18: Problema abz6, Solução-alvo=970, Método GRASP+PR

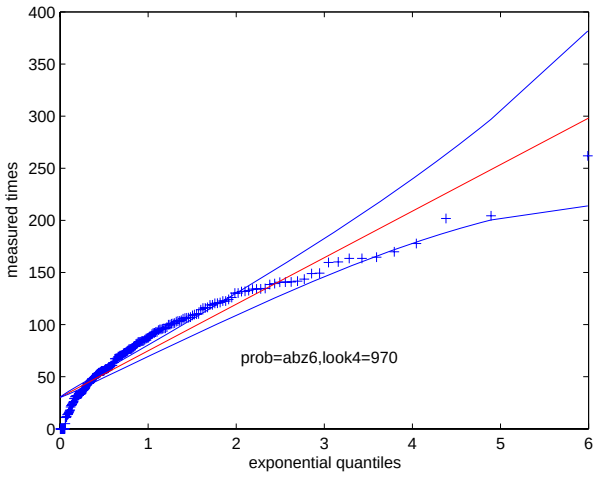

Figura 5.19: Problema abz6, Solução-alvo=970, Método GRAPR

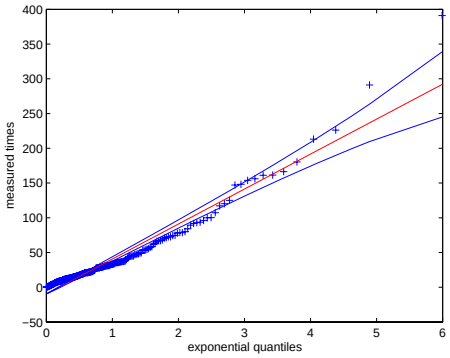

Figura 5.20: Problema mt10, Solução-alvo=970, Método GRASP

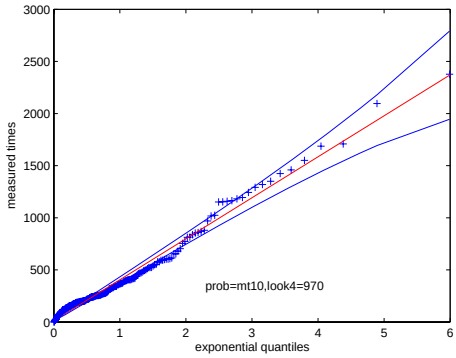

Figura 5.21: Problema mt10, Solução-alvo=970, Método GRASP+PR

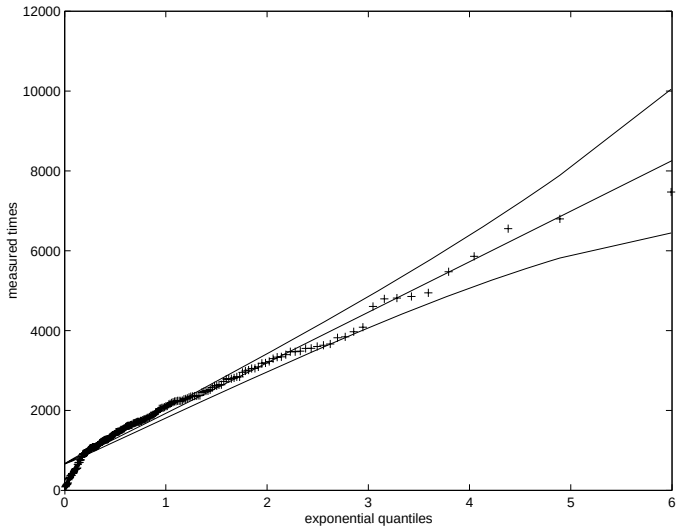

Figura 5.22: Problema mt10, Solução-alvo=970, Método GRAPR

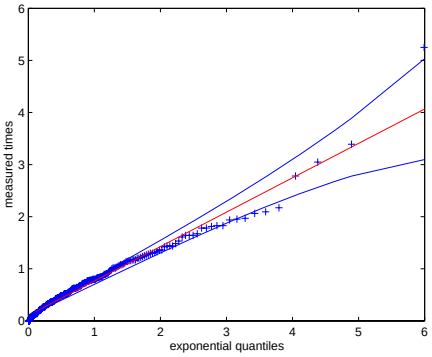

Figura 5.23: Problema orb5, Solução-alvo=930, Método GRASP

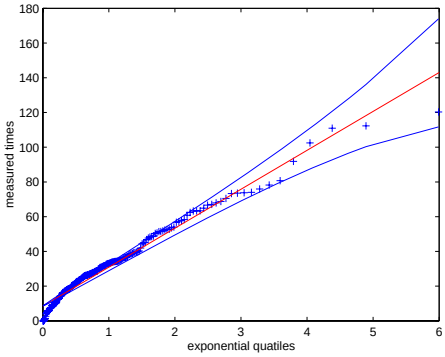

Figura 5.24: Problema orb5, Solução-alvo=930, Método GRASP+PR

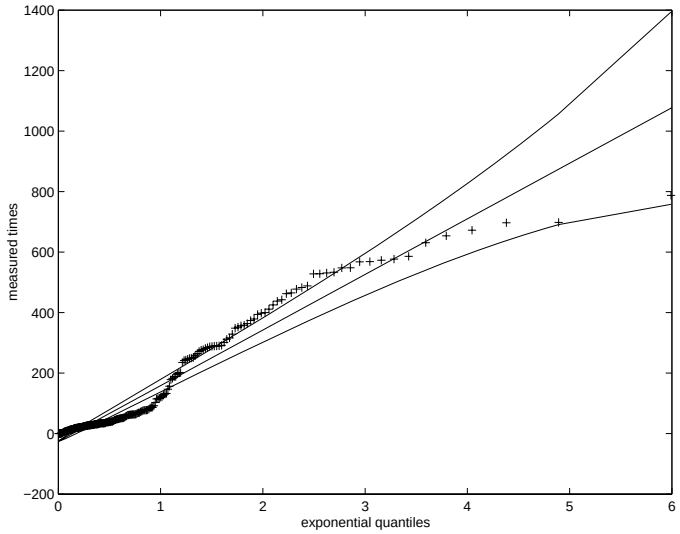

Figura 5.25: Problema orb5, Solução-alvo=930, Método GRAPR

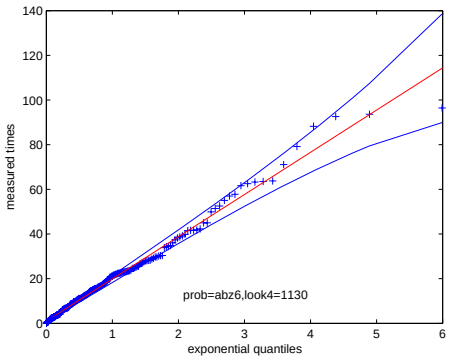

Figura 5.26: Problema la21, Solução-alvo=1130, Método GRASP

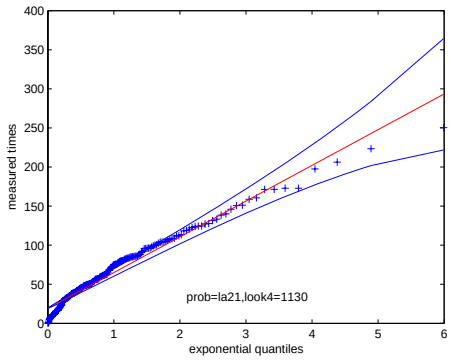

Figura 5.27: Problema la21, Solução-alvo=1130, Método GRASP+PR

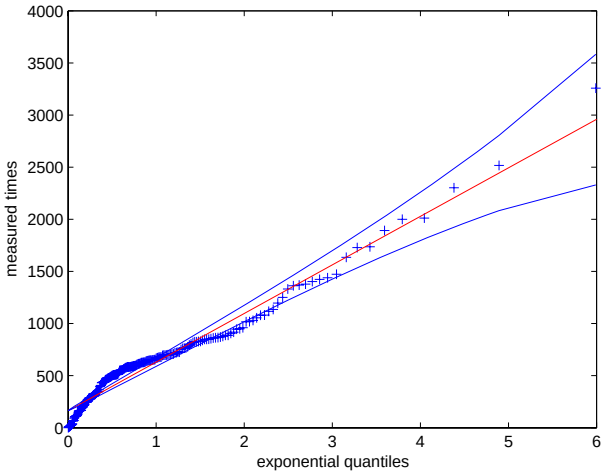

Figura 5.28: Problema la21, Solução-alvo=1130, Método GRAPR

Os parâmetros estimados para todos os casos são mostrados nas tabelas a seguir. Cada tabela mostra os resultados para um determinado método. Estimativas negativas para o parâmetro $\hat{\mu}$, observadas nas tabelas, são esperadas sempre que o deslocamento que esse parâmetro provoca na distribuição exponencial é pequeno, já que $\hat{\mu}$ é o ponto de interseção da reta definida pelos primeiro e terceiro quartiles do gráfico Q-Q. Podemos fazer uma análise qualitativa, considerando que o parâmetro $\hat{\lambda}$ é o tempo médio estimado para se atingir a solução-alvo e o parâmetro $\hat{\mu}$ é o tempo mínimo estimado para se atingir a solução-alvo. Como o tempo mínimo para se atingir a solução-alvo é o tempo correspondente a uma iteração, $\hat{\mu}$ é uma estimativa de tempo de uma iteração.

Tabela 5.1: Resultados para os problemas analisados mostrando a melhor solução conhecida, a solução-alvo e os parâmetros estimados - Método GRASP

GRASP				
			estimativas	
Problema	bks	solução alvo	$\hat{\mu}$	$\hat{\lambda}$
abz6	943	970	-0,021	0,448
mt10	930	970	-9,629	33,153
la21	1047	1130	1,002	17,209
orb5	887	930	0,105	0,683

Tabela 5.2: Resultados para os problemas analisados mostrando a melhor solução conhecida, a solução-alvo e os parâmetros estimados - Método GRASP+PR

GRASP+PR				
			estimativas	
Problema	bks	solução alvo	$\hat{\mu}$	$\hat{\lambda}$
abz6	943	970	3,578	5,647
mt10	930	970	12,142	299,46
la21	1047	1130	19,344	50,39
orb5	887	930	8,545	22,024

Tabela 5.3: Resultados para os problemas analisados mostrando a melhor solução conhecida, a solução-alvo e os parâmetros estimados - Método GRAPR

GRAPR				
			estimativas	
Problema	bks	solução alvo	$\hat{\mu}$	$\hat{\lambda}$
abz6	943	970	30,298	59,395
mt10	930	970	662,92	1273,4
la21	1047	1130	164,56	443,75
orb5	887	930	-25,257	225,01

5.2.2 Problema de Planejamento da Expansão de Sistemas de Transmissão

Os testes para o problema de planejamento da expansão de sistemas de transmissão foram realizados nos sistemas reduzidos da região Sul e da região Sudeste Brasileira. Os métodos metaheurísticos analisados foram os seguintes:

1. GRASP: GRASP puro.
2. GRASP+Path Relinking: GRASP com uma fase de Religamento de Caminhos após a fase de busca local
3. GRAPR

Duzentas execuções independentes de cada método para cada caso, com uma determinada solução alvo, foram executadas. Os algoritmos retornam o tempo de CPU total assim que um valor tão bom quanto a solução alvo é encontrado. Esses tempos de execução são usados na geração dos gráficos Q-Q com informação de desvio padrão e dos gráficos de distribuição de probabilidade.

Os valores sub-ótimos utilizados nos estudos da Região Sul foram 157, 165 e 170. Esses valores estão distantes, aproximadamente, $1,7\%$; $7,0\%$ e $10,2\%$ do valor ótimo (bks), que é de $\mathsf{US\$}154,26$ milhões. Os valores sub-ótimos utilizados nos estudos da Região Sudeste foram 451, 465 e 470, que também estão distantes aproximadamente, $6,8\%$; $10,2\%$ e $11,3\%$ do valor ótimo (bks), que é de $\mathsf{US\$}422$ milhões.

Os parâmetros estimados para todos os casos são mostrados nas tabelas 5.4, 5.5 e 5.6.

Tabela 5.4: Resultados para os sistemas analisados mostrando a melhor solução conhecida, a solução-alvo e os parâmetros estimados - Método GRASP

GRASP				
			estimativas	
Problema	bks	solução alvo	$\hat{\mu}$	$\hat{\lambda}$
Região Sul	154,26	170	-0,085	8,107
		165	0,366	11,037
		157	3,908	89,776
Região Sudeste	422	470	0,2793	8,278
		465	-2,409	16,029
		451	-1,2801	73,552

Tabela 5.5: Resultados para os sistemas analisados mostrando a melhor solução conhecida, a solução-alvo e os parâmetros estimados - Método GRASP+PR

GRASP+PR				
			estimativas	
Problema	bks	solução alvo	$\hat{\mu}$	$\hat{\lambda}$
Região Sul	154,26	170	0,298	7,431
		165	0,099	11,023
		157	0,845	77,281
Região Sudeste	422	470	1,184	8,306
		465	2,146	13,796
		451	11,538	58,639

Observamos das tabelas 5.4, 5.5 e 5.6, que mostram os valores dos parâmetros μ e λ das distribuições exponenciais a dois parâmetros, para cada par de instância/solução-alvo analisada, que as razões λ/μ calculadas com os valores encontrados nas tabelas para o método de GRAPR são menores que as razões calculadas com os parâmetros estimados para o método GRASP+PR, que, por sua vez, são menores que os valores das razões para o método GRASP. Apesar do método GRAPR achar a solução-alvo mais rapidamente do que GRASP e GRASP+PR, suas iterações precisam de tempos de CPU maiores, o que corresponde a valores maiores para o parâmetro μ. O religamento de caminhos também acelera o método GRASP, reduzindo o espalhamento do tempo de solução, isto é, o parâmetro λ. Por isso, valores de μ são maiores e valores de λ são menores para os métodos GRASP+PR e GRAPR com relação aos parâmetros do GRASP. Por essa razão, as distribuições plotadas para GRAPR não podem ser aproximadas por uma distribuição exponencial simples.

Tabela 5.6: Resultados para os sistemas analisados mostrando a melhor solução conhecida, a solução-alvo e os parâmetros estimados - Método GRAPR

GRAPR				
			estimativas	
Problema	bks	solução alvo	$\hat{\mu}$	$\hat{\lambda}$
Região Sul	154,26	170	-1,459	9,892
		165	-1,624	15,090
		157	-12,742	114,850
Região Sudeste	422	470	1,079	8,299
		465	1,489	14,116
		451	1,927	70,139

Os gráficos de distribuição de probabilidade acumulada e os gráficos Q-Q para os sistemas das regiões Sul e Sudeste utilizando-se os algoritmos citados são mostrados a seguir.

Os gráficos de distribuição de probabilidade acumulada, com as distribuições empíricas e teóricas superpostas, são mostrados lado a lado, para um determinado valor de solução-alvo e método utilizado.

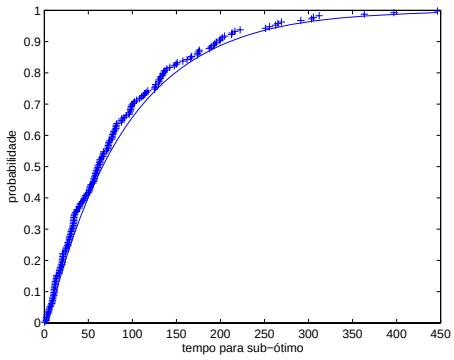

Figura 5.29: Região Sul, Solução-alvo=157, Método GRASP

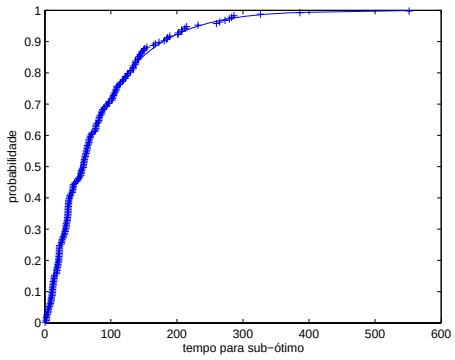

Figura 5.30: Região Sul, Solução-alvo=157, Método GRASP+PR

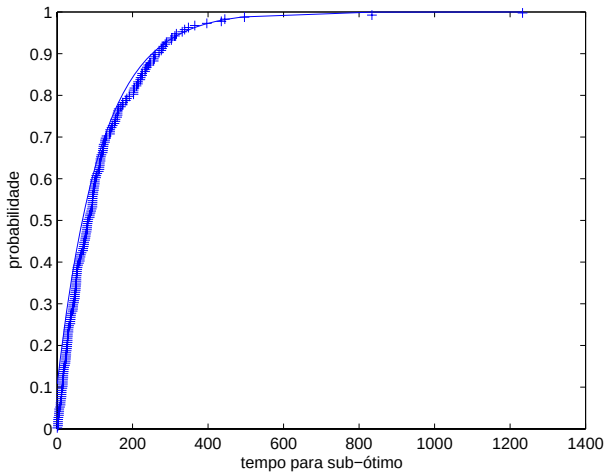

Figura 5.31: Região Sul, Solução-alvo=157, Método GRAPR

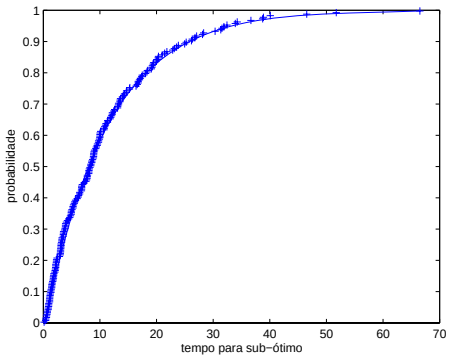

Figura 5.32: Região Sul, Solução-alvo=165, Método GRASP

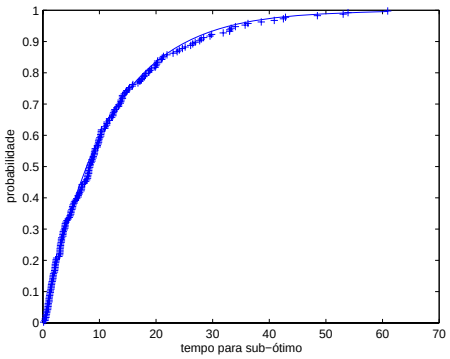

Figura 5.33: Região Sul, Solução-alvo=165, Método GRASP+PR

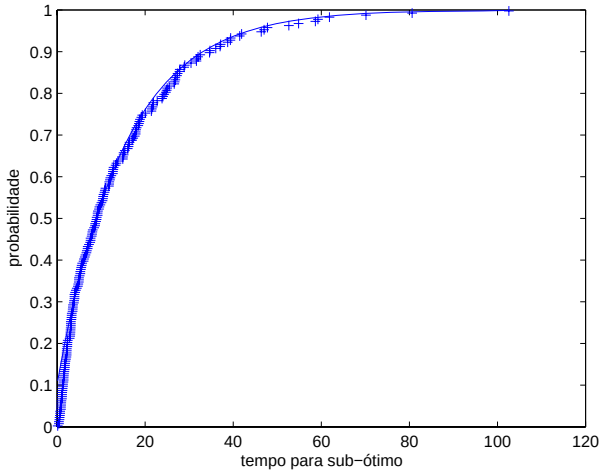

Figura 5.34: Região Sul, Solução-alvo=165, Método GRAPR

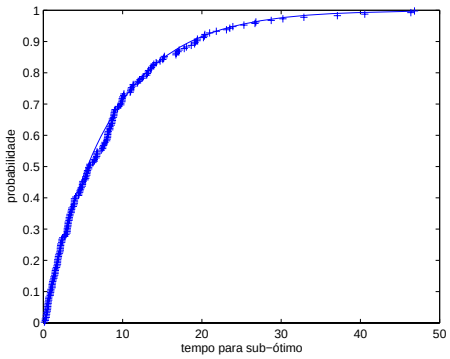

Figura 5.35: Região Sul, Solução-alvo=170, Método GRASP

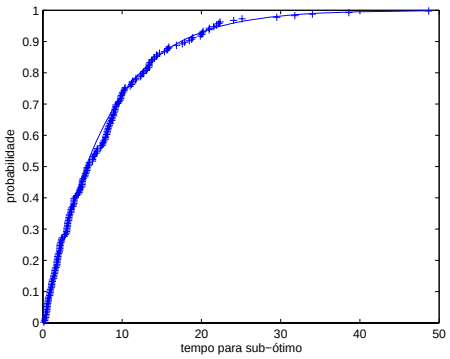

Figura 5.36: Região Sul, Solução-alvo=170, Método GRASP+PR

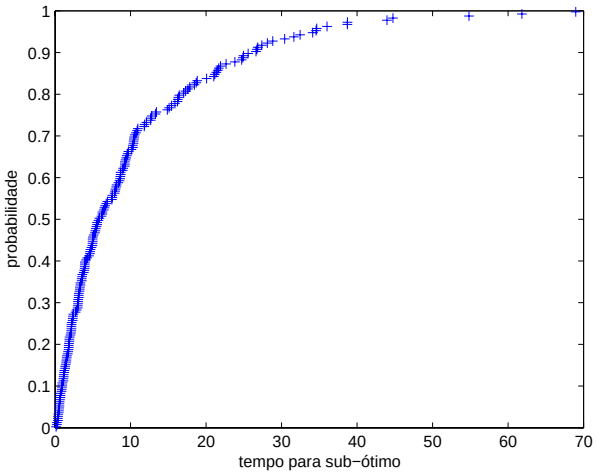

Figura 5.37: Região Sul, Solução-alvo=170, Método GRAPR

Uma comparação das distribuições exponenciais de cada método para cada valor alvo é mostrada nos gráficos a seguir.

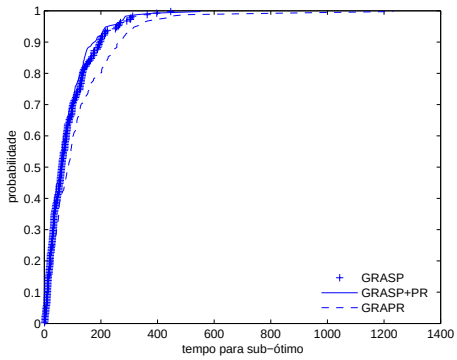

Figura 5.38: Região Sul, Solução-alvo=157, Distribuições de probabilidade para os métodos analisados

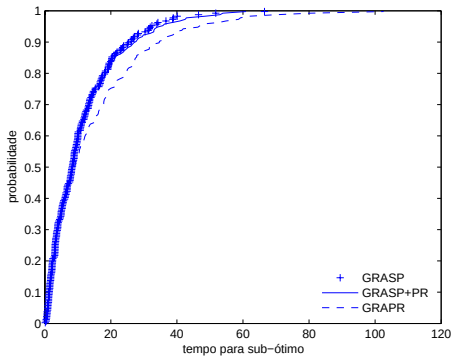

Figura 5.39: Região Sul, Solução-alvo=165, Distribuições de probabilidade para os métodos analisados

Verifica-se que as distribuições dos métodos comparados são bastante parecidas. Como o método GRAPR é o mais custoso computacionalmente, os gráficos indicam que o método encontra a solução-alvo em um número de iterações menor que os outros métodos.

Os gráficos Q-Q com informação de desvio padrão também são mostrados, da mesma forma que os gráficos de distribuição de probabilidade. Para cada ponto do gráfico, mostramos o desvio padrão para mais e para menos, com relação à reta traçada no gráfico Q-Q.

Verificamos que ocorre pouco desvio da linearidade nos gráficos Q-Q ilustrados, para todos os métodos analisados. Também verificamos que os pontos traçados ficam mais próximos da reta estimada, para valores de solução-alvo mais difíceis. Isso indica que as distribuições de probabilidade se encaixam bem em uma distribuição exponencial a dois parâmetros.

Os gráficos de distribuição de probabilidade do sistema da região Sudeste são mostrados através das figuras 5.50 a 5.58.

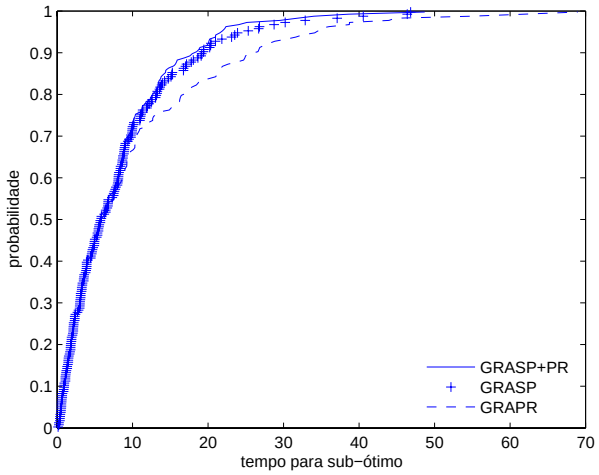

Figura 5.40: Região Sul, Solução-alvo=170, Distribuições de probabilidade para os métodos analisados

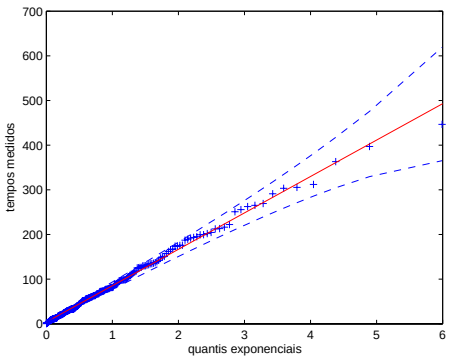

Figura 5.41: Região Sul, Solução-alvo=157, Método GRASP

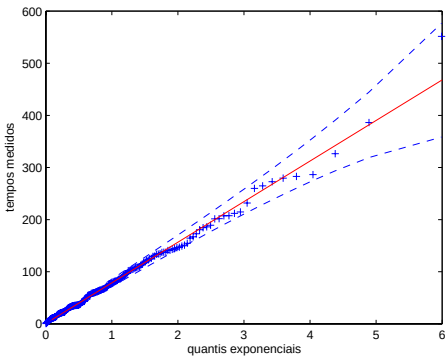

Figura 5.42: Região Sul, Solução-alvo=157, Método GRASP+PR

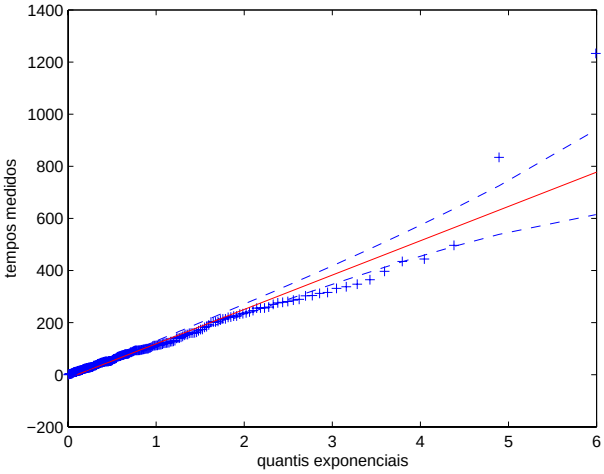

Figura 5.43: Região Sul, Solução-alvo=157, Método GRAPR

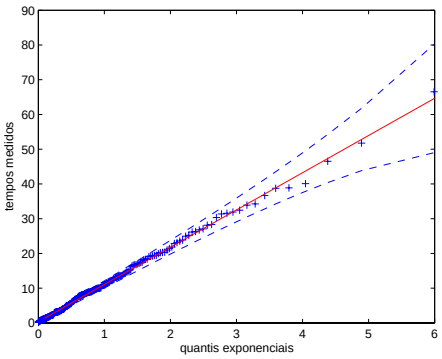

Figura 5.44: Região Sul, Solução-alvo=165, Método GRASP

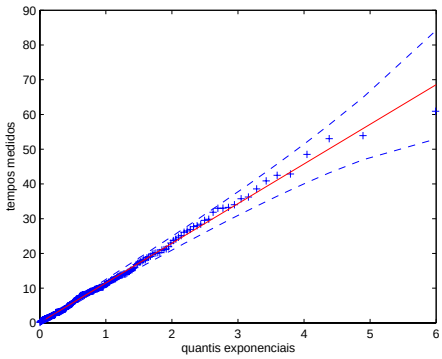

Figura 5.45: Região Sul, Solução-alvo=165, Método GRASP+PR

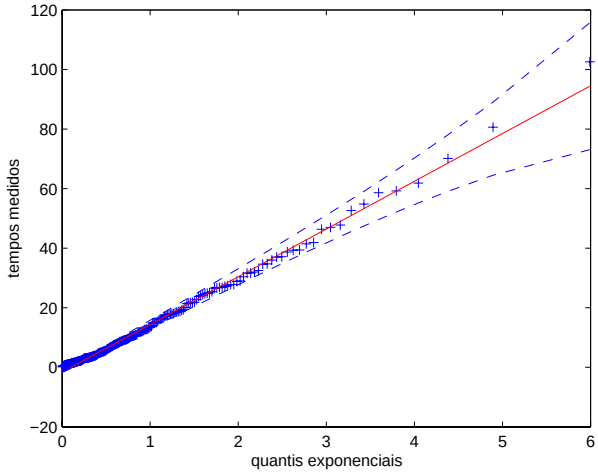

Figura 5.46: Região Sul, Solução-alvo=165, Método GRAPR

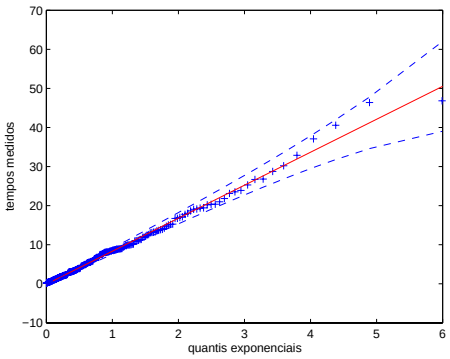

Figura 5.47: Região Sul, Solução-alvo=170, Método GRASP

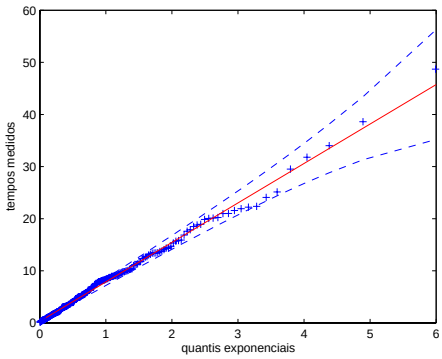

Figura 5.48: Região Sul, Solução-alvo=170, Método GRASP+PR

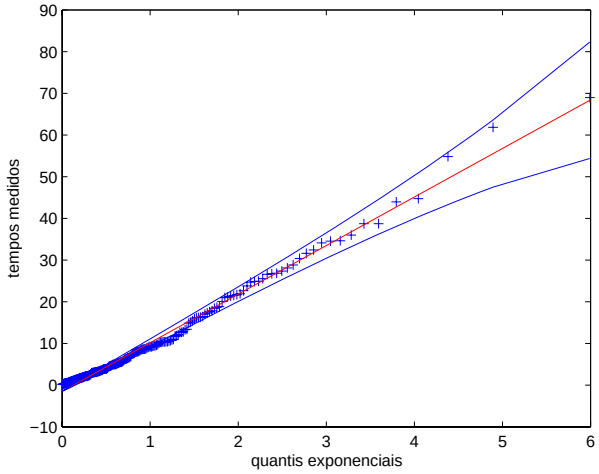

Figura 5.49: Região Sul, Solução-alvo=170, Método GRAPR

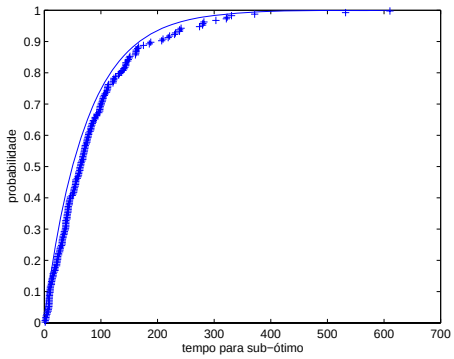

Figura 5.50: Região Sudeste, Solução-alvo=451, Método GRASP

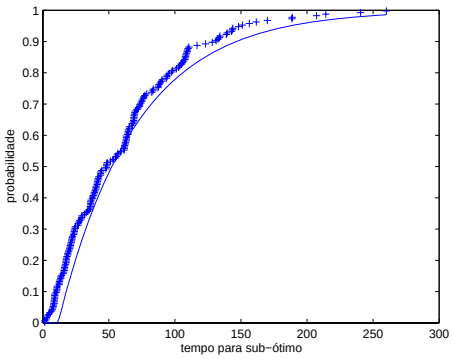

Figura 5.51: Região Sudeste, Solução-alvo=451, Método GRASP+PR

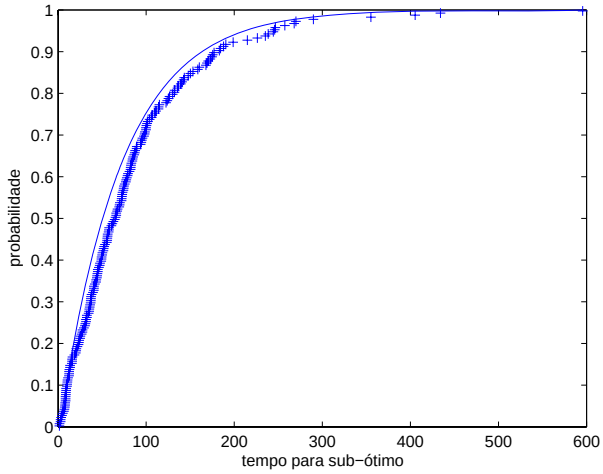

Figura 5.52: Região Sudeste, Solução-alvo=451, Método GRAPR

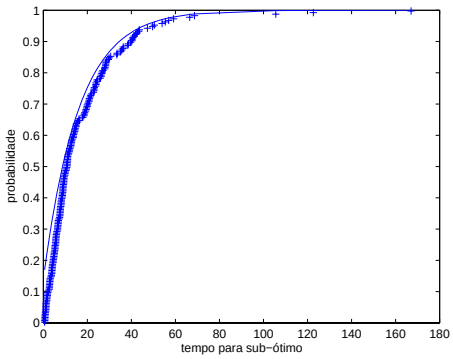

Figura 5.53: Região Sudeste, Solução-alvo=465, Método GRASP

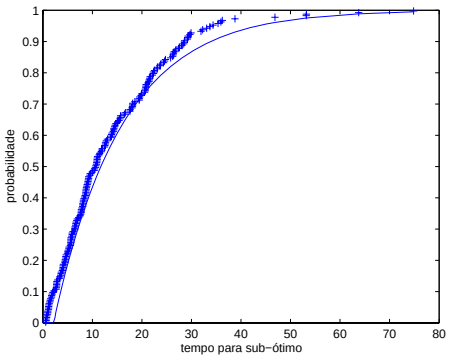

Figura 5.54: Região Sudeste, Solução-alvo=465, Método GRASP+PR

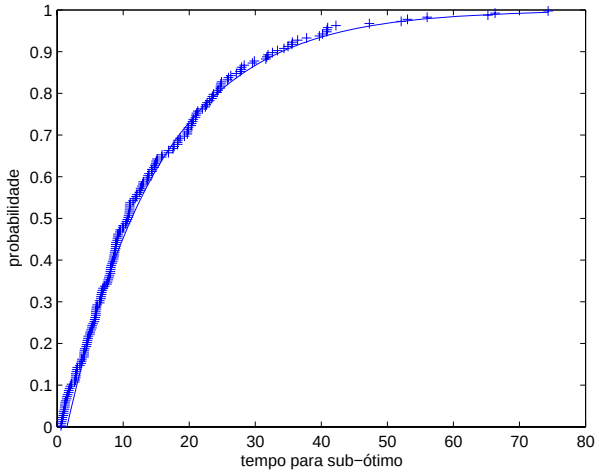

Figura 5.55: Região Sudeste, Solução-alvo=465, Método GRAPR

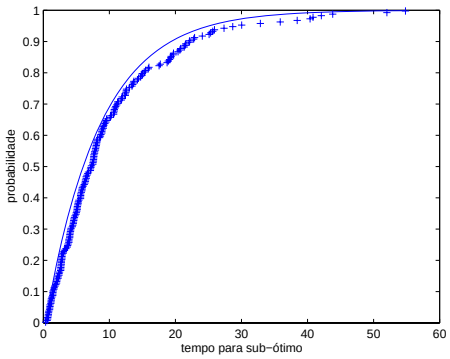

Figura 5.56: Região Sudeste, Solução-alvo=470, Método GRASP

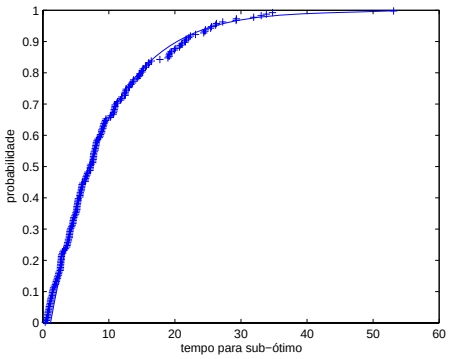

Figura 5.57: Região Sudeste, Solução-alvo=470, Método GRASP+PR

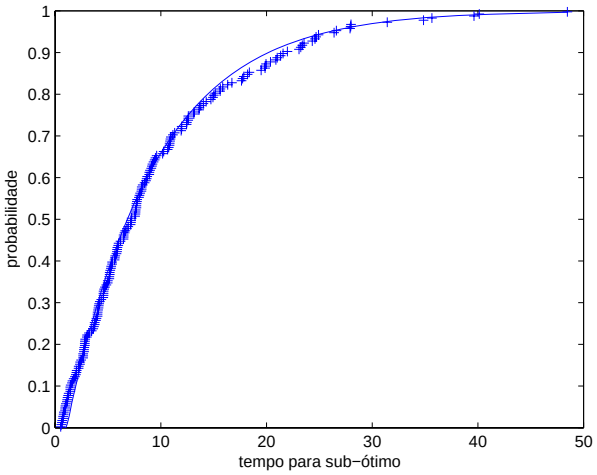

Figura 5.58: Região Sudeste, Solução-alvo=470, Método GRAPR

Os gráficos Q-Q com informação de desvio padrão também são mostrados da mesma forma que os gráficos de distribuição de probabilidade.

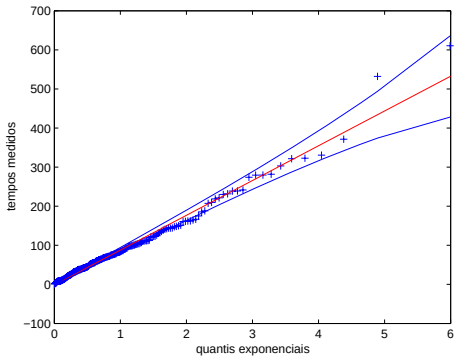

Figura 5.59: Região Sudeste, Solução-alvo=451, Método GRASP

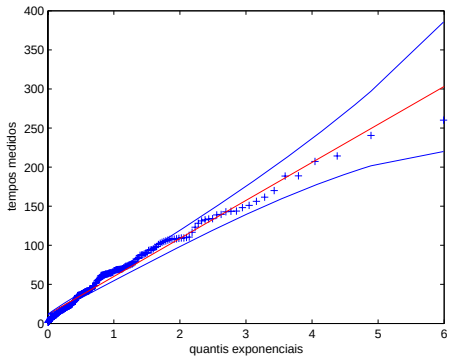

Figura 5.60: Região Sudeste, Solução-alvo=451, Método GRASP+PR

Uma comparação das distribuições exponenciais de cada método para cada valor alvo é mostrada a seguir.

Este capítulo apresentou um estudo experimental para se determinar a distribuição de probabilidade do tempo de solução em metaheurísticas do tipo GRASP. Este tipo de estudo foi inicialmente proposto para explicar acelerações lineares observadas em implementações paralelas da metaheurística GRASP, para problemas de otimização combinatória. As acelerações lineares são justificadas se a variável aleatória tempo de solução para valor alvo é distribuída exponencialmente. Os passos realizados para a obtenção dos gráficos de distribuição de probabilidade da variável aleatória tempo de solução para se atingir um determinado valor alvo, foram descritos. Também foi explicado como se obter os gráficos Q-Q, usados para se determinar os parâmetros das distribuições exponenciais. Os resultados obtidos para o problema de Planejamento da Expansão de Sistemas de Transmissão e para o Escalonamento de Tarefas foram exibidos. Foi verificado, para os dois problemas analisados, (JSP e Planejamento da Expansão de Sistemas), que a variável tempo de solução para se achar uma solução sub-ótima, se encaixa bem em uma distribuição exponencial a dois parâmetros, para as metaheurísticas GRASP, GRASP+PR e GRAPR.

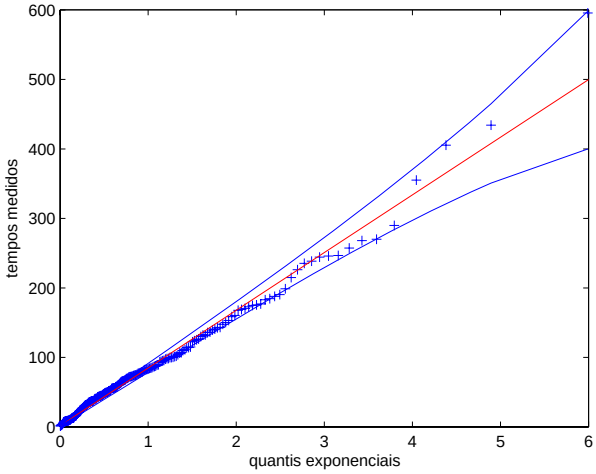

Figura 5.61: Região Sudeste, Solução-alvo=451, Método GRAPR

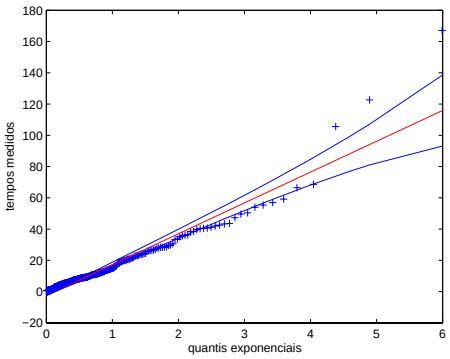

Figura 5.62: Região Sudeste, Solução-alvo=465, Método GRASP

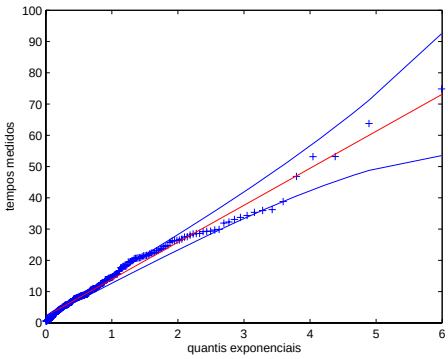

Figura 5.63: Região Sudeste, Solução-alvo=465, Método GRASP+PR

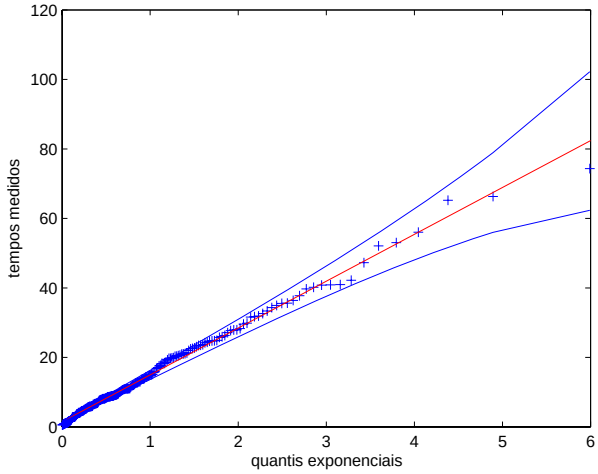

Figura 5.64: Região Sudeste, Solução-alvo=465, Método GRAPR

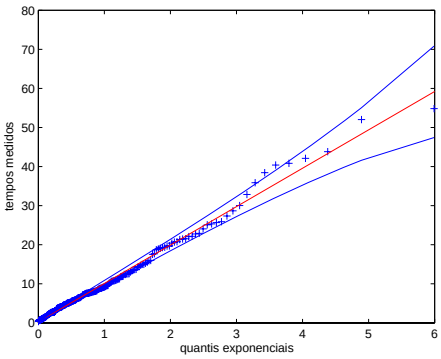

Figura 5.65: Região Sudeste, Solução-alvo=470, Método GRASP

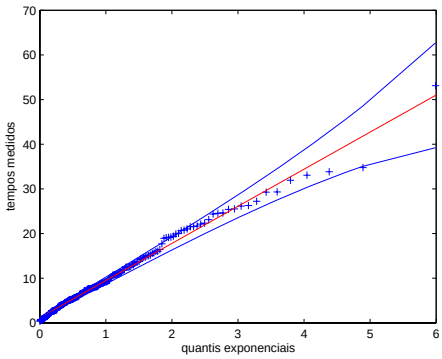

Figura 5.66: Região Sudeste, Solução-alvo=470, Método GRASP+PR

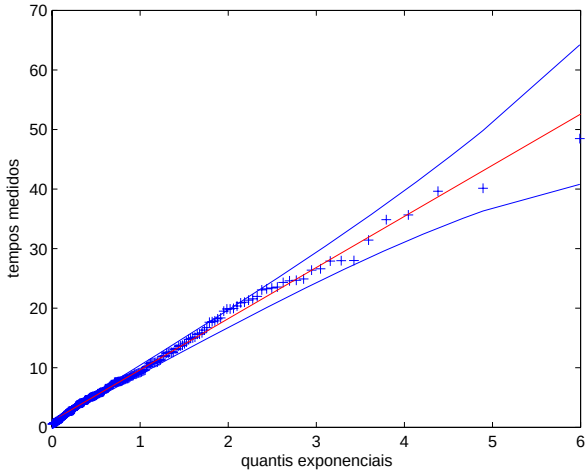

Figura 5.67: Região Sudeste, Solução-alvo=470, Método GRAPR

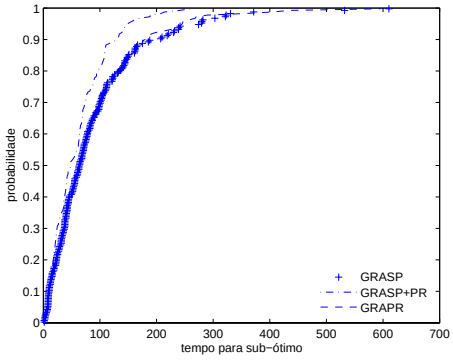

Figura 5.68: Região Sudeste, Solução-alvo=451, Distribuições de probabilidade para os métodos analisados

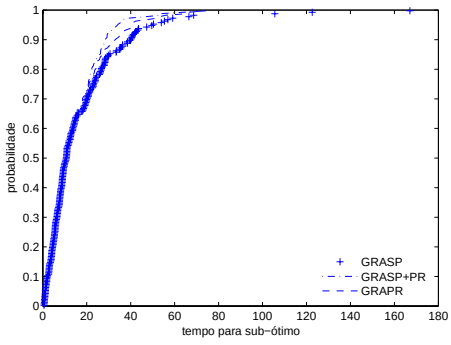

Figura 5.69: Região Sudeste, Solução-alvo=465, Distribuições de probabilidade para os métodos analisados

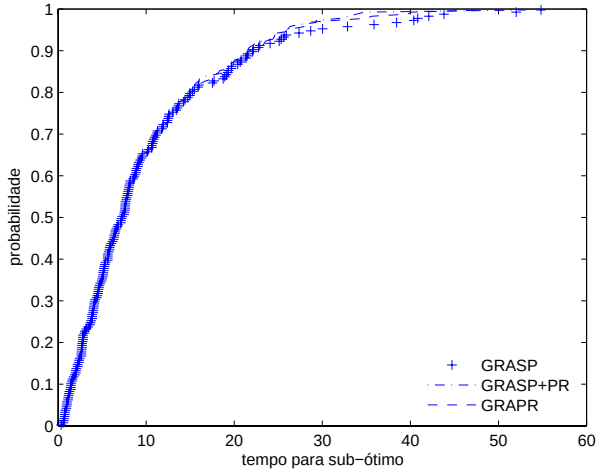

Figura 5.70: Região Sudeste, Solução-alvo=470, Distribuições de probabilidade para os métodos analisados

CAPÍTULO 6

IMPLEMENTAÇÃO PARALELA

6.1 Por que Paralelizar?

O desenvolvimento de computadores com poder de processamento cada vez maior é incessante. As pesquisas para a construção de processadores que operam a frequências cada vez maiores, memórias mais rápidas, barramentos de comunicação mais velozes, entre outros equipamentos não param. A ciência, a engenharia moderna e até a indústria de entretenimento fornecem motivação para esse desenvolvimento computacional. Os softwares também ficam mais complexos, com mais recursos e precisam de maior poder computacional. No início da era computacional, cientistas e engenheiros não imaginavam que seria possível desenvolver um computador capaz de realizar poucas centenas de operações aritméticas por segundo. Mas, assim que este poder de processamento foi alcançado, começaram a desejar mais e o objetivo atual é de um trilhão (10^{12}) de operações por segundo.

É possível enumerar problemas, nas mais diversas áreas do conhecimento, que demandam um poder computacional maior que o disponível atualmente. Problemas de previsão climática, por exemplo, requerem uma modelagem detalhada da região de estudo e um grande número de cálculos para a sua realização. O tempo disponível para esses estudos também é limitado pelo fato de que, normalmente, se objetiva uma previsão para as próximas 24 ou 48 horas. Simulações a nível atômico de bio-moléculas também estão entre esses grandes problemas desafiadores da ciência e engenharia cujas soluções darão uma grande contribuição para nosso conhecimento. É importante ressaltar que o maior poder de processamento deve ser acompanhado de uma maior capacidade de armazenagem de dados já que grande poder de cálculo sobre uma massa de dados pequena

não é muito útil.

Na área da Engenharia Elétrica, mais especificamente nos estudos dos Sistemas de Potência, também encontramos problemas que seriam adequados à aplicação da computação paralela. Problemas como o Planejamento da Expansão do sistema de transmissão, que é abordado nesta tese e que é um problema de grande porte com uma grande quantidade de dados. Metaheurísticas como Algoritmos Genéticos e GRASP têm sido aplicadas com êxito nesses problemas e, por serem altamente paralelizáveis, a utilização da computação paralela é muito indicada. A obtenção de soluções ótimas ou sub-ótimas em tempos computacionais reduzidos é um grande incentivo para a utilização desses métodos. A computação de alto desempenho apresenta a vantagem de permitir uma modelagem mais detalhada dos sistemas de potência devido aos seus recursos (maior poder de processamento), e outros tipos de estudos seriam beneficiados.

Enquanto não se constrói um processador capaz de executar (10^{12}) operações aritméticas por segundo, podemos obter esse processamento conectando vários processadores e módulos de memória em paralelo colocando-os para trabalharem juntos na resolução de um problema. Isso se chama processamento paralelo, ou seja, um computador (ou uma coleção de computadores) com múltiplos processadores trabalhando juntos. Esses processadores são conectados através de uma rede de interconexão assim como os módulos de memória. Até pouco tempo atrás, não existia um padrão para a programação de máquinas paralelas. Cada máquina possuía seu software próprio e seu uso exigia um treinamento específico. Era necessário o aprendizado de uma linguagem de programação ou um sistema de passagem de mensagem para cada plataforma. Dois grupos resolveram, então, desenvolver padrões para essa programação. O primeiro grupo, ou Forum do Fortran de Alta Performance, desenvolveu um conjunto de extensões ao Fortran-90 que permitia aos programadores implementar paralelismo de dados nos programas. Os dados (vetores e matrizes) são divididos entre os processadores e cada um aplica as mesmas operações ao seu conjunto de dados. No entanto, os compiladores não são capazes de converter todos os algoritmos paralelos eficientemente em linguagem de máquina. O segundo grupo, ou Forum da Interface de Passagem de Mensagem (MPI) especificou uma biblioteca de funções que podem ser chamadas de um programa em C ou Fortran para a programação de máquinas paralelas. Uma função de passagem de mensagem transmite dados de um processo para outro e é um método geral e eficiente de se obter programas paralelos, sendo, atualmente, a forma mais usada de programação paralela.

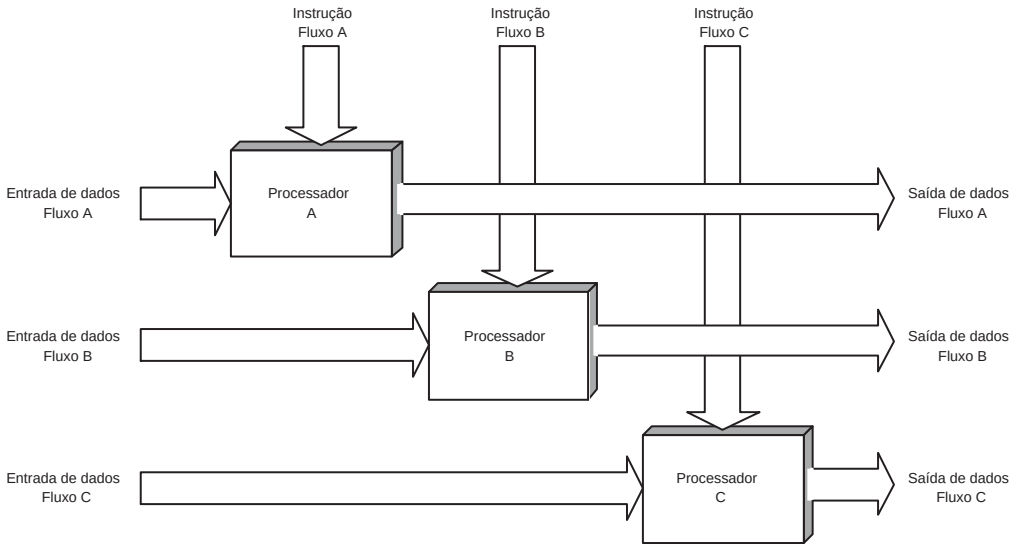

Figura 6.1: Fluxo de instruções e dados em uma arquitetura MIMD.

De acordo com a classificação original de computadores de Flynn (Taxonomia de Flynn), baseada no número de instruções e fluxo de dados, podemos ter sistemas SISD, SIMD, MISD e MIMD. O modelo SISD (Fluxo de instrução e dados únicos) corresponde ao computador convencional, ou seja, PC, Macintosh ou Estação de Trabalho. É serial e determinístico. Um único fluxo de instrução é processado sobre uma única entrada de dados durante um ciclo da CPU. O modelo SIMD (Fluxo de instrução único e dados múltiplos) é adequado ao paralelismo de dados. O sistema possui uma única CPU que controla uma série de unidades funcionais que possuem uma pequena quantidade de memória. É um modelo sícrono(lock-step) e determinístico. A categoria mais geral corresponde à arquitetura MIMD (Fluxo de instrução e dados múltiplos). Os processadores são autônomos e não existe a necessidade de sincronismo, podendo ser determinístico ou não. Esse tipo de arquitetura é dividido em sistemas de memória compartilhada e distribuída. O diagrama de uma arquitetura MIMD é mostrado na figura 6.1.

A máquina de memória compartilhada consiste de uma coleção de processadores e módulos de memória interconectados por uma rede como mostra a figura 6.2.

Nessas máquinas, o paralelismo é alcançado através do uso de estruturas de dados compartilhadas ou através da emulação da semântica de passagem de mensagem por software. Em sistemas de memória distribuída, cada processador tem sua própria memória. Uma representação dessa arquitetura é dada na figura 6.3.

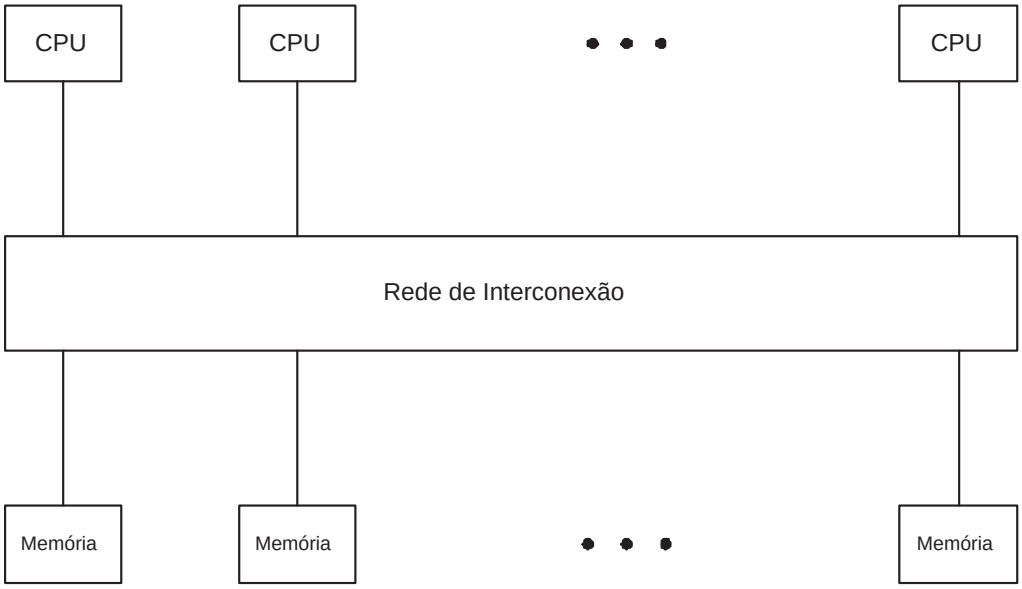

Figura 6.2: Diagrama de uma arquitetura de memória compartilhada.

Figura 6.3: Diagrama de uma arquitetura de memória distribuída.

Nestes sistemas, cada nó (par formado por CPU/memória) recebe uma cópia do programa paralelo, que enviam e recebem mensagens para coordenar o processamento. As mensagens contidas em um programa para uma máquina de memória distribuída contém, geralmente, dados, informações de sincronismo e outros dados que controlam a execução do programa paralelo.

6.2 Cluster para Computação de Alto Desempenho

Cluster é um sistema paralelo/distribuído que consiste de uma coleção de computadores independentes interligados, que cooperam como se fossem um recurso computacional único. É classificado como uma arquitetura MIMD. Os clusters de estações de trabalho ou PC's são uma alternativa barata e prontamente disponível para plataformas específicas de computação de alto desempenho. Clusters também permitem um crescimento gradual do sistema de acordo com as necessidades das aplicações. Podemos citar como vantagens do cluster:

- Baixo custo : Estações de trabalho e PC's são prontamente disponíveis a baixo custo.
- Facilidade de Upgrade : Últimas tecnologias em processadores podem ser facilmente incorporadas.
- Possibilidade de uso de softwares e sistemas operacionais já conhecidos.
- Facilidade de expansão. Os clusters permitem um crescimento gradual do sistema de acordo com as necessidades das aplicações.
- Heterogeneidade : computadores diferentes podem ser integrados.

Como conseqüência, apresentam também como benefício uma maior disponibilidade do sistema devido à redundância do hardware, sistema operacional e aplicações. O hardware também apresenta uma maior tolerância a falhas devido à redundância da maioria dos componentes e software (servidores, aplicativos, serviços, etc.). A desvantagem é que apresentam uma rede de conexão menos eficiente que um computador paralelo, mas a velocidade de comunicação entre computadores está aumentando graças a novas tecnologias de rede e protocolos implementados em LAN's e WAN's.

O cluster utilizado nas simulações desta tese foi o cluster Mercury localizado no NACAD (Núcleo de Atendimento em Computação de Alto Desempenho) da COPPE/UFRJ. O cluster possui oito(8) nós dual processados com tecnologia Intel Pentium III de 1 GHz, memória real de 512MB e memória cache de 256KB. Memória RAM distribuída de 4.0GB e rede interna dedicada com tecnologia Fast-Ethernet (100MB/s). A plataforma é Linux distribuição Red Hat com suporte às linguagens Fortran-77, Fortran-90 e C/C++.

6.3 Implementação Paralela para o Planejamento da Expansão de Sistemas de Transmissão de Energia Elétrica

Descrevemos a seguir como foi feita a implementação do algoritmo paralelo para o planejamento da expansão de sistemas de transmissão de energia elétrica. O principal objetivo desta implementação é reduzir o tempo computacional de execução do algoritmo GRAPR que, devido à sua maior capacidade de exploração do espaço de busca, requer um tempo de execução maior para cada iteração. Espera-se que, com a execução em paralelo de vários algoritmos, a solução almejada seja encontrada mais cedo, isto é, em um número reduzido de iterações, com um tempo computacional aceitável.

Mostraremos como foi feita a implementação de um esquema de paralelização mostrado no trabalho [2]. Esse esquema, chamado de não-colaborativo, limita a comunicação entre os processadores para a detecção de término de processo.

6.3.1 Esquema Não-Colaborativo

Esse esquema básico de paralelização foi proposto em [4]. Nesses testes, os algoritmos interrompem a sua execução somente quando uma solução melhor ou igual à solução-alvo é encontrada. A figura 6.4 mostra o pseudo-código para esse esquema. A comunicação entre os processadores é feita utilizando-se passagem de mensagens e está limitada ao término do programa. Processos mandam mensagem para todos os outros quando eles terminam a execução por terem achado uma solução no mínimo igual a solução-alvo. O algoritmo não-colaborativo é construído baseado no algoritmo seqüêncial e cada processo executa uma cópia do programa. Na linha 2 da Figura 6.4, o número ou rank do

processo e o número de processos são determinados. Cada fase de construção do GRAPR é inicializada com uma semente para o gerador de número aleatório. Para assegurar a independência dos processos, cada processo deve utilizar uma semente para o gerador de número aleatório $rand()$ distinta. A semente inicial para o processo my_rank é calculado nas linhas 3 a 5. O loop que se inicia na linha 6 e termina na linha 27 executa as iterações. As fases de construção, busca local e GRAPR são idênticas a do algoritmo seqüêncial. Na linha 16, se um processo acha uma solução com custo menor ou igual à solução alvo ($look4$), ele envia uma mensagem para todos os outros processos indicando que ele encontrou a solução.

6.4 Resultados Computacionais para as Implementações Paralelas do Planejamento da Expansão de Sistemas de Transmissão

Nesta seção, mostraremos os resultados obtidos com as implementações paralelas dos algoritmos para o planejamento da expansão de sistemas de transmissão. Os algoritmos paralelos usados no teste foram:

1. GRASP puro
2. GRASP com Religamento de Caminhos não colaborativo
3. GRAPR não colaborativo

Os parâmetros utilizados nos testes foram os mesmos dos testes com o algoritmo seqüêncial. GRAPR realizou 10 iterações a cada execução do método. A implementação paralela foi executada em $1;2;4;8$ e 16 processadores, utilizando a biblioteca de passagem de mensagem do MPI. Nem sempre tínhamos todos os nós disponíveis para as execuções do programa, de forma que indicaremos nos gráficos quando os tempos foram obtidos utilizando apenas 14 processadores. Os gráficos foram gerados com 60 execuções independentes dos algoritmos para cada número de processadores considerados no estudo. Os tempos de cpu foram medidos utilizando-se a função MPI_WT do MPI. Essas medidas de tempo excluem os tempos gastos para a leitura de dados, inicialização das sementes do gerador de número aleatório e envio de dados para o terminal de saída. Os tempos de aceleração médios foram calculados dividindo-se a soma dos tempos de processamento

```
procedimento GRAPR_NÃO_COLAB(Maxiter,Semente)
 1  LêDados();
 2  meu_rank = PEGA_RANK(); nprocs = PEGA_NO_PROCS();
 3  para i = 1,...,(Maxiter/nprocs) * meu_rank faça
 4   semente = rand(semente);
 5  fim-para;
 6  para i = 1,...,Maxiter faça
 7   ConstruçãoAleatóriaGulosa(semente,x̂);
 8   BuscaLocal(x̂);
 9   x̂^S = x̂;
 10   AtualizaConjuntoElite(x̂,E);
 11   SelecionaSolução(E,E_i)
 12   x̂^T = E_i;
 13   GRAPR(bias,x̂^S,x̂^T,x̂^R,Semente)
 14   x̂ = x̂^R;
 15   AtualizaSolução(x̂,x̄)
 16   se Custo(x̄) ≤ look4 então EnviaTodos(look4_fim);
 17   se i == maxitr então;
 18    num_stop = num_stop + 1;   EnviaTodos(maxitr_fim);
 19   fim-se;
 20   recebido = VerificaRecebimento(flag);
 21   se recebido então;
 22    se flag == look4_stop então pare;
 23    senão-se flag == maxitr_stop então num_stop = num_stop + 1
 24    fim-se;
 25   fim-se;
 26   if num_stop == nprocs então pare fim-se;
 27  fim-para
 28  retorna x̄;
fim GRAPR_NÃO_COLAB;
```

Figura 6.4: Pseudo-código de GRAPR Não-Colaborativo

do programa paralelo executado em 1 processador, pela soma dos tempos de processamento do programa paralelo executado em 2; 4; 8; e 16 processadores, para 60 execuções do código.

6.4.1 Sistema da Região Sul

As figuras 6.5 e 6.7 mostram gráficos de distribuição de probabilidade para a implementação paralela do sistema da Região Sul. Curvas de aceleração correspondentes também são exibidas.

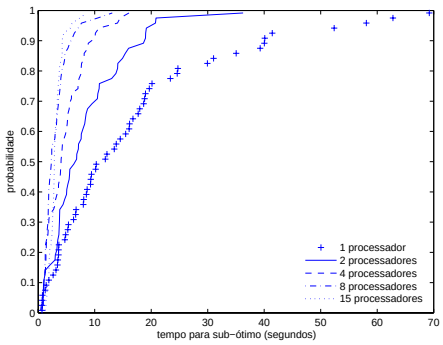

Figura 6.5: Região Sul, Solução-alvo=165, Método GRASP

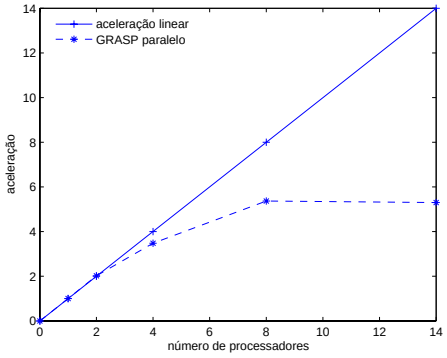

Figura 6.6: Região Sul, Solução-alvo=165, Método GRASP

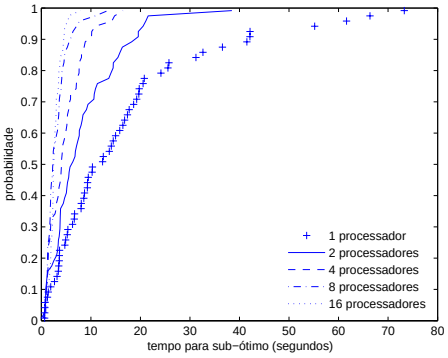

Figura 6.7: Região Sul, Solução-alvo=165, Método GRASP+PR

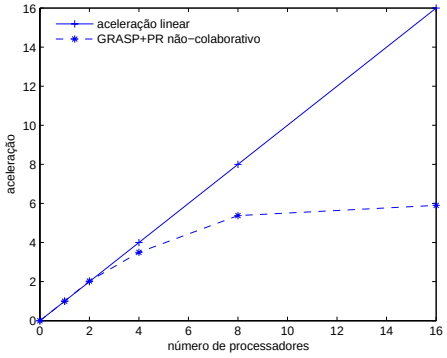

Figura 6.8: Região Sul, Solução-alvo=165, Método GRASP+PR

A tabela 6.1 mostra, para determinados tempos de processamento, a probabilidade de se achar uma solução de qualidade no mínimo igual à solução-alvo, nesse tempo pré-fixado, como função do número de processadores. O algoritmo utilizado foi o GRASP

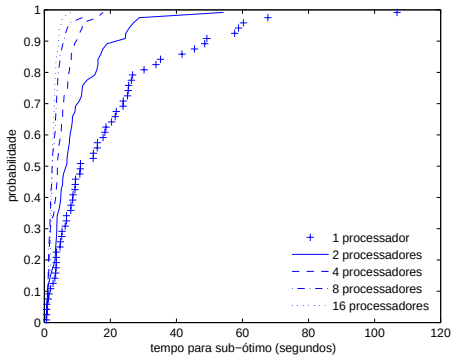

Figura 6.9: Região Sul, Solução-alvo=165, Método GRAPR

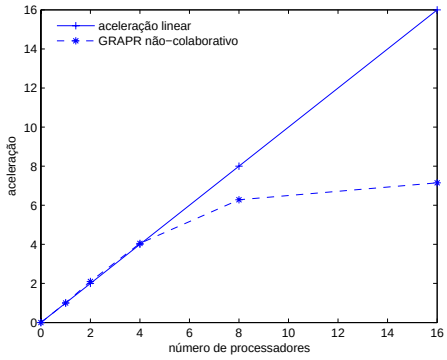

Figura 6.10: Região Sul, Solução-alvo=165, Método GRAPR

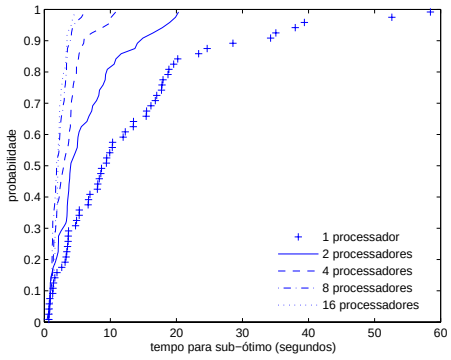

Figura 6.11: Região Sul, Solução-alvo=170, Método GRASP

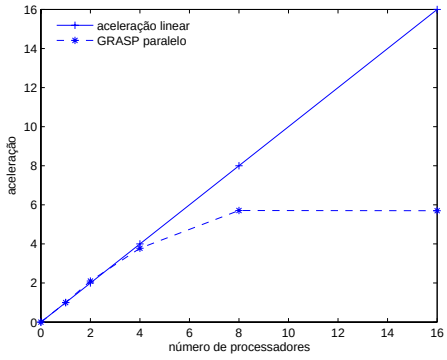

Figura 6.12: Região Sul, Solução-alvo=170, Método GRASP

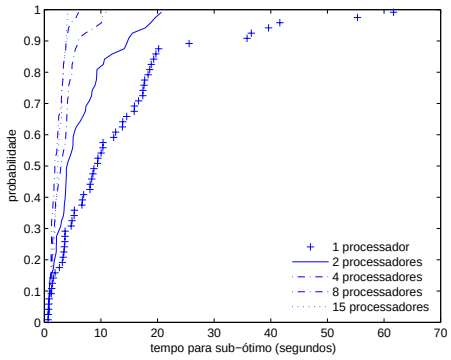

Figura 6.13: Região Sul, Solução-alvo=170, Método GRASP+PR

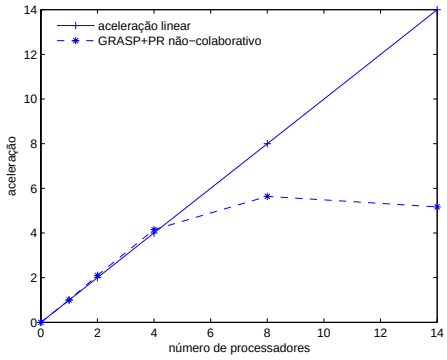

Figura 6.14: Região Sul, Solução-alvo=170, Método GRASP+PR

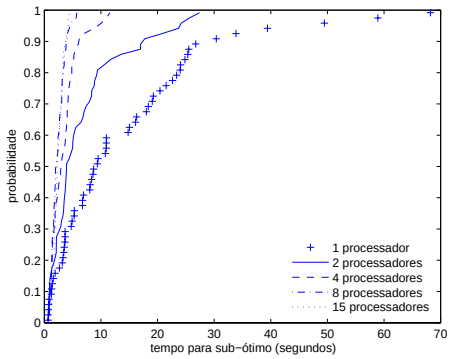

Figura 6.15: Região Sul, Solução-alvo=170, Método GRAPR

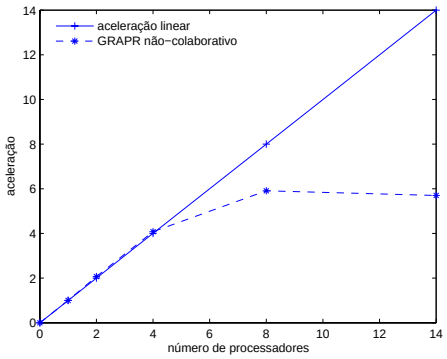

Figura 6.16: Região Sul, Solução-alvo=170, Método GRAPR

Tabela 6.1: Estimativas de probabilidade para a Região Sul. Solução-alvo = 165, Método GRASP

		probabilidade GRASP paralelo número de processadores				
Problema	Tempo	1	2	4	8	16
Região Sul	10s	.46	.67	.9	.93	1.0
	20s	.75	.92	1.0	1.0	1.0
	30s	.82	.98	1.0	1.0	1.0

paralelo. A tabela mostra, por exemplo, que a probabilidade de se achar uma solução de valor US$165 milhões na Região Sul em no máximo 10s, varia de 46% para um processador a 100% para 16 processadores.

A tabela 6.2 mostra estimativas de probabilidade para o sistema da região sul, utilizando o algoritmo de GRAPR.

Podemos verificar das tabelas de estimativas de probabilidade, que o método GRAPR apresenta maior probabilidade de achar a solução-alvo no tempo pré-fixado no estudo para a maior parte das simulações com número variável de processadores.

Tabela 6.2: Estimativas de probabilidade para a Região Sul. Solução-alvo = 165, Método GRAPR

		probabilidade GRAPR paralelo número de processadores				
Problema	Tempo	1	2	4	8	16
Região Sul	10s	.46	.67	.9	.97	1.0
	20s	.63	.94	1.0	1.0	1.0
	30s	.8	1.0	1.0	1.0	1.0

Os gráficos de distribuição de probabilidade mostram o aumento na probabilidade de se achar uma determinada solução-alvo com o aumento no número de processadores. Podemos observar pelos gráficos que acelerações sub-lineares são obtidas para todos os casos.

Esse comportamento da metaheurística GRASP paralela, implementada utilizando-se o modelo não-colaborativo, poderia ter sido previsto baseado nas razões dos parâmetros $|\mu|/\lambda$ da implementação seqüencial, como mostrado no trabalho [3]. Os parâmetros estimados para os pares de instância/valor-alvo analisados, encontrados no capítulo 5, apresentam valores elevados para as razões $|\mu|/\lambda$, justificando as acelerações sub-lineares. Essa estimativa não pode ser feita para uma implementação paralela com cooperação, já que a proposição para a distribuição exponencial a dois parâmetros não leva em conta a troca de informações entre processadores. Como foi observado no estudo da distribuição de probabilidade a dois parâmetros, a medida que o valor de ρ (número de processadores) aumenta, o termo $\rho|\mu|$ aumenta e a aceleração diminui.

6.4.2 Sistema da Região Sudeste

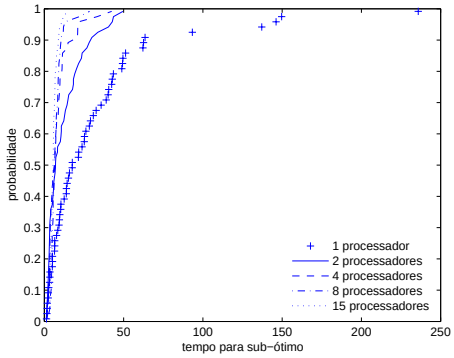

Figura 6.17: Região Sudeste, Solução-alvo=465, Método GRASP

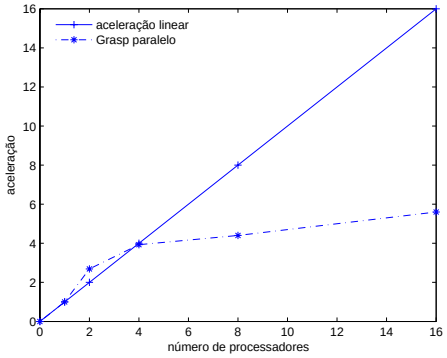

Figura 6.18: Região Sudeste, Solução-alvo=465, Método GRASP

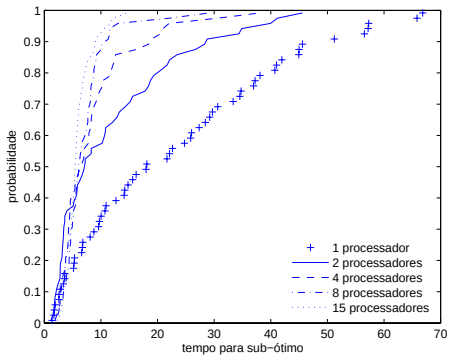

Figura 6.19: Região Sudeste, Solução-alvo=465, Método GRASP+PR

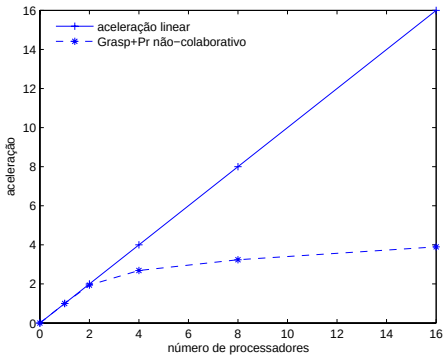

Figura 6.20: Região Sudeste, Solução-alvo=465, Método GRASP+PR

As tabelas 6.3 e 6.4 mostram acelerações com relação a um único processador e eficiência (aceleração dividida pelo número de processadores) para as implementações paralelas dos algoritmos de GRASP e GRAPR. As soluções-alvo utilizadas são de US$\$170$ milhões na Região Sul e US$\$465$ milhões na Região Sudeste.

Verificamos que o método GRAPR e o método GRASP apresentam acelerações e eficiências semelhantes para simulações em vários processadores.

Este capítulo apresentou uma pequena introdução sobre processamento paralelo, e mostrou as vantagens de se utilizar um cluster de PC's para computação de alto desempenho. Foi mostrado como foi feita a implementação computacional do esquema paralelo não-colaborativo, para o problema de Planejamento da Expansão de Sistemas de Trans-

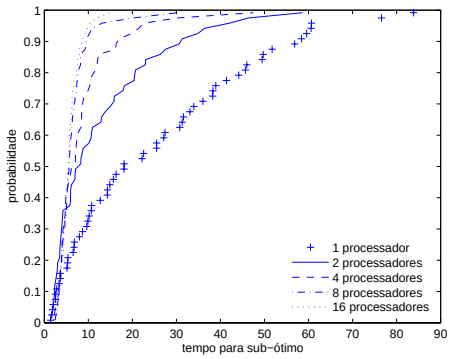

Figura 6.21: Região Sudeste, Solução-alvo=465, Método GRAPR

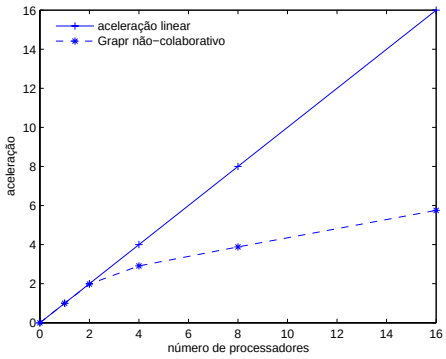

Figura 6.22: Região Sudeste, Solução-alvo=465, Método GRAPR

Tabela 6.3: Acelerações com respeito a um único processador e eficiência. Método GRASP

	probabilidade GRASP paralelo							
	2		4		8		16	
Problema	aceleração	efic.	aceleração	efic.	aceleração	efic.	aceleração	efic.
Região Sul	2.26	1.13	3.08	.77	4.88	.61	4.33	.27
Região Sudeste	2.69	1.34	3.93	.98	4.40	.55	5.60	.35
média:	2.47	1.23	3.50	.87	4.64	.58	4.96	.31

missão, juntamente com o pseudo-código do algoritmo. Os gráficos de distribuição de probabilidade da variável aleatória tempo de solução, utilizando 1; 2; 4; 8 e 16 processadores foram obtidos e comparados. As acelerações e eficiências alcançadas com a implementação paralela também foram exibidas.

Tabela 6.4: Acelerações com respeito a um único processador e eficiência. Método GRAPR

	probabilidade GRAPR paralelo							
	2		4		8		16	
Problema	aceleração	efic.	aceleração	efic.	aceleração	efic.	aceleração	efic.
Região Sul	2.30	1.15	3.02	.75	4.40	.55	3.94	.25
Região Sudeste	1.98	.99	2.91	.73	3.88	.48	5.75	.36
média:	2.14	1.07	2.96	.74	4.14	.51	4.84	.30

CAPÍTULO 7

CONCLUSÕES E DESENVOLVIMENTOS FUTUROS

Este trabalho de tese apresentou um novo método metaheurístico para solução de problemas de otimização combinatória, que combina conceitos de dois métodos conhecidos e bastante utilizados neste campo de pesquisa. O novo método recebeu a denominação de GRAPR (Greedy Randomized Adaptive Path Relinking), pois aplica conceitos de GRASP (Greedy Randomized Adaptive Search Procedure) na geração de trajetórias do método de Religamento de Caminhos (Path Relinking).

Foram descritos os dois tipos de problemas de otimização combinatória utilizados para testar a eficiência do novo método metaheurístico, desenvolvido especialmente para a solução destes dois tipos de problemas. O Planejamento da Expansão de Redes de Transmissão de Energia Elétrica e o problema de Escalonamento de Tarefas (Job Shop Scheduling). A formulação matemática dos problemas foi apresentada, juntamente com as principais dificuldades associadas à sua resolução.

Os principais métodos de otimização já aplicados para resolver estes problemas foram especificados no capítulo de Revisão Bibliográfica. A qualidade dos resultados obtidos para cada trabalho também foi citada. As metaheurísticas GRASP e Religamento de Caminhos, utilizadas na elaboração do método de resolução proposto, são descritas em detalhes. Os termos e conceitos do GRASP são introduzidos e é explicado como a generalização desses conceitos, aplicada ao método de Religamento de caminhos, deu origem ao GRAPR (Greedy Randomized Adaptive Path Relinking). A implementação computacional dos dois métodos é mostrada em detalhes. Pseudo-códigos para os algoritmos desenvolvidos mostram os passos seguidos pelo método.

O método proposto foi aplicado com sucesso aos problemas de otimização escolhidos para estudo. O método teve excelente desempenho quando aplicado ao Planejamento da Expansão de Sistemas de Transmissão. Para o sistema da região sul, a solução ótima foi encontrada em todos os casos analisados, com um aumento modesto do tempo computacional. O método também foi capaz de encontrar a solução ótima para o sistema da região sudeste, mostrando maior precisão que os métodos GRASP e GRASP+PR, que não foram capazes de encontrar uma solução com tal qualidade. O método também teve bom desempenho quando aplicado ao problema de Escalonamento de Tarefas, com uma eficiência menor do que quando aplicado ao problema de Planejamento.

Os métodos do tipo GRASP apresentados nesta tese, mostraram que são uma alternativa robusta e eficiente para a aplicação em problemas de otimização combinatória. As metaheurísticas GRASP e suas formas híbridas, GRASP+PR e GRAPR são simples de implementar, exigem um ajuste de poucos parâmetros e encontram boas soluções em tempos computacionais bastante aceitáveis. GRAPR, particularmente, possui um grande potencial de exploração do espaço de busca e é capaz de reduzir o número de iterações necessárias para se encontrar a solução ótima de um determinado problema. É necessário ressaltar que não possuímos nenhuma garantia de que a solução encontrada por uma metaheurística é a solução ótima do problema analisado, mas, certamente, é uma boa solução que pode auxiliar o planejador ou engenheiro de planejamento na sua tarefa diária, indicando candidatos promissores, no caso de problemas de expansão da transmissão.

Uma investigação experimental da distribuição de probabilidade da variável tempo de solução também foi realizada, mostrando-se os gráficos de distribuição de probabilidade e gráficos Q-Q para as instâncias analisadas nos dois problemas estudados. Verificou-se, pelo estudo da distribuição de probabilidade da variável aleatória tempo de solução para se atingir uma determinada solução-alvo, para as metaheurísticas analisadas na tese, que as distribuições se encaixam em uma distribuição exponencial a dois parâmetros, verificando-se também que o método GRAPR é capaz de encontrar uma determinada solução-alvo em um número de iterações menor que os demais métodos na maioria dos casos.

Os algoritmos utilizados neste trabalho foram implementados utilizando as linguagens de programação de alto nível Fortran77 e C. Uma versão paralela do algoritmo para processamento em máquinas de memória distribuída, mais especificamente em um cluster de PC's localizado na COPPE/UFRJ, utilizando a biblioteca de passagem de mensagens

MPI foi desenvolvida.

Neste trabalho, implementou-se um esquema de paralelização denominado de *multiple independent walks* não-colaborativo, onde cada processador possui uma cópia dos dados do problema e do programa. O esquema não-colaborativo limita a comunicação entre os processadores à detecção de término de processo. Todos os processadores interrompem suas execuções quando um deles encontra uma solução igual ou melhor que uma determinada solução-alvo. Sementes diferentes devem ser fornecidas aos geradores de números aleatórios das iterações de cada processador para evitar a geração de seqüências de soluções idênticas.

Gráficos das distribuições de probabilidade do tempo de solução para o algoritmo paralelo são exibidos, juntamente com os gráficos de aceleração do tempo de execução, com o número de processadores variando de um a dezesseis. Acelerações sub-lineares são obtidas para todos os casos.

Como trabalhos futuros, pretende-se aplicar a metaheurística proposta em outros casos exemplos. Melhoramentos na representação da rede, para o caso de Planejamento da Expansão de Sistemas de Transmissão, considerando a modelagem de outros tipos de equipamentos, é uma meta a ser alcançada. Também pretendemos considerar o Planejamento com critérios de confiabilidade, considerando contingências e perdas. Implementar um esquema de paralelização colaborativo em que os processadores trocam informações de seus conjuntos de elite, além da detecção de término de processo, deve ser desenvolvida para se conseguir uma implementação paralela ainda mais eficiente.

Bibliografia

[1] J. Adams, E. Balas, and D. Zawack. "The Shifting Bottleneck Procedure for Job Shop Scheduling". *Management Science*, 34, pp. 391–401, 1988.

[2] R. M. Aiex, S. Binato, and M. G. C. Resende. "Parallel GRASP with Path-Relinking for Job-Shop Scheduling". *Parallel Computing*, pp. 393–430, 2003.

[3] R. M. Aiex and M. G. C. Resende. "Parallel Strategies for GRASP with Path-Relinking". K. Nonobe T. Ibaraki and M. Yagiura, editors, *Metaheuristics : Progress as real problem solvers*, pp. 301–331. Springer, 2005.

[4] R. M. Aiex, M. G. C. Resende, P. M. Pardalos, and G. Toraldo. *GRASP with Path Relinking for the Three-Index Assignment Problem*. Technical report, AT&T Research, 2000.

[5] R.M. Aiex, M.G.C. Resende, and C.C. Ribeiro. "Probability Distribution of Solution Time in GRASP: An Experimental Investigation". *Journal of Heuristics*, 8, pp. 343–373, 2002.

[6] D. Applegate and W.Cook. "A Computational Study of the Job-Shop Scheduling Problem". *ORSA Journal on Computing*, 3, pp. 149–156, 1991.

[7] L. Bahiense, G. C. Oliveira, M. V. F. Pereira, and S. Granville. "A Mixed Integer Disjunctive Model for Transmission Network Expansion". *IEEE Transactions on Power Systems*, 16(3), pp. 560–565, Aug 2001.

[8] K. R. Baker. *Introduction to Sequencing and Scheduling*. John Wiley, 1974.

[9] J. F. Benders. "Partitioning Methods for Solving Mixed Variables Programming Problems". *Numerische Mathematik*, 4, pp. 238–252, 1962.

[10] R. J. Bennon and J. A. Juves. “Use of Sensitivity Analysis in Automated Transmission Planning”. *Proceedings of the conference on Power Industry Computer Applications*, Philadelphia, PA, US, May 1981.

[11] S. Binato. *Expansão Ótima de Sistemas de Transmissão Através de Decomposição de Benders e Técnicas de Planos Cortantes*. Tese de D.Sc., COPPE/UFRJ, Rio de Janeiro, RJ, Brasil, Abril 2000.

[12] S. Binato, H. Faria Jr., and M. G. C. Resende. “Greedy Randomized Adaptive Path Relinking”. *Proceedings of the Fourth Metaheuristic International Conference*, pp. 393–397, Porto, Portugal, Jul 2001.

[13] S. Binato and G. C. Oliveira. “A Heuristic Approach to Cope with Multi-Year Transmission Expansion Planning”. *Proceedings of the Stockholm Power Tech Conference*, Stockholm, Sweden, 1995. paper SPT S01-03-277.

[14] S. Binato and G. C. Oliveira. “A Reactive GRASP for Transmission Network Expansion Planning”. C.C. Ribeiro and P. Hansen, editors, *Essays and surveys on metaheuristics*, pp. 81–100. Kluwer Academic Publishers, 2001.

[15] S. Binato, G. C. Oliveira, and J. L. Araújo. *A Greedy Adaptive Search Procedure for Transmission Expansion Planning*. Technical report, Cepel internal report, 1998.

[16] S. Binato, G. C. Oliveira, and J. L. Araujo. “A Greedy Randomized Adaptive Search Procedure for Transmission Network Expansion Planning”. *IEEE Transactions on Power Systems*, 16(2), pp. 247–253, May 2001.

[17] S. Binato, M. V. F. Pereira, and S. Granville. “A New Benders Decomposition Approach to Solve Power Transmission Network Design Problems”. *IEEE Transactions on Power Systems*, 16(2), pp. 235–240, May 2001.

[18] S. Binato and S. P. Roméro. “Power Transmission Network Expansion Planning by a Hybrid Genetic Algorithm”. *Proceedings of the IX Latin-IberoAmerican Congress on Operations Research (IX CLAIO)*, Buenos Aires, Arg., Jul/Aug 1998.

[19] J. L. Bresina. “Heuristic-Biased Stochastic Sampling”. *Proceedings of the AAAI-96*, pp. 271–278, 1996.

[20] P. Brucker, B. Jurisch, and B. Sievers. “A Branch and Bound Algorithm for the Job-Shop Scheduling Problem”. *Discrete Applied Mathematics*, 49, pp. 105–127, 1994.

[21] J. Carlier and E. Pinson. “An Algorithm for Solving the Job-Shop Problem”. *Management Science*, 35, pp. 164–176, 1989.

[22] J. Carlier and E. Pinson. “A Practical Use of Jackson’s Preemptive Schedule for Solving the Job-Shop Problem”. *Annals of Operations Research*, 26, pp. 269–287, 1990.

[23] F. Della Croce, R. Tadei, and R. Rolando. “Solving a Real World Project Scheduling Problem with a Genetic Approach”. *Belgium Journal of Operations Research, Statistics and Computer Science*, 33, pp. 65–78, 1994.

[24] L. Davis. “Job Shop Scheduling with Genetic Algorithms”. *Proceedings of the First International Conference on Genetic Algorithms and their Applications*, pp. 136–140. Morgan Kaufmann, 1985.

[25] H. de Faria Jr. e S. Binato e M. G. C. Resende e D. M. Falcão. “GRASP com Path-Relinking para o Planejamento da Expansão de Redes de Transmissão”. *Anais do XIV Congresso Brasileiro de Automática*, pp. 599–604, Set 2002.

[26] H. de Faria Jr. e S. Binato e M. G. C. Resende e D. M. Falcão. “Power Transmission Network Design by Greedy Randomized Adaptive Path Relinking”. *IEEE Transactions on Power Systems*, v. 20(n. 1), pp. 43–49, Feb 2005.

[27] C. DeChamps, J. Vankelecom, and E. Jamoulle. “TRANEX – An Interactive Computer Program for Transmission Expansion”. *IEEE Summer Power Metting*, paper A79 476-3, Jul/Aug 1979.

[28] Y. P. Dusonchet and A. H. El-Abiad. “Transmission Planning using Discrete Dynamic Optimization”. *IEEE Transactions on Power Apparatus and Systems*, PAS-92, pp. 1358–1371, Jul/Aug 1973.

[29] EPRI. *Mathematical Decomposition Techniques for Power System Expansion Planning – Analysis of the Linearized Power Flow Model using the Benders Decomposition Technique*. Technical Report RP 2473-6, EPRI, 1988.

[30] EPRI. *Mathematical Decomposition Techniques for Power System Expansion Planning – Decomposition Methods and Uses*. Technical Report RP 2473-6, EPRI, 1988.

[31] T. A. Feo and M. G. C. Resende. "A Probabilistic Heuristic for a Computationally Difficult Set Covering Problem". *Operations Research Letters*, 8, pp. 67–71, 1989.

[32] T. A. Feo and M. G. C. Resende. "Greedy Randomized Adaptive Search Procedures". *Journal of Global Optimization*, 6, pp. pp. 109–133, 1995.

[33] R. Fischl and W. R. Puntel. "Computer Aided Design of Electric Power Transmission Network". *IEEE Winter Power Meeting*, paper C72-168-8, Jan/Feb 1973.

[34] H. Fisher and G. L. Thompson. "Probabilistic Learning Combinations of Local Job-Shop Scheduling Rules". J. F. Muth and G. L. Thompson, editors, *Industrial Scheduling*, pp. 225–251. Prentice Hall, Englewood Cliffs, NJ, 1963.

[35] M. L. Fisher. "Optimal Solution of Scheduling Problems Using Lagrange Multipliers, Part I". *Operations Research*, 21, pp. 1114–1127, 1973.

[36] C. Fleurent and F. Glover. "Improved constructive multistart strategies for the quadratic assignment problem using adaptive memory". *INFORMS Journal on Computing*, 11, pp. 198–204, 1999.

[37] S. French. *Sequencing and Scheduling:An Introduction to the Mathematics of the Job-Shop*. Ellis Horwood, 1982.

[38] R. A. Gallego, A. B. Alves, A. Monticelli, and R. Romero. "Parallel Simulated Annealing Applied to Long Term Transmission Expansion Planning". *IEEE Transactions on Power Systems*, 12(1), pp. 181–187, Feb 1997.

[39] R. A. Gallego, A. Monticelli, and R. Romero. "Comparative Studies on Non-Convex Optimization Methods for Transmission Network Expansion Planning". *IEEE Transactions on Power Systems*, 13(3), pp. 822–828, Aug 1998.

[40] R. A. Gallego, A. Monticelli, and R. Romero. "Transmission System Expansion Planning by Extended Genetic Algorithm". *IEE Proceedings – Generation, Transmission and Distribution*, volume 145, pp. 329–335, May 1998.

[41] L. L. Garver. “Transmission Network Estimation Using Linear Programming”. *IEEE Transactions on Power Apparatus and Systems*, PAS-89(7), pp. 1688–1687, Sep/Oct 1970.

[42] B. Giffler and G. L. Thompson. “Algorithms for Solving Production Scheduling Problems”. *Operations Research*, 8, pp. 487–503, 1960.

[43] F. Glover. “Tabu Seach - Part I”. *ORSA Journal on Computing*, 1, pp. 90–206, 1989.

[44] F. Glover. “Tabu Seach - Part II”. *ORSA Journal on Computing*, 2, pp. 4–32, 1990.

[45] F. Glover. “Genetic Algorithms and Scatter Search: Unsuspected Potentials”. *Statistics and Computing*, pp. pp. 131–140, 1994.

[46] F. Glover, M. Laguna, and R. Martí. “Fundamentals of Scatter Search and Path Relinking”. *Control and Cybernetics*, 39(3), pp. pp. 653–684, 2000.

[47] D. Goldberg. *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley, 1989.

[48] C. C. Gonzaga. *Estudos de Algoritmos de Busca em Grafos e sua Aplicação a Problemas de Planejamento*. Tese de D.Sc., COPPE – Federal University of Rio de Janeiro, Rio de Janeiro, RJ, Brasil, 1973.

[49] S. Granville and M. V. F. Pereira. *Analisys of the Linearized Power Flow Model in Benders Decomposition*. Technical Report SOL 85-04, System Optimization Lab, Dept. of Operations Research, Stanford University, 1985.

[50] J. Holland. *Adaptation in Natural and Artificial System*. University of Michigan Press, 1975.

[51] A. S. Jain and S. Meeran. *A State-of-the-Art Review of Job-Shop Scheduling Techniques*. Technical report, Department of Applied Physics, Electronic and Mechanical Engineering, University of Dundee, Dundee, Scotland, 1998.

[52] J.A. Jardini, D.S. Ramos, J.S.C. Martini, L.B. Reis, and C.M.V. Tahan. “Brazilian Energy Crisis”. *IEEE Power Engineering Review*, pp. 21–24, April 2002.

[53] S. M. Johnson. “Optimal Two- and Three-Stage Production Schedules with Set-Up Times Included”. *Naval Research Logistics Quarterly*, 1, pp. 61–68, 1954.

[54] J. C. Kaltenbatch, J. Peshon, and E. H. Gehrig. "A Mathematical Optimization Technique for the Expansion of Electrical Power Transmission Systems". *IEEE Transactions on Power Apparatus and Systems*, PAS-89(1), pp. 113–119, Jan 1970.

[55] S. Kirkpatrick. "Optimization by Simulated Annealing: Quantitative Studies". *Journal of Statistical Physics*, 34, pp. 975–986, 1984.

[56] U.G.W. Knight. "The Logical Design of Electrical Networks Using Linear Programming Methods". *IEE Proceedings*, volume 107A, pp. 306–314, 1960.

[57] M. Kolonko. "Some New Results on Simulated Annealing Applied to the Job Shop Scheduling Problem". *European Journal of Operational Research*, 1998. To appear.

[58] G. Lattore-Bayona and I. J. Péres-Arriaga. "CHOPIN, A Heuristic Model for Long Term Transmission Expansion Planning". *IEEE Transaction on Power Systems*, 9(4), pp. 1886–1894, Nov 1994.

[59] S. Lawrence. *Supplement to Resource Constrained Project Scheduling: An Experimental Investigation of Heuristic Scheduling Techniques*. Technical report, Graduate School of Industrial Administration, Carnegie-Mellon University, Pittsburg, October 1994.

[60] S. T. Y. Lee, K. L. Hicks, and E. Hnylicza. "Transmission Expansion by Branch-and-Bound Integer Programming with Optimal Cost-Capacity Curves". *Proceedings of IEEE PES Winter Meeting*, New York, NY, US, 1974.

[61] V. A. Levi and M. S. Calovic. "A New Decomposition Based Method for Optimal Expansion Planning of Large Transmission Networks". *IEEE Transactions on Power Systems*, 6(3), pp. 937–943, Aug 1991.

[62] H.R. Lourenço. "Local Optimization and The Job-Shop Scheduling Problem". *European Journal of Operational Research*, 83, pp. 347–364, 1995.

[63] H.R. Lourenço and M. Zwijnenburg. "Combining the Large-Step Optimization with Tabu-Search: Application to the Job-Shop Scheduling Problem". I.H. Osman and J.P. Kelly, editors, *Meta-Heuristics: Theory and Apllications*, pp. 219–236. Kluwer Academic Publishers, 1996.

[64] H. Matsuo, C. J. Suh, and R. S. Sullivan. *A Controlled Search Simulated Annealing Method for the General Job-Shop Scheduling Problem*. Technical Report 03-04-88, Graduate School of Business, University of Texas at Austin, Austin, TX, March 1988.

[65] M. Minoux. *Mathematical Programming: Theory and Algorithms*. John Wiley and Sons, New York, NY, US, 1986.

[66] A. Monticelli, A. Santos Jr., M. V. F. Pereira, S. H. F. Cunha, J. G. Praça, and B. Park. "Interactive Transmission Network Planning Using a Least-Effort Criterion". *IEEE Transactions on Power Apparatus and Systems,* PAS-101(10), pp. 3919–3925, Oct 1982.

[67] T. E. Morton and D. W. Pentico. *Heuristic Scheduling Systems*. Wiley, 1993.

[68] E. Nowicki and C. Smutnicki. "A Fast Taboo Search Algorithm for the Job Shop Problem". *Management Science*, 42, pp. 797–813, 1996.

[69] G. C. Oliveira, A. P. Costa, and S. Binato. "Large Scale Transmission Network Planning using Optimization and Heuristic Techniques". *IEEE Transactions on Power Systems*, 10(4), pp. 1828–1834, Nov 1995.

[70] J. M. A. Ortiz. *Metodologia de Expansão Automática da Transmissão Utilizando um Algoritmo de Busca Tabu*. Tese de M.Sc., Universidade Federal de Santa Catarina, Florianópolis, SC, Brasil, dezembro 1997.

[71] I. H. Osman and J. P. Kelly. *Meta-Heuristics: Theory and Applications*. Kluwer Academic, 1996.

[72] C. H. Papadimitriou and S. Steiglitz. *Combinatorial Optimization*. Prentice-Hall, 1982.

[73] M. V. F. Pereira. *Aplicação de Análise de Sensibilidade no Planejamento da Expansão de Sistemas de Geração-Transmissão*. Tese de D.Sc., COPPE – Universidade Federal do Rio de Janeiro, Rio de Janeiro, RJ, Brasil, 1985.

[74] M. V. F. Pereira, L. M. V. G. Pinto, G. C. Oliveira, and S. H. F. Cunha. *Composite Generation-Transmission Expansion Planning*. Technical Report EPRI Report RP 2473-9, EPRI, 1987.

[75] M. Pinedo. *Scheduling - Theory, Algorithms and Systems*. Prentice Hall, 1995.

[76] E. Pinson. "The Job Shop Scheduling Problem: A Concise Survey and Some Recent Developments". P. Chrétienne, E.G. Coffman Jr., J.K. Lenstra, and Z. Liu, editors, *Scheduling theory and its applications*, pp. 277–293. John Wiley and Sons, 1995.

[77] L. M. V. G. Pinto and A. Nunes. "A Model for Optimal Transmission Expansion Planning". *Proceedings of the Power System Computing Conference*, pp. 13–23, Graz, Austria, Oct 1990.

[78] M. G. C. Resende. "A GRASP for Job Shop Scheduling". *INFORMS Spring Meeting*, 1997.

[79] R. Romero. *Planejamento da Expansão de Sistemas de Transmissão por Decomposição de Benders Hierarquizada*. Tese de M.Sc., UNICAMP, Campinas, SP, Brasil, dezembro 1989.

[80] R. Romero. *Um Método de Decomposição para o Planejamento a Longo Prazo de Sistemas de Transmissão*. Tese de D.Sc., UNICAMP, Campinas, SP, Brasil, Agosto 1993.

[81] R. Romero, R. A. Gallego, and A. Monticelli. "Transmission System Expansion Planning by Simmulated Annealing". *Proceedings of the conference on Power Industry Computer Applications*, Salt Lake City, US, May 1995.

[82] R. Romero and A. Monticelli. "A Hierarchical Decomposition Approach for Transmission Expansion Planning". *IEEE Transactions on Power Systems*, 9(1), pp. 373–380, Feb 1994.

[83] R. Romero and A. Monticelli. "A Zero-One Implicit Enumeration Method for Optimizing Investments in Transmission Expansion Planning". *IEEE Transactions on Power Systems*, 9(3), Aug 1994.

[84] B. Roy and B. Sussmann. Les problèmes d'ordonnancement avec contraintes disjonctives, 1964.

[85] Thomsen S. *Meta-Heuristics Combined with Branch and Bound*. Technical report, Copenhagen Business School, Copenhagen, Denmark, 1997.

[86] E. L. Silva, J. M. Areiza, G. C. Oliveira, and S. Binato. “Transmission Network Expansion Planning Under a Tabu Search Approach”. *IEEE Transactions on Power Systems*, 16(1), pp. 62–68, Feb 2001.

[87] E. D. Taillard. “Parallel Taboo Search Techniques for The Job Shop Scheduling Problem”. *ORSA Journal on Computing*, 6, pp. 108–117, 1994.

[88] R. J. M. Vaessens, E. H. L. Aarts, and J. K. Lenstra. “Job Shop Scheduling by Local Search”. *INFORMS Journal on Computing*, 8, pp. 302–317, 1996.

[89] P. J. M. Van Laarhoven, E. H. L. Aarts, and J. K. Lenstra. “Job Shop Scheduling by Simulated Annealing”. *Operations Research*, 40, pp. 113–125, 1992.

[90] R. Villasana. *Transmission Network Planning Using Linear and Linear Mixed Integer Programming*. Ph.D. thesis, Ressenlaer Polytechnic Institute, 1984.

[91] Cheung J. Y. “Scheduling”. C. H., editor, *Artificial Neural Networks for Intelligent Manufactoring*, pp. 159–193. Chapman and Hall, 1994.

APÊNDICE A

DADOS DOS SISTEMAS DE TRANSMISSÃO UTILIZADOS

A.1 Sistema Equivalente da Região Sul Brasileira

O sistema equivalente da região Sul brasileira foi, inicialmente, utilizado em [66]. Posteriormente, vários trabalhos utilizaram este sistema teste para ilustrar resultados, veja por exemplo [15, 74, 30, 29, 38, 39, 39, 49, 70, 73, 81, 82, 83, 80, 86].

Tabela A.1: Sistema equiv. da região Sul Brasileira.
Dados das barras.

Barra	Ger.Atual (MW)	Lim.Ger. (MW)	Carga (MW)
1			
2			443.1
4			300.7
5			238.
7			
8			72.2
9			
12			512.
13			185.8
14	944.	1257.	
16	1366.	2000.	
17	1000.	1050.	
18			
19	773.	1670.	
20			1091.
21			
22			81.9

Tabela A.1: continuação.

Barra	Ger.Atual (MW)	Lim.Ger. (MW)	Carga (MW)
23			458.1
24			478.2
26			231.9
27	54.	220.	
32	450.	500.	
33			229.1
34	221.	748.	
35			216.
36			90.1
37	212.	300.	
38			216.
39	221.	600.	
40			262.1
42			1608.
43			
44			79.1
45			86.7
46	599.	700.	
3			
6			
10			
11			
15			
25			
28	730.	800.	
29			
30			
31	310.	700.	
41			

Tabela A.2: Sistema equiv. da região Sul Brasileira.
Dados dos circuitos existentes.

Circuito	Reatância (% base 100MVA)	Lim. Transmissão (MW)
18–20	19.97	200.
27–38	20.80	200.
27–38	20.80	200.
35–38	19.80	200.
37–42	21.05	200.
39–42	20.30	200.

Tabela A.2: continuação.

Circuito	Reatância (% base 100MVA)	Lim. Transmissão (MW)
39–42	20.30	200.
39–42	20.30	200.
44–45	18.64	200.
9–14	17.56	220.
9–14	17.56	220.
13–18	18.05	220.
14–26	16.14	220.
24–34	16.47	220.
8–13	13.48	240.
14–18	15.14	240.
14–18	15.14	240.
24–33	14.48	240.
1–7	6.16	270.
1–2	10.65	270.
1–2	10.65	270.
4–9	9.24	270.
5–9	11.73	270.
5–8	11.32	270.
7–8	10.23	270.
4–5	5.66	270.
4–5	5.66	270.
2–5	3.24	270.
2–5	3.24	270.
12–14	7.40	270.
12–14	7.40	270.
13–20	10.73	270.
14–22	8.40	270.
22–26	7.90	270.
20–23	9.32	270.
20–23	9.32	270.
23–24	7.74	270.
23–24	7.74	270.
26–27	8.32	270.
26–27	8.32	270.
33–34	12.65	270.
27–36	9.15	270.
36–37	10.57	270.
34–35	4.91	270.
34–35	4.91	270.
37–39	2.83	270.
37–40	12.81	270.
40–42	9.32	270.

Tabela A.2: continuação.

Circuito	Reatância (% base 100MVA)	Lim. Transmissão (MW)
38–42	9.07	270.
38–42	9.07	270.
38–42	9.07	270.
42–44	12.06	270.
32–43	3.09	1400.
19–21	2.78	1500.
19–32	1.95	1800.
46–19	2.22	1800.
46–16	2.03	1800.
16–17	0.78	2000.
17–19	0.61	2000.
18–19	1.25	600.
20–21	1.25	600.
42–43	1.25	600.

Tabela A.3: Sistema equiv. da região Sul Brasileira.
Dados dos circuitos candidatos.

Circuito	Reatância (% base 100MVA)	Lim. Transmissão (MW)	Custo (10^3 US$)	Lim. Adições
40–45	22.05	180.	14.00	3
18–20	19.97	200.	12.73	3
27–38	20.80	200.	13.24	3
35–38	19.80	200.	12.63	3
37–42	21.05	200.	13.39	3
39–42	20.30	200.	12.94	3
44–45	18.64	200.	11.93	3
9–14	17.56	220.	11.27	3
13–18	18.05	220.	11.57	3
14–26	16.14	220.	10.41	3
24–34	16.47	220.	10.61	3
8–13	13.48	240.	8.795	3
14–18	15.14	240.	9.805	3
24–33	14.48	240.	9.400	3
4–11	22.46	240.	14.25	3
1–7	6.16	270.	4.350	3
1–2	10.65	270.	7.075	3
4–9	9.24	270.	6.215	3
5–9	11.73	270.	7.730	3
5–8	11.32	270.	7.480	3
7–8	10.23	270.	6.825	3

Tabela A.3: continuação.

Circuito	Reatância (% base 100MVA)	Lim. Transmissão (MW)	Custo (10^3 US$)	Lim. Adições
4–5	5.66	270.	4.045	3
2–5	3.24	270.	2.580	3
12–14	7.40	270.	5.105	3
13–20	10.73	270.	7.125	3
14–22	8.40	270.	5.710	3
22–26	7.90	270.	5.410	3
20–23	9.32	270.	6.270	3
23–24	7.74	270.	5.310	3
26–27	8.32	270.	5.660	3
33–34	12.65	270.	8.290	3
27–36	9.15	270.	6.165	3
36–37	10.57	270.	7.025	3
34–35	4.91	270.	3.590	3
37–39	2.83	270.	2.330	3
37–40	12.81	270.	8.390	3
40–42	9.32	270.	6.270	3
38–42	9.07	270.	6.115	3
42–44	12.06	270.	7.935	3
2–4	8.82	270.	5.965	3
14–15	3.74	270.	2.885	3
5–11	9.15	270.	6.165	3
27–29	9.98	270.	6.670	3
26–29	5.41	270.	3.895	3
28–43	4.06	1200.	46.700	3
28–41	3.39	1300.	39.290	3
16–32	3.11	1400.	36.220	3
32–43	3.09	1400.	35.960	3
19–25	3.25	1400.	37.750	3
25–32	3.19	1400.	37.110	3
32–41	3.09	1400.	35.960	3
19–21	2.78	1500.	32.630	3
31–41	2.78	1500.	32.630	3
17–32	2.32	1700.	27.520	3
19–32	1.95	1800.	23.430	3
46–19	2.22	1800.	26.370	3
46–16	2.03	1800.	24.320	3
46–3	2.03	1800.	24.320	3
16–28	2.22	1800.	26.370	3
16–17	0.78	2000.	10.510	3
17–19	0.61	2000.	8.715	3
46–10	0.81	2000.	10.890	3
46–6	1.28	2000.	16.010	3

Tabela A.3: continuação.

Circuito	Reatância (% base 100MVA)	Lim. Transmissão (MW)	Custo (10^3 US$)	Lim. Adições
21–25	1.74	2000.	21.120	3
31–32	0.46	2000.	7.050	3
28–31	0.53	2000.	7.820	3
28–30	0.58	2000.	8.330	3
41–43	1.39	2000.	17.290	3
18–19	1.25	600.	8.160	3
20–21	1.25	600.	8.160	3
42–43	1.25	600.	8.160	3
15–16	1.25	600.	8.160	3
46–11	1.25	600.	8.160	3
24–25	1.25	600.	8.160	3
29–30	1.25	600.	8.160	3
40–41	1.25	600.	8.160	3
2–3	1.25	600.	8.160	3
5–6	1.25	600.	8.160	3
9–10	1.25	600.	8.160	3

A.2 Sistema Equivalente da Região Sudeste Brasileira

O sistema equivalente da região Sudeste brasileira não foi tão estudado como o sistema equivalente do Sul brasileiro mas já conta também com inúmeros trabalhos, veja por exemplo: [13, 16, 69, 86, 18]. Este sistema foi proposto pela Eletrobrás em 1987 para estudos de expansão da região Sudeste brasileira para os anos de 1995, 2000 e 2005. Os dados de geração e carga utilizados correspondem ao ano de 2000 e estão ilustrados na tabela a seguir.

Tabela A.4: Sistema equiv. da região Sudeste Brasileira.
Dados das barras.

Barra	Nome	Ger.Atual (MW)	Lim.Ger. (MW)	Carga (MW)
12	itaipu	750	0.0	0.0
21	ivaipora	750	1740.0	0.0
27	itaipu	500	3390.0	958.0
33	rosana	440	272.0	0.0
37	assis	230	0.0	62.0
38	xavantes	230	352.0	95.0
39	jurumirim	230	185.0	203.0

Tabela A.4: continuação.

Barra	Nome	Ger.Atual (MW)	Lim.Ger. (MW)	Carga (MW)
40	l.n.garcez	230	60.0	178.0
41	botucatu	230	120.0	633.0
43	p.primaver	500	0.0	0.0
53	itabera	750	0.0	0.0
54	itabera	500	0.0	0.0
60	botucatu	500	0.0	0.0
200	sao paulo	500	936.0	9580.0
201	p.primaver	440	1531.0	499.0
202	taquarucu	440	425.0	45.0
203	jupia	440	4982.0	718.0
204	capivara	440	544.0	147.0
205	assis	440	0.0	134.0
206	a.vermelha	440	1174.0	592.0
207	a.vermelha	500	0.0	0.0
208	bauru	440	234.0	391.0
209	s.simao	500	1429.0	0.0
210	botucatu	440	0.0	0.0
211	itumbiara	500	2953.0	0.0
212	marimbondo	500	1266.0	0.0
213	marimbondo	345	0.0	0.0
214	p.colombia	345	279.0	216.0
215	araraquara	500	0.0	0.0
216	araraquara	440	0.0	828.0
217	campinas	500	0.0	0.0
218	campinas	345	0.0	669.0
219	v.grande	345	323.0	0.0
220	emborcacao	500	1937.0	372.0
221	pocos	345	0.0	542.0
222	pocos	500	0.0	0.0
223	taubate	440	0.0	1007.0
224	estreito	345	939.0	0.0
225	mascarenha	345	407.0	401.0
226	jaguara	500	0.0	0.0
227	jaguara	345	531.0	398.0
228	c.paulista	500	0.0	8.0
229	pimenta	345	0.0	151.0
230	itutinga	345	0.0	0.0
231	b.h.	500	0.0	470.0
232	angra	500	2182.0	67.0
233	taubate	500	0.0	0.0
234	adriano	500	1257.0	7423.0
235	barb-jfora	345	278.0	1107.0

Tabela A.4: continuação.

Barra	Nome	Ger.Atual (MW)	Lim.Ger. (MW)	Carga (MW)
236	gafanhoto	345	0.0	151.0
237	adriano	345	231.0	0.0
238	campos	345	409.0	287.0
239	vitoria	345	143.0	1156.0
240	mesquita	500	88.0	829.0
241	furnas	345	1116.0	0.0
242	n.ponte	500	766.0	0.0
243	b.h.	345	97.0	0.0
244	jequitinho	500	317.0	0.0
245	vitoria	500	0.0	0.0
246	itumbiara	345	549.0	882.0
247	bandeirant	345	0.0	807.0
248	bandeirant	500	0.0	0.0
249	brasilia	345	118.0	994.0
250	brasilia	500	0.0	0.0
251	a.tocantis	500	2798.0	0.0
252	t. sul	345	0.0	633.0
253	doce	500	499.0	0.0
254	conchal	440	0.0	647.0
255	rib.preto	440	188.0	848.0
256	sta.barb.	440	0.0	1712.0
257	t.sul	500	0.0	0.0
259	r.preto	500	0.0	0.0
260	sao paulo	440	0.0	0.0
261	sao paulo	750	0.0	0.0
262	sao paulo	345	0.0	0.0
263	sao paulo	230	0.0	0.0
267	t.marias	345	954.0	1159.0
272	t.oeste	345	0.0	0.0
273	t.oeste	500	0.0	0.0

Tabela A.5: Sistema equiv. da região Sudeste Brasileira. dados dos circuitos existentes.

Circuito	Reatância (% base 100MVA)	Lim. Transmissão (MW)
12–21	0.91	2200.0
12–21	0.91	2200.0
12–21	0.91	2200.0
12–27	0.34	1650.0
12–27	0.34	1650.0

Tabela A.5: continuação.

Circuito	Reatância (% base 100MVA)	Lim. Transmissão (MW)
12–27	0.34	1650.0
12–27	0.34	1650.0
21–53	0.78	2200.0
21–53	0.78	2200.0
21–53	0.78	2200.0
37–40	4.92	200.0
37–40	4.92	200.0
37–205	6.67	150.0
37–205	6.67	150.0
38–39	5.30	200.0
38–39	5.30	200.0
38–40	3.37	200.0
38–41	13.20	200.0
39–41	8.48	200.0
200–217	1.47	1200.0
200–228	2.47	1200.0
200–233	1.41	1200.0
200–260	0.02	9999.0
200–261	0.02	9999.0
200–262	0.02	9999.0
200–263	0.02	9999.0
201–33	0.34	1100.0
201–202	1.85	1100.0
201–202	1.85	1100.0
202–203	3.70	1100.0
202–204	1.34	1100.0
202–205	3.07	1100.0
203–206	2.45	1100.0
203–208	5.12	1100.0
203–208	5.12	1100.0
203–208	5.12	1100.0
203–208	5.12	1100.0
203–216	6.21	1100.0
203–216	6.21	1100.0
204–205	1.78	1100.0
205–208	2.30	1100.0
205–210	3.53	1100.0
205–210	3.53	1100.0
206–216	5.46	1100.0
206–255	5.32	1100.0
207–206	1.33	750.0
207–209	1.47	1200.0

Tabela A.5: continuação.

Circuito	Reatância (% base 100MVA)	Lim. Transmissão (MW)
207–212	2.10	1200.0
208–216	1.75	1100.0
209–211	2.47	1200.0
209–226	5.21	1200.0
210–256	1.71	1100.0
210–256	1.71	1100.0
211–220	2.16	1200.0
211–246	2.14	560.0
212–213	2.14	560.0
212–215	3.10	1200.0
212–215	3.10	1200.0
213–214	2.45	600.0
214–219	1.38	600.0
214–246	6.27	600.0
215–217	2.58	1200.0
215–222	2.64	1200.0
216–254	2.69	1100.0
216–256	2.42	1100.0
217–218	2.14	560.0
217–228	3.27	1200.0
218–221	3.95	600.0
219–224	3.51	600.0
219–227	2.76	600.0
220–242	1.32	1200.0
220–273	2.98	1200.0
221–222	2.14	560.0
221–224	6.18	600.0
221–224	6.18	600.0
221–241	4.23	600.0
221–241	4.23	600.0
222–228	2.86	1200.0
224–225	1.03	600.0
224–227	0.75	600.0
224–241	4.14	600.0
225–241	3.29	600.0
226–227	2.25	400.0
226–227	2.25	400.0
226–227	2.25	400.0
226–231	4.87	1200.0
226–242	1.47	1200.0
226–257	6.26	1200.0
227–229	5.70	600.0

Tabela A.5: continuação.

Circuito	Reatância (% base 100MVA)	Lim. Transmissão (MW)
227–229	5.70	600.0
228–232	1.51	1200.0
228–233	1.21	1200.0
228–234	2.55	1350.0
228–234	2.55	1350.0
229–235	7.24	600.0
229–236	3.29	600.0
229–241	2.00	600.0
229–243	6.74	600.0
230–237	6.21	600.0
230–237	6.21	600.0
230–241	6.24	600.0
230–241	6.24	600.0
231–240	2.52	1200.0
231–243	2.25	400.0
231–243	2.25	400.0
231–243	2.25	400.0
231–273	3.03	1200.0
232–234	1.72	1200.0
232–234	1.72	1200.0
232–234	1.72	1200.0
233–223	1.11	900.0
234–237	2.14	560.0
234–237	2.14	560.0
235–252	2.17	600.0
236–243	3.80	600.0
237–238	8.65	600.0
237–238	8.65	600.0
238–239	7.43	600.0
238–239	7.43	600.0
240–257	2.24	1200.0
242–273	2.69	1200.0
243–252	3.95	600.0
243–267	6.81	600.0
245–239	2.14	560.0
246–247	6.05	600.0
246–247	6.05	600.0
247–249	5.62	600.0
247–249	5.62	600.0
249–250	2.14	560.0
249–250	2.14	560.0
249–250	2.14	560.0

Tabela A.5: continuação.

Circuito	Reatância (% base 100MVA)	Lim. Transmissão (MW)
250–251	2.36	1300.0
250–251	2.36	1300.0
255–256	4.08	1100.0
257–252	2.25	400.0
257–252	2.25	400.0
260–208	3.98	1100.0
260–208	3.98	1100.0
260–208	3.98	1100.0
260–208	3.98	1100.0
260–216	4.92	1100.0
260–223	2.44	1100.0
260–223	2.44	1100.0
260–254	2.69	1100.0
260–256	2.52	1100.0
261–53	0.92	2200.0
261–53	0.92	2200.0
261–53	0.92	2200.0
262–218	2.82	600.0
262–221	6.58	600.0
262–221	6.58	600.0
262–221	6.58	600.0
263–41	19.20	200.0
267–272	4.93	600.0
272–273	2.25	400.0
272–273	2.25	400.0

Tabela A.6: Sistema equiv. da região Sudeste Brasileira. Dados dos circuitos candidatos.

Circuito	Reatância (% base 100MVA)	Lim. Transmissão (MW)	Custo (10^3 US$)	Lim. Adições
12–21	0.91	2200.0	85.9	3
12–27	0.34	1650.0	11.7	3
21–53	0.78	2200.0	73.7	3
27–200	4.37	1200.0	67.6	3
37–40	4.92	200.0	4.1	3
37–205	6.67	150.0	3.6	3
38–39	5.30	200.0	4.4	3
38–40	3.37	200.0	2.8	3
38–41	13.20	200.0	11.0	3
39–41	8.48	200.0	7.0	3

Tabela A.6: continuação.

Circuito	Reatância (% base 100MVA)	Lim. Transmissão (MW)	Custo (10^3 US$)	Lim. Adições
53–54	0.34	1650.0	11.7	3
54–60	0.72	1200.0	25.8	3
200–60	1.11	1200.0	39.5	3
200–217	1.47	1200.0	17.6	3
200–228	2.47	1200.0	29.6	3
200–233	1.41	1200.0	16.8	3
200–260	0.02	9999.0	51.0	3
200–261	0.02	9999.0	51.0	3
200–262	0.02	9999.0	51.0	3
200–263	0.02	9999.0	51.0	3
201–33	0.34	1100.0	3.6	3
201–43	1.33	750.0	6.1	3
201–202	1.85	1100.0	19.6	3
201–203	5.34	1100.0	56.8	3
202–203	3.70	1100.0	39.3	3
202–204	1.34	1100.0	14.3	3
202–205	3.07	1100.0	32.7	3
203–206	2.45	1100.0	26.1	3
203–208	5.12	1100.0	54.5	3
203–216	6.21	1100.0	66.1	3
204–205	1.78	1100.0	18.9	3
205–208	2.30	1100.0	24.5	3
205–210	3.53	1100.0	37.5	3
206–216	5.46	1100.0	58.1	3
206–255	5.32	1100.0	56.6	3
207–206	1.33	750.0	6.1	3
207–209	1.47	1200.0	17.6	3
207–212	2.10	1200.0	25.1	3
208–216	1.75	1100.0	18.6	3
209–211	2.47	1200.0	29.6	3
209–212	3.15	1200.0	37.6	3
209–226	5.21	1200.0	62.3	3
210–41	6.67	150.0	3.6	3
210–60	1.33	750.0	6.1	3
210–256	1.71	1100.0	18.2	3
211–212	3.22	1200.0	38.5	3
211–220	2.16	1200.0	25.8	3
211–246	2.14	560.0	5.0	3
211–248	2.76	1200.0	33.0	3
211–250	4.79	1200.0	57.3	3
212–213	2.14	560.0	5.0	3
212–215	3.10	1200.0	37.1	3

Tabela A.6: continuação.

Circuito	Reatância (% base 100MVA)	Lim. Transmissão (MW)	Custo (10^3 US$)	Lim. Adições
212–226	2.70	1200.0	32.2	3
213–214	2.45	600.0	9.7	3
214–219	1.38	600.0	5.5	3
214–246	6.27	600.0	24.9	3
215–217	2.58	1200.0	30.8	3
215–222	2.64	1200.0	31.5	3
215–226	3.30	1200.0	39.4	3
216–215	1.33	750.0	6.1	3
216–254	2.69	1100.0	28.6	3
216–255	2.69	1100.0	28.6	3
216–256	2.42	1100.0	25.7	3
217–218	2.14	560.0	5.0	3
217–222	2.25	1200.0	26.9	3
217–228	3.27	1200.0	39.0	3
218–221	3.95	600.0	15.7	3
219–224	3.51	600.0	14.0	3
219–227	2.76	600.0	11.0	3
220–242	1.32	1200.0	17.4	3
220–250	4.49	1200.0	53.7	3
220–273	2.98	1200.0	39.4	3
221–222	2.14	560.0	5.0	3
221–224	6.18	600.0	24.6	3
221–241	4.23	600.0	16.8	3
222–226	3.75	1200.0	44.8	3
222–228	2.86	1200.0	34.2	3
224–225	1.03	600.0	4.1	3
224–227	0.75	600.0	3.0	3
224–241	4.14	600.0	16.4	3
225–241	3.29	600.0	13.1	3
226–227	2.25	400.0	4.6	3
226–231	4.87	1200.0	64.4	3
226–242	1.47	1200.0	19.4	3
226–257	6.26	1200.0	82.9	3
226–259	2.25	1200.0	26.9	3
227–229	5.70	600.0	22.7	3
228–232	1.51	1200.0	18.1	3
228–233	1.21	1200.0	14.5	3
228–234	2.55	1350.0	30.4	3
229–235	7.24	600.0	28.8	3
229–236	3.29	600.0	13.1	3
229–241	2.00	600.0	8.0	3
229–243	6.74	600.0	26.8	3

Tabela A.6: continuação.

Circuito	Reatância (% base 100MVA)	Lim. Transmissão (MW)	Custo (10^3 US$)	Lim. Adições
230–237	6.21	600.0	24.7	3
230–241	6.24	600.0	24.8	3
231–240	2.52	1200.0	33.3	3
231–243	2.25	400.0	4.6	3
231–250	9.36	1200.0	111.9	3
231–257	1.68	1200.0	22.2	3
231–273	3.03	1200.0	40.1	3
232–234	1.72	1200.0	20.6	3
233–223	1.11	900.0	6.4	3
234–237	2.14	560.0	5.0	3
234–257	70.18	1200.0	12.6	3
235–237	3.62	600.0	12.9	3
235–238	6.74	600.0	24.1	3
235–252	2.17	600.0	7.7	3
236–243	3.80	600.0	15.1	3
237–238	8.65	600.0	30.9	3
238–239	7.43	600.0	26.5	3
240–244	7.49	1200.0	89.5	3
240–245	3.75	1200.0	44.8	3
240–253	2.45	1200.0	32.4	3
240–257	2.24	1200.0	29.6	3
242–273	2.69	1200.0	41.6	3
243–252	3.95	600.0	14.1	3
243–267	6.81	600.0	27.1	3
244–245	7.49	1200.0	89.5	3
245–239	2.14	560.0	5.0	3
245–253	1.50	1200.0	17.9	3
246–247	6.05	600.0	21.6	3
246–249	10.52	600.0	37.5	3
247–249	5.62	600.0	20.1	3
248–247	2.14	560.0	5.0	3
248–250	1.87	1200.0	22.4	3
248–251	7.71	1200.0	92.2	3
249–250	2.14	560.0	5.0	3
250–251	2.36	1300.0	56.7	3
255–256	4.08	1100.0	43.4	3
255–259	1.33	750.0	6.1	3
257–252	2.25	400.0	4.6	3
260–208	3.98	1100.0	42.3	3
260–216	4.92	1100.0	52.3	3
260–223	2.44	1100.0	25.9	3
260–254	2.69	1100.0	28.6	3

Tabela A.6: continuação.

Circuito	Reatância (% base 100MVA)	Lim. Transmissão (MW)	Custo (10^3 US$)	Lim. Adições
260–256	2.52	1100.0	26.8	3
261–53	0.92	2200.0	86.8	3
262–218	2.82	600.0	11.2	3
262–221	6.58	600.0	23.5	3
263–41	19.20	200.0	16.0	3
267–272	4.93	600.0	17.6	3
272–273	2.25	400.0	4.6	3

Tabela A.7: Sistema equiv. da região Sudeste Brasileira.
Conjunto reduzido de circuitos candidatos.

Circuito	Reatância (% base 100MVA)	Lim. Transmissão (MW)	Custo (10^3 US$)
207–206	1.33	750.0	6.1
207–209	1.47	1200.0	17.6
210–41	6.67	150.0	3.6
210–41	6.67	150.0	3.6
211–246	2.14	560.0	5.0
216–215	1.33	750.0	6.1
220–242	1.32	1200.0	17.4
220–242	1.32	1200.0	17.4
220–250	4.49	1200.0	53.7
221–224	6.18	600.0	24.6
224–227	0.75	600.0	3.0
224–227	0.75	600.0	3.0
226–227	2.25	400.0	4.6
226–242	1.47	1200.0	19.4
226–242	1.47	1200.0	19.4
226–259	2.25	1200.0	26.9
234–237	2.14	560.0	5.0
244–245	7.49	1200.0	89.5
245–239	2.14	560.0	5.0
245–253	1.50	1200.0	17.9
249–250	2.14	560.0	5.0
250–251	2.36	1300.0	56.7
255–259	1.33	750.0	6.1
255–259	1.33	750.0	6.1
204–205	1.78	1100.0	18.9
209–211	2.47	1200.0	29.6
214–219	1.38	600.0	5.5
214–246	6.27	600.0	24.9

Tabela A.7: continuação.

Circuito	Reatância (% base 100MVA)	Lim. Transmissão (MW)	Custo (10^3 US$)
248–247	2.14	560.0	5.0
248–247	2.14	560.0	5.0
248–250	1.87	1200.0	22.4
37–205	6.67	150.0	3.6
38–40	3.37	200.0	2.8
39–41	8.48	200.0	7.0